databricks

14 DAYS

AI CHALLENGE

DAY 01

Topic:

Platform Setup & First Steps

Challenge:

1. Create Databricks Community Edition account
2. Navigate Workspace, Compute, Data Explorer
3. Create first notebook
4. Run basic PySpark commands

#DatabricksWithIDC

Day 1 Reflection

1. **Experienced collaborative workflows using shared Databricks notebooks**

2. **Understood how a unified platform connects data, analytics, and AI**

3. **Built a strong foundation for scalable, team-driven data projects**

By: Ruchi Wange

CONTEXT & GOALS

Challenge Overview

14 Days AI Challenge

An intensive program designed to learn and apply modern AI technologies directly on the Databricks Lakehouse platform.

Organized By

A collaborative effort by industry leaders:
Indian Data Club • Databricks • Codebasics

Daily Format

Structured for consistency:

1. Short, focused learning goal
2. Hands-on notebook execution

Today's Focus (Day 01)

- ✓ Set up Databricks Community Edition
- ✓ Explore Workspace Interface
- ✓ Run First PySpark Commands

PLATFORM OVERVIEW

What is Databricks?

Unified Data + AI Platform

Built on the Lakehouse architecture, it unifies your data warehousing and data science workloads into a single platform, eliminating data silos.

Core Components

- **Compute:** Apache Spark (distributed processing)
- **Storage:** Delta Lake (reliability & performance)
- **Governance:** Unity Catalog

Integrated Workflows

Seamlessly combines data engineering, real-time analytics, and ML/LLM workflows, allowing teams to collaborate on the same data.

The Outcome

Enables faster pipelines, fully governed data, and production-grade AI solutions, all within a single collaborative environment.

KEY BENEFITS

Why use Databricks for AI/ML?

Scalability

Seamlessly handle massive datasets with auto-scaling clusters and serverless compute options. Scale from single-node testing to distributed training effortlessly.

Productivity

A unified workspace bringing data, code, and visualizations together. Features collaborative notebooks, integrated git repos, and built-in dashboards.

Performance

Achieve lightning-fast query execution with the Photon engine and Delta Lake storage optimizations, including caching and data skipping.

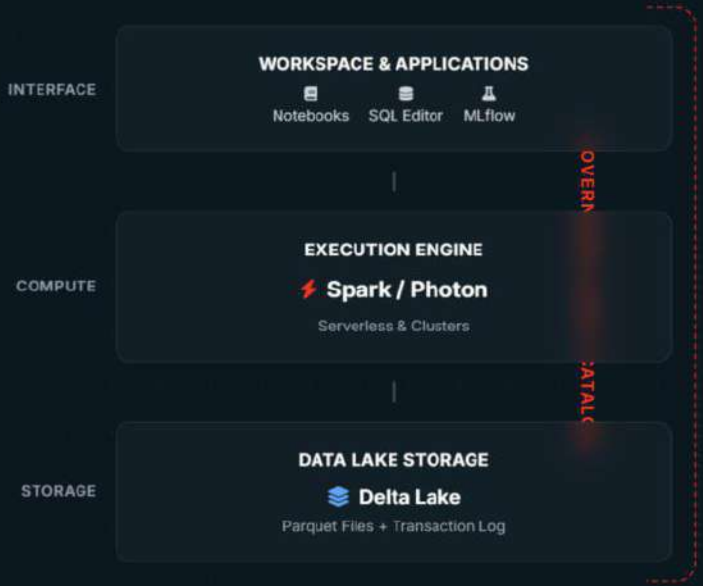
Governance

Secure your AI assets with **Unity Catalog**. Manage permissions, audit logs, and data lineage across all workspaces from a central point.

ARCHITECTURE OVERVIEW

How Databricks Works

Lakehouse Platform



Unified Storage Layer

Data resides in your cloud account (S3/ADLS/GCS) in open formats. **Delta Lake** adds ACID transactions and versioning on top of Parquet files.

Decoupled Compute

Compute resources (clusters) spin up on demand to process data, then shut down to save costs. **Photon engine** accelerates queries.

Collaborative Workspace

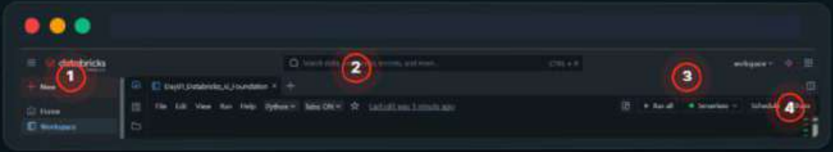
Data Engineers, Scientists, and Analysts work in the same environment using **Python, SQL, R, or Scala** within interactive notebooks.

Unity Catalog Governance

A single centralized layer to manage permissions, audit logs, and data lineage across all workspaces and clouds.

INTERFACE OVERVIEW

Explore the Workspace



The Databricks workspace is your unified command center for data engineering, data science, and analytics tasks.

1 Sidebar Navigation

Access core areas: **Workspace** (files), **Catalog** (data), **Workflows**, and Compute.

2 Global Search Bar

Quickly find notebooks, tables, dashboards, or documentation across the entire environment.

3 Notebook Controls

Manage compute attachment, schedule jobs, and share your work with collaborators.

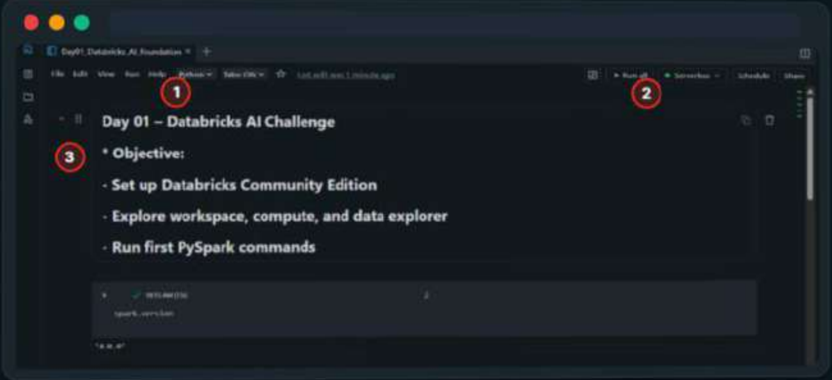
4 Contextual Right Rail

Access comments, version history, variable explorer, and performance metrics.

Tip: Use `Ctrl + P` to open the search palette instantly.

ENVIRONMENT SETUP

First Steps: Create a Notebook



Day 01 – Databricks AI Challenge

* Objective:

- Set up Databricks Community Edition

- Explore workspace, compute, and data explorer

- Run first PySpark commands

```
spark.version
```

Before analyzing data, you need to set up your environment. Notebooks are the core workspace for mixing code, visuals, and text.

1 **Create a Python Notebook**
Navigate to **Workspace > Users > Your Folder**, right-click and select **Create > Notebook**. Set the default language to Python.

2 **Attach Compute**
Use the dropdown in the top-right toolbar to attach your notebook to an active cluster (e.g., Community Edition cluster).

3 **Define Objectives**
Start with a Markdown cell (`%md`) to document your goals, making your analysis easy to follow.

4 **Execute Cells**
Run code using `Shift + Enter` or the "Run" button. Keep "Tabs: ON" for easier multitasking.

Tip: Use the "Run All" button to execute the entire notebook from top to bottom when reviewing your work.

CORE CONCEPTS

PySpark Basics You'll Use Today

SparkSession

The entry point to programming Spark with the Dataset and DataFrame API. In Databricks notebooks, this is automatically available as the variable `spark` .

DataFrame

A distributed collection of data organized into named columns. Think of it like a table in a relational database or a pandas DataFrame, but optimized for distributed processing.

Lazy Evaluation

Spark doesn't compute transformations immediately. Instead, it builds a "logical plan" and waits until an action is called. This optimization is key to performance.

The Common Pattern

→ **Create:** Load data or create DataFrame

→ **Transform:** `select` , `filter` (Builds Plan)

✓ **Action:** `show()` , `count()` (Executes)

FIRST STEPS

Command 1: Check Spark Version

Python 3

Goal

Before running complex transformations, verify that your environment is ready. The `spark` object is automatically available in Databricks notebooks.

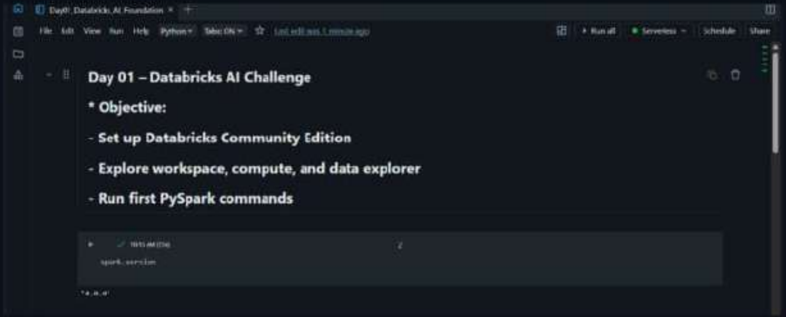
Cmd 1

```python
# Check the active Spark version
spark.version
```

>_ OUTPUT

```
'4.0.0'
```

Note: Seeing a version number confirms that the distributed computing cluster is attached and responsive.



Day 01 – Databricks AI Challenge

* Objective:

- Set up Databricks Community Edition

- Explore workspace, compute, and data explorer

- Run first PySpark commands

```
spark.version
```

```
'4.0.0'
```

● Cluster: Active

DATA STRUCTURE BASICS

Command 2: Create a DataFrame

Python 3

Goal

Build a small table in memory to understand Spark's structure. We define raw data (rows) and a schema (columns), then convert it into a distributed `DataFrame`.

Cmd 2

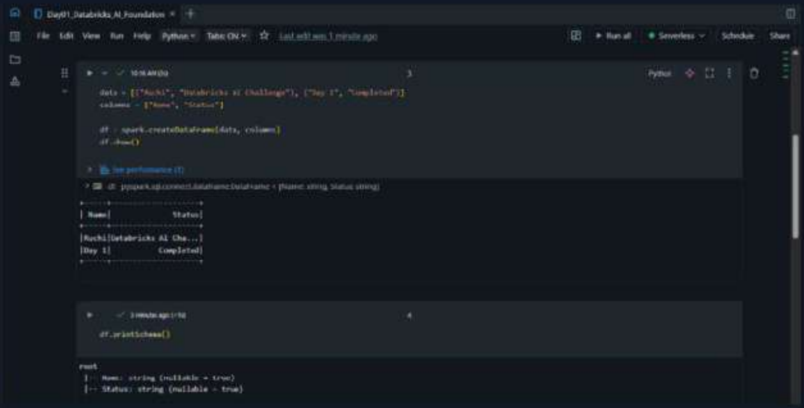
```python
data = [("Ruchi", "Databricks AI Challenge"), ...]
```

>_ OUTPUT PREVIEW

```
+-----+--------------------+
| Name|              Status|
+-----+--------------------+
|Ruchi|Databricks AI Challe...|
|Day 1|           Completed|
+-----+--------------------+
```

Note: `df.show()` is an Action that triggers the job execution.

RESULT



```python
data = [("Ruchi", "Databricks AI Challenge"), ("Day 1", "Completed")]
columns = ["Name", "Status"]

df = spark.createDataFrame(data, columns)
df.show()
```

```
df: pyspark.sql.connect.dataframe.DataFrame = [Name: string, Status: string]
+-----+--------------------+
| Name|              Status|
+-----+--------------------+
|Ruchi|Databricks AI Cha...|
|Day 1|           Completed|
+-----+--------------------+
```

```python
df.printSchema()
```

```
root
 |-- Name: string (nullable = true)
 |-- Status: string (nullable = true)
```

● Job: Completed (2s)

FIRST STEPS

Command 3: Inspect Schema

Python 3

Goal

Spark DataFrames have a defined schema (column names and data types). Using `printSchema()` helps you understand the structure of your data and verify type inference.

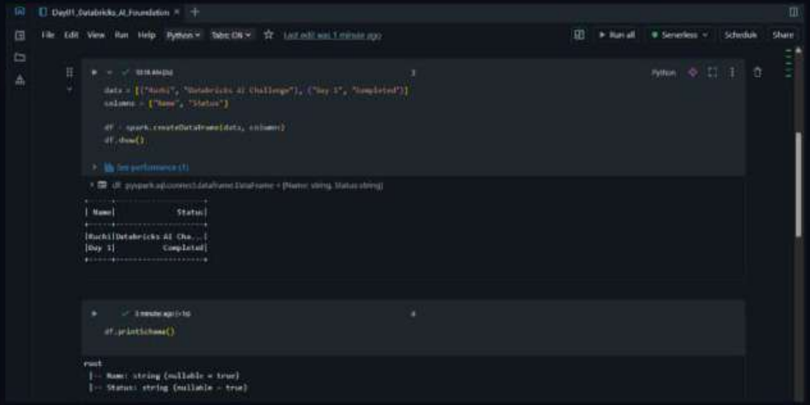
Cmd 3

```python
# View the structure of the DataFrame
df.printSchema()
```

>_ OUTPUT

```
root
 |-- Name: string (nullable = true)
 |-- Status: string (nullable = true)
```

Insight: Unlike Pandas, Spark is strongly typed. Knowing if a column is `nullable` is crucial for data quality checks.



Day01_Databricks_AI_Foundation
File Edit View Run Help Python Tabs: ON
Last edit was 1 minute ago
Run all
Serverless
Schedule
Share
Python

```python
data = [("Ruchi", "Databricks AI Challenge"), ("Day 1", "Completed")]
columns = ["Name", "Status"]

df = spark.createDataFrame(data, columns)
df.show()
```

See performance (1)

df: pyspark.sql.connect.dataframe.DataFrame = [Name: string, Status: string]

```
+-----+--------------------+
| Name|              Status|
+-----+--------------------+
|Ruchi|Databricks AI Cha...|
|Day 1|           Completed|
+-----+--------------------+
```

1 minute ago (<1s)

```python
df.printSchema()
```

```
root
 |-- Name: string (nullable = true)
 |-- Status: string (nullable = true)
```

● Schema: Inferred

TRANSFORMATIONS & ACTIONS

Command 4: Select and Show

Python 3

Goal

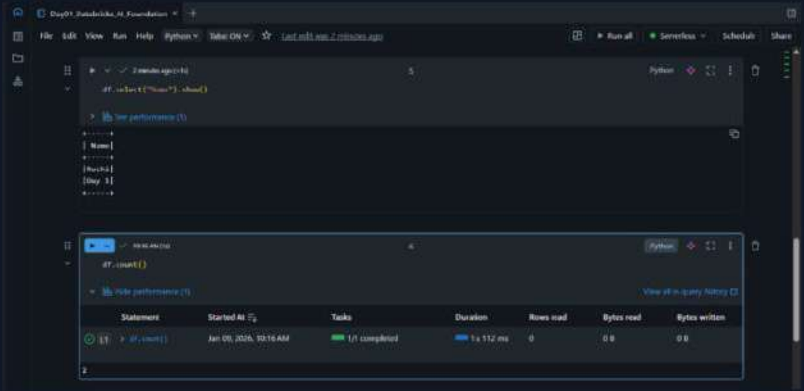
Demonstrate how to project specific columns using a transformation (`select`) and trigger execution with an action (`show`).

Cmd 4

```python
# Select "Name" column and show content
df.select("Name").show()
```

OUTPUT

```
+-----+
| Name|
+-----+
|Ruchi|
|Day 1|
+-----+
```



```
df.select("Name").show()
```

See performance (1)

```
+-----+
| Name|
+-----+
|Ruchi|
|Day 1|
+-----+
```

```
df.count()
```

Hide performance (1)

View all in query history

Statement	Started At	Tasks	Duration	Rows read	Bytes read	Bytes written
df.count()	Jan 09, 2026, 10:16 AM	1/1 completed	1 s 112 ms	0	0 B	0 B

```
2
```

● Execution Time: < 1s

⚡ **Insight:** Spark lazily builds a plan for `select()` but only executes it when you call `show()`.

TRANSFORMATIONS VS ACTIONS

Command 5: Count Rows

Python 3

Goal

Execute a simple Action to trigger computation. Unlike transformations (which are lazy), `count()` forces Spark to process the data and return a single value to the driver.

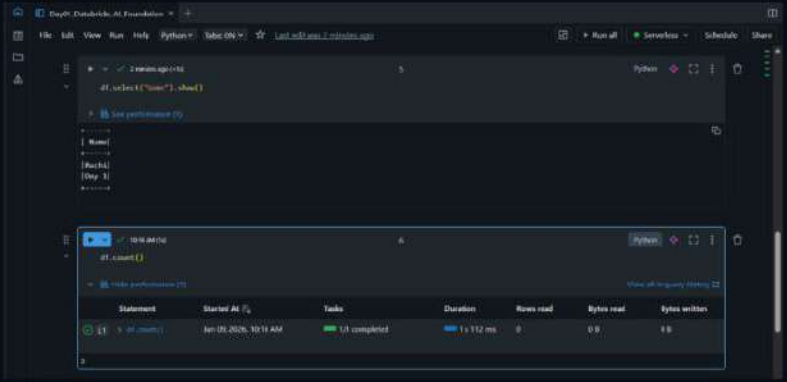
Cmd 6

```python
# Trigger an action to count rows
df.count()
```

>_ OUTPUT

```
2
```

⚡ **Insight:** This is where the actual work happens. The Spark job is launched, tasks are distributed, and the result is sent back.



```
df.select("Name").show()
```

```
+------+
|  Name|
+------+
|Ruchi|
|Day  1|
+------+
```

```
df.count()
```

Statement	Started At	Tasks	Duration	Rows read	Bytes read	Bytes written
df.count()	Jan 09, 2026, 10:11 AM	1/1 completed	1 s 112 ms	0	0 B	0 B

```
2
```

● Spark Job: Completed (0.4s)

Thank You!!

Presented by: Ruchi Wange