DATA ENGINEER SYLLABUS

General Overview

- Total Duration: 6-8 weeks (1.5–2 months)
- Focus Areas:
    - Core Concepts: 40% (Data Engineering, Databases, Cloud Platforms)
    - Hands-On Tools & Technologies: 40% (SQL, Python, PySpark, Linux, Azure)
    - Real-Time Project: 20% (practical application to solidify knowledge)

Here's a suggested breakdown for each topic along with time allocation:

Week 1-2: Introduction & Core Concepts of Data Engineering

Objective: Build a strong foundation for DE tasks.

Focus Areas:

1. Introduction to Data Engineering (5–10 hours)
    - What is Data Engineering? Key responsibilities of Data Engineers.
    - Data pipeline architecture: Batch vs. real-time data processing.
    - Key skills and tools used in data engineering.
    - Overview of cloud platforms, databases, and distributed systems.
2. Linux Fundamentals (8–12 hours)
    - Linux basics: file system, commands (e.g., ls, cp, mv, grep, find).
    - Process management, file permissions.
    - Shell scripting basics.
    - Introduction to managing databases on Linux (e.g., installing PostgreSQL, MySQL).
3. SQL Basics (10–12 hours)
    - SQL syntax: SELECT, JOIN, WHERE, GROUP BY, ORDER BY.
    - Data manipulation: INSERT, UPDATE, DELETE.
    - Query optimization basics.
    - Data types, normalization, and indexing.

Outcome for Week 2:

- Students should be comfortable with Linux commands, working with files, and writing basic SQL queries.

Week 3-4: Programming, Data Processing, and Cloud Platforms

Objective: Get comfortable with Python, PySpark, and understand cloud platforms.

Focus Areas:

1. Python Basics (10–12 hours)
    - Python syntax, data types, control flow, functions, and error handling.
    - Introduction to libraries for Data Engineering: pandas, numpy, requests, etc.
    - Working with files (CSV, JSON) and basic data manipulation.
    - Writing simple scripts for data extraction, cleaning, and transformation.
2. PySpark Fundamentals (12–15 hours)
    - Introduction to Spark and its ecosystem.
    - Setting up PySpark and writing basic PySpark programs.
    - Working with RDDs and DataFrames.
    - Data transformations and actions in PySpark (e.g., map, filter, reduce).
    - Performance tuning and managing large datasets.
3. Cloud Platforms (Azure Focus) (8–10 hours)
    - Introduction to cloud computing and Data Engineering on cloud platforms.
    - Overview of Azure Data services: Azure Storage, Azure SQL, Azure Databricks, Azure Data Factory.
    - Working with Azure Storage (Blob, Data Lake).
    - Creating and managing resources on Azure (Azure Portal, CLI).

Outcome for Week 4:

- Students should be comfortable with basic Python programming and basic PySpark operations, and have an introductory understanding of Azure.

Week 5-6: Advanced Topics and Project Work

Objective: Build deeper skills in data pipeline development and deploy knowledge in a project.

Focus Areas:

1. Advanced SQL for Data Engineering (8–10 hours)
    - Complex joins, subqueries, window functions, and CTEs.
    - Query optimization for large datasets.
    - Working with time-series data.
    - Data warehousing concepts: OLAP vs OLTP.
2. Data Pipelines and Workflow Automation (10–12 hours)
    - ETL (Extract, Transform, Load) pipelines with Python and PySpark.
    - Introduction to orchestration tools: Apache Airflow, Azure Data Factory.
    - Scheduling and automating data workflows.
3. Data Engineering Best Practices (6–8 hours)
    - Data quality and validation checks.

- Handling missing or corrupted data.
- Version control using Git.
- Introduction to testing (unit tests, integration tests).

Week 7-8: Project Work and Finalization

Objective: Apply all learned concepts to a real-time data engineering project.

Project:

1. Project Scope (12–15 hours)
   - The project should cover end-to-end Data Engineering tasks:
     - Data ingestion from multiple sources (APIs, Databases, Files).
     - Data transformation using Python and PySpark.
     - Building a real-time or batch data pipeline.
     - Storing data in Azure storage (Blob, Data Lake).
     - Building a dashboard or report using the processed data (optional).
     - Automating the pipeline using Azure Data Factory or Apache Airflow.
2. Project Deployment & Testing (5–8 hours)
   - Testing the pipeline with sample data.
   - Deploying the pipeline to the cloud.
   - Monitoring and logging for data pipelines.
3. Code Review & Final Presentation (5–8 hours)
   - Final project presentation to the class (or mentors), explaining the architecture and design.
   - Code review and feedback for improvement.

Project Focus:

- The project is essential for ensuring practical application of all the concepts learned.
- It should include:
  - Data Ingestion: Using APIs, FTP, and cloud services.
  - Data Transformation: Using Python and PySpark.
  - Storage & Management: Using cloud storage (Azure).
  - Orchestration: Automating the workflow with tools like Azure Data Factory or Airflow.
  - Real-Time Data: If possible, include a stream of real-time data.

Suggested Week-by-Week Breakdown:

Week	Topic	Hours/Focus
1	Core Concepts, Linux, SQL	25–30 hours
2	Python Basics, SQL Advanced	25–30 hours
3	PySpark Basics, Azure Overview	25–30 hours
4	Advanced SQL, PySpark, Cloud (Azure)	30–35 hours
5	Data Pipelines, Automation, Best Practices	30–35 hours
6	Project Work, Code Implementation	30–35 hours
7	Project Completion, Testing, Deployment	25–30 hours
8	Final Presentation, Code Review	15–20 hours

Total Training Hours: 180–220 hours

Key Tips for Success:

1. Emphasize Real-World Scenarios: Focus on creating real-world data pipelines and handling production-level data.
2. Hands-on Labs: Ensure that each topic includes practical exercises.
3. Project is Critical: The project should be comprehensive and well-documented.
4. Iterate: If the students get stuck, let them work through the issues while guiding them, as debugging is a key skill in Data Engineering.

1. Core Concepts of Data Engineering

Objective: Understand the role of a Data Engineer, data pipeline architecture, and tools.

Topics to Cover:

1. What is Data Engineering?
    - Training Material: Introduction to Data Engineering (slides, articles, videos).
        - Roles of a Data Engineer vs Data Scientist vs Data Analyst.
        - Key responsibilities: building pipelines, handling large datasets, data processing.
        - Tools: Databases, Big Data, Cloud Platforms, ETL.
    - Scenario: Case study of a Data Engineering team in an organization.
        - Scenario: A company needs to integrate their sales data from multiple sources (CRM, ERP systems) and provide a report every day. How does a Data Engineer design the process?
2. Data Pipeline Basics
    - Training Material: Articles and documentation on data pipelines.
        - Overview of ETL and ELT processes.
        - Batch vs real-time processing.
        - Tools: Apache Kafka, Apache Spark, Azure Data Factory, Airflow.
    - Scenario: Build a simple pipeline for ingesting data from a database, transforming it (e.g., cleaning, aggregating), and storing it in a data warehouse.
3. Databases: SQL and NoSQL
    - Training Material: Articles/videos explaining SQL (relational) and NoSQL (key-value, document, column store).
        - Types of databases: RDBMS, Columnar, Graph, Key-Value, Document.
        - Use cases: When to use SQL vs NoSQL.
    - Scenario: Design a database schema for storing user transaction data and compare how you'd structure it in an SQL database vs a NoSQL database (e.g., MongoDB).

---

2. Linux Basics

Objective: Learn basic Linux commands and file handling.

Topics to Cover:

1. Linux File System & Basic Commands
    - Training Material: Online tutorials or slides (e.g., tutorials on LinuxCommand.org).
        - File structure: /home, /etc, /var, /tmp.
        - Commands: ls, cd, cp, mv, rm, cat, grep, find.

- Scenario: Scenario: You have a folder of 1,000 log files. Write a script that finds all logs containing the word “error” and moves them to a new folder.

2. Process Management
    - Training Material: Guides on ps, top, kill, nohup, cron.
        - Managing processes and jobs in Linux.
        - Background jobs and scheduled tasks.
    - Scenario: Schedule a task to run a Python data cleaning script every night at 2 AM using cron jobs.
3. Basic Shell Scripting
    - Training Material: Online tutorials or books like "The Linux Command Line" by William Shotts.
        - Writing bash scripts: loops, conditionals, and functions.
        - Example script: Monitoring disk usage and sending an alert email if usage exceeds 90%.
    - Scenario: Write a bash script that takes a log file as input, processes it by extracting relevant data, and outputs it into a CSV.

---

3. SQL Basics and Advanced

Objective: Master SQL for data extraction, manipulation, and optimization.

Topics to Cover:

1. SQL Basics (SELECT, WHERE, JOIN, GROUP BY)
    - Training Material: Interactive SQL tutorials (e.g., Mode Analytics SQL Tutorial).
        - Simple queries: SELECT, FROM, WHERE, ORDER BY.
        - Joins: INNER JOIN, LEFT JOIN, RIGHT JOIN.
        - Aggregations: COUNT(), SUM(), AVG().
    - Scenario: Query a customer database to find the total sales for each customer in the last quarter.
2. SQL Advanced (Subqueries, Window Functions, Optimization)
    - Training Material: Articles/videos on advanced SQL topics (e.g., "SQL Performance Explained").
        - Subqueries, CTEs (Common Table Expressions), and window functions.
        - Indexing and optimization.
    - Scenario: Given a table of sales data, write a query that ranks salespeople based on their total sales, but only considering their sales for the last 6 months.
3. Database Design & Schema Optimization
    - Training Material: Books or slides on normalization (1NF, 2NF, 3NF).
        - Data modeling: designing tables, primary/foreign keys.
        - Normalization vs. denormalization for performance.

- Scenario: Normalize a database schema that currently stores customer orders in a single table, and discuss performance trade-offs.

---

4. Python Basics

Objective: Use Python for scripting and data manipulation.

Topics to Cover:

1. Python Syntax & Data Structures
    - Training Material: Python documentation, tutorials (e.g., W3Schools, Real Python).
        - Variables, loops, functions, and classes.
        - Lists, dictionaries, sets, and tuples.
    - Scenario: Write a Python script that loads data from a CSV, cleans the data (e.g., removing duplicates), and outputs it to another file.
2. Libraries for Data Engineering: Pandas, Numpy
    - Training Material: Documentation on pandas and numpy (e.g., official websites).
        - Data manipulation with pandas: filtering, grouping, merging.
        - Numerical computations with numpy.
    - Scenario: Use pandas to load a dataset, perform basic cleaning (e.g., handling missing values), and aggregate the data by category.

---

5. PySpark

Objective: Handle large-scale data processing with Spark.

Topics to Cover:

1. Introduction to Spark & PySpark
    - Training Material: Spark documentation and PySpark tutorials.
        - RDDs, DataFrames, and SparkSQL.
        - Basic transformations: map(), filter(), flatMap().
    - Scenario: Load a CSV file into PySpark, apply some transformations (e.g., filtering rows, grouping), and store the result.
2. Advanced PySpark (Join, GroupBy, Window Functions)
    - Training Material: PySpark documentation, hands-on labs.
        - Optimizing Spark jobs using partitionBy, repartition.
        - Using window functions for ranking and partitioning data.
    - Scenario: Given a large dataset of transactions, use PySpark to calculate the monthly total sales per store.

---

6. Cloud Platforms (Azure Focus)

Objective: Work with cloud services, especially Azure for data storage and processing.

Topics to Cover:

1. Azure Overview & Data Services

- Training Material: Microsoft Learn or Azure Docs (Azure Data Engineer learning path).
  - Services: Azure Blob Storage, Data Lake, Azure Databricks, Azure SQL Database.
- Scenario: Set up an Azure Storage account, upload a dataset to Blob Storage, and retrieve it using Azure SDK for Python.

2. Building Data Pipelines with Azure Data Factory
   - Training Material: Azure Data Factory tutorials and documentation.
     - Create pipelines to ingest, transform, and load data (ETL).
   - Scenario: Create a simple Azure Data Factory pipeline that pulls data from a REST API, cleans it using a Python script, and stores it in an Azure SQL Database.

---

7. Real-Time Project

Objective: Apply all learned concepts to a real-world data engineering project.

Topics to Cover:

1. Project Overview: Building a Data Pipeline
   - Training Material: Guides and templates for building end-to-end pipelines.
     - Designing and implementing data pipelines using the tools covered (PySpark, Azure, SQL).
     - Automating the pipeline.
   - Scenario: Build an ETL pipeline where data is fetched from an API, transformed in Spark, and loaded into Azure Data Lake. The pipeline should be automated to run daily.
2. Testing and Deployment
   - Training Material: Testing practices for data pipelines (unit tests, integration tests).
     - Deployment and monitoring in the cloud (using Azure).
   - Scenario: Deploy your data pipeline to production and monitor its performance and errors using Azure monitoring tools.

Conclusion:

For each topic, it's important to:

- Provide hands-on exercises and scenarios that closely mimic what they will encounter in the job.
- Use real-world examples to make concepts relatable.
- Focus on both coding skills (Python, SQL, PySpark) and cloud skills (Azure) since they are vital for Data Engineers.
- Ensure the project work ties everything together into a cohesive learning experience

◆◆ NEWGEN CORPORATE TRAINING CENTER ◆◆

OFFLINE / ONLINE BATCH

Address: NEWGEN CORPORATE TRAINING CENTER, Bremen Chowk, Office No.217, West Avenue, Above Atithi Hotel, Aundh, Pune – 411027

URL: http://www.newgensoftech.com

Mail Id: balkrishna8588@gmail.com

Insta Id: @Newgen_Softech