**Hexaware Data Engineering Project**

**Student Name:** Abhilash Gowda S

Mahesh Ratneshwar Godale

**Trainer:** Harinya Ma'am

**Title:** Agricultural Crop Yield in Indian States Analysis using Azure

**Problem Statement:**

Data Lake Exploration and Optimization:

Use PySparkSQL to explore and optimize data stored in a data lake. Leverage Azure Databricks for efficient querying and analysis of data lake contents..

**Project Overview**

**Data Lake Exploration and Optimization using PySparkSQL and Azure**

This project explores and optimizes agricultural data stored in a data lake using PySparkSQL and Azure Databricks. The "Agricultural Crop Yield in Indian States Dataset" (1997-2020) serves as the basis for analyzing trends in crop production, rainfall, fertilizer usage, and pesticide application. Optimization techniques like filtering, partitioning, and caching are applied to improve query performance and identify factors influencing crop yields across Indian states.

Key dataset features include crop types, seasons, cultivation areas, production quantities, rainfall, and input usage. The project emphasizes the importance of leveraging modern data platforms like Azure Databricks for handling vast datasets efficiently. Advanced querying and data transformations ensure deeper insights into patterns and correlations, empowering stakeholders with actionable intelligence.

In addition, the project aims to provide a scalable and reusable framework for analyzing agricultural datasets in big data environments. By addressing critical questions on crop yield trends and factors, the findings contribute to better decision-making in agricultural resource planning and policy development. The outcomes include an efficient workflow for data lake analysis, detailed insights into yield patterns, and practical recommendations to enhance agricultural productivity and resource management.

**Objective**

The primary objectives of this project are to:

1. Explore the Agricultural Crop Yield dataset stored in a data lake using PySparkSQL.
2. Optimize data processing and querying within the data lake using techniques like partitioning, indexing, and caching.
3. Leverage Azure Databricks for efficient data analysis and visualization of the dataset.
4. Enhance query performance and reduce processing time for large-scale data exploration.
5. Ensure the optimized solution is scalable and can handle large datasets effectively.

**Data Overview**

The dataset used in this project is sourced from **Kaggle**, a popular platform for sharing and exploring datasets. It contains agricultural data for multiple crops cultivated across various states in India, spanning from the year **1997 to 2020**. This dataset is crucial for understanding crop yield prediction, as it includes key features such as crop types, crop years, cropping seasons, states, areas under cultivation, production quantities, annual rainfall, fertilizer usage, pesticide usage, and calculated yields.

The dataset provides valuable insights into the relationship between various factors like weather conditions, resource usage, and crop yield. This enables better decision-making in agricultural planning and policy formulation.

**Columns Description:**

* **Crop**: The name of the crop cultivated.
* **Crop\_Year**: The year in which the crop was grown.
* **Season**: The specific cropping season (e.g., Kharif, Rabi, Whole Year).
* **State**: The Indian state where the crop was cultivated.
* **Area**: The total land area (in hectares) under cultivation for the specific crop.
* **Production**: The quantity of crop production (in metric tons).
* **Annual\_Rainfall**: The annual rainfall received in the crop-growing region (in mm).
* **Fertilizer**: The total amount of fertilizer used for the crop (in kilograms).
* **Pesticide**: The total amount of pesticide used for the crop (in kilograms).
* **Yield**: The calculated crop yield (production per unit area).

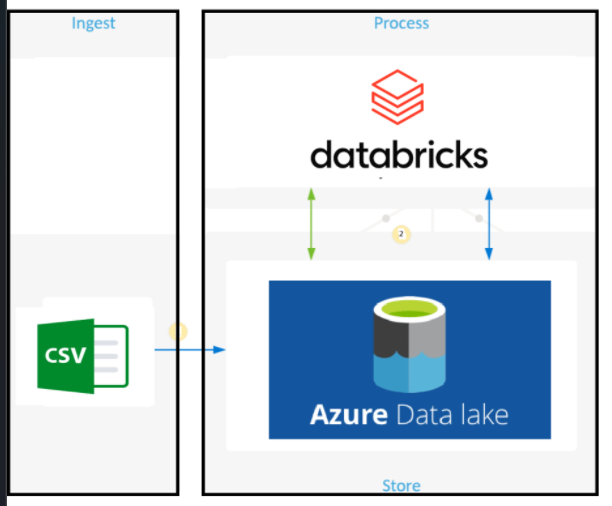
The dataset provides a rich source of data for performing detailed analysis of agricultural trends and patterns. By examining how different variables interact—such as the effect of rainfall on crop yield or the relationship between fertilizer use and productivity—the project aims to uncover valuable insights into improving agricultural productivity. Furthermore, by utilizing advanced data lake technologies and optimization techniques, the project seeks to improve the efficiency of querying and analyzing large volumes of agricultural data, making it easier to derive actionable insights for stakeholders in the agriculture industry.

This data serves as a foundation for further exploration into precision agriculture, resource management, and policy decisions aimed at optimizing crop production and sustaining agricultural growth in India.

**Architecture diagram**

Below is the architecture diagram illustrating the connection between Azure Databricks, the CSV file, and Azure Data Lake. In this setup, the CSV file containing agricultural crop yield data is ingested into Azure Data Lake, providing a centralized storage solution.

Azure Databricks is then used to process and analyze the data stored in the lake, leveraging its distributed computing capabilities for efficient querying and data optimization.



**How it works:**

**Source Data File Name**

The source data file, **crop\_yield.csv**, is stored in Azure Data Lake. This CSV file contains essential data related to crop yields, such as crop types, yield measurements, geographical locations, and timestamps.

These attributes provide valuable insights for analyzing crop production trends, regional variations, and yield forecasts. The data is accessed from Azure Data Lake and loaded into **Azure Databricks** using **PySpark** for exploration, cleaning, and processing. This setup ensures efficient handling of large datasets for further analysis and decision-making.

**Workflow:**

**1. Data Access with Azure Databricks:**:

* Use **Azure Databricks** to connect to **Azure Data Lake** and access the data using the SAS Token for secure access.
* Access the data stored in a CSV file format (e.g., crop\_yield.csv) from the Data Lake using PySpark

**2. Data Exploration and Analysis**:

* **PySparkSQL** is used to load, explore, and analyze the data from the Data Lake.
* Perform basic operations like viewing the data schema and displaying sample records using Spark DataFrame methods.

**3. Data Optimization**:

* Leverage Spark to perform any necessary optimizations such **as filtering, transforming,** or **aggregating** the data efficiently for analysis.
* Perform additional optimization by leveraging Spark's built-in functions to improve the overall performance of the data processing tasks, ensuring faster execution times during exploratory data analysis and subsequent transformations.

**Azure Resources Used for this Project**

**Azure Databricks** and **Azure Data Lake** are essential components for large-scale data processing and analysis. Together, they enable efficient exploration and optimization of data stored in a data lake.

**Azure Databricks**

Azure Databricks is an Apache Spark-based analytics platform optimized for Azure. It facilitates collaborative data engineering and data science workflows, providing a unified environment for real-time data processing, advanced analytics, and machine learning.

* **Unified Workspace**: Supports interactive development in languages like PySpark,SQL.
* **Scalable Data Processing**: Leverages Apache Spark for distributed data processing and analytics.
* **Seamless Integration**: Fully integrated with Azure services, including Azure Data Lake Storage.

**Azure Data Lake Storage**

Azure Data Lake Storage is a highly scalable data storage service designed for large datasets. It provides a secure and cost-effective environment for storing and managing data.

* **Scalable Storage**: Supports large volumes of unstructured data.
* **Hierarchical Namespace**: Facilitates efficient data organization and access.
* **Optimized for Analytics**: Works seamlessly with analytics tools like Azure Databricks.

**Integration of Azure Databricks with Data Lake**

Azure Databricks and Azure Data Lake work together to enable efficient data exploration and optimization. Databricks can process data directly stored in Data Lake, enabling fast, distributed data analytics without the need for data migration. This integration ensures seamless workflows for big data projects, including real-time processing and optimization of large datasets like the Agricultural Crop Yield dataset.

**Project Requirements & Tasks Performed**

**Project Requirements:**

**1. Azure Services**

* **Azure Databricks:** To perform serverless data processing, including data transformation and analysis, using notebooks.
* **Azure Data Lake:** To store raw input data and processed output data efficiently for scalable access.

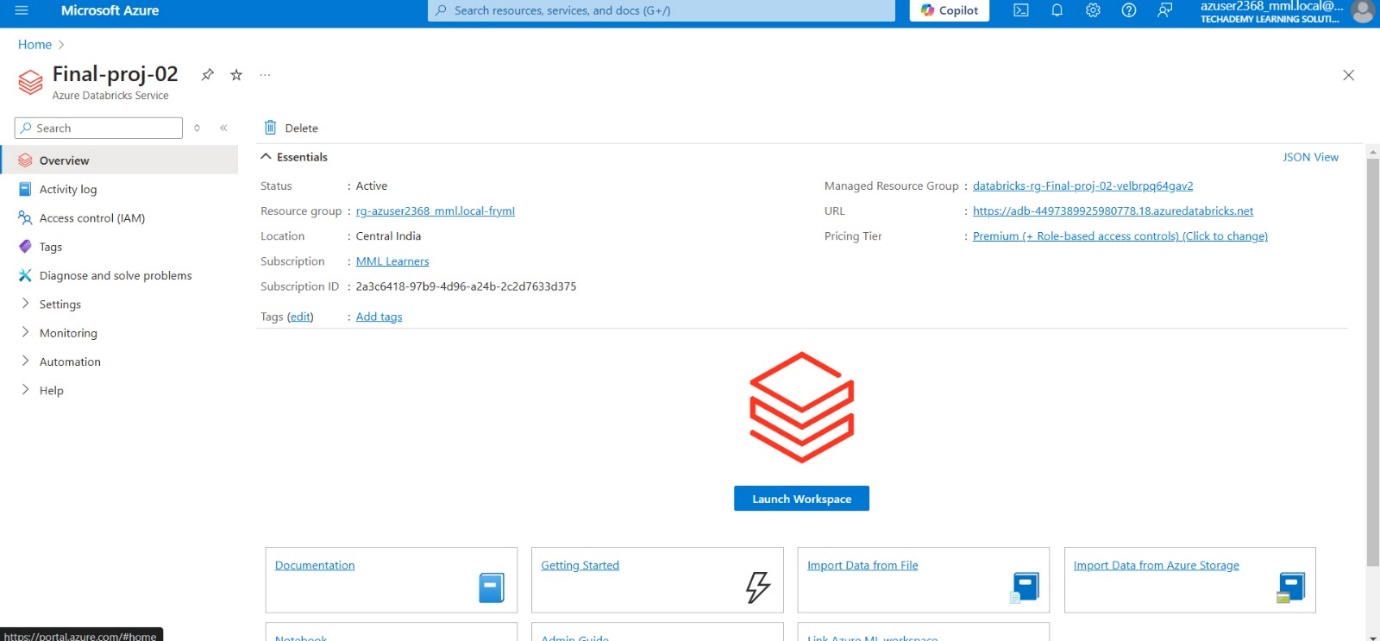
**2. Input Data**

* **File Format:** CSV
* **Source:** [Kaggle](https://www.kaggle.com/datasets/akshatgupta7/crop-yield-in-indian-states-dataset)
* **Description:** The CSV file contains raw data to be ingested, processed, and transformed into meaningful insights.

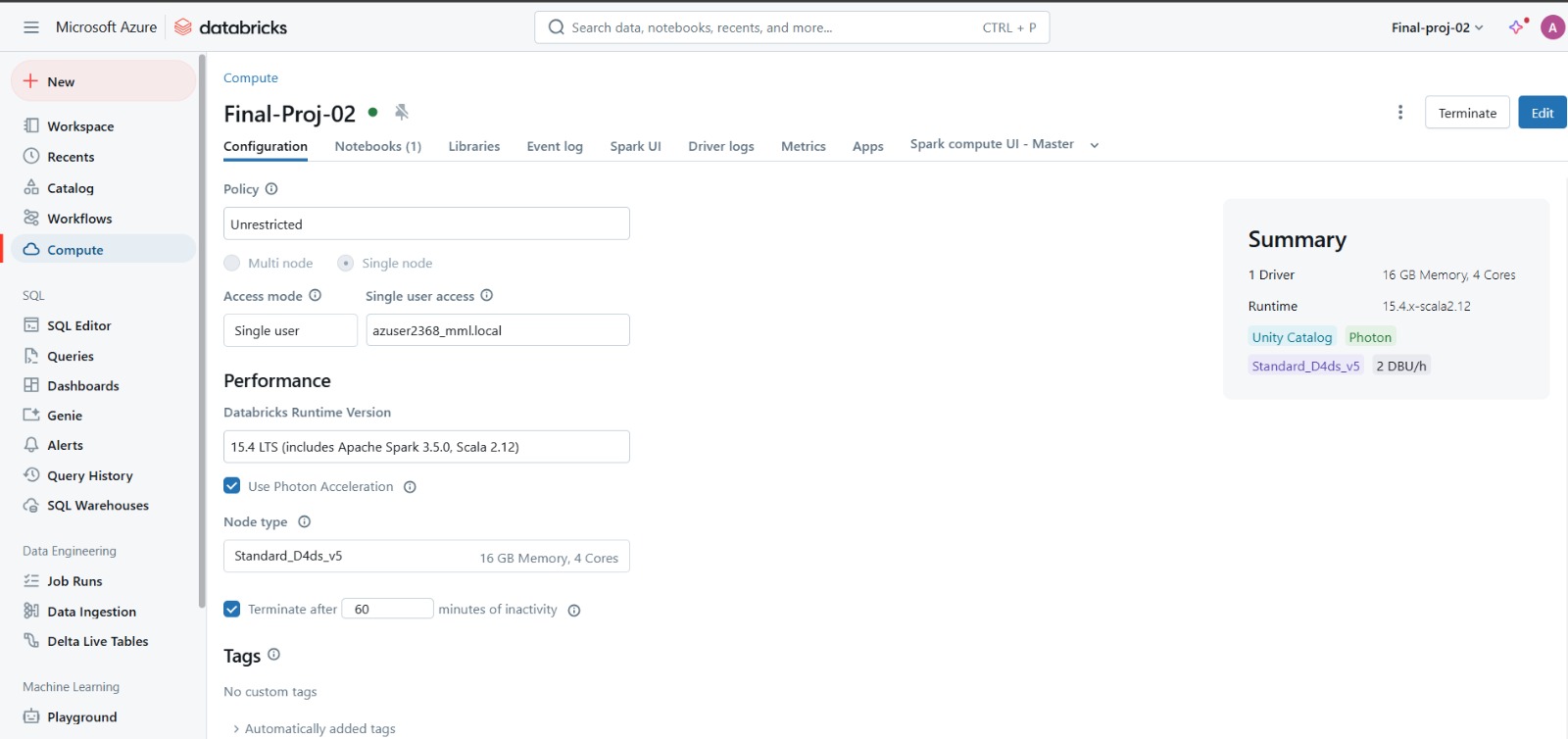
**Tasks Performed:**

**Task 1: Set Up the Environment**

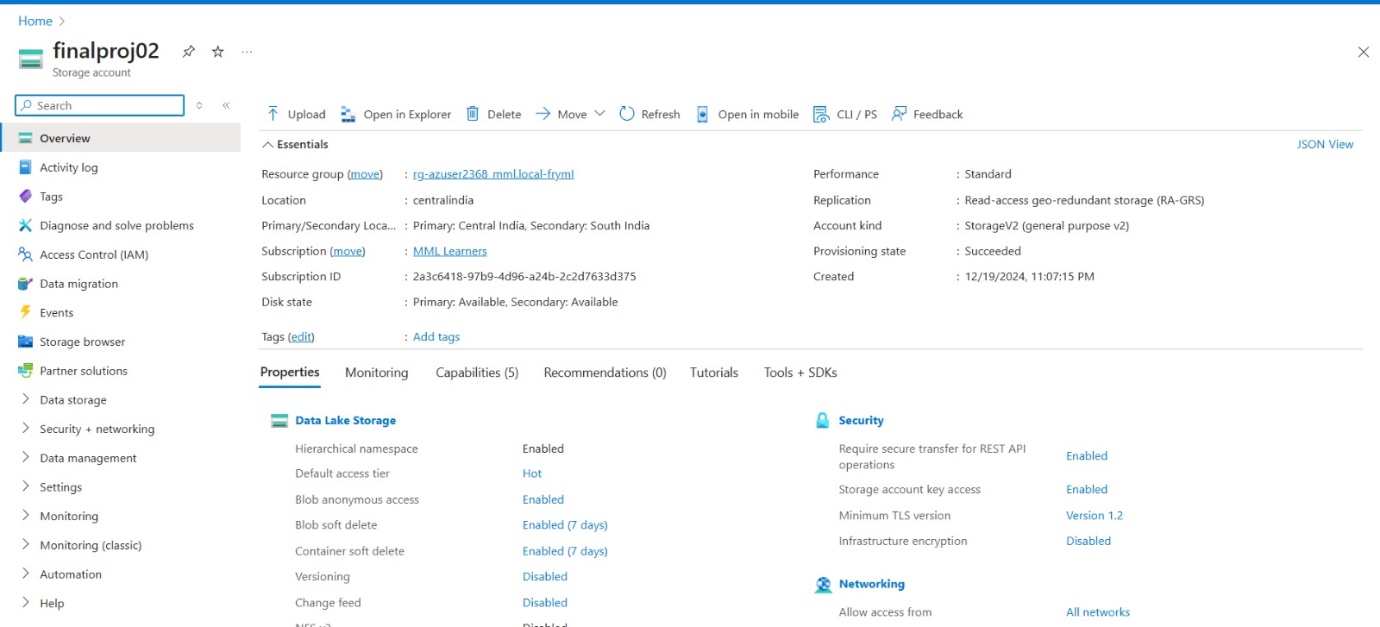
**1.Create a Databricks Workspace:**

****

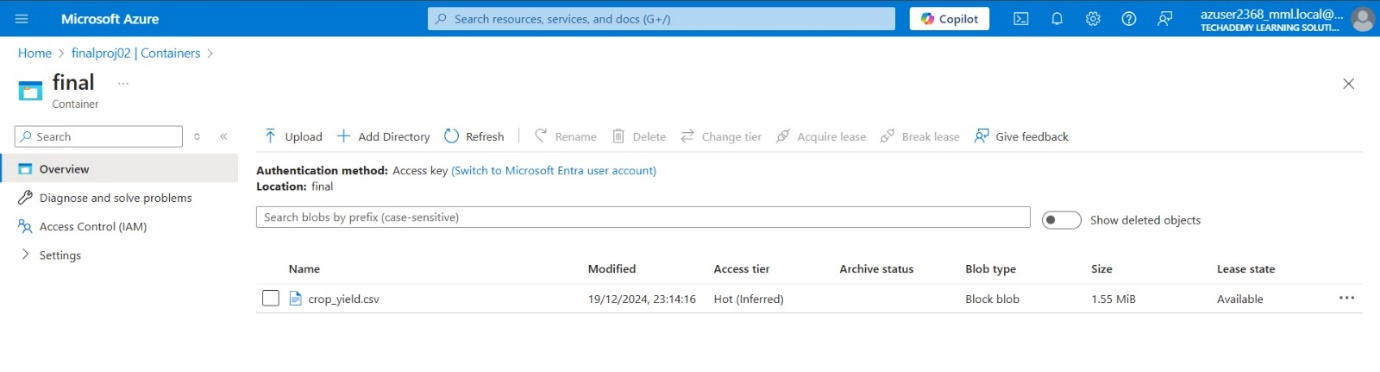
**2.Create and Start a Cluster:**

****

**3.Create a Data lake:**

****

**4.Upload Data to the Data Lake:**

****

**Task 2: Mount ADLS Gen2 to Databricks**

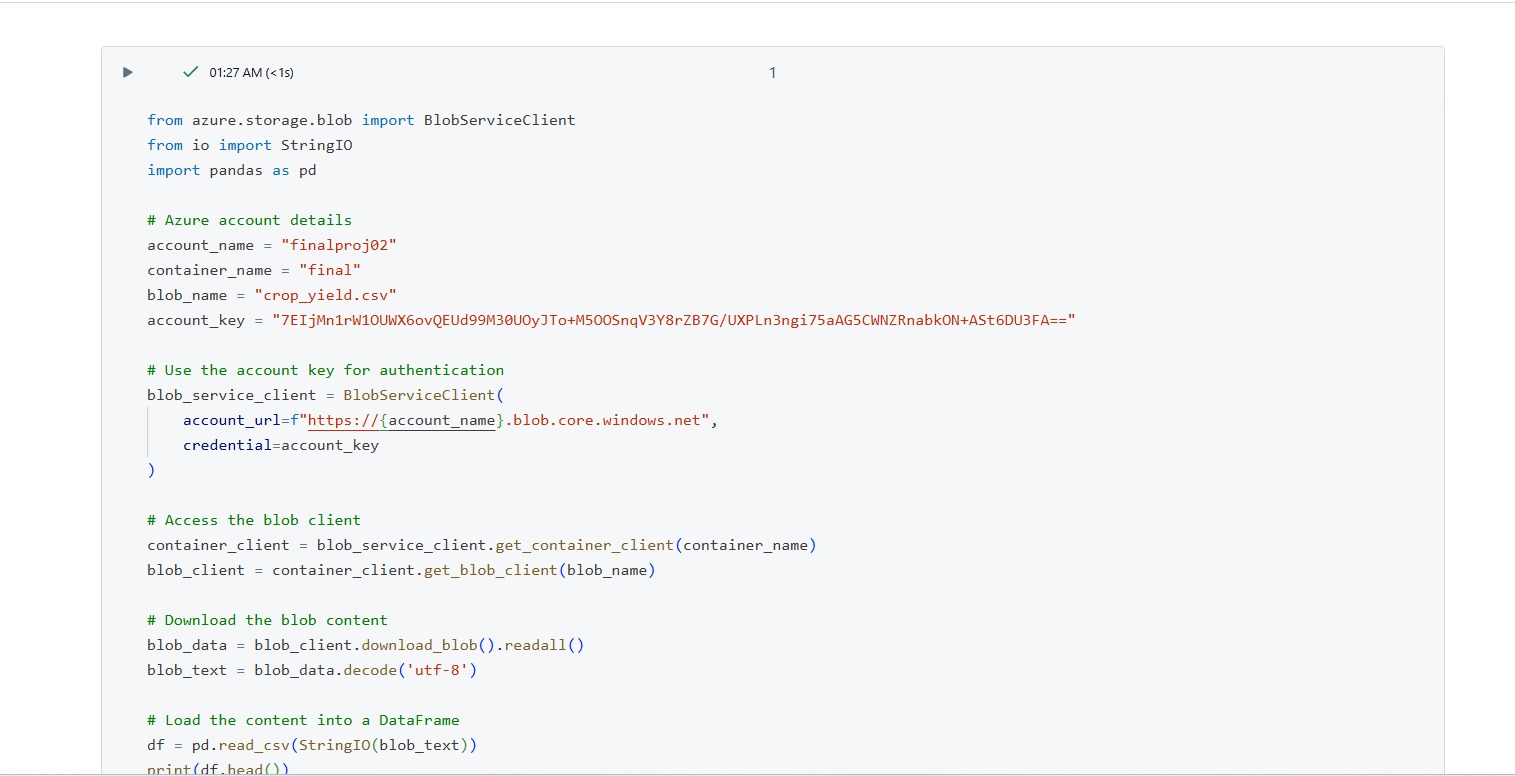
**1.Collect Required Details:**

**Storage Account Name**: Found in your Azure Storage Account settings.

**Container Name**: The name of the container where you uploaded the file.

**File Path**: Path to your uploaded file in the container.

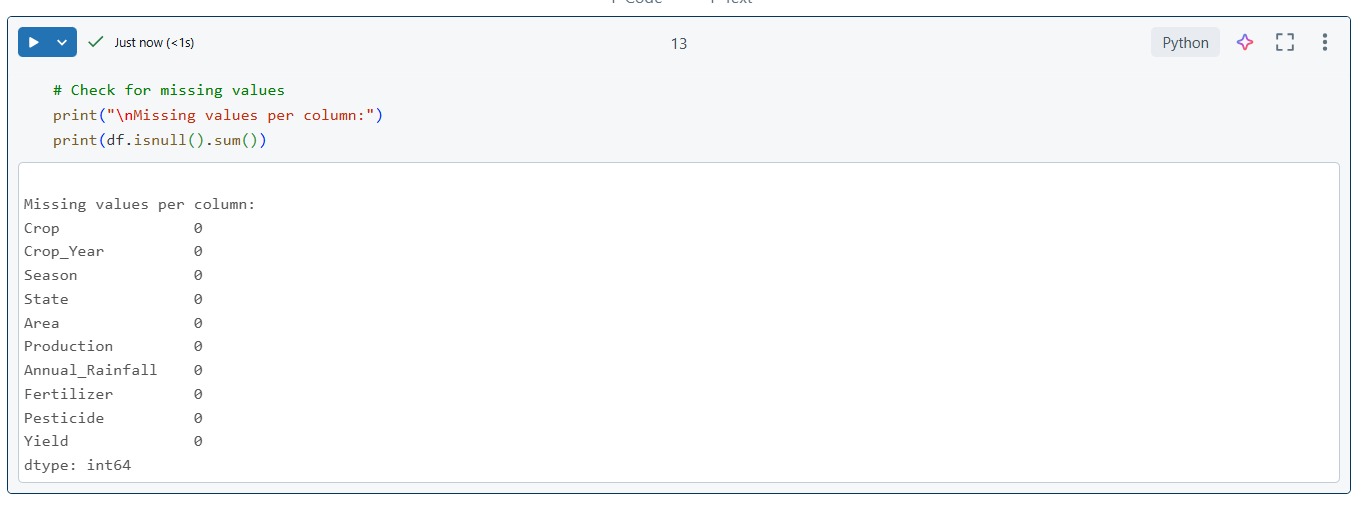
**Access Key or Service Principal**: Obtain the storage account access key or set up a service principal for authentication.



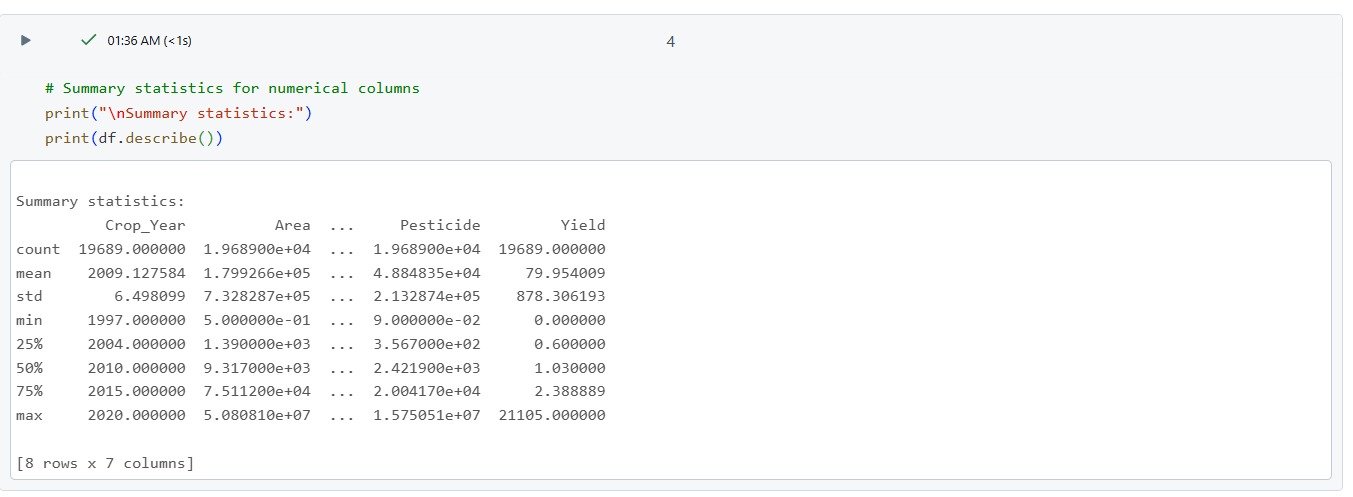
**Task 3: Data Exploration with Optimized Techniques**

exploring the data involves performing some basic analysis to understand its structure, quality, and insights. You can use various techniques such as:

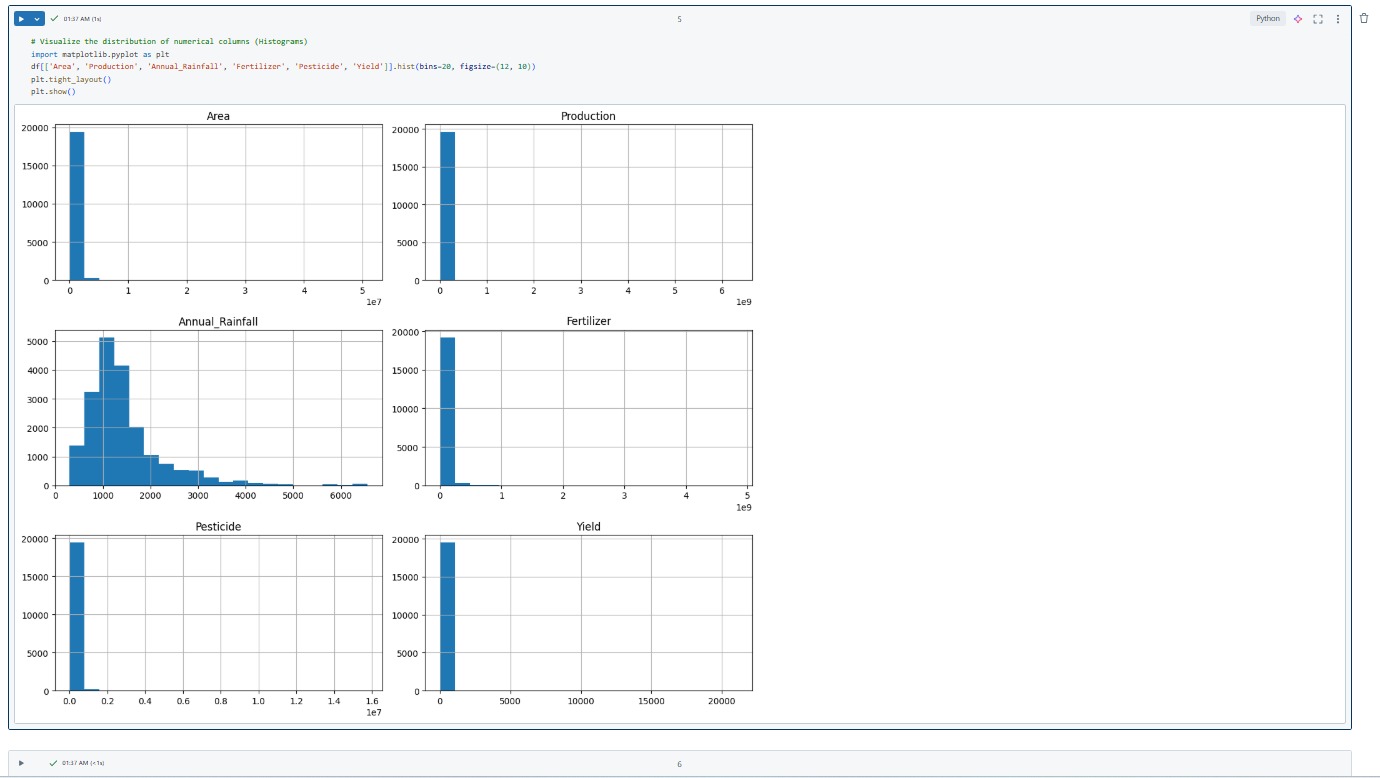
* **Checking for Missing Values**: Identify any missing or null values in your dataset.



* **Descriptive Statistics**: Get a summary of numerical columns, such as mean, median, standard deviation, etc.



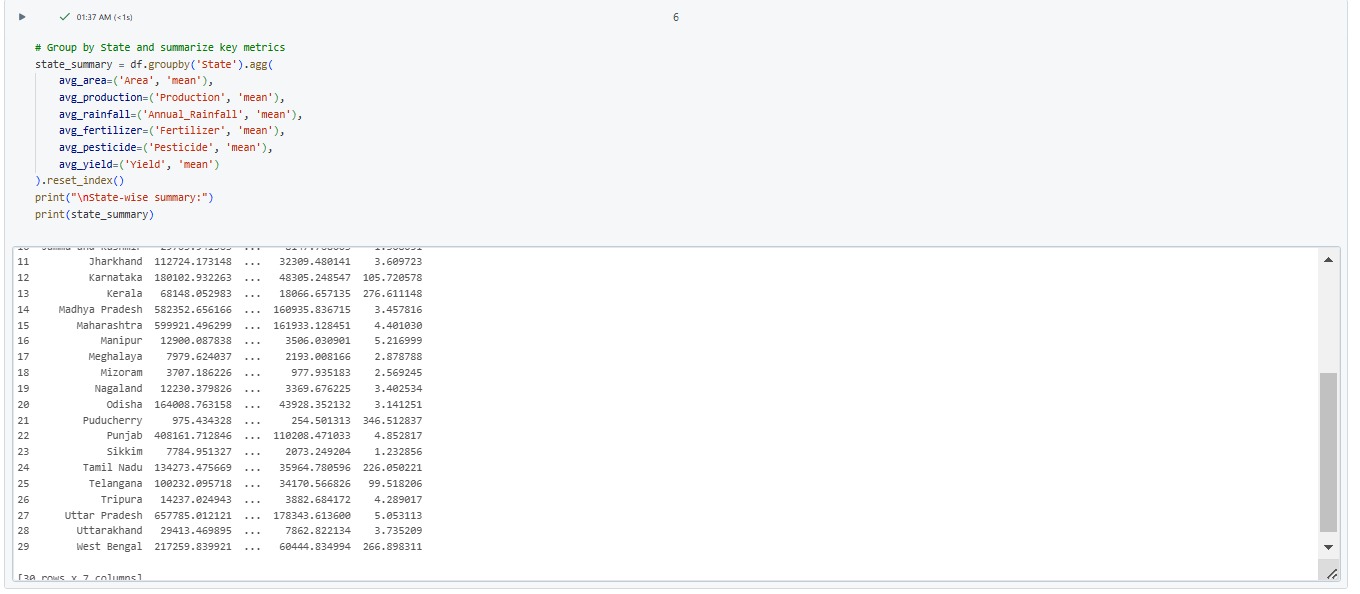
* **Data Distribution**: Visualize the distribution of key variables to check for any patterns.



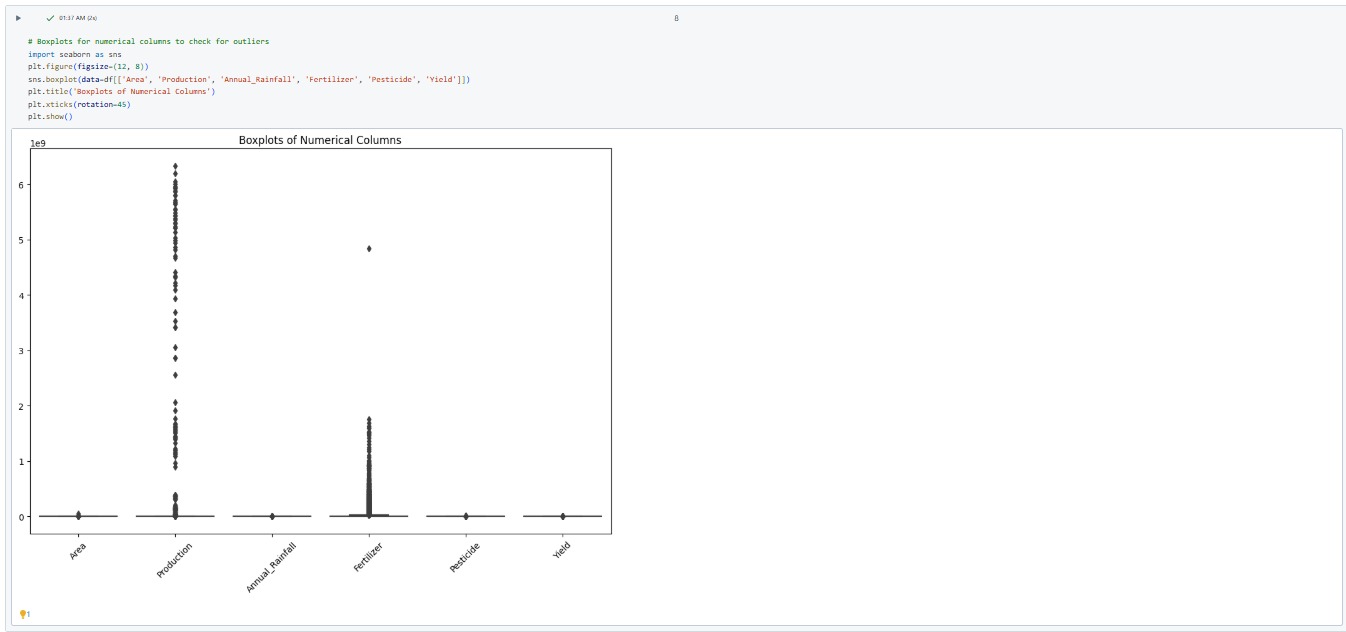
* **Data Types and Schema**: Confirm the data types of columns and inspect the schema.



* **Group Analysis**: Compare statistics across different regions (State) and seasons (Season).



* **Outliers**: Boxplots can reveal outliers that may need further investigation or cleaning.

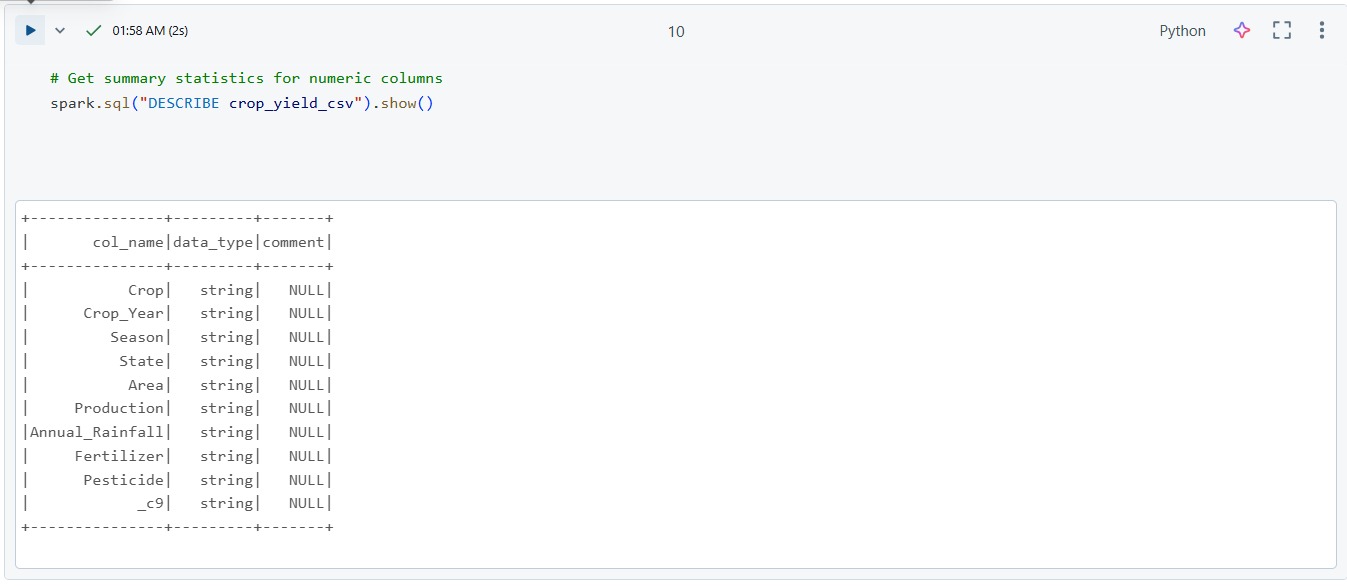


**Task 4: Perform Data Exploration Using PySparkSQL**

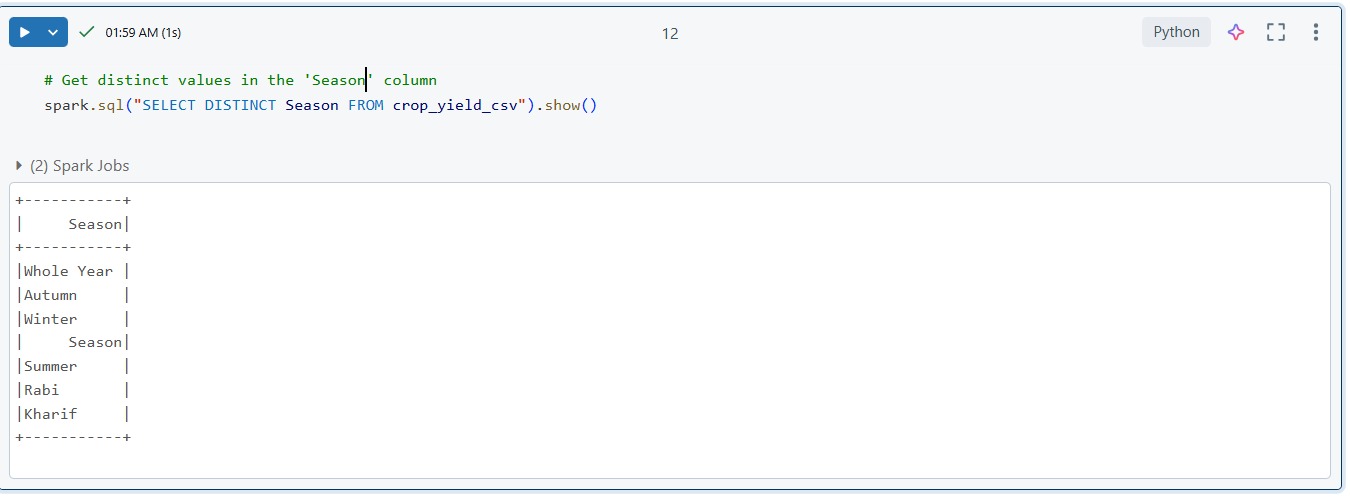
**1.Get the total number of records in the dataset**

****

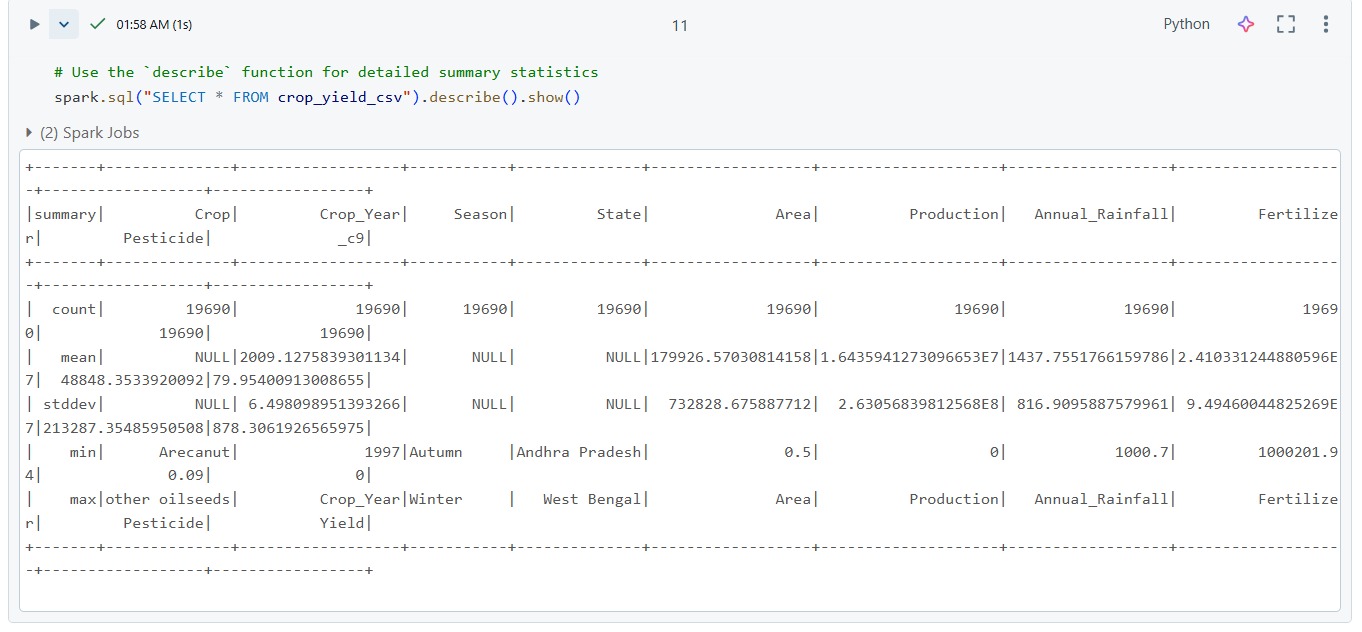
**2.Get the summary statistics for numeric columns**

****

**3.** **Get unique values in a specific column**

****

**4.Using the describe function for detailed summary statistics**

****