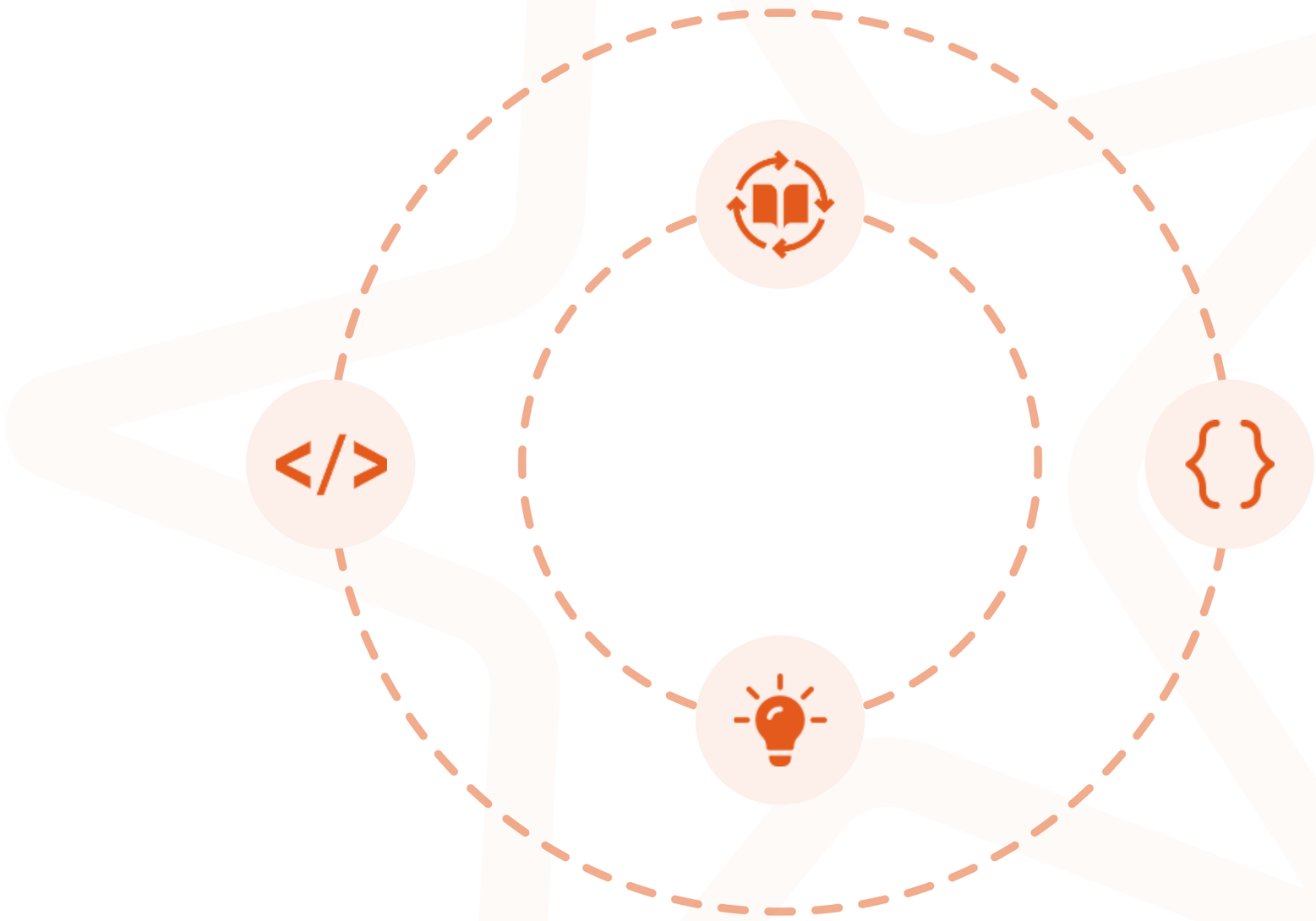
BOSSCODER ACADEMY

LEARN

PySpark

APACHE Spark™



In Just 30 Days

Disclaimer

Everyone has their own way of learning. The key is focusing on the core elements of PySpark to build a strong understanding.

This guide is designed to assist you in that journey.

Introduction:

PySpark is the Python API for Apache Spark, a framework for large-scale data processing.

This 30-day guide is designed to take you from a beginner to intermediate level, introducing key PySpark concepts and providing practice questions to solidify learning. Each day will include both theoretical learning and hands-on practice, helping you to grasp the concepts thoroughly.

Week 1: Introduction to PySpark and Environment Setup

Day 1

Introduction to Big Data and Spark Ecosystem

OBJECTIVE:

Learn about the Big Data ecosystem and how Apache Spark fits into it.

TOPICS:

- Big Data challenges
- Hadoop vs. Spark
- Spark architecture (driver, executor, tasks)
- PySpark and its advantages

PRACTICE:

- Write an explanation of Spark's advantages over Hadoop.
- List various components of Spark.

Day 2

Setting Up PySpark Environment

OBJECTIVE:

Set up your local development environment for PySpark.

TOPICS:

- Installing PySpark on Windows/Mac/Linux
- Running Spark in Standalone mode
- Introduction to Jupyter Notebooks for PySpark development

PRACTICE:

- Set up a virtual environment and install PySpark.
- Write a PySpark script to print "Hello, Spark!".

Day 3

SparkContext and SparkSession

OBJECTIVE:

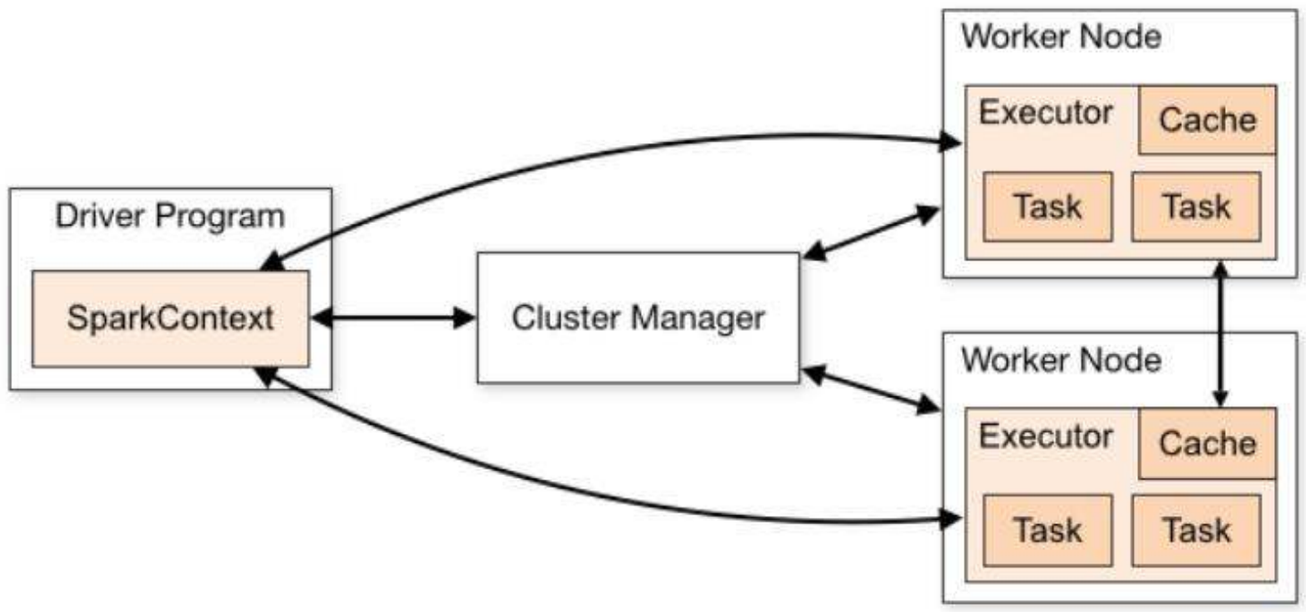
Understand the role of `SparkContext` and `SparkSession`.

TOPICS:

- Importance of SparkContext
- Creating and using SparkSession
- Differences between SparkContext and SparkSession

PRACTICE:

- Create a simple SparkSession.
- Write code to access SparkContext through SparkSession.



Day 4

RDDs (Resilient Distributed Datasets) Basics

OBJECTIVE:

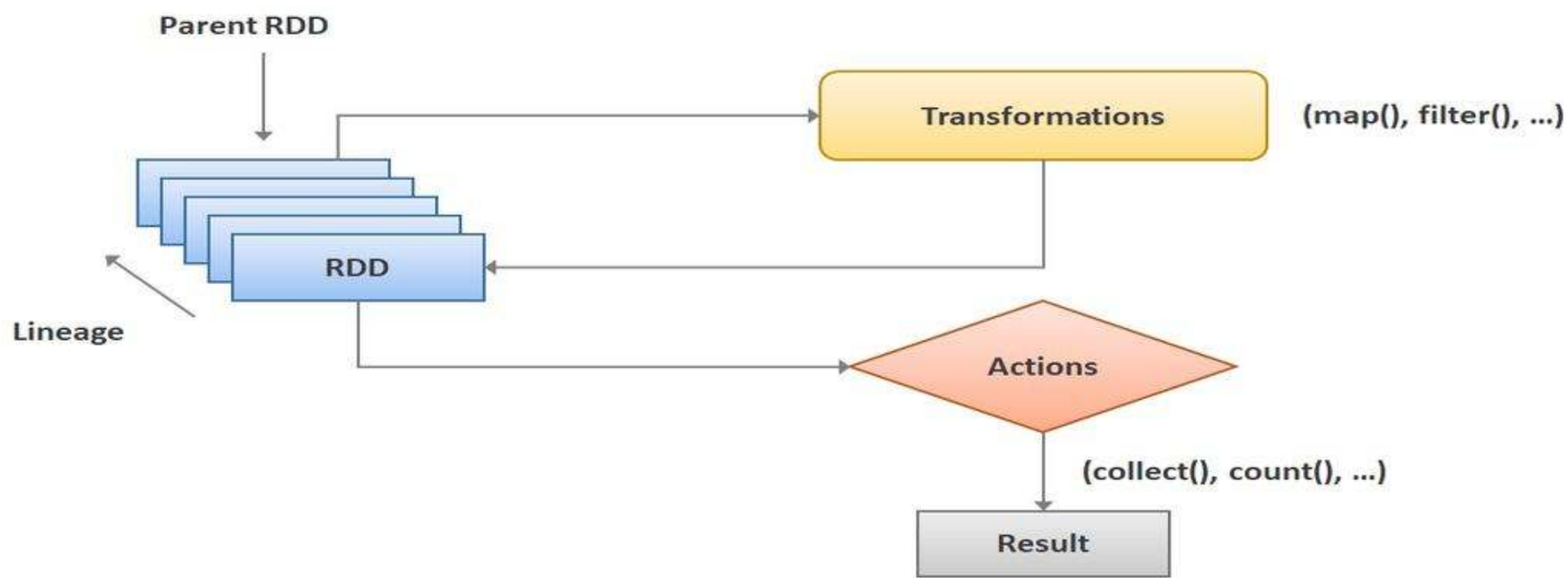
Learn about RDDs and their fundamental operations.

TOPICS:

- What are RDDs?
- Creating RDDs from files and collections
- Lazy evaluation in Spark

PRACTICE:

- Create an RDD from a local Python list.
- Perform basic transformations like `map()` and `filter()` on the RDD.



Day 5

RDD Transformations and Actions

OBJECTIVE:

Perform various RDD transformations and actions.

TOPICS:

- Transformation vs. Action
- Common RDD transformations: `map()`, `flatMap()`, `filter()`, `union()`
- Actions: `collect()`, `count()`, `reduce()`

PRACTICE:

- Write a PySpark program to find the average of a dataset using RDDs.
- Use different transformations and actions on an RDD of integers.

Day 6

Key-Value Pair RDDs

OBJECTIVE:

Explore key-value pair RDDs for aggregations and joins.

TOPICS:

- Creating key-value RDDs
- Transformations on key-value pairs: `reduceByKey()`, `groupByKey()`
- Key-value pair operations: `join()`, `cogroup()`

PRACTICE:

- Write a program to count the frequency of words in a text file.
- Perform joins on two key-value pair RDDs.

Day 7

Data Persistence and Partitioning

OBJECTIVE:

Learn about persistence and partitioning in RDDs.

TOPICS:

- Caching and persistence in PySpark
- Partitioning RDDs and its impact on performance
- Custom partitioning

PRACTICE:

- Cache an RDD and measure the performance difference.
- Write code to repartition an RDD.

Day 8

Introduction to DataFrames

OBJECTIVE:

Learn the basics of DataFrames in PySpark.

TOPICS:

- Difference between RDDs and DataFrames
- Creating DataFrames from RDDs, files (CSV, JSON)
- Viewing DataFrame schema and data

PRACTICE:

- Load a CSV file into a DataFrame.
- Print the schema of a DataFrame.

Day 9

DataFrame Operations

OBJECTIVE:

Perform basic operations on DataFrames.

TOPICS:

- Selecting columns and rows
- Filtering rows, applying conditions
- Sorting and ordering data

PRACTICE:

- Perform filtering and sorting operations on a dataset.
- Select specific columns from a DataFrame.

Day 10

Aggregations in DataFrames

OBJECTIVE:

Learn how to aggregate data using DataFrames.

TOPICS:

- Grouping DataFrames
- Aggregating functions: `count()`, `sum()`, `avg()`
- Performing groupBy operations

PRACTICE:

- Group and aggregate a dataset by a specific column.
- Calculate the average of a numeric column in the DataFrame.

Day 11

DataFrame Joins

OBJECTIVE:

Perform various join operations on DataFrames.

TOPICS:

- Types of joins: inner, outer, left, right
- Using `join()` function in DataFrames

PRACTICE:

- Perform inner and left joins on two DataFrames.
- Practice merging two datasets based on a common column.

Day 12

DataFrame Functions and Expressions

OBJECTIVE:

Use built-in functions for DataFrame manipulation.

TOPICS:

- Common functions: `withColumn()`, `selectExpr()`
- Using SQL expressions for filtering and transformations

PRACTICE:

- Use `withColumn()` to add and modify columns.
- Apply SQL expressions for complex transformations.

Day 13

Working with Dates and Timestamps

OBJECTIVE:

Learn to work with date and timestamp data in PySpark.

TOPICS:

- Date and time functions
- Formatting and parsing dates
- Performing date arithmetic

PRACTICE:

- Extract year, month, and day from a timestamp.
- Perform date arithmetic like adding or subtracting days.

Day 14

Spark SQL Introduction

OBJECTIVE:

Run SQL queries on PySpark DataFrames using Spark SQL.

TOPICS:

- Registering DataFrames as SQL tables
- Running SQL queries on DataFrames
- Introduction to `spark.sql()`

PRACTICE:

- Register a DataFrame as a temporary table.
- Write SQL queries to filter and aggregate data from DataFrames.

Week 3: Advanced Data Processing and Optimization

Day 15

Working with Complex Data Types

OBJECTIVE:

Handle complex data types in PySpark.

TOPICS:

- Nested columns (Structs, Arrays)
- Accessing and transforming nested columns

PRACTICE:

- Load a JSON file with nested structures.
- Write code to access and modify complex columns.

Day 16

User-Defined Functions (UDFs)

OBJECTIVE:

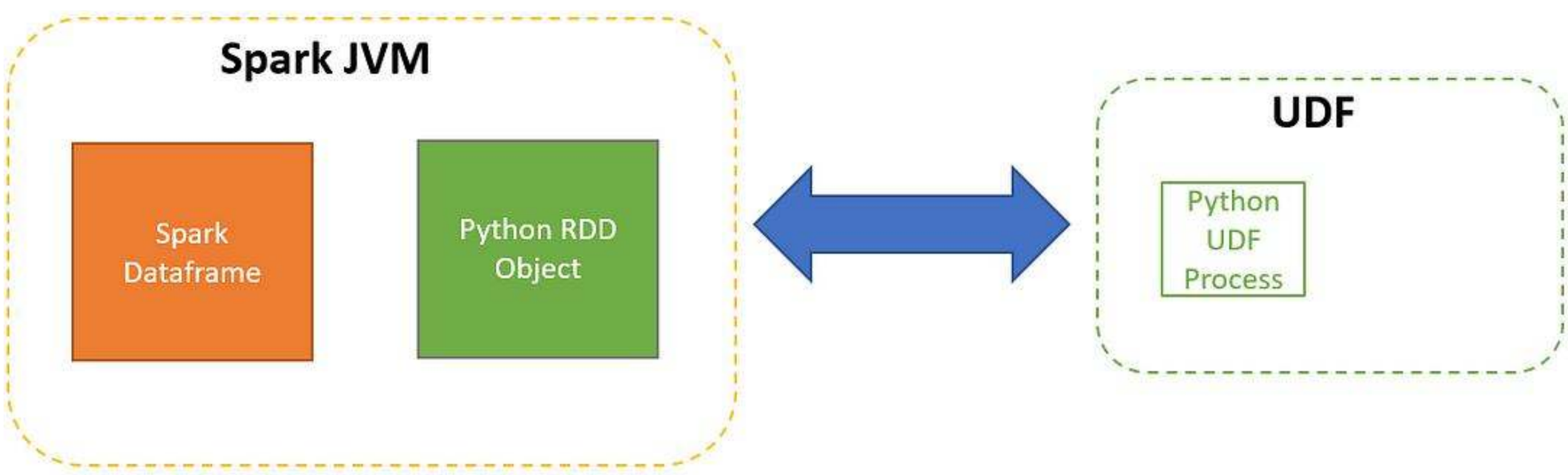
Define and use custom UDFs in PySpark.

TOPICS:

- Creating Python functions as UDFs
- Using UDFs in DataFrames

PRACTICE:

- Create a custom UDF to manipulate DataFrame columns.
- Write a UDF to clean up text data in a dataset.



Day 17

Broadcasting Variables and Accumulators

OBJECTIVE:

Use broadcast variables and accumulators for performance optimization.

TOPICS:

- Importance of broadcasting
- Using broadcast variables
- Working with accumulators

PRACTICE:

- Broadcast a large lookup table to all workers.
- Create an accumulator to count values across partitions.

Day 18

Performance Tuning and Optimization

OBJECTIVE:

Learn performance tuning techniques in PySpark.

TOPICS:

- Common performance bottlenecks
- Caching, partitioning, and memory optimization
- Tungsten and Catalyst optimizers

PRACTICE:

- Optimize a PySpark job using caching and repartitioning.
- Profile and analyze job performance using Spark UI.

Day 19

Working with Parquet and ORC Formats

OBJECTIVE:

Learn about optimized file formats for faster processing.

TOPICS:

- Introduction to Parquet and ORC formats
- Reading and writing Parquet/ORC files
- Advantages of columnar formats

PRACTICE:

- Convert a CSV dataset to Parquet format.
- Read and analyze data from Parquet files.

Day 20

Window Functions

OBJECTIVE:

Use window functions for complex aggregations.

TOPICS:

- Introduction to window functions
- Performing operations like `rank()`, `dense_rank()`, `row_number()`
- Windowing over partitions

PRACTICE:

- Calculate the rank of rows in a dataset.
- Use window functions to calculate moving averages.

Day 21

Handling Missing Data

OBJECTIVE:

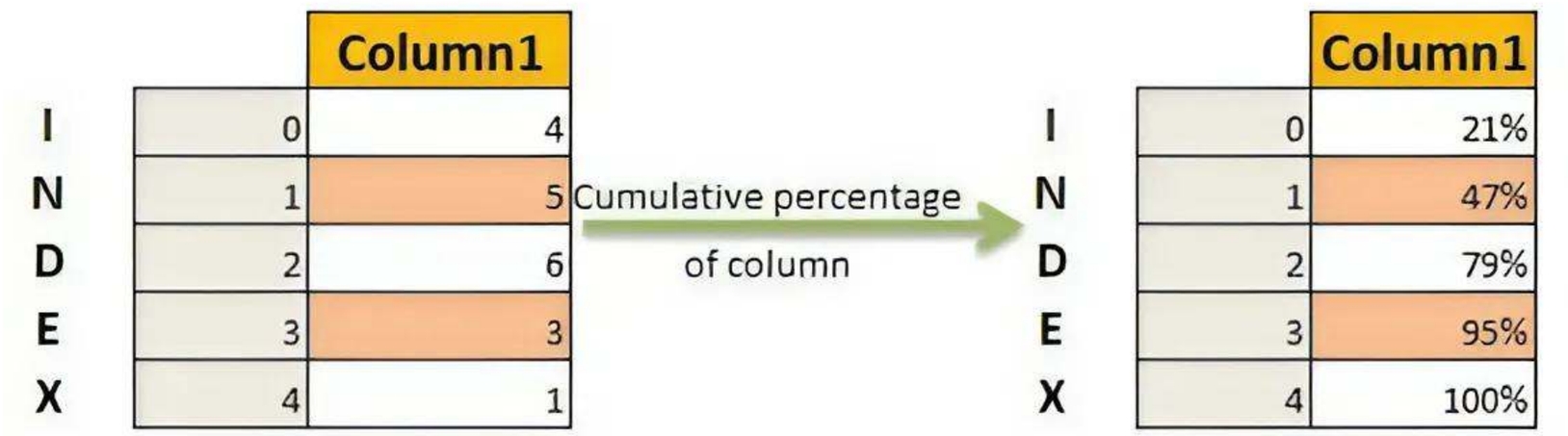
Learn to handle missing or null data in PySpark.

TOPICS:

- Detecting nulls
- Dropping nulls or filling missing values

PRACTICE:

- Detect and fill missing values in a dataset.
- Drop rows with null values in specified columns.



Week 4: Machine Learning and PySpark MLlib

Day 22

Introduction to PySpark MLlib

OBJECTIVE:

Learn about PySpark's MLlib for machine learning.

TOPICS:

- Overview of MLlib
- Pipeline concept in MLlib
- Supported algorithms in PySpark MLlib

PRACTICE:

- Create a basic PySpark ML pipeline.

Day 23

Feature Engineering

OBJECTIVE:

Perform feature engineering using PySpark.

TOPICS:

- Techniques for feature scaling and selection
- Handling categorical data
- Using `VectorAssembler` for feature transformation

PRACTICE:

- Apply one-hot encoding to a categorical column.
- Normalize a numeric column in a dataset.

Day 24

Classification Models in PySpark

OBJECTIVE:

Train and evaluate classification models using PySpark.

TOPICS:

- Logistic regression, decision trees, random forests
- Model training and evaluation

PRACTICE:

- Train a decision tree classifier on a dataset.
- Evaluate the performance of a classifier using accuracy metrics.

Day 25

Regression Models in PySpark

OBJECTIVE:

Train and evaluate regression models using PySpark.

TOPICS:

- Linear regression, generalized linear models
- Model tuning using cross-validation

PRACTICE:

- Train a linear regression model.
- Use cross-validation to tune hyperparameters.

Day 26

Clustering Models in PySpark

OBJECTIVE:

Learn clustering techniques in PySpark.

TOPICS:

- K-means clustering
- Evaluation metrics for clustering

PRACTICE:

- Perform K-means clustering on a dataset.
- Analyze cluster centroids and labels

Day 27

Dimensionality Reduction Techniques

OBJECTIVE:

Apply dimensionality reduction in PySpark.

TOPICS:

- PCA (Principal Component Analysis)
- Singular Value Decomposition (SVD)

PRACTICE:

- Perform PCA on a dataset to reduce features.
- Analyze the output of PCA and interpret results.

Day 28

Model Persistence and Deployment

OBJECTIVE:

Learn how to save and load models in PySpark.

TOPICS:

- Saving models using `save()` and `load()`
- Deploying models for predictions

PRACTICE:

- Train a model and save it to disk.
- Load a saved model and use it for predictions.

Day 29

Spark Streaming Basics

OBJECTIVE:

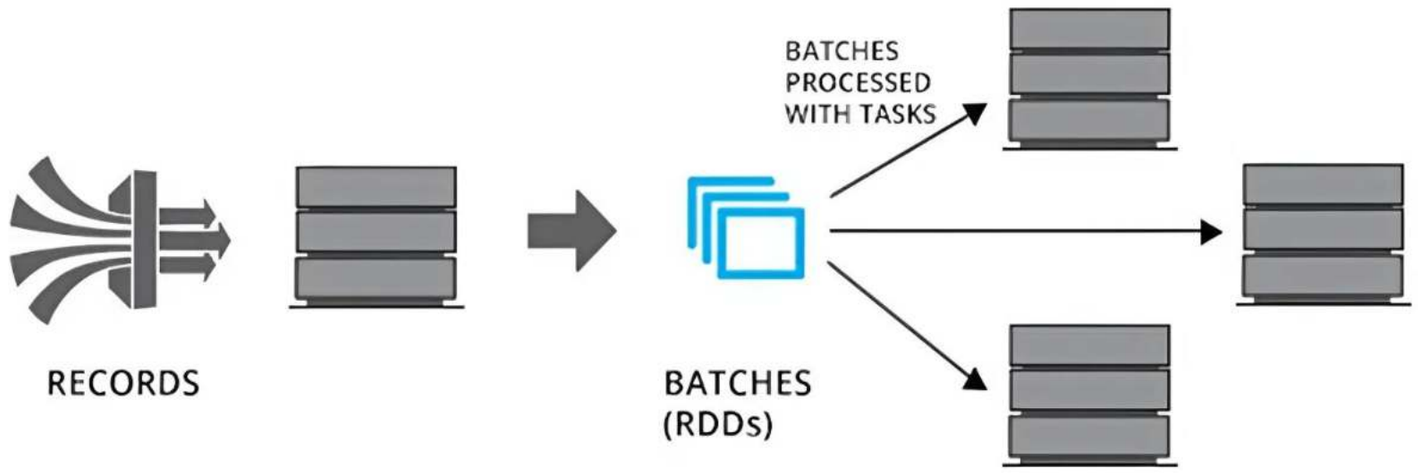
Introduction to real-time data processing with Spark Streaming.

TOPICS:

- Spark Streaming architecture
- Creating a streaming context
- Processing real-time data

PRACTICE:

- Set up a Spark Streaming context.
- Process real-time data from a socket or file stream.



Day 30

Spark Structured Streaming

OBJECTIVE:

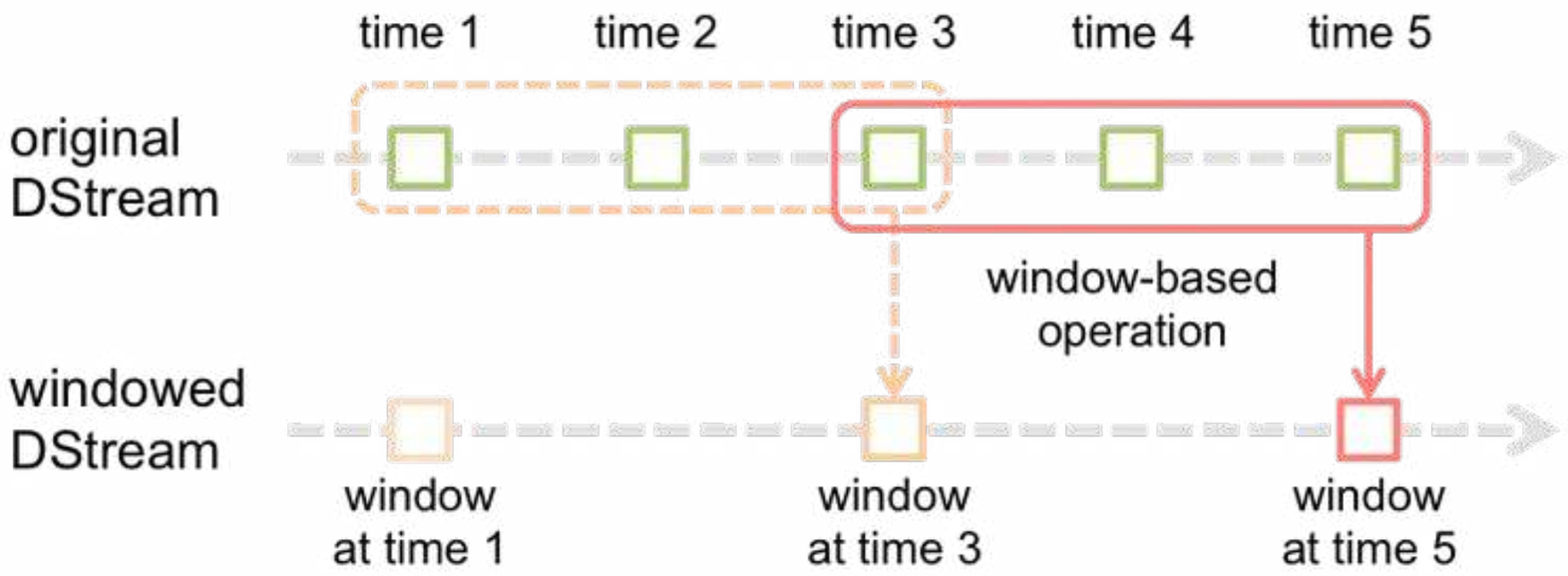
Learn about structured streaming in Spark.

TOPICS:

- Difference between Spark Streaming and Structured Streaming
- Writing structured streaming queries
- Output modes in structured streaming

PRACTICE:

- Create a structured streaming job.
- Write data to a streaming sink (console, file, etc.).



Conclusion:

By following this structured 30-day guide, you will have developed a solid understanding of PySpark and be ready to tackle real-world problems using Spark's powerful data processing engine.

Keep practicing the concepts, and explore more advanced topics as you grow comfortable with PySpark.

BOSSCODER ACADEMY

WHY BOSSCODER?

1000+ Alumni placed at Top Product-based companies.

More than **136% hike** for every 2 out of 3 Working Professional.

Average Package of **24LPA**.

> The syllabus is most **up-to-date** and the list of problems provided covers **all important topics.**
>
> **Lavanya**
> Meta

> Course is very well **structured** and **streamlined** to crack any **MAANG** company .
>
> **Rahul**
> Google

EXPLORE MORE