Documento generado de: chapter1.pptx

# Slide 1

Introduction to PySpark I N T R O D U C T I O N TO P YS PA R K

Benjamin Schmidt Data Engineer



object 3

# Slide 2

Meet your instructor

Almost a Decade of Data Experience with PySpark Used PySpark for Machine Learning, ETL tasks, and much more more Enthusiastic teacher of new tools for all!  
-

INTRODUCTION TO PYSPARK



object 3



object 4



object 5

# Slide 3

What is PySpark?

Distributed data processing: Designed to handle large datasets across clusters Supports various data formats including CSV, Parquet, and JSON SQL integration allows querying of data using both Python and SQL syntax Optimized for speed at scale

INTRODUCTION TO PYSPARK



object 3



object 5



object 6



object 7

# Slide 4

When would we use PySpark?

Big data analytics Distributed data processing Real-time data streaming Machine learning on large datasets ETL and ELT pipelines Working with diverse data sources:  
CSV JSON Parquet Many Many More

INTRODUCTION TO PYSPARK



object 3



object 5



object 6



object 7



object 8



object 9

# Slide 5

Spark cluster

Master Node Manages the cluster, coordinates tasks,  
and schedules jobs

Worker Nodes Execute the tasks assigned by the master Responsible for executing the actual computations and storing data in memory or disk

INTRODUCTION TO PYSPARK



object 3



object 5



object 6

# Slide 6

SparkSession

SparkSessions allow you to access your Spark cluster and are critical for using PySpark.  
  
# Import SparkSession from pyspark.sql import SparkSession  
  
# Initialize a SparkSession spark = SparkSession.builder.appName("MySparkApp").getOrCreate()

.builder() sets up a session getOrCreate() creates or retrieves a session  
.appName() helps manage multiple sessions

INTRODUCTION TO PYSPARK



object 6



object 8



object 10



object 12

# Slide 7

PySpark DataFrames

Similar to other DataFrames but Optimized for PySpark  
# Import and initialize a Spark session from pyspark.sql import SparkSession spark = SparkSession.builder.appName("MySparkApp").getOrCreate()  
  
# Create a DataFrame census\_df = spark.read.csv("census.csv",  
["gender","age","zipcode","salary\_range\_usd","marriage\_status"])  
  
# Show the DataFrame census\_df.show()

INTRODUCTION TO PYSPARK



object 3



object 4

# Slide 8

Let's practice!  
I N T R O D U C T I O N TO P YS PA R K

# Slide 9

Introduction to PySpark DataFrames I N T R O D U C T I O N TO P YS PA R K

Benjamin Schmidt Data Engineer



object 3

# Slide 10

About DataFrames

DataFrames: Tabular format (rows/columns)  
Supports SQL-like operations Comparable to a Pandas Dataframe or a SQL TABLE Structured Data

INTRODUCTION TO PYSPARK



object 3



object 5



object 6



object 7

# Slide 11

Creating DataFrames from filestores

# Create a DataFrame from CSV census\_df = spark.read.csv('path/to/census.csv', header=True, inferSchema=True)

INTRODUCTION TO PYSPARK

# Slide 12

Printing the DataFrame

# Show the first 5 rows of the DataFrame census\_df.show()

INTRODUCTION TO PYSPARK

# Slide 13

Printing DataFrame Schema

# Show the schema census\_df.printSchema() Output:  
root

INTRODUCTION TO PYSPARK

# Slide 14

Basic analytics on PySpark DataFrames

# .count() will return the total row numbers in the DataFrame row\_count = census\_df.count()  
print(f'Number of rows: {row\_count}')  
  
# groupby() allows the use of sql-like aggregations census\_df.groupBy('gender').agg({'salary\_usd': 'avg'}).show()  
  
Other aggregate functions are:  
  
sum()  
min()  
max()

INTRODUCTION TO PYSPARK



object 5



object 7



object 9

# Slide 15

Key functions for PySpark analytics

.select() : Selects specific columns from the DataFrame  
.filter() : Filters rows based on specific conditions  
.groupBy() : Groups rows based on one or more columns  
.agg() : Applies aggregate functions to grouped data

INTRODUCTION TO PYSPARK



object 3



object 5



object 7



object 9

# Slide 16

Key Functions For Example

# Using filter and select, we can narrow down our DataFrame

> 50).select('age', 'occupation')

filtered\_census\_df = census\_df.filter(df['age'] filtered\_census\_df.show()  
Output  
+ + +

|age| occupation |  
+ + +

+ + +

INTRODUCTION TO PYSPARK

# Slide 17

Let's practice!  
I N T R O D U C T I O N TO P YS PA R K

# Slide 18

More on Spark DataFrames I N T R O D U C T I O N TO P YS PA R K

Benjamin Schmidt Data Engineer



object 3

# Slide 19

Creating DataFrames from various data sources

CSV Files: Common for structured, delimited data JSON Files: Semi-structured, hierarchical data format Parquet Files: Optimized for storage and querying, often used in data engineering

Example:  
spark.read.csv("path/to/file.csv")

Example:  
spark.read.json("path/to/file.json")

1 https://spark.apache.org/docs/latest/api/python/reference/pyspark.pandas/api/pyspark.pandas.read\_csv

Example:  
spark.read.parquet("path/to/file.parquet")

INTRODUCTION TO PYSPARK



object 3



object 5



object 6



object 9



object 11



object 13

# Slide 20

Schema inference and manual schema definition

Spark can infer schemas from data with inferSchema=True Manually define schema for better control - useful for fixed data structures

INTRODUCTION TO PYSPARK



object 3



object 6

# Slide 21

DataTypes in PySpark DataFrames

IntegerType : Whole numbers E.g., 1 , 3478 , -1890456 LongType: Larger whole numbers E.g., 8-byte signed numbers, 922334775806 FloatType and DoubleType: Floating-point numbers for decimal values E.g., 3.14159 StringType: Used for text or string data E.g., "This is an example of a string."  
...

INTRODUCTION TO PYSPARK



object 3



object 5



object 9



object 10



object 12



object 13



object 15



object 16



object 19

# Slide 22

DataTypes Syntax for PySpark DataFrames

# Import the necessary types as classes from pyspark.sql.types import (StructType,  
StructField, IntegerType, StringType, ArrayType)  
  
# Construct the schema schema = StructType([  
StructField("id", IntegerType(), True), StructField("name", StringType(), True), StructField("scores", ArrayType(IntegerType()), True)  
])  
  
# Set the schema df = spark.createDataFrame(data, schema=schema)

INTRODUCTION TO PYSPARK

# Slide 23

DataFrame operations - selection and filtering

Use .select() to choose specific columns Use .filter() or .where() to filter rows based on conditions Use .sort() to order by a collection of columns  
# Select and show only the name and age columns df.select("name", "age").show()  
  
# Filter on age > 30 df.filter(df["age"] > 30).show()  
  
# Use Where to filter match a specific value df.where(df["age"] == 30).show()

INTRODUCTION TO PYSPARK



object 6



object 8



object 11

# Slide 24

Sorting and dropping missing values

Order data using .sort() or .orderBy()  
Use na.drop() to remove rows with null values  
# Sort using the age column df.sort("age", ascending=False).show()  
  
# Drop missing values df.na.drop().show()

INTRODUCTION TO PYSPARK



object 4



object 7

# Slide 25

Cheatsheet

spark.read\_json() : Load data from JSON spark.read.schema() : Define schemas explicitly  
.na.drop() : Drop rows with missing values  
.select() , .filter() , .sort() , .orderBy() : Basic data manipulation functions

INTRODUCTION TO PYSPARK



object 3



object 5



object 7



object 9

# Slide 26

Let's practice!  
I N T R O D U C T I O N TO P YS PA R K