1. Foundational Skills Building (1-3 Months)

Python Programming

- **Objective**: Gain proficiency in Python, focusing on data manipulation and analysis.
- **Activities**: Practice with exercises on control structures, functions, and data structures.

Computing Fundamentals

- **Objective**: Understand essential computer science concepts such as algorithms, data structures, and memory management.
- **Activities**: Solve algorithmic problems and study computational theories.

SQL (Structured Query Language)

- **Objective**: Learn how to query and manipulate data in relational databases.
- **Activities**: Practice writing and optimizing SQL queries.

2. **Data modeling and database management**

A strong understanding of databases and data modeling ensures that your data systems are efficient and scalable, which is a must for data engineers!

Here's what you need to know:

Relational databases

Relational databases like **PostgreSQL**, **MySQL**, and **Microsoft SQL Server** are the backbone of structured data storage. As a data engineer, you'll need to:

- Design schemas that define how data is organized.

- Optimize queries for performance and efficiency.
- Understand indexing to speed up data retrieval.

For hands-on practice, check out the **Creating PostgreSQL Databases** course. If you're new to Microsoft SQL Server, the **Introduction to SQL Server** course is a great resource to get started.

NoSQL databases

NoSQL systems like **MongoDB** and **Cassandra** are designed for unstructured or semi-structured data. They're essential in scenarios where:

- Flexibility in schema design is important.
- Applications need to handle large volumes of data at scale, such as real-time analytics or social media data.

The **NoSQL Concepts** course is an excellent resource for learning the fundamentals and understanding where and how to use these powerful databases effectively.

Data warehouses

Data warehouses are specialized systems optimized for analytical queries and reporting. Tools like **Snowflake**, **Amazon Redshift**, and **Google BigQuery** are commonly used by data engineers to:

- Store and analyze large volumes of historical data.
- Aggregate data from multiple sources for business intelligence.
- Ensure fast query performance for complex analytics.

DataCamp provides courses on all of these data warehouses, as well as **data warehousing** in general, for you to get started:

- **Introduction to Snowflake**
- **Introduction to Redshift**
- **Introduction to BigQuery**

Data lakes

Data lakes, such as those built on **Amazon S3**, **Azure Data Lake**, or **Google Cloud Storage**, are designed for storing raw, unprocessed data. Unlike data warehouses, data lakes handle both structured and unstructured data, making them ideal for:

- Storing large datasets for machine learning or AI applications.
- Supporting use cases like log storage, IoT data, and streaming data.

3. Core Data Engineering Skills

1. **ETL vs. ELT**
    - Understand trade-offs: ETL (on-premise), ELT (cloud data warehouses)
2. **Batch Processing**
    - Build pipelines with Apache Spark or AWS Glue.
3. **Stream Processing**
    - Implement real-time data flows using Kafka, Flink, or Kinesis.
4. **Data Warehousing**
    - OLAP vs. OLTP, star/snowflake schemas, slowly changing dimensions

4. Cloud Technologies Exploration (1-2 Months)

Cloud Computing Platforms (AWS, Google Cloud)

- **Objective**: Gain a basic understanding of cloud services.
- **Activities**: Experiment with cloud-based storage and compute services.

5. **Big Data Ecosystem**

- **Hadoop & Spark**: Distributed storage (HDFS) vs. in-memory compute (Spark).
- **Kafka & Pub/Sub**: Event streaming fundamentals, partitioning, consumer groups.
- **Lakehouse**: Merge data lake flexibility with warehouse performance (e.g., Delta Lake).

Phase 6: Orchestration & Automation

- **Airflow**: DAG design, XComs, Airflow sensors.
- **Emerging Tools**: Prefect for dynamic workflows, Dagster for typed pipelines.

Phase 7: Governance, Security & Quality

- **Data Catalogs**: Leverage tools like DataHub or Amundsen for lineage.
- **Compliance**: GDPR/CCPA considerations, encryption at rest/in transit.
- **Testing & Monitoring**: Implement data quality checks (Great Expectations), pipeline observability (Prometheus, Grafana).

Phase 8: MLOps & Analytics Integration

- **Feature Stores**: Feast or Tecton for consistent feature pipelines.
- **Model Serving**: Integrate with TensorFlow Serving or TorchServe.
- **BI Connectivity**: Expose curated data to Tableau, Looker, or Power BI dashboards

9. Building Practical Experience and Applying Skills

Beginner Projects (1-2 Months)

- **Examples**: Web scraping, data cleaning challenges, basic data pipelines.

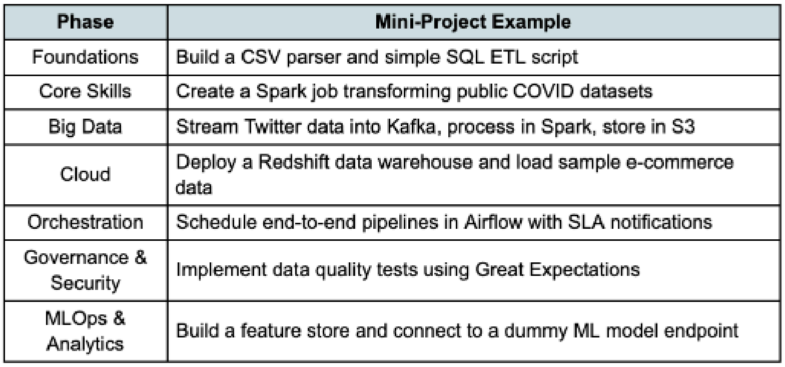
Intermediate Projects (2-4 Months)

- **Examples**: Real-time sensor data analysis, recommendation systems, cloud-based data warehouses.

Advanced Projects (4+ Months)

- **Examples**: Machine learning pipelines, real-time analytics dashboards, big data analysis with Spark.

Phase 9: Projects & Portfolio



Phase	Mini-Project Example
Foundations	Build a CSV parser and simple SQL ETL script
Core Skills	Create a Spark job transforming public COVID datasets
Big Data	Stream Twitter data into Kafka, process in Spark, store in S3
Cloud	Deploy a Redshift data warehouse and load sample e-commerce data
Orchestration	Schedule end-to-end pipelines in Airflow with SLA notifications
Governance & Security	Implement data quality tests using Great Expectations
MLOps & Analytics	Build a feature store and connect to a dummy ML model endpoint