

**Data Engineering with Python and PySpark**

**Course Number:** PYTH-260  
**Duration:** 5 days

**Overview**

Data engineering sits at the start of every data science project and is concerned with compiling, cleaning, harmonizing, and exploring data for downstream analysis. Analysts often do this crucial preparatory work, which becomes challenging as projects scale.

This Python and PySpark for Data Engineering training course teaches aspiring data engineers, data scientists, data science managers, and other quantitative professionals how to prepare data in a repeatable and scalable manner. Students learn the pain points that arise as data scales and how to construct a scalable data engineering pipeline. Attendees use Python, PySpark, and Databricks Community Edition for processing on a cloud-based, scalable cluster.

**Prerequisites**

Attendees must have some working experience in any modern programming language and knowledge of SQL and data processing concepts, including data grouping and aggregation.

**Materials**

All Python and PySpark training students receive comprehensive courseware.

**Software Needed on Each Student PC**

* Windows, Mac, or Linux with at least 8 GB RAM
  + Most class activities will create Spark code and visualizations in a browser-based notebook environment. The class also details how to export these notebooks and how to run code outside of this environment.
* A current version of Anaconda for Python 3.x
* Related lab files that Accelebrate will provide
* Internet access

**Objectives**

* Manual inspection for data quality and reliability
* Understand the key data ingestion methods
* Understand the key database types
* Articulate the use cases for SQL, NoSQL, and graph databases
* Inspect data with univariate and bivariate inspection methods
* Flag and quantify the severity of outliers
* Inspect data and flag data for deviation from normality
* Inspect and flag missing data
* Generate standard reports for data quality issues
* Describe the four levels of cloud services (PaaS, SaaS, IaaS, DaaS)
* Become aware of the major cloud providers and their offerings
* Understand trade-offs between cloud services and on-premise solutions
* Articulate use cases for pure Python vs. PySpark
* Articulate use cases for local vs. cloud-based analytics pipelines
* Implement a prototype data pipeline with python in a Jupyter notebook
* Implement a scalable data pipeline with PySpark
* Migrate a data pipeline to the cloud using Databricks Community Edition
* Build an end-to-end solution culminating in a data visualization

**Outline**

* What is Data Engineering?
* The Data Lifecycle
  + Scoping and selecting data
  + Staging and harmonization
  + Staging and saving data
  + Analysis and summarization
  + Insight
  + Revise and repeat
  + Data Engineering in the organization
    - Prepares data for downstream consumers
  + Core data engineering responsibilities
    - Stage
    - Cleanse
    - Conform
    - Deliver
    - Plan for scaling and automation
  + Data engineering toolkit
    - PySpark for big data
    - Cloud cluster distributed computing
    - Storage systems
    - Automation and orchestration
  + Next steps
    - Processes scale
    - Track failures and successes
    - Organize growing collections of logs
    - Automate system processes and checks
  + Process orchestration
    - Splunk
    - ELK stack
    - Airflow
* Challenges of Modern Data Engineering
  + Data size
    - The four V’s
    - Volume
    - Velocity
    - Variety
    - Voracity
    - How big?
  + Strategies for dealing with big data
    - Streaming
    - Chunking
    - Batching
    - Sampling
  + Types of data
    - Structured
    - Partially structured
    - Unstructured
    - Form or survey data
    - Tweet stream / text blobs
    - Image or sound data
    - Numeric measurement data
    - Sensor/IOT
    - Dirty or clean
    - Set format
    - Text data
    - Interpretation
    - Contradictory data (summary rules)
  + Resiliency
    - Sampling
    - Eventual consistency
    - Real-time decisions
    - Single or multiple pass processing
  + Presenting and analysis
    - Summarizing your data
    - Data granularity and drill down
    - Defining the question
    - Operationalization
  + Delivery format
    - Data visualization
    - Data summarization
    - Data life cycle
    - Data persistence
* The Data Science Pipeline
  + The seven steps to data science
    - 1) Collect and clean: Extraction Transformation and Loading (ETL)
    - 2) Understand the data: Exploratory Data Analysis (EDA)
    - 3) Modeling and evaluation
    - 4) Interpretation and presentation
    - 5) Revision
    - 6) Productionalization
    - 7) Maintenance
* Data Engineering on the Cloud
  + Local assets vs. the cloud
  + Compute asset management
  + Service architecture
  + Cloud providers
    - Amazon AWS
    - GPI
    - Azure
    - DigitalOcean
  + Four levels of cloud service
    - Software as a service (SaaS)
    - Platform as a service (PaaS)
    - Desktop as a service (DaaS)
    - Infrastructure as a service (IaaS)
  + Types of clouds
* Python for Data Analytics
  + Alternative analytics coding languages
    - Excel VBA
    - C
    - Java
    - R
    - Golang
    - SPSS
    - SAS
  + Why Python?
    - Python as the glue!
  + PyData ecosystem
    - Scikit-learn
    - Jupyter (notebook and lab)
    - Python platforms
    - Shell
    - Notebooks
    - IDEs
    - Visual Studio
    - SQL connections
    - PySpark
  + The Python community
    - The popularity of the language
    - Pep8 standards
  + The "Pythonic" code ethic
    - Readability
    - Clear function
    - Least effort
* The Data Science Flow Using Python
  + Import dependencies
  + Import data
  + Check data quality
  + Data code book
  + Data dictionary
  + Missing data
  + Bias
  + Variance
  + Data distribution
  + Sanity checking
  + Check experimental design
  + Experimental protocol
  + Non-random sampling issues
  + Data cleaning
  + Imputation
  + Unbalanced samples
  + Data exploration
  + Univariate
  + Bivariate
  + corrplots
  + data visualization
  + Matplotlib
  + Seaborn
  + Holoviews for really big data
  + Dashboard with panel
* PySpark for Big Data
  + Why use PySpark?
    - Distributed clusters
    - Single context
  + Alternatives
  + Pandas (Python)
    - R (datatable and Tidyverse)
    - DASK
    - Hadoo
  + Spark comprehensive components
    - Spark architecture
    - Spark session
    - Spark schema
    - Transformations
    - Actions
    - Leveraging Spark
* Using the PySpark API
  + What is PySpark?
    - A Java Virtual machine (JVM py4J)
    - A Python wrapper for Spark Scala
    - When to use Spark Scala instead
  + Spark APIs
    - DataFrames
    - Dataset
    - RDD
    - Speed considerations Python DF vs. Python RDD
    - Return types RDD vs. other
  + PySpark coding
    - Data exploration
    - Functions
    - Spark DF to pandas DataFrame
* Data Pipelines with PySpark locally
  + Build a project using Spark in a Jupyter notebook
  + Install Spark locally
    - Spark shell
    - PySpark drivers
    - PySpark env vars
  + Introduction to notebooking
    - Collaboratory introduction
    - Data scientist go-to toolkit
    - Iterative coding
    - Testing and prototyping
    - Communication
    - Markdown and code
  + open collaboratory notebook for shared analysis
    - Code cells and markdown
    - Markdown use, take notes use latex
    - Kernel definition and intro
  + Get CSV from the web
    - !wget
  + Explore the file system structure.
    - !ls, !pwd
  + Walkthrough the data science workflow on Spark
    - Data ingestion
    - EDA exploratory data analysis
    - ETL (extract transform and load)
    - Iterative data exploration
    - Data visualization
* Databricks as an End-to-End Cloud Solution
  + Databricks community
    - A free offering to build a cloud cluster
    - Try and troubleshoot the first iteration of a workflow
  + Why Databricks?
    - Easy repeatable setup
    - Cross-organization standardized platform
    - End-to-end solution
    - Automatic Spark dependencies and cluster generation
    - Databricks history
    - Hadoop map-reduce
    - Berkeley amp lab
    - Databricks history
  + Why use Spark on the cloud?
    - Scale with clusters
    - On-demand resources
    - Speed acceleration
  + Setting up Databricks
  + Databricks Can Use Different Backends
    - AWS, GPI, Azure, or Community Edition
    - Plan selection
  + Start with Databricks community edition
    - Community.cloud.Databricks.com
  + Databricks tour
    - Databricks concepts
    - Workspaces
    - Notebooks
    - Clusters
    - Libraries
    - Tables
    - Jobs (scheduling)
    - The DBC file format
    - Databricks demo gallery
  + Make a Notebook in Workspace
    - Markdown and Python in Databricks notebook
    - Make Spark context
    - Automatic spark setup and cluster generation
    - Managing Databricks clusters
* Machine Learning Workflow with PySpark on the Cloud
  + Set up and execute Machine Learning (ML) flows
  + Use Databricks to run ML flow on student’s data
  + Increased performance with Spark on a cluster
  + Setting up a PySpark notebook on Databricks
    - Demonstrate NLP and clustering workflows on Twitter data
    - Demonstrate cluster use and optimization on the same analysis
    - Student workshop to bring all concepts together and present result
* Conclusion