MASTERING DATABRICKS DATA ENGINEERING

Using AWS ● PySpark ● Delta Lake ● Kafka ● Airflow ● Generative AI

Complete Industry-Ready Curriculum | Hands-On Projects | Placement Support

Introduction to Big Data and Hadoop

- What is Big Data? — 5 Vs (Volume, Velocity, Variety, Veracity, Value)
- What is Hadoop? — HDFS, YARN, MapReduce Architecture
- What is Spark? — In-Memory Processing vs Disk-Based MapReduce
- What are NoSQL Databases? — Types: Document, Key-Value, Columnar, Graph
- Difference Between Hadoop and Spark — Performance, Use Cases
- Common Big Data Problems — Data Silos, Quality, Security, Scalability
- Hadoop Ecosystem — HDFS, YARN, Hive, HBase, Sqoop, Flume, Oozie

AWS Introduction (50 Hours)

EC2 (Elastic Compute Cloud)

- Create Windows/Mac/Linux Servers
- Create a Sample Website
- Autoscaling — Horizontal vs Vertical Scaling
- Create and Use AMIs (Amazon Machine Images)
- Security Groups and Network Configuration
- EC2 Instance Types — General Purpose, Compute, Memory, Storage Optimized
- Spot Instances vs On-Demand vs Reserved Instances

S3 (Simple Storage Service)

- Store Data in S3 — Buckets, Objects, Folders
- S3 Storage Classes — Standard, IA, Glacier, Intelligent Tiering
- Submit Commands in Client Mode — AWS CLI
- Get Data from Various Sources and Store in S3
- S3 Bucket Policies — Public vs Private Access
- S3 Versioning and Lifecycle Policies
- S3 Event Notifications — Trigger Lambda on File Upload

- S3 Transfer Acceleration for Large Files
- S3 Select — Query Data Without Downloading
- Cross-Region Replication (CRR) for Disaster Recovery

Athena (Serverless Query Service)

- What is Serverless Computing? — Pay-Per-Query Model
- Process JSON and CSV Data with Athena
- Create External Tables on S3 Data
- Partition Pruning for Performance Optimization
- Athena Query Optimization — Columnar Formats (Parquet, ORC)
- Athena CTAS (Create Table As Select)
- Athena Workgroups for Cost Control

RDS (Relational Database Service)

- Create Different Databases — MySQL, PostgreSQL, Oracle, SQL Server
- Create Sample Tables and Process Data
- Best Practices for Cost Optimization
- Practice Oracle and MySQL Using RDS
- RDS Multi-AZ for High Availability
- RDS Read Replicas for Scalability
- Automated Backups and Point-in-Time Recovery
- RDS Performance Insights — Query Monitoring

EMR (Elastic MapReduce)

- Practice PySpark and Hive on EMR
- Create EMR Clusters and Process Data
- EMR vs EC2 — When to Use Each
- Hive Internals and Sample Programs
- Import Data from RDS to S3 Using Sqoop
- EMR Cluster Sizing — Master, Core, Task Nodes
- EMR Spot Instances for Cost Savings
- EMR Notebook for Interactive Analysis
- EMR with Spark vs EMR with Hive — Performance Comparison
- EMR Security — Encryption, IAM Roles, Security Groups

AWS Glue (Serverless ETL)

- Glue Architecture — Glue Catalog, Crawlers, Jobs, Triggers
- Process CSV and JSON Data Using Glue
- Retrieve Data from Athena Using Glue
- Use Crawlers to Auto-Detect Schema
- Execute PySpark/Scala Jobs in Glue
- Glue DynamicFrame vs Spark DataFrame
- Glue ETL Best Practices — Job Bookmarks, Pushdown Predicates
- Glue Data Quality — Validation Rules

- Glue Change Data Capture (CDC) — Track Inserts, Updates, Deletes
- Implementing SCD Type 1 and Type 2 in Glue
- Glue Workflows — Orchestrate Multiple Jobs
- Glue Triggers — Event-Driven vs Scheduled
- Glue Connection — JDBC to RDS, Redshift, Oracle
- Incremental Load Patterns with Glue Job Bookmarks
- Glue Custom Transformations — Python Shell vs PySpark
- Partitioning Strategies for Glue Output
- Glue Cost Optimization — DPU Sizing, Job Metrics

Lambda & Boto3

- Access AWS Resources Using Boto3 from PyCharm
- Use Boto3 in Lambda Functions
- Integrate Lambda with Glue and Redshift
- Connect Boto3 with Services Like EC2, EMR, Glue, Redshift
- Lambda Triggers — S3, CloudWatch, API Gateway, SQS, SNS
- Lambda Layers — Reusable Code and Dependencies
- Lambda Environment Variables and Secrets Manager
- Error Handling and Retries in Lambda
- Lambda with Step Functions for Complex Workflows

Redshift (Data Warehouse)

- Load and Process Data from S3
- SortKey and DistKey Optimization
- Redshift Architecture — Leader Node, Compute Nodes
- Compare Snowflake vs Redshift
- Redshift Spectrum — Query S3 Directly
- COPY Command — Efficient Data Loading
- UNLOAD Command — Export to S3
- Redshift Concurrency Scaling
- Workload Management (WLM) — Query Queues
- Redshift Materialized Views
- Redshift Data Sharing Across Clusters

CloudWatch (Monitoring & Logging)

- How to Monitor Resources
- Debugging Application Failures
- Autoscaling Based on CloudWatch Metrics
- Usage Across AWS Services (EC2, RDS, Glue)
- CloudWatch Logs Insights — Query Log Data
- CloudWatch Alarms — Email/SMS Notifications
- CloudWatch Dashboards — Custom Metrics

IAM (Identity and Access Management)

- Users, Groups, and Roles
- Custom Policies — JSON Policy Documents
- Importance of IAM Keys in Snowflake, Databricks, PyCharm Use Cases
- IAM Best Practices — Least Privilege Principle
- IAM Role Assumption for Cross-Account Access
- IAM Policy Simulator for Testing
- MFA (Multi-Factor Authentication) Setup

Introduction to Apache Spark

Spark Core Fundamentals

- Why Use Spark Instead of Hadoop MapReduce?
- Importance of HDFS/YARN in Spark
- Spark Architecture — Driver, Executors, Cluster Manager
- Types of APIs: RDD, DataFrame, Dataset
- Use Cases for Spark — Batch, Streaming, ML, Graph Processing
- Why Spark is Faster Than MapReduce — DAG, In-Memory Processing
- In-Memory Processing in Spark — Memory vs Disk Persistence
- Spark Deployment Modes — Local, Standalone, YARN, Mesos, Kubernetes

RDD Internals

- Properties of RDD: Immutability, Laziness, Fault Tolerance
- SparkContext, SQLContext, SparkSession Internals
- Create RDDs in Different Ways — parallelize, textFile, wholeTextFiles
- Transformations and Actions — Narrow vs Wide Transformations
- Debugging Transformations — toDebugString, explain()
- Spark Web UI — Jobs, Stages, Tasks, Storage, Environment
- RDD Partitioning — Default Partitioning, Custom Partitioners
- RDD Persistence — cache() vs persist() with Storage Levels

RDD Hands-On Programming

- Map, FlatMap, Filter, Distinct
- ReduceByKey vs GroupByKey — Performance Comparison
- Spark-submit Examples — Client vs Cluster Mode
- 20 RDD Use Case Programs
- Word Count, Top N Records, Joins, Set Operations
- Aggregations — sum, avg, count, min, max
- Sorting — sortBy, sortByKey, top, takeOrdered

Spark SQL & DataFrames

- Convert RDD to DataFrame

- Python DataFrame vs Spark DataFrame
- DataFrame Reader — read.csv, read.json, read.parquet
- Processing Data in Different Formats: CSV, JSON, XML, Avro, ORC, Text, Parquet
- Database Integration: Oracle, MySQL, PostgreSQL
- Sqoop vs Spark JDBC — Which to Use When
- NoSQL Integration: HBase, Cassandra, MongoDB
- DataFrame Operations — select, filter, groupBy, agg, join
- Window Functions — rank, dense_rank, row_number, lag, lead
- UDF (User Defined Functions) — Python UDF vs Pandas UDF
- Spark SQL Catalyst Optimizer — Logical and Physical Plans

PySpark Advanced Concepts

- Dataset API Importance — Type Safety
- Spark Memory Management — Execution vs Storage Memory
- Resource Optimization — Executor Memory, Cores, Shuffle Partitions
- Spark Debugging with Client Mode and Web UI
- Automate Spark with Oozie and Airflow
- Spark-Snowflake Integration
- Broadcast Variables and Accumulators
- Spark Partitioning Strategies — repartition vs coalesce
- Handling Data Skew — Salting Technique
- Adaptive Query Execution (AQE) — Dynamic Partition Pruning
- Spark on Kubernetes — Container Orchestration
- Delta Lake Integration with PySpark

Change Data Capture (CDC) with PySpark

- What is CDC? — Tracking Inserts, Updates, Deletes
- CDC Sources — Database Logs, Triggers, Timestamps
- Implementing CDC with AWS DMS (Database Migration Service)
- SCD Type 1 — Overwrite Existing Records
  - Update current values without history tracking
  - Use MERGE INTO for Type 1 updates
- SCD Type 2 — Maintain Full History
  - Add effective_start_date and effective_end_date columns
  - Add is_current flag for active records
  - Implement with Delta Lake MERGE statement
  - Handle late-arriving data scenarios
- SCD Type 3 — Track Limited History (Previous + Current)
- Delta Lake MERGE for CDC — Upsert Pattern
- CDC Performance Optimization — Partition by Date, Z-Ordering
- Incremental CDC Processing vs Full Snapshot

Spark Streaming

Introduction to Spark Streaming

- Micro-Batch vs Stream Processing — Batch Intervals
- D-Stream API Internals — Discretized Streams

- Live Data Processing — Windowing, Sliding Intervals

Structured Streaming

- Real-World Examples — IoT, Clickstream, Logs
- Integration with Kafka — Kafka Consumer API
- Log Analysis — Real-Time Error Detection
- Export to Databases — JDBC Sink
- Snowflake Integration with Streaming
- Watermarking — Handling Late Data
- Output Modes — Append, Update, Complete
- Checkpointing for Fault Tolerance
- Exactly-Once Semantics with Kafka + Delta Lake

Apache Kafka

- Kafka Architecture — Brokers, Topics, Partitions, Offsets
- Producer API — Send Messages to Kafka
- Consumer API — Read Messages from Kafka
- Consumer Groups — Parallel Processing
- Integration with Spark — Structured Streaming with Kafka
- End-to-End Workflow with AWS, Databricks, and Cloudera
- Kafka Connect — Source and Sink Connectors
- Kafka Streams — Stream Processing Framework
- Schema Registry — Avro, Protobuf, JSON Schema
- Kafka Performance Tuning — Batch Size, Compression, Partitioning

Apache NiFi

- NiFi Internals — Flow-Based Programming
- Data Flow Examples (Local to S3, API to S3)
- Integration with Kafka and Spark
- Templates & Most Frequently Used Processors
  - GetFile, PutFile, GetHTTP, InvokeHTTP
  - ConvertRecord, UpdateAttribute, RouteOnAttribute
  - PutS3Object, FetchS3Object
- NiFi Registry — Version Control for Flows
- NiFi Cluster Setup for High Availability

Apache Airflow

- Airflow Installation in EC2
- Data Pipeline Creation — DAGs (Directed Acyclic Graphs)
- DAG Management — Scheduling, Dependencies, Retries
- Airflow-Spark-Snowflake Integration
- Airflow Operators — BashOperator, PythonOperator, SparkSubmitOperator
- Airflow Sensors — FileSensor, S3KeySensor, TimeDeltaSensor
- Airflow XComs — Cross-Task Communication
- Airflow Variables and Connections — Secure Credentials

- Airflow with AWS — EMR, Glue, Redshift Operators
- Monitoring Airflow — UI, Logs, Alerts

Introduction to Databricks

- Databricks vs Spark vs Snowflake — Architecture Comparison
- Databricks Architecture — Control Plane, Data Plane
- Working in Databricks Workspace
- Using Databricks Notebooks — Python, Scala, SQL, R
- Databricks Community Edition vs AWS Databricks

Databricks File System (DBFS)

- What is DBFS? — Abstraction Layer over S3
- DBFS Commands — mkdirs, cp, mv, head, put, rm, rmdir
- Magic Commands — %sh, %fs, %scala, %python, %sql, %md
- Mounting S3 Buckets to DBFS
- DBFS vs Direct S3 Access — Performance Implications

Databricks Utilities (dbutils)

- Credentials Utility — dbutils.credentials
- FileSystem Utility — dbutils.fs
- Notebook Utility — dbutils.notebook.run, exit
- Secrets Utility — dbutils.secrets.get
- Widgets Utility — dbutils.widgets.text, dropdown

Databricks Cluster Management

- Creating and Configuring Clusters
- Managing Clusters — Start, Terminate, Delete
- Cluster Information and Logs
- Types of Clusters: All-Purpose, Job Clusters
- Cluster Modes: Standard, High Concurrency, Autoscaling
- Cluster Libraries — Install PyPI, Maven, JAR, Wheel
- Cluster Policies — Cost Control, Resource Limits
- Spot Instances in Databricks Clusters

Databricks Integration with AWS

- Integration with S3 — Read/Write Data
- Integration with RDS — JDBC Connections
- Integration with Redshift — Databricks-Redshift Connector
- Integration with Glue Catalog — External Tables
- Integration with Kinesis — Real-Time Streaming
- Integration with DynamoDB — NoSQL Data Access
- IAM Roles for Databricks — Instance Profiles

- AWS Secrets Manager Integration

Databricks Streaming API

- Introduction to Structured Streaming in Databricks
- Handling Bad Records — badRecordsPath
- Regular Expression Parsing in Streaming
- Streaming Data into S3 and Delta Tables
- Trigger Modes — Once, ProcessingTime, Continuous
- Foreachbatch for Custom Sink Logic

Databricks Lakehouse (Delta Lake)

- Data Lake vs Delta Lake — ACID Transactions
- Delta Lake Best Practices
- Delete, Update, Alter Tables — DML Operations
- Optimization Steps — OPTIMIZE, VACUUM, Z-ORDER
- Handling SCD (Type 1 & Type 2) with Delta MERGE
  - SCD Type 1 Implementation — Overwrite Pattern
  - SCD Type 2 Implementation — Historical Tracking
- Deduplication Strategies — Distinct, Window Functions
- Streaming Data Handling with Delta Lake
- Delta Lake Time Travel — VERSION AS OF, TIMESTAMP AS OF
- Change Data Feed (CDF) — Track Row-Level Changes
- Delta Lake Schema Evolution — mergeSchema Option
- Delta Lake Constraints — NOT NULL, CHECK Constraints

Databricks Unity Catalog

- Create Schema and Table Using Unity Catalog
- Access Controls, User Management, and Metastore
- Row-Level Access Control — Dynamic Views
- Masking Columns — Redaction for PII
- Roles, Users, and Groups — RBAC
- Managing External Tables — S3-Backed Tables
- Lakehouse Federation — Query External Databases
- Data Lineage — Track Data Flow
- Audit Logging — User Actions, Data Access

Databricks Workflows

- Introduction to Workflows — Job Orchestration
- Creating, Running, and Managing Jobs
- Scheduling and Monitoring Jobs — Cron Expressions
- Create Dependency Between Multiple Jobs — Task Dependencies
- Job Parameters — Dynamic Execution
- Job Notifications — Email, Webhooks, Slack
- Retry Policies — Max Retries, Timeout

Delta Live Tables (DLT)

- Introduction to Delta Live Tables — Declarative ETL
- Creating and Configuring Delta Pipelines
- Real-Time Streaming with Delta Live Tables
- Error Handling and Recovery in Delta Live Tables
- Delta Live Tables Best Practices
- Expectations — Data Quality Rules
- Materialized Views vs Streaming Tables
- DLT Pipeline Observability — Event Log, Lineage Graph

Generative AI for Data Engineering

AI-Powered Development Fundamentals

- Introduction to Large Language Models (LLMs) for Code Generation
- Prompt Engineering for Data Engineering Tasks
  - Zero-shot, Few-shot Prompting for PySpark Code
  - Chain-of-Thought Prompting for Complex ETL Logic
- Security Best Practices — Never Share Credentials or PII with AI Tools

Claude.ai for Data Engineering

- Using Claude for PySpark Development
  - Generate PySpark transformation code from requirements
  - Convert SQL queries to PySpark DataFrames
  - Debug PySpark errors with stack traces
- Using Claude for AWS Glue
  - Generate Glue ETL scripts from data flow diagrams
  - Create Glue Catalog schema definitions
  - Optimize Glue DPU usage with AI suggestions
- Claude for Databricks Notebooks
  - Auto-generate Delta Lake MERGE statements for CDC
  - Create DLT pipeline code from requirements
- Claude Artifacts — Interactive Code Visualizations

GitHub Copilot for PySpark & AWS

- Installing GitHub Copilot in VS Code
- Autocomplete for PySpark DataFrame Operations
  - Inline suggestions for window functions, joins, aggregations
  - Generate pytest unit tests for PySpark functions
- GitHub Copilot for Boto3 — AWS SDK Autocomplete
  - S3, Glue, EMR, Lambda code completion
- GitHub Copilot Chat
  - /explain — Explain complex PySpark code

- /fix — Debug PySpark errors
- /tests — Generate unit tests

ChatGPT for Data Engineering

- Using ChatGPT for PySpark Code Generation
  - Generate complete ETL pipelines from natural language
- Using ChatGPT for AWS CloudFormation/Terraform
  - Generate infrastructure-as-code templates
- ChatGPT Code Interpreter
  - Upload CSV and generate PySpark transformation code
- Custom GPTs for Data Engineering — PySpark Style Guide GPT

Cursor IDE for Data Engineering

- Cursor Overview — AI-First Code Editor
- Cmd+K — Inline Code Generation and Editing
  - Generate entire PySpark ETL pipeline from requirements
  - Refactor legacy Spark code to modern patterns
- Cursor Composer — Multi-File AI Editing
- Codebase Indexing — AI Learns Your Project Structure

Claude Code — Agentic CLI Development

- What is Claude Code — Terminal-Based AI Agent
- Using Claude Code for Data Engineering
  - 'claude code create a PySpark CDC pipeline with SCD Type 2'
  - Auto-generate Glue jobs, Airflow DAGs, DLT pipelines
- Autonomous Testing and Debugging

AI Workflows for Data Engineers

- Workflow 1: Generate PySpark CDC Code with Claude → Test with Copilot
- Workflow 2: Debug Glue Job Errors with ChatGPT
- Workflow 3: Convert Hive SQL to PySpark with Cursor
- Workflow 4: Auto-Generate Databricks DLT Pipeline
- Best Practices
  - Always review and test AI-generated code
  - Never paste production data or credentials into AI tools
  - Use enterprise AI tools with data residency guarantees

🎯 WHAT YOU'LL MASTER

✓ Build production-grade PySpark data pipelines on AWS EMR and Databricks

- ✓ Implement CDC with SCD Type 1 and Type 2 using Delta Lake MERGE
- ✓ Master AWS Glue for serverless ETL with incremental loads and job bookmarks
- ✓ Design real-time streaming pipelines with Kafka, Spark Structured Streaming, and Delta Lake
- ✓ Orchestrate complex workflows with Apache Airflow and Databricks Workflows
- ✓ Use Generative AI tools (Claude, Copilot, Cursor) to 10x development speed
- ✓ Implement Unity Catalog for data governance with row-level security and PII masking
- ✓ Deploy Delta Live Tables for declarative ETL with data quality expectations

📞 ENROLLMENT & CONTACT

🌐 **Website:** sreyobhilashiIT.com

📧 **Email:** info@databrickstraining.in

📱 **Phone:** +91-8500002025

💬 **WhatsApp:** +91-9247159150

Training & Placement Excellence | Industry-Ready Curriculum