DATA ENGINEERING | INTERMEDIATE | 5 DAYS

Enterprise Data Engineering

Build a strong foundation in modern Data Engineering using enterprise tools and proven architecture patterns, from ingestion to transformation and serving. Learn how to design reliable pipelines with governance, quality, orchestration, and performance best practices to power analytics and business reporting at scale.

Why Enterprise Data Engineering?

Faster decision-making

Enable trusted dashboards and analytics by standardizing pipelines and data models.

AI readiness

Prepare clean, validated datasets required for successful AI/ML initiatives.

Operational efficiency

Reduce manual reporting effort with automated batch and streaming pipelines.

Governance and compliance

Improve audit readiness with repeatable pipelines, validation checks, and lineage practices.

Scalable foundations

Build a platform that supports growth, high data volumes, and advanced analytics.

What you'll be able to do

- ✓ Write production-grade SQL and Python for data extraction, transformation, and validation.
- ✓ Design scalable data models for analytics and business reporting.
- ✓ Build reliable ETL/ELT pipelines using orchestration and modular transformations.
- ✓ Process high-volume datasets using Apache Spark (PySpark) distributed processing patterns.
- ✓ Work effectively with cloud data warehouses and understand performance and cost tradeoffs.
- ✓ Build real-time streaming pipelines using Kafka fundamentals.
- ✓ Apply enterprise best practices for data quality, observability, and governance readiness.
- ✓ Deliver an end-to-end mini-capstone data platform solution.

Who this is for

- Software Engineers transitioning into Data Engineering
- Data Analysts moving toward pipeline development
- ETL Developers and BI Engineers modernizing data stacks
- Junior to Intermediate Data Engineers
- Cloud Engineers and DevOps Engineers supporting data platforms
- Tech Leads and Data Architects who need scalable platform patterns

Curriculum

01 Spark for Data Engineering (Big Picture)

- Why Spark is used for ETL
- Distributed processing
- Scale-out workloads
- Fault tolerance
- ETL vs ELT in the real world
- Spark execution basics (driver and executors, lazy evaluation, transformations vs actions)
- Spark UI overview (what engineers monitor)
- **Hands-on:** Identify which workloads require Spark vs SQL-only tools

02 Databricks Environment Overview

- Databricks workspace components
- Notebook workflow
- Interactive clusters vs job clusters
- Cluster sizing basics
- **Hands-on:** Lab 1: Setup workspace and create first notebook
- **Hands-on:** Lab 2: Create cluster and validate Spark session

03 Reading Data into Spark DataFrames

- Read CSV with schema inference and explicit schema
- Read JSON nested data
- Read Parquet
- Schema management (inferSchema pitfalls, specifying schema for stability)
- **Hands-on:** Lab 3: Read raw CSV orders dataset and inspect schema
- **Hands-on:** Lab 4: Read nested JSON events and flatten selected fields

04 Core Transformations (Real ETL Patterns)

- Selecting and renaming columns
- Filtering rows
- Type casting
- Handling null values
- Adding derived columns
- **Hands-on:** Lab 5: Clean raw orders table (standardize dates, handle missing fields, fix numeric types)

05 Joins and Aggregations in Spark

- Join types and pitfalls (row explosion)
- Aggregations using groupBy
- Rollups (intro)
- Performance hint concepts (intro)
- **Hands-on:** Lab 6: Join customers + orders
- **Hands-on:** Lab 7: Create daily revenue aggregates

06 Why Delta Lake (Lakehouse Foundations)

- Problems with raw Parquet only (inconsistent writes, partial failures, no table history)
- Delta Lake benefits (ACID transactions, schema enforcement, time travel)
- **Hands-on:** Lab 8: Write cleansed dataset as Delta table
- **Hands-on:** Lab 9: Query Delta table and view versions

07 Incremental ETL Patterns (Production Requirement)

- Full refresh vs incremental
- Partition-based incremental loads
- Watermark strategy concepts
- Handling late arriving data
- **Hands-on:** Lab 10: Implement incremental append into Delta (partitioned by date)
- **Hands-on:** Lab 11: Validate incremental load row counts across runs

08 Merge/Upsert Workflows (CDC Style)

- The need for MERGE (updates, dedup, CDC-like merges)
- Using Delta MERGE INTO
- Building dimension tables using upserts
- **Hands-on:** Lab 12: MERGE customers into curated customer dimension table
- **Hands-on:** Lab 13: Update records and verify current state changes correctly

09 Data Validation and Quality Checks in Spark Pipelines

- Schema checks
- Null checks
- Duplicate checks
- Range checks
- Stop pipeline on quality failure vs warn only
- **Hands-on:** Lab 14: Add validation stage before publishing curated output
- **Hands-on:** Lab 15: Generate a data quality report output

10 Debugging Broken Data Pipelines

- Schema mismatch
- Bad data types
- Corrupt records
- Missing columns
- Join explosion issues
- **Hands-on:** Lab 16: Fix the broken pipeline exercise (wrong schema, invalid join key, corrupted file handling, missing partition folder)

11 Spark Performance Fundamentals (Must-Know)

- Why Spark pipelines become slow (shuffles, skewed joins, too many small files)
- Partitioning concepts (repartition vs coalesce)
- Caching and persistence strategy
- Broadcast joins (intro)
- **Hands-on:** Lab 17: Optimize job by reducing shuffle
- **Hands-on:** Lab 18: Fix slow join performance using broadcast join

12 File Optimization and Lakehouse Maintenance

- Small files problem
- Compaction strategies concept
- Z-Ordering concept (Databricks)
- VACUUM and retention concept
- Optimize table operations overview
- **Hands-on:** Lab 19: Run OPTIMIZE on Delta table and compare query performance
- **Hands-on:** Lab 20: Run VACUUM safely and explain retention impact

13 Building Reusable ETL Pipelines

- Notebook to modular pipeline approach
- Using parameters (processing date, source path, target path)
- Configuration-driven pipelines
- Pipeline logging patterns
- **Hands-on:** Lab 21: Add parameters to ETL notebook (run_date, env)
- **Hands-on:** Lab 22: Generate run metrics (rows processed, runtime, failed records)

14 Orchestration Options (Databricks Jobs + Airflow Overview)

- Databricks Jobs (scheduling, retries, notifications)
- Airflow integration overview (triggering Databricks job run, dependency chaining)
- **Hands-on:** Lab 23: Convert notebook into a Databricks job and schedule it
- **Hands-on:** Lab 24: Simulate failure and validate retries

15 Security and Access Control (Intro)

- Workspace permissions (concept)
- Secure handling of credentials
- Secrets management concept (Databricks secrets)
- **Hands-on:** Define access policy for data engineers, analysts, external contractors

16 End-to-End Pipeline Design (Batch ELT Architecture)

- Architecture: raw → cleansed → curated → marts
- Choosing Delta vs Parquet
- Partition keys
- Incremental vs full
- SLA and validation strategy
- **Hands-on:** Workshop: Design the capstone pipeline architecture

17 Capstone Implementation (Hands-on Build)

- Capstone goal: Build an end-to-end ELT pipeline using Databricks + Delta
- Ingest raw datasets (customers, orders, payments)
- Build cleansed Delta tables
- Build curated tables (customer dimension MERGE, daily revenue fact table)
- Implement incremental processing (run-by-date parameter)
- Add data quality checks (unique order_id, non-null customer_id, valid amounts)
- Produce reporting tables (top customers, revenue by region)
- **Hands-on:** Deliverables: notebooks/scripts, Delta tables, validation report, KPI queries

18 Fix the Broken ETL Challenge

- Broken scenario: wrong schema
- Broken scenario: missing partition folder
- Broken scenario: join explosion
- Broken scenario: duplicates causing incorrect revenue
- **Hands-on:** Lab 25: Troubleshoot and fix the broken ETL pipeline

19 Production Readiness Checklist

- Idempotency checklist
- Backfill strategy
- Late arriving data plan
- Observability plan (logs, run metrics, alerts)
- Cost optimization checklist
- **Hands-on:** Workshop: Create a production readiness checklist for your team