Job Oriented Program (Self Taught) to Become an Azure Data Engineer for Absolute Beginners to Pro In 2026-27

Prepared By: **Ganesh. R**

Azure Senior Data Engineer

@rg_data_talks

@rganesh0203

Azure DE Roadmap

1

"Cloud" means using online services like Amazon Web Services, Google Cloud, or Microsoft Azure to build and manage systems for storing, processing, and analyzing data, instead of using their own physical computers. This

2

Python, in simple terms, is a versatile programming language that data engineers use to build and manage systems for handling data.

3

Azure Storage offers highly available, massively scalable, durable, and secure storage for a variety of data objects in the cloud.

4

SQL, or Structured Query Language, in simple words for a data engineer, is the language used to talk to and manage data in relational databases.

5

ADF is a cloud-based service for data integration that allows data engineers to build, schedule, and orchestrate data workflows for tasks like ETL and data movement at scale

6

It's the way to use the popular Python to work with Apache Spark, a powerful engine analyzing massive datasets.

7

a low-code, serverless workflow and integration service used for automating tasks like data orchestration, process automation, and connecting disparate systems

8

"Big data" is about handling extremely large, complex, and fast-moving datasets that traditional systems can't manage

Continues

9

ADB is aa unified data and AI platform that helps engineers build and maintain data solutions, including ETL pipelines and machine learning workflows

10

Snowflake is a cloud-based data warehouse that is used to store, process, and analyze large amounts of data efficiently

11

Airflow is a workflow management platform that acts as a smart manager to programmatically schedule, monitor, and orchestrate complex data pipelines and tasks

12

Kafka is an event streaming platform, essentially a message bus for handling large volumes of real-time data

13

Azure Functions, in simple terms for a data engineer, is a serverless compute service that allows you to run small pieces of code

14

It is a cloud-based data platform large volumes of log and performance data generated by Azure and non-Azure resources.

15

Is most commonly refers to Microsoft Fabric, a unified analytics platform that simplifies the entire data workflow from ingestion and transformation to analysis and reporting

16

Key Vault is a secure digital "safe box" in the cloud used by data engineers to store and manage sensitive information like passwords, API keys, and encryption keys

Continues

17

It is a data catalog and governance tool that helps you find, understand, and manage all your data assets across different systems, whether they are on-premises or in the cloud.

18

Azure Cosmos DB can be understood as a globally distributed, multi-model NoSQL database service that offers elastic scalability and guaranteed low latency for handling large volumes of diverse data.

19

Power BI is a business intelligence (BI) tool used to connect to data, transform it, and then visualize it to create interactive reports and dashboards

20

DBT (data build tool) is a tool that helps transform raw data into clean, analysis-ready data using SQL

21

Azure DevOps, CI/CD (Continuous Integration/Continuous Delivery or Deployment) refers to the automation of the entire lifecycle of data pipelines, data transformations, and related infrastructure.

22

Finally designing, building, and maintaining data pipelines and systems to collect, transform, store, and analyze data

You Should Cover

- Python
- SQL
- DataBricks
- Airflow
- Snowflake
- PySpark
- Kafka
- Storage
- Azure Data Factory
- Power BI
- Azure Devops
- Cosmos DB
- Logic Apps
- Log Analytics
- Fabric
- Purview
- DBT
- Key Vault
- Big Data
- Azure Functions

Azure Cloud Computing

Labs to Perform

LAB 1 - Explore Compute and storage options for Data Engineering Workloads

LAB 2 - Load and Save Data through RDD and Data Frame in PySpark

LAB 3 - Configuring Single Node Single Cluster in Kafka

LAB 4 - Run Interactive Queries using Azure Functions Analytics Server-less and confiure data masking

LAB 5 - 5 Data Exploration and Transformation in Azure Databricks and working with delta live tables

LAB 6 - Explore Transform and Load Data into the Data Warehouse using Pyspark

LAB 7 involves importing and transferring data to the data warehouse and then presenting it visually on PowerBI.

LAB 8 - Transform Data with Azure Snowflake with Airflow Pipeline

LAB 9 - Real Time Stream Processing with Stream Analytics

LAB 10 - Create a Stream Processing Solution with Kafka and Databricks

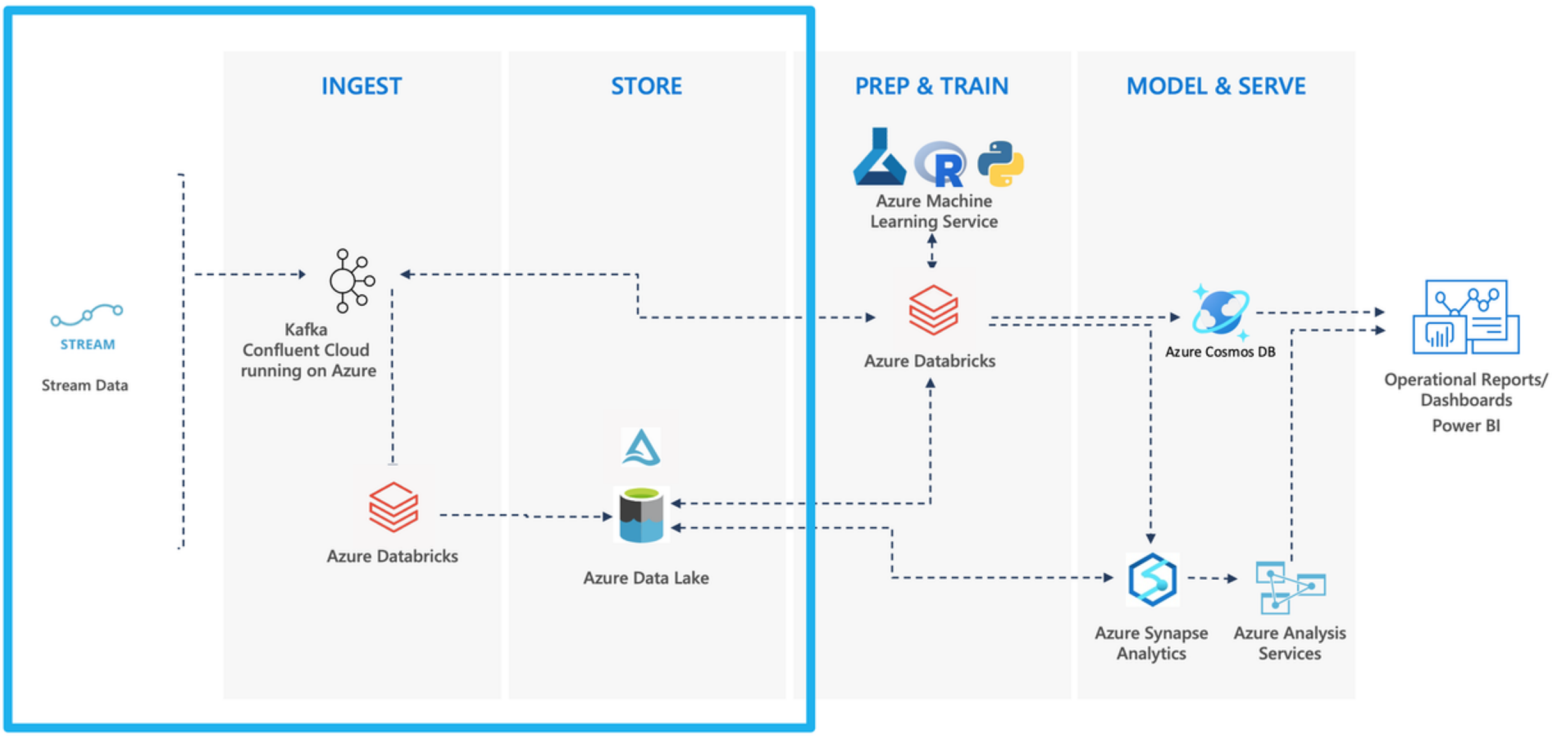
LAB 11- Data Catalog & Data Governance

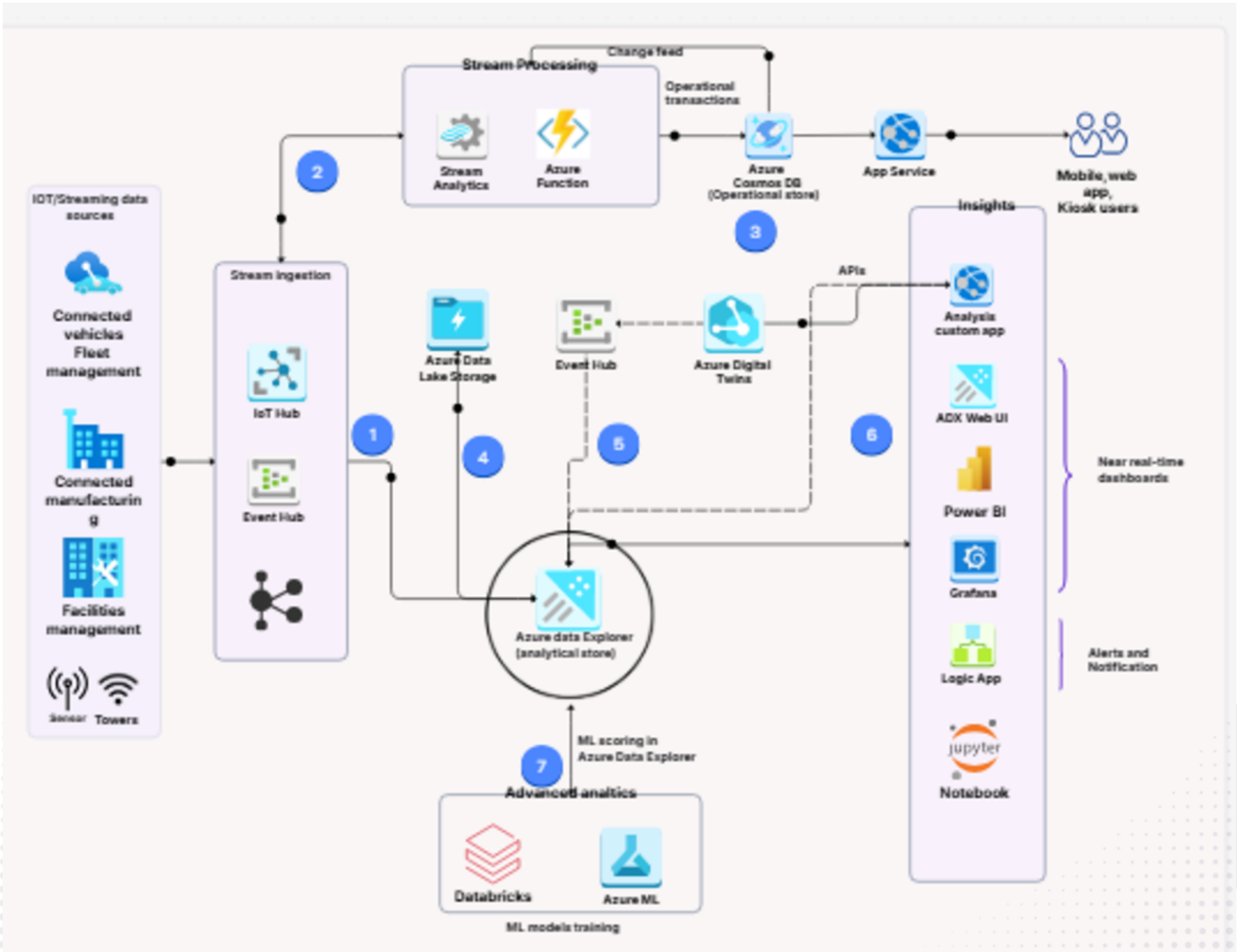
LAB 11- dbt on Databricks Project

LAB 12- Real-Time Flight Delay Analytics (Batch + Streaming)

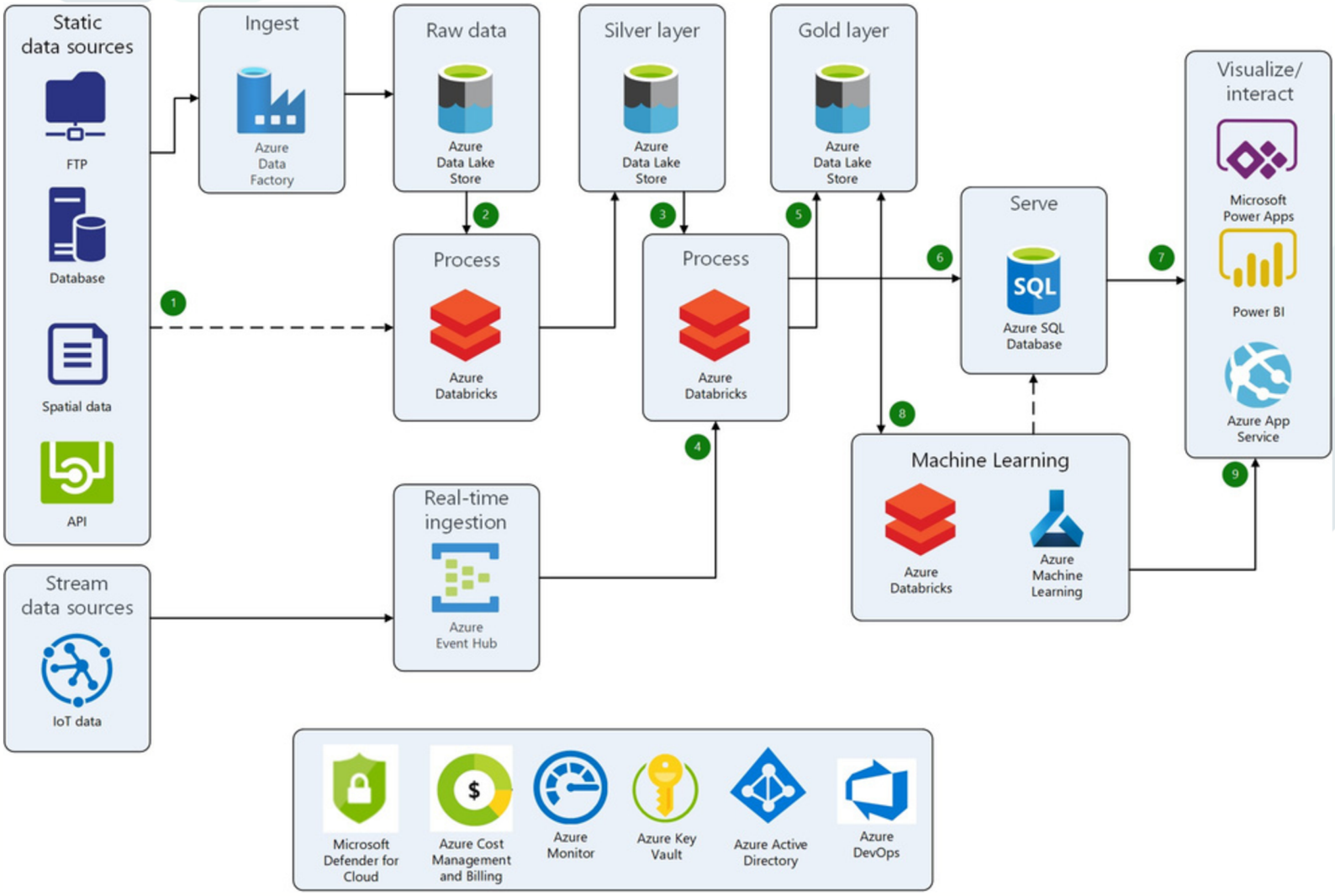
LAB 13- E-commerce Open Dataset)

Streaming Architecture





Batch Architecture



Module 1: Python

Introduction
- What is Python?
- Why Python?
- Who uses Python?
- History of Python
- Features of Python
- Where Python can be used?
- Installing Python from the command line
- IDLE
- Running Python scripts on Windows/Unix/Linux

Basic Python Syntax
- Basic Syntax
- Identifiers
- Rules to define identifiers
- Single-line comments
- Multi-line comments
- Reserved Keywords
- Naming Conventions
- print(), type() and id() functions
- input() and raw_input() function
- Type Conversion functions
- del keyword

Data Types
- What is a Datatype?
- Built-in Datatypes
- None Type
- Numeric Types
- Sequences
- Sets
- Mappings
- Bool Datatype

Operators
- Arithmetic operators
- Assignment operators
- Unary minus operators
- Relational operators
- Logical operators
- Membership operators

String Handling
- What is string?
- Reading Data From Strings in Python
- String slicing
- String concatenation
- String formatting in Python
- Working with string functions

Flow Control Statements
- Conditional Statements
- Transfer Statements
- Iterative Statements

Conditional Statements
- if statement (One-way decisions)
- if...else statement (Two-way decisions)
- if...elif...else statement (Multi-way)
- Nested if else
- Single line if else statement
- Values as Conditions
- Using Logical Operators

Transfer Statements
- Break statements
- Continue statements
- Pass statements

Iterative Statements
- While loop
- while...else
- For loop
- Infinite loop
- Nested loops
- for...else

Collections

- Introduction to collections
- Lists
- Tuples
- Sets
- Dictionaries
- Collections indexing and slicing
- Working with methods of collections
- Iterating through collections
- Nested collections
- Getting dictionary values
- Counting with dictionaries
- Differences between List, Tuple, and Set

Working with Lists

- Declaration and accessing lists
- Data updation in lists
- Data deletion from lists
- Understanding basic list operations
- Python list built-in functions
- Alias and clone
- Sorting list elements
- Nested lists
- List comprehensions

Working with Tuples

- Declaration and accessing tuples
- Data updation in Tuples
- Data deletion from Tuples
- Understanding the basic Tuples operations
- Understanding tuple indexing, slicing, and matrices
- Python tuple built-in functions
- Experiencing tuples with methods

Working with Dictionary

- Declaration and accessing dictionary
- How to access elements from a dictionary
- Built-in functions on dictionaries
- How to add an element to a dictionary
- How to delete elements from a dictionary
- Dictionary operations

Functions

- Defining a function
- Calling a function
- Function parameters
- Types of arguments
- Handling return values
- Scope of variable – global, local
- Passing collections to a function
- Lambda functions / anonymous functions
- filter() and map() functions

Understanding Function Arguments

- Positional arguments
- Keyword arguments
- Default arguments
- Variable-length arguments

Modules

- What is a module?
- Types of modules
- The import statement
- from...import
- Reloading modules
- dir() function
- main() function

Packages

- Introduction to packages
- __init__.py file
- Defining packages
- Importing from packages
- Defining sub-packages
- Importing from sub-packages

Date, OS, Sub-Process Modules

- Definitions of OS
- Date format conversions
- Usage of date, OS in real time
- Definition of date modules
- Definitions of sub-process modules

File Handling

- What is a file?
- Opening a file
- Reading data from a file
- Writing data to a file
- Closing a file

Advance Python

- Working with the methods of file objects
- Replacing the content of a file
- Working with directories
- Working with CSV files
- Assignments

OOPs Concepts

- Introduction to OOP's programming
- Features of OOPs
- Classes and Objects
- Encapsulation
- Abstraction
- Inheritance
- Polymorphism

Classes and Objects

- Creating a class
- Constructor
- Namespaces
- Types of Methods
- Passing Members of one Class to Another Class
- Inner classes

Inheritance and Polymorphism

- Constructors in inheritance
- The super() method
- Types of inheritance
- Polymorphism
- Operator overloading
- Method overloading
- Method overriding

Abstract Classes and Interfaces

- Abstract method and abstract class
- Interfaces in Python
- Abstract classes vs interfaces
- Assignments

Exception Handling

- Syntax errors
- Runtime errors
- What is an exception?
- Need of exception handling
- Predefined exceptions
- try, except, and finally clauses
- Handling multiple exceptions
- Nested try, except, and finally blocks
- raise, assert statements
- Assignments

Regular Expressions

- Regular expressions syntax
- Understanding regular expressions
- Special characters
- Character classes
- Forming regular expressions
- Matching at beginning or end
- match(), search() and sub() functions
- Splitting a string
- Replacing text
- Flags

Database Connectivity

- Introduction to RDBMS
- Installation of Oracle database access
- Creating Oracle database instances
- Establishing connection with Oracle
- Executing SQL queries
- Assignments

Python JSON Parsers

- How to read JSON
- How to load JSON
- How to parse JSON
- Assignments

Data Analytics

- Introduction to Big Data
- Pandas
- NumPy
- Matplotlib

Resources:

https://www.youtube.com/watch?v=QXeEoD0pB3E&list=PLsyeobzWxl7poL9JTVyndKe62ieoN-MZ3

Module 2: MS SQL Server

1) Introduction to SQL Server

- What is Data
- What is Database
- What is DBMS
- What is RDBMS
- What is SQL Server
- Installing SQL Server Management Studio
- SQL Server Authentication mode
- SQL Server Windows Authentication mode

2) SQL Commands

- i) Data Definition Language Commands (DDL)
- CREATE
- ALTER
- DROP
- TRUNCATE
- ii) Data Manipulate Language Commands (DML)
- INSERT
- DELETE
- UPDATE
- LOCK
- iii) Data Query Language Commands (DQL)
- SELECT
- iv) Data Control Language Commands (DCL)
- GRANT
- INVOKE
- v) Transaction Control Language Commands (TCL)
- COMMIT
- TRANSACTION
- ROLLBACK

3) Database

- SYSTEM DATABASES
- USER DEFINED DATABASES
- CREATE DATABASE

4) Tables

- SYSTEM DEFINED TABLE
- USER DEFINED TABLE
- DEFINE SCHEMA

5) SQL Datatypes

- INT
- BIGINT
- SMALLINT
- CHAR
- VARCHAR
- NVARCHAR
- DATETIME
- DATE
- TIME
- XML
- DECIMAL
- NUMERIC

6) SQL Keys

- Primary Key
- Foreign Key
- Unique Key
- Super Key
- Natural Key
- Alternative Key
- Candidate Key
- Composite Key
- Surrogate Key

7) SQL Constraints

- Null
- Not Null
- Default
- Primary Key
- Unique Key
- Foreign Key

8) SQL Joins

- LEFT JOIN / LEFT OUTER JOIN
- RIGHT JOIN / RIGHT OUTER JOIN
- FULL JOIN / FULL OUTER JOIN
- SELF JOIN
- INNER JOIN / JOIN
- CROSS JOIN

9) SQL Clause

- DISTINCT
- TOP
- WHERE
- ORDER BY
- IN
- NOT IN
- IS NULL
- IS NOT NULL

10) SQL Operators

- AND
- OR
- <
- <=
- =
- <>
- !=
- LIKE
- WILDCARD
- BETWEEN
- UNION
- UNION ALL
- EXCEPT
- INTERSECT

11) SQL Window Functions

- ROW_NUMBER()
- RANK()
- DENSE_RANK()
- NTILE()
- LAG()
- LEAD()
- FIRST_VALUE()
- LAST_VALUE()

12) SQL Datetime Functions

- DATEDADD()
- DATEDIFF()
- DATENAME()
- DATETIME()
- DATEADD()
- GETDATE()
- EOMONTH()
- DAY
- MONTH
- YEAR

13) SQL String Functions

- LTRIM
- RTRIM
- RIGHT
- LEFT
- SUBSTRING
- CHARINDEX
- PATINDEX
- UPPER
- LOWER
- REPLACE
- STUFF
- REVERSE
- LEN
- REPLICATE
- CONCAT

14) SQL Aggregate Functions

- SUM
- COUNT
- MAX
- MIN
- AVG

15) SQL Conditional Statements

- IIF
- CASE
- WHILE
- IF
- BREAK
- CONTINUE

16) SQL Advanced Functions

- ISNULL
- COALESCE
- ROLLUP
- CUBE
- PIVOT
- UNPIVOT
- SUB QUERIES
- CO RELATED SUB QUERIES
- DERIVED TABLES
- TABLE VARIABLES
- VARIABLES
- COMMON TABLE EXPRESSIONS
- TEMP TABLES

17) Views in SQL
- SIMPLE VIEW
- COMPLEX VIEW
- ENCRYPTION AND SCHEMA BINDING

18) Stored Procedure in SQL
- Stored Procedure without parameter
- Stored Procedure input parameter
- Stored Procedure output parameter
- Stored Procedure Return Value
- Drop Stored Procedure
- Transaction, Commit, Rollback
- Try-Catch Exception Handling
- Encrypt Stored Procedure

19) Functions in SQL
- Multi Table Valued Function
- Inline Function
- Scalar Function

20) Merge in SQL
- MERGE

21) Dynamic SQL
- Dynamic SQL

22) Indexes in SQL
- Create Index
- Cluster Index
- Non-Cluster Index
- Filter Index
- Covered Index
- Column Stored Index
- Drop Index
- Rebuild Index

23) Triggers in SQL
- DDL Triggers
- DML Triggers

24) Cursors in SQL
- Life Cycle of Cursor
- Declare Cursor Statement
- Open Cursor Statement
- Fetch Cursor Statement
- Close Cursor Statement

Resources:

https://www.youtube.com/watch?v=D_0Veya2XAY&list=PLNcg_FV9n7qZY_2eAtUzEUulNjTJREhQe

Module 3: PySpark

Why Spark was developed.

- Low level APIs RDDs
- Creating & Manipulating RDDs
- Transformations & Actions
- Connecting to Data Sources
- What is Spark and its features.
- Spark Main Components.
- Introduction to Spark.
- HDFS Commands
- Introduction to SparkSession
- RDD Fundamentals
- What is RDD
- RDD Properties
- When to use RDD
- RDD Problems
- Create RDD
- Different Ways to Create RDDs
- RDD Operations
- Transformations - Low Level
- Transformations - Join Types
- Actions - Total Aggregations
- Shuffle and Combiner
- Transformations - Key Aggregations
- Transformations - Sorting
- Transformations - Ranking
- Transformations - Set
- Transformations - Sampling
- Transformations - Partition
- Transformations - Repartition
- Transformations - Repartition and Sort
- Transformations - Coalesce
- Transformations - Repartition Vs Coalesce
- Extraction
- Spark Cluster Execution Architecture_Full Architecture
- Spark Cluster Execution Architecture_YARN As Spark Cluster Manager
- Spark Cluster Execution Architecture_JVMs across Clusters
- Spark Cluster Execution Architecture-Commonly Used Terms in Execution Framework
- Spark Cluster Execution Architecture - Narrow and Wide Transformations
- Spark Cluster Execution Architecture - DAG Scheduler
- Spark Cluster Execution Architecture - Task Scheduler
- RDD Persistence
- Spark Shared Variables
- SparkSQL Architecture
- Detailed SparkSession Features
- DataFrame Fundamentals

Structured APIs

- DataFrames
- Spark data types
- Operations : Filters, Column manipulations, Unions, Joins, Aggregations, Sorting
- I/O operations with different types of data sources (TEXT,CSV,JSON,PARQUET,DB)
- Spark SQL
- Datasets
- DataFrame Rows
- DataFrame Columns
- DataFrame ETL
- DataFrame ETL_Introduction to Transformations and Extraction
- DataFrame ETL_DataFrame APIs Introduction Extraction
- DataFrame ETL_DataFrame APIs Selection

- DataFrame ETL_DataFrame APIs Filter or Where
- DataFrame ETL_DataFrame APIs Sorting
- DataFrame ETL_DataFrame APIs Set
- DataFrame ETL_DataFrame APIs Join
- DataFrame ETL_DataFrame APIs Aggregations
- DataFrame ETL_DataFrame APIs GroupBy
- DataFrame ETL_DataFrame APIs Windows
- DataFrame ETL_DataFrame Built-in Functions Introduction
- Performance and Optimization

Productionizing applications:

- How Spark runs on a cluster
- Submitting Spark applications
- Deployment Modes
- Monitoring & Debugging
- Performance Tuning

Running Spark jobs on Cluster

- Spark Runtime Architecture
- Spark Driver
- Executors
- Cluster Managers
- Connecting Spark To Different File System and Perform ETL ,(Extration Transformation and Loading)
- Connecting Spark To DataBases and Perform ETL (Extration Transformation and Loading)
- Spark StorageLevel
- Spark Serializers
- Spark-Submit and Cluster Explanation
- Performance Tuning

PySpark Streaming at Scale

- Introduction to Spark Streaming
- PySpark Streaming with Apache Kafka
- Real-world Practical use cases
- Operations On Streaming
- Window Operations

Advanced Spark Programming

- Spark Shared Variables
- Custom Accumulator
- Spark and Fault Tolerance
- Broadcast variables
- Numeric RDD Operations
- Per-Partition Operations

Processing Big Data with PySpark in Databricks

- Databricks environment and Platform
- ETL, Dataframes and data visualization in Databricks
- PySpark in Databricks with RDDs, Spark Dataframes API or Spark SQL
- Spark Column Expresions and Dataframe Agregations
- Spark Data Sources and Format types
- Spark Architecture Concepts and Query Optimization
- Advanced analytics and data visualization with Databricks
- Machine Learning with Spark at Databricks
- Spark Streaming at Databricks

Resources:

https://www.youtube.com/watch?v=wNRjR6Cds5s&list=PL2IsFZBGM_IHCl9zhRVC1EXTomkEp_1zm

Module 4: Introduction to Big Data

1. What is Big Data?

Big Data refers to extremely large and complex datasets that cannot be stored, processed, or analysed using traditional databases or tools like Excel or a single machine.

These datasets require distributed storage and parallel processing systems.

2. The 5 V's of Big Data

Big Data is best understood through the 5 V characteristics:

✅ 1. Volume

Massive amount of data—terabytes, petabytes, even exabytes.

✅ 2. Velocity

Speed at which data is generated and processed (real-time, streaming).

✅ 3. Variety

Different forms of data:

- Structured (SQL tables)
- Semi-structured (JSON, XML)
- Unstructured (images, audio, logs)

✅ 4. Veracity

Data quality—accuracy, consistency, trustworthiness.

✅ 5. Value

Useful insights or business benefits extracted from data.

3. Sources of Big Data

Big data is generated from various sources:

- Social media (Instagram, Twitter, YouTube)
- Sensors / IoT devices (smart devices, wearables)
- E-commerce transactions
- Mobile apps
- System logs and enterprise apps
- Machine-generated data

4. Why Traditional Systems Fail

Traditional RDBMS systems struggle due to:

- Limited storage on a single machine
- Slow processing for huge datasets
- Inability to handle unstructured data
- Difficulty scaling beyond a certain limit
- This led to the rise of distributed systems like Hadoop and Spark.

5. Big Data Technologies & Ecosystem

🔹 Hadoop Ecosystem

- HDFS – Distributed file system
- MapReduce – Distributed processing framework
- YARN – Resource management
- Hive – SQL on Hadoop
- HBase – NoSQL database

🔹 Apache Spark

- Next-generation big data engine:
- Faster than Hadoop (in-memory processing)
- Supports SQL, ML, Streaming, Graph processing

🔹 Cloud Big Data Services

Azure:

- Azure Data Lake Storage (ADLS)
- Azure Databricks
- Azure Synapse Analytics
- Azure Event Hubs / Stream Analytics

AWS:

- S3
- EMR
- Glue
- Kinesis

GCP:

- BigQuery
- Dataproc
- Pub/Sub

6. Types of Big Data Processing

🔸 **Batch Processing**

- Processing huge volumes of data at scheduled intervals.
- Example: Daily ETL job on Databricks.

🔸 Real-Time / Stream Processing

- Processing continuous data streams.
- Example: Kafka + Spark Streaming.

🔸 Interactive / Ad hoc Processing

- Running on-demand queries.
- Example: SQL queries on BigQuery or Synapse.

7. Career Roles in Big Data

- Data Engineer
- Big Data Engineer
- Machine Learning Engineer
- Cloud Data Architect
- Streaming Engineer (Kafka)

8. Real-World Examples

- Movie recommendations
- real-time pricing
- Fraud detection in banking
- IoT device monitoring

Resources:

https://www.youtube.com/watch?v=H5SHmiKTFsM&t=6020s&pp=ygUSQmlnIERhdGEgcGxheWxpc3Qg

Module 5: Azure Cloud Basics — Portal & Subscription

What is Azure?

- Microsoft's cloud computing platform.
- Offers services across Compute, Storage, Networking, Databases, Analytics, AI, DevOps, and Security.
- Pay-as-you-go pricing model.
- Global network of data centers.

Azure Account vs Tenant vs Subscription

Understanding these three concepts is critical:

✔ Azure Account

- Your individual login identity (Microsoft account/Entra ID user).
- Used to access Azure Portal and manage resources.

✔ Azure AD / Microsoft Entra Tenant

- A dedicated identity and directory service for your organization.
- Stores users, groups, roles, and policies.
- Each tenant can have multiple subscriptions.

✔ Azure Subscription

- Billing boundary + Logical container for Azure resources.
- All resources (VMs, Storage Accounts, Databases, etc.) live inside a subscription.
- Has usage limits, RBAC permissions, cost management, and policies.

Azure Portal Overview

- The Azure Portal is the web-based UI to manage Azure resources.

🔹 Key Features

- Dashboard for quick access to services.
- Create/manage resources (VMs, Storage, Functions, Databricks, etc.).
- Subscription management.
- Access Control (IAM).
- Cost Management + Billing.
- Monitoring and Alerts.

🔹 Common Areas

- Home → Quick shortcuts to major services.
- All Services → Complete list of Azure offerings.
- Resource Groups → Logical grouping of resources.
- Marketplace → Deploy tools, OS images, and services.

Azure Subscription – In Detail

🔸 Why do we need a subscription?

- Provides billing and usage tracking.
- Acts as a security boundary for resource access.
- Supports policies, compliance rules, and quotas.

🔸 Types of Azure Subscriptions

- Subscription Type
- Description
- Free Trial
- $200 credits for 30 days + 12 months free services.
- Pay-As-You-Go
- Most common; pay only for what you use.
- Student Subscription
- Free credits without credit card.
- Enterprise Agreement (EA)
- For large organizations.
- MSDN / Azure for Dev/Test
- For development/testing workloads at reduced cost.

Subscription Management Concepts

✔ Resource Groups (RG)

- Logical container within a subscription.
- Helps organize and manage resources.
- Lifecycle management (delete RG → deletes all resources inside).

✔ Management Groups

- Used when you have multiple subscriptions.
- Apply policies, RBAC, or budgets across subscriptions.

✔ RBAC (Role-Based Access Control)

- Assign permissions at: Management Group → Subscription → RG → Resource.
- Built-in roles: Owner, Contributor, Reader.

✔ Azure Policies

- Enforce compliance.
- Example: restrict VM sizes, enforce tags, block public IPs.

✔ Cost Management

- Set budgets, cost alerts, forecasting.
- Monitor service usage + optimize resources.

How Many Subscriptions Do You Need?

- Best practice setup:
- For Organizations:
- Dev
- Test / QA
- UAT
- Production
- For Individuals:
- Personal Projects → 1
- Certifications → 1 (Free Trial / Student)

Key Terms to Remember

- Term
- Meaning
- Subscription ID
- Unique GUID for subscription identification.
- Tenant ID
- Identity boundary.
- Billing Account
- Tracks costs and invoices.
- Resource Provider
- API namespaces (ex: Microsoft.Storage, Microsoft.Compute).

Azure Portal Navigation for Subscription

- When you open the Azure Portal:
- Search for Subscriptions
- Select your subscription
- You can manage:
- Overview
- Access Control (IAM)
- Cost Analysis
- Resource Providers
- Tags
- Policies
- Security & Compliance

Resources:

https://www.youtube.com/watch?v=10jm7Waan8M&list=PLdpzxOOAlwvIcxgCUyBHVOcWs0Krjx9xR

Module 6: Azure Storage Account: Blob Storage & ADLS Gen2

1. Azure Storage Account

An Azure Storage Account is a fundamental Azure resource that provides highly available, secure, scalable cloud storage for various data types.

Storage Account Services

1. Blob Storage – store unstructured object data
2. File Shares – SMB/NFS file shares
3. Queue Storage – message-based communication
4. Table Storage – NoSQL key-value storage
5. (Optional) ADLS Gen2 features (Hierarchical namespace)

Key Features

- Global availability & durability
- Multiple replication types (LRS, ZRS, GRS, RA-GRS)
- High security (RBAC, SAS, Access keys, Private endpoints)
- Lifecycle management policies
- Hot, Cool, Archive storage tiers

2. Azure Blob Storage

- Azure Blob Storage is object storage for unstructured data.
- Blob Types
- Blob Type
- Description
- Block Blob
- For storing files such as images, documents, backups
- Append Blob
- Optimized for append-only logs
- Page Blob
- Used for Azure VM disks
- Blob Storage Structure

Storage Account
→ Container
→ Blobs (files)

Use Cases

- Application data storage
- Big Data ingestion landing zone
- Backup and restore
- Serving static website content
- Logs, videos, images, binaries

3. ADLS Gen2 (Azure Data Lake Storage Gen2)

- ADLS Gen2 is a data lake solution built on top of Blob Storage, designed for large-scale analytics.
- Key Features
- Hierarchical Namespace (HNS): Folder structure like a file system
- POSIX-style ACLs for precise access control
- Optimized for big data analytics workloads
- Seamless integration with Databricks, Synapse, ADF, HDInsight
- Faster directory operations (move, rename, delete)
- ADLS Gen2 Structure
- Storage Account (StorageV2)

→ Container
→ Folders
→ Subfolders
→ Files

- Use Cases
- Data Lake (Bronze, Silver, Gold layers)
- ETL/ELT pipelines
- Real-time & batch processing
- Machine Learning feature store
- Analytics workloads in Spark/Synapse

4. Blob Storage vs ADLS Gen2 — Comparison

- Feature
- Blob Storage
- ADLS Gen2
- Namespace
- Flat
- Hierarchical
- Folders
- Virtual
- Physical
- Performance
- Standard
- Optimized for analytics
- Access Control
- Containers
- File & folder-level ACLs
- Best For
- General storage
- Big data/analytics
- Rename/Move

- Slow (copy + delete)
- Fast (atomic ops)

5. When to Use?

- Use Blob Storage When:
- Storing documents, images, media, backups
- Simple object storage is enough
- Hosting static websites
- Application-level storage
- Use ADLS Gen2 When:
- Building a data lake
- Working with Spark/Databricks/Synapse
- Need folder-level access control
- Large file ingestion pipelines
- Big Data workloads & analytics
- A Storage Account is the parent container for Azure storage services.
- Blob Storage is unstructured object storage.
- ADLS Gen2 is an upgraded Blob Storage with analytics-optimized features.
- Use Blob for general storage, ADLS Gen2 for big data & analytics.

✔ Quick Summary

- A Storage Account is the parent container for Azure storage services.
- Blob Storage is unstructured object storage.
- ADLS Gen2 is an upgraded Blob Storage with analytics-optimized features.
- Use Blob for general storage, ADLS Gen2 for big data & analytics.

Resources:

https://www.youtube.com/watch?v=_Qlkvd4ZQuo&pp=ygUWYXp1cmUgU3RvcmFnZSBjb25lY3B0cw%3D%3D

Module 7: Azure Kay Vaults

📌 **Module 1: Introduction to Azure Key Vault**
- What is Azure Key Vault?
- Why use Key Vault?
- Introduction to Key Vault Objects

📌 **Module 2: Creating Key Vault**
- Create Key Vault in Portal
- How to access course source code?
- Create Key Vault Using PowerShell
- Useful Resources

📌 **Module 3: Key Vault Secrets**
- Key Vault Secrets
- Key Vault Secrets

📌 **Module 4: Key Vault Keys**
- Key Vault Keys
- Key Vault
- Useful Resources

📌 **Module 5: Key Vault Certificates**
- Key Vault Certificates
- Demo: Key Vault Certificates
- Useful Resources

📌 **Module 6: Using Key Vault with Applications**
- Managing App Secrets with Key Vault
- Useful Resources
- Key Operations with Key Vault
- Useful Resources
- Symmetric Key Wrap/Unwrap
- Useful Resources
- Azure Disk Encryption (ADE) with Key Vault

📌 **Module 7: Access Control & Security**
- Management Plane vs Data Plane
- Management Plane Authorization using RBAC
- Demo: Role Based Access Control with Built-in Roles
- Useful Resources
- Role Based Access Control with Custom Roles

📌 **Module 8: Access Policies & Managed Identity**
- About Access Policy
- Configuring Access Policy
- Managed Identity
- Demo: Assign Managed Identity to Function App
- Network Restrictions

📌 **Module 9: Logging & Monitoring**
- Logging and Monitoring Overview
- Enable Logging for Key Vault
- Logging to Storage Account
- Key Vault Analytics (format without time) –

Resources:

https://www.youtube.com/watch?v=kP7KpfToMkg&pp=ygUYYXp1cmUga2V5IHZhdWx0IHBsYXlsaXN0

Module 8: Azure Data Factory (ADF)

🔹 **1. What is Azure Data Factory?**
✨ Definition
✨ ETL vs ELT
✨ Why ADF? Use cases
✨ Key features
🔹 **2. Azure Data Factory Architecture**
⚡ Control plane
⚡ Data plane
⚡ Integration runtime
⚡ ADF workflow overview
🔹 **3. Azure Data Factory Portal UI**
🖌️ Author tab
📊 Monitor tab
⚙️ Manage tab
🔄 Git vs LIVE mode
🔹 **4. Top-level Concepts**
🟢 Pipelines – Orchestration container
🟢 Activities – Tasks within pipelines
🟢 Linked Services – Connection configuration
🟢 Datasets – Metadata representation of data
🟢 Triggers – Pipeline automation
🟢 Data Flows – Visual ETL transformations
🟢 Integration Runtimes – Compute for data movement & transformation

Pipeline

🔹 1. What is a Pipeline?
🟡 Logical container for activities
🔹 2. Create a New Pipeline
🔹 3. Organize Pipelines into Folders
🔹 4. Debug Pipeline
🔹 5. Publish Pipeline
🔹 6. Parameters / Pipeline Parameters
🟡 Pipeline parameters
🟡 Global & system parameters
🟡 Passing parameters to datasets and activities

Linked Service

🔹 1. What is a Linked Service?
🟠 Connection info for data sources
🔹 2. Create Linked Service For:
🔷 Azure Blob Storage
🔷 Azure SQL Database
🔷 SQL Server (On-prem)
🔷 ADLS Gen1 / Gen2
🔷 SFTP / REST API
🔹 3. Linked Service Parameterization
🔷 Dynamic connections
🔷 Key Vault integration

Datasets

🔹 1. What is a Dataset?
🟣 Representation of data structures
🔹 2. Create a Dataset For:
📂 File Formats in Blob / ADLS: Avro, Binary, CSV, Excel, JSON, ORC, Parquet, XML
📂 Tabular Datasets: SQL Database, SQL Server, Oracle, PostgreSQL, Snowflake
🔹 3. Dataset Parameterization
🔹 Folder path, file names, table names

Activities

🔹 1. Wait Activity
🔹 2. Variables
🔹 Create variable
🔹 Set variable
🔹 Append variable
🔹 3. Copy Data Activity
🔹 Source
🔹 Sink
🔹 Mapping
🔹 Settings
🔹 User properties
🔹 4. Copy Files in Blob
📄 One file
📂 All files from folder
📂 All files & folders recursively
🔹 5. Copy from Blob to SQL / ADLS
⚡ File formats: CSV, TSV, Parquet, Avro, ORC
🔹 6. Databricks Notebook Activity
🔹 7. Azure Function Activity
🔹 8. Lookup / Stored Procedure Activity
🔹 9. Get Metadata / Delete Activity
🔹 10. Execute Pipeline Activity
🔹 11. Validation / Fail Activity
🔹 12. Iteration & Conditionals
🔹 Filter
🔹 ForEach
🔹 If Condition
🔹 Switch
🔹 Until

Triggers

🔹 1. Types
⏰ Schedule Trigger
🧩 Tumbling Window Trigger
📦 Event-based Trigger (Blob events)
🔹 2. Triggers with Parameters
🔹 Pass runtime values into pipelines

Integration Runtime (IR)

- 🔹 1. Azure Auto Resolve Integration Runtime
- 🔹 2. Azure Managed Virtual Network
- 🔹 3. Self-Hosted Integration Runtime
- 🔹 4. Linked Self-Hosted IR
- 🔹 Disaster recovery
- 🔹 Hybrid scenarios

Source Control

- 🔹 1. Git Configuration
- 🔹 Connect ADF to GitHub / Azure DevOps
- 🔹 Branching strategy
- 🔹 2. ARM Templates
- 🔹 Export / Import
- 🔹 3. Azure DevOps Repos
- 🔹 CI/CD pipelines for ADF

Global Parameters

🌐 Environment configuration

🔟 Credentials

🔑 Managed identity
🔑 User-assigned identity
🔑 Key Vault integration

1 Monitoring ADF Jobs

📈 Pipeline monitoring
📊 Activity runs
⚡ Trigger runs

1 Alerts

- 📢 Email / Metric-based alerts

1 Failure Notifications via Logic Apps

- 💡 Event-based failure alerts

1 Data Flows

- 🔹 1. What is Data Flow?
- 🔹 2. Mapping Data Flow
- 🔹 3. Data Flow Debug
- 🔹 4. Transformations
  - 🔹 Filter, Aggregate, Join
  - 🔹 Conditional Split, Derived Column
  - 🔹 Exists, Union, Lookup, Sort
  - 🔹 GroupBy, Pivot, Unpivot, Flatten
  - 🔹 Parse, Stringify
  - 🔹 Alter Row, Assert
  - 🔹 Flowlet
- 🔹 5. Validate Schema / Schema Drift
- 🔹 6. Remove Duplicate Rows

1 Azure DevOps

- 🔹 1. Repos
  - 🔹 Branching & Pull Requests
  - 🔹 YAML pipelines for ADF CI/CD

Resources:

https://www.youtube.com/watch?v=Mc9JAra8WZU&list=PLMWaZteqtEaLTJffbbBzVOv9C0otal1FO

Module 9: Logic Apps

Module 1: Introduction to Azure Logic Apps

- What is Azure Logic Apps?
- Logic Apps vs Azure Functions vs ADF vs Power Automate
- Types of Logic Apps: Consumption, Standard
- Triggers, Actions, Connectors
- Typical data engineering use cases

Module 2: Logic Apps Architecture

- Logic Apps runtime
- Managed connectors vs Built-in connectors
- Workflow Definition Language (WDL)
- Integration Account overview
- Run history, workflow state

Module 3: Core Building Blocks

- Trigger types: Recurrence, HTTP, Event Grid, Event Hub, Service Bus, Blob-based triggers
- Actions:
- Condition
- Switch
- For-each
- Until
- Scope
- Variables (Initialize/Set)
- Compose
- Parse JSON

Module 4: Connectors for Data Engineering

- Azure Blob Storage
- ADLS Gen2
- Azure Data Factory
- Azure SQL / SQL Server
- Synapse Analytics connector
- Cosmos DB connector
- REST API & HTTP connector
- Service Bus / Event Hub connectors

Module 5: Data Engineering Use Cases

- Trigger ADF pipelines on blob upload
- Event-driven orchestration
- Detect duplicate files in storage
- File validation before processing
- Schema validation (JSON/XML)
- Folder creation, file movement
- Email notifications on failures
- Trigger Databricks jobs via REST API
- Execute SQL stored procedures

Module 6: Working with Files

- Read blob metadata
- List files in folder
- Filter arrays (file-level filtering)
- Extract file name, extension, timestamp
- Rename files
- Move/copy files
- Process CSV / JSON / XML
- Liquid templates for transformations

Module 7: Integration with Azure Data Factory

- Trigger ADF pipelines
- Passing parameters
- Capturing ADF responses
- Handling success/failure
- Post-processing actions (email, logging)

Module 8: Integration with Azure Databricks

- Trigger Databricks Jobs API
- Pass notebook parameters
- Wait for job run completion
- Retry and exception patterns

Module 9: Event-Driven Architecture

- Event Grid + Logic Apps
- Event Hub + Logic Apps
- Service Bus triggers
- Real-time alerting patterns
- Serverless event workflows

Module 10: Advanced Workflow Capabilities

- Error handling with Scopes
- Configure Run After
- Terminate action
- Retry policies
- Concurrency control
- Workflow chaining (child workflows)
- Custom connectors

Module 11: Monitoring & Logging

- Run history inspection
- Integration with Log Analytics
- Diagnostic settings
- Alerts and action groups
- End-to-end tracking

Module 12: Security & Compliance

- Managed identities
- Role assignments
- Key Vault integration
- Access policies for Storage, SQL, ADF
- Private endpoints and VNet
- IP restrictions

Module 13: Cost Optimization

- Consumption vs Standard pricing
- Use of built-in connectors to reduce cost
- Reducing unnecessary triggers
- Workflow optimization for fewer actions

Module 14: Real Projects for Data Engineers

- Event Grid trigger → Validate file → Trigger ADF → Notify
- File monitoring Logic App with duplicate detection
- Databricks notebook automation workflow
- Logic App for SQL-based metadata logging
- Service Bus → Logic Apps → Cosmos DB ingestion
- Automated pipeline failure alerting via Logic Apps
- Logic App for schema & format validation before ETL

Resources:

https://www.youtube.com/watch?v=Vn-ACdYe36Q&list=PLm0VCJgNxnpf3hdvhAqTtk9BF8p2K6oxF&pp=0gcJCbAEOCosWNin

Module 10: Azure Databricks

1) Introduction to Big Data

Data Concepts

- What is Data?
- What is a Database?

Big Data Concepts

- What is Big Data?
- Challenges of Big Data
- Why Traditional Databases Can't Handle Big Data

2) Introduction to Hadoop

- What is Hadoop?
- How Hadoop Overcomes Big Data Challenges

Hadoop Architecture

- Hadoop Daemons
- HDFS
- YARN
- MapReduce

3) Introduction to Spark

- Spark Architecture
- Spark Internals
- Spark RDD
- Spark DataFrame
- Spark Streaming

4) Introduction to Databricks

- What is Databricks?
- Databricks Architecture
- Working in Databricks Workspace
- Working with Databricks Notebook

5) Working with Databricks File System (DBFS)

- What is DBFS?

DBFS Commands: mkdirs, cp, mv, head, put, rm, rmdir

- Handling Multiple Files in DBFS
- Processing Files in DBFS
- Archiving Files in DBFS

6) Databricks Utilities

- Credentials Utility
- FileSystem Utility
- Notebook Utility
- Secrets Utility
- Widgets Utility

7) Databricks Cluster Management

- Creating & Configuring Clusters
- Managing Clusters
- Displaying Clusters
- Starting a Cluster
- Terminating a Cluster
- Deleting a Cluster
- Cluster Information & Logs

Types of Clusters:

- All-Purpose Clusters
- Job Clusters

Cluster Modes:

- Standard
- High Concurrency
- Autoscaling
- Databricks Runtime Versions

8) Databricks – Batch Processing Integrations:

- Blob Storage
- Azure Data Lake Storage Gen2
- Azure SQL Database
- Synapse
- Azure Key Vault

9) Databricks – Streaming API

- What is Streaming?
- Processing Streaming Using PySpark API
- Handling Bad Records
- Streaming Data into Gen2 Lake
- Loading Data into Tables

10) Databricks – Lakehouse (Delta Lake)

- Difference Between Data Lake & Delta Lake
- Introduction to Delta Lake
- Features of Delta Lake
- Creating Delta Tables
- DML Operations in Delta Tables
- Merge Statements
- Handling SCD Type 1 & Type 2
- Handling Data Deduplication
- Handling Streaming Data in Delta Lake

11) Delta Lake: Medallion Architecture

- Bronze Layer – Raw Data
- Silver Layer – Cleansed & Transformed Data
- Gold Layer – Curated, Business-Ready Data

12) Workflows in Databricks

- Introduction to Workflows
- Creating, Running, & Managing Databricks Jobs
- Scheduling Databricks Jobs
- Monitoring Databricks Jobs

13) Azure DevOps – Repos

- What are DevOps Repos
- Integrating Databricks Notebooks with Repos
- Commit & Sync Notebooks to/from Repos

Resources:

https://www.youtube.com/watch?v=XOSuR8g2SfQ&list=PL2IsFZBGM_IGiAvVZWAEKX8gg1ItnxEEb&pp=0gcJCbAEOCosWNin

Module 11: Azure Snowflake:

1. Introduction to Snowflake
- What is Snowflake?
- Key features: Multi-cloud, Elasticity, Scalability
- Snowflake Editions overview
- Use cases in data engineering

2. Snowflake Architecture
- Shared Data Architecture
- Storage Layer, Compute Layer (Virtual Warehouses), Cloud Services Layer
- Cloning, Zero-Copy Cloning, Data Sharing

3. Snowflake Setup & Administration
- Snowflake account setup
- Interfaces: Web UI, SnowSQL, APIs
- Role-Based Access Control (RBAC)
- Resource monitoring & management
- Warehousing best practices: scaling, concurrency, auto-suspend

4. Data Loading & Unloading
- Supported file formats: CSV, JSON, Avro, Parquet, ORC
- COPY command for bulk loading
- External stages & cloud storage integration (AWS S3, Azure Blob)
- Snowflake Streams & File Formats
- Data unloading

5. Data Modelling & Schema Design
- Schema types: Star, Snowflake, Data Vault
- Choosing schema for data engineering needs
- Optimal table structures
- Time Travel & Data Retention

6. Snowflake SQL & Query Optimization
- Core SQL: DDL, DML, TCL
- Semi-structured data handling (JSON, XML)
- Query performance tuning & optimization
- Query Profile analysis
- Materialized Views & Caching

7. Data Transformation & Pipelines
- Snowflake Streams & Tasks overview
- Creating & managing pipelines
- Snowflake Python Connector & Snowpark
- Integration with ETL/ELT tools (Apache Airflow)
- Orchestration with Azure Data Factory

8. Data Governance & Security
- Data encryption: at-rest & in-transit
- Role hierarchy & access control
- Masking policies for sensitive data
- Data Sharing & Collaboration
- Auditing & logging: ACCESS_HISTORY, QUERY_HISTORY

9. DevOps in Snowflake
- Version control: Git for DDL & scripts
- CI/CD pipelines: Azure DevOps, GitHub Actions, Jenkins
- Automated deployment: Dev → QA → Prod
- Infrastructure-as-Code (IaC) for Snowflake
- Automated testing: SQL unit tests, data validation
- Monitoring & alerting for pipeline failures
- Terraform or other DevOps tool integration

10. Advanced Features & Integrations
- Time Travel & Fail-safe
- External tables & External Functions
- Data replication & disaster recovery
- BI tool integration: Power BI

11. Real-World Projects & Use Cases
- End-to-end ETL pipeline with Snowflake + DevOps
- Optimizing data warehouse performance
- Real-time data streaming pipelines
- Automated deployments, monitoring & version control

Resources:

https://www.youtube.com/watch?v=AR88dZG-hwo&list=PLba2xJ7yxHB7SWc4Sm-Sp3uGN74ulI4pS

Module 12: Apache Airflow

🚀 Airflow Introduction
- Apache Airflow overview
- Comparison with other workflow tools
- Course prerequisites
- Installing Conda (virtual environment)

Components of Airflow
- Install Airflow (MacOS, Linux, Windows, Docker)
- Run Airflow locally
- Airflow UI & CLI introduction

Core Concepts
- DAGs and default arguments
- Tasks and operators
- Defining dependencies

Advanced Concepts
- Use cases & setup
- Connections & variables
- Load data to BigQuery, run SQL queries
- Hooks, XComs, Jinja templating, macros

Advanced DAG Operations
- Create Dataproc Hadoop cluster
- Branching & submit PySpark jobs
- SubDAGs & trigger rules
- DAG documentation

Custom Plugins
- Create custom operators & sensors
- Run custom plugins

Testing
- Load test DAGs
- Unit test DAGs, operators, and custom operators

Airflow in Production
- Executors: Local & Celery
- Service Level Agreements (SLAs)
- Security: Authentication, roles, encryption
- Remote logging
- Monitoring: StatsD, Prometheus, Grafana
- Error tracking: Sentry
- Managed Airflow services

Resources:

https://www.youtube.com/watch?v=K9AnJ9_ZAXE&list=PLwFJcsJ61oujAqYpMp1kdUBcPG0sE0QMT

Module 13: Apache Kafka

Kafka Fundamentals

- Kafka Fundamentals
- Topics, Partitions and Offsets
- Producers and Message Keys
- Consumers & Deserialization
- Consumer Groups & Consumer Offsets
- Brokers and Topics
- Topic Replication
- Producer Acknowledgements & Topic Durability
- Zookeeper
- Kafka KRaft - Removing Zookeeper

Kafka Setup

- Important: Starting Kafka & Lectures Order
- FAQ for Setup Problems
- Starting Kafka with Conduktor - Multi Platform
- Mac OS X
- Download and Setup Kafka in PATH
- Start Kafka in KRaft mode
- Using brew
- Start Zookeeper & Kafka
- Linux
- Download and Setup Kafka in PATH
- Start Kafka in KRaft mode
- Start Zookeeper & Kafka
- Windows (WSL2 & Non-WSL2)
- WSL2 Setup
- Download Kafka and PATH Setup
- Start Kafka in KRaft mode
- How to Fix Problems
- Extra Instructions
- Start Zookeeper & Kafka

Kafka CLI

- CLI Introduction
- WINDOWS WARNING: PLEASE READ
- What to include for Bootstrap Servers?
- Kafka Topics CLI
- Kafka Console Producer CLI
- Kafka Console Consumer CLI
- Kafka Consumers in Group
- Kafka Consumer Groups CLI
- Resetting Offsets

Kafka with Conduktor

- Conduktor - Demo

Kafka Programming in Java

- Kafka SDK List
- Creating Kafka Project
- Java Producer
- Java Producer Callbacks
- Java Producer with Keys
- Java Consumer
- Java Consumer - Graceful Shutdown
- Java Consumer inside Consumer Group
- Java Consumer Incremental Cooperative Rebalance & Static Group Membership
- Java Consumer Incremental Cooperative Rebalance - Practice
- Java Consumer Auto Offset Commit Behavior
- Programming - Advanced Tutorials

Producer Advanced Concepts

- IMPORTANT: Start Local Kafka with Conduktor using Docker
- Wikimedia Producer Project Setup
- Wikimedia Producer Implementation
- Wikimedia Producer Run
- Wikimedia Producer - Producer Config Intros
- Producer Acknowledgements Deep Dive
- Producer Retries
- Idempotent Producer
- Safe Kafka Producer Settings
- Wikimedia Producer Safe Producer Implementation
- Kafka Message Compression
- linger.ms and batch.size Producer settings
- Wikimedia Producer High Throughput Implementation
- Producer Default Partitioner & Sticky Partitioner
- [Advanced] max.block.ms and buffer.memory

OpenSearch Consumer Project

- OpenSearch Consumer - Project Overview
- OpenSearch Consumer - Project Setup
- Setting up OpenSearch on Docker
- Setting up OpenSearch on the Cloud
- OpenSearch 101
- OpenSearch Consumer Implementation - Part 1
- OpenSearch Consumer Implementation - Part 2
- Consumer Delivery Semantics
- OpenSearch Consumer Implementation Part 3 - Idempotence
- Consumer Offsets Commit Strategies
- OpenSearch Consumer Implementation Part 4 - Delivery Semantics
- OpenSearch Consumer Implementation Part 5 - Batching Data
- Consumer Offset Reset Behavior
- OpenSearch Consumer Implementation Part 6 - Replaying Data
- Consumer Internal Threads
- Consumer Replica Fetching - Rack Awareness

Kafka Extended APIs

- Kafka Extended APIs - Overview
- Kafka Connect Introduction
- Kafka Connect Hands On: Warning
- Kafka Connect Wikimedia & ElasticSearch Hands On
- Kafka Streams Introduction
- Kafka Streams Hands-On
- Kafka Schema Registry Introduction
- Kafka Schema Registry Hands On
- Which Kafka API should I use?

Kafka Architecture & Best Practices

- Choosing Partition Count & Replication Factor
- Kafka Topics Naming Convention
- Case Studies
- MovieFlix
- GetTaxi
- MySocialMedia
- MyBank
- Big Data Ingestion
- Logging and Metrics Aggregation
- Operations & Security
- Kafka Cluster Setup High Level Architecture Overview
- Kafka Monitoring & Operations
- Kafka Security
- Kafka Multi Cluster & MirrorMaker
- Advertised Listeners: Kafka Client & Server Communication Protocol

Advanced Kafka

- Changing a Topic Configuration
- Segment and Indexes
- Log Cleanup Policies
- Log Cleanup Delete
- Log Compaction Theory
- Log Compaction Practice
- Unclean Leader Election
- Large Messages in Kafka

Resources:

https://www.youtube.com/watch?v=xGwzuz8F9k0&list=PLVz2XdJiJQxwpWGoNokohsSW2CysI6lDc

Module 14: Azure Functions

💡 Introduction

- ⚡ Serverless compute service that lets you run code without managing servers
- 📌 Automatically scales based on event triggers
- 💰 Pay only for execution time, not for idle resources

🧱 Core Concepts

- Function App → Container for one or more functions
- Function → Your code + trigger + bindings
- Triggers → Event that starts execution
- Bindings → Input/output integrations (no boilerplate code)
- Runtime Stack → Python, C#, Java, Node.js, PowerShell, etc.

⏳ Triggers

- ⏰ Timer Trigger (CRON-based scheduling)
- 📨 HTTP Trigger (REST API)
- 🔁 Queue Trigger (Azure Storage Queue)
- 📦 Blob Trigger (Blob Storage file arrival)
- 📊 Event Hub Trigger (real-time streams)
- 💬 Service Bus Trigger (queues + topics)

🔗 Bindings

- 📥 Input bindings: Blob, Queue, Table, SQL, Cosmos DB
- 📤 Output bindings: Blob, Queue, Event Hub, Service Bus, Email
- 🔄 Reduces boilerplate; no need to write connection code

Hosting Plans

- Consumption Plan
  - Auto-scale
  - Pay for execution time
  - Best for event-driven workloads
- Premium Plan
  - Always warm instances
  - VNET integration
  - High performance
- Dedicated App Service Plan
  - Fixed VM cost
  - Good for long-running or steady workloads

🔍 Observability & Monitoring

- 📊 Azure Monitor + Application Insights
- 🔎 Live Metrics, Logs, Traces
- 🛑 Failure & retry tracking (Queue, Service Bus)

🛠️ Use Cases

- 🗂️ Event-driven ETL pipelines
- 📨 Serverless APIs
- 🔄 Real-time file processing
- 🔁 Scheduled jobs (CRON, cleanup, alerts)
- 🔀 Stream processing with Event Hubs
- 🤝 Custom connectors for Logic Apps / ADF

🔐 Security

- 🔐 Managed Identity for secure resource access
- 🔑 Key Vault integration
- 🛡️ Access restrictions + VNET integration (Premium)
- 🔀 OAuth2 / AAD for HTTP functions

🚀 Deployment

- GitHub Actions
- Azure DevOps Pipelines
- Zip Deploy / VS Code Deploy
- Container-based deployment

Module 15: Azure Purview

1. Introduction to Microsoft Purview
- What is Purview?
- Role in data governance & cataloging

2. Purview Core Components
- Data Map
- Data Catalog
- Governance Portal
- Collections

3. Metadata Scanning & Classification
- Registering data sources
- Scanning (Azure, on-prem, multi-cloud)
- Automatic & custom classifications

4. Data Lineage
- End-to-end lineage tracking
- ADF, Synapse, Databricks, Power BI lineage
- Impact analysis

5. Business Glossary & Metadata Management
- Creating glossary terms
- Assigning terms to assets
- Custom metadata

6. Access Control & Governance Policies
- RBAC roles in Purview
- Data access policies
- Integration with MIP sensitivity labels

7. Architecture & Integration Runtimes
- Purview account and collections
- Network & security (private endpoints)
- Azure IR vs Self-hosted IR

8. Integrations
- Azure Data Factory
- Synapse Analytics
- Databricks
- Power BI
- Snowflake, AWS S3 (optional)

9. Monitoring & Reports
- Scan insights
- Classification insights
- Lineage insights

10. Real-World Use Cases
- Data discovery
- Compliance & PII detection
- Centralized data catalog
- Enterprise data governance

Resources:

https://www.youtube.com/watch?v=WUYNBFDRyFU&list=PLU1w_BFZFd2r_hvhw-c3Q6KsohNc8UeiT

Module 16: Azure Log Analytics

⭐ Introduction to Log Analytics
- What is Log Analytics
- Firewall requirements
- Create a Log Analytics workspace in Azure Portal
- Explore workspace overview
- Connect data sources to the workspace

⭐ VM Connectivity & Agents
- Connect VMs using VM extensions
- Connect Windows VM using agent
- Configure Windows agent to send data to multiple workspaces
- Connect Linux VM using agent
- Verify that VMs are connected and sending data
- Supported OS for Windows agents
- Supported OS for Linux agents

⭐ Environment & First Steps
- Free demo environment
- Run your first query
- View collected data
- Collect Performance and Event logs
- Query window settings
- Save queries
- Load saved queries
- Export query results

⭐ KQL Basics (Kusto Query Language)
- Basic queries
- where clause
- Logical operators
- equals and notequals
- has operator
- contains
- startswith and endswith
- in and has_any
- TimeGenerated
- summarize operator
- distinct and count
- sort and top
- Column selection: project, project-away
- extend operator (plus iff, round)
- Computer groups
- Setting query scope
- String comparison operators
- Arithmetic operators
- More practice
- Other useful KQL operators

⭐ Data Collection Integrations
- Connect Storage Account
- Collect Syslog data

⭐ Monitoring Solutions
- What are monitoring solutions
- List available solutions
- Install solutions
- Review “Agent Health” solution
- Delete a solution
- Install solutions (alternative method)
- Solution targeting
- Practice section

⭐ Service Map
- Introduction to Service Map
- What is Service Map
- Create Service Map solution
- Install Dependency Agent
- Access Service Map solution
- Service Map UI overview
- Set time range
- Status badges & color codes
- Machine groups
- Process filter
- Find failed connections
- Change focus machine
- Self links
- Machine summary
- Process summary
- Integration with other solutions
- Course conclusion

⭐ Change Tracking & Inventory

- Enable Change Tracking (Part 1 & Part 2)
- Explore Change Tracking solution
- Change Tracking options
- Track Linux file changes
- Test Linux file monitoring
- Inventory
- Manage machines onboarded to solutions

⭐ Alerts

- Create alerts
- Manage alerts
- Challenge exercise

Resources:

https://www.youtube.com/watch?v=2tSLfd02VLc&list=PL8wOlV8Hv3o9OwWe0QWji1KjVQtEyk_dL

Module 17: Microsoft Fabric

1) Introduction to Big Data
- What is Data?
- What is a Database?
- What is Big Data?
- Challenges of Big Data
- Why Traditional Databases Cannot Handle Big Data?

2) Introduction to Microsoft Fabric
- What is Microsoft Fabric?
- How to Enable Fabric in Your Organization
- Fabric Workspace Structure
- Big Data Analytics with Microsoft Fabric
- Microsoft Fabric – A Unified Platform
- Advantages of Microsoft Fabric

3) Introduction to Fabric Components
- OneLake
- Real-time Intelligence
- Data Factory
- Fabric Data Science
- Fabric Data Engineering
- Data Activator
- Fabric Data Warehouse
- Power BI

4) Fabric Account Creation
- How to Sign Up for Microsoft Fabric
- How to Activate a Microsoft Fabric Trial
- How to Create and Manage Workspaces
- Workspace Configurations & Settings
- Managing Capacity Units

5) OneLake Concepts
- Unified Storage Layer
- Logical Data Organization
- Delta Tables & Open Data Formats
- Shortcuts (Virtual Data Access)
- Integration with Fabric Workloads
- Security & Access Control
- Performance & Cost Optimization

6) Data Factory in Microsoft Fabric
- Unified Data Integration Platform
- Pipeline Activities & Dataflows
- Connectivity & Data Ingestion
- Integration with OneLake & Other Fabric Services
- Data Transformation & Orchestration
- Debugging & Monitoring Pipelines

7) Fabric Data Engineering – Lakehouse
- Difference: Data Lake vs Data Lakehouse
- Understanding the Fabric Lakehouse
- Lakehouse Architecture
- Creating a Lakehouse
- Lakehouse Features & Interface Overview
- Uploading Files into the Lakehouse
- DevOps Repos & Deployment Pipelines

8) Fabric Data Engineering – Notebooks
- Creating & Using Notebooks
- Developing & Running Notebooks
- Using Python in Notebooks
- Notebook Utilities
- Notebook Visualizations
- Exploring Data in the Lakehouse
- Loading Data into the Lakehouse
- Notebook Source Control & Deployment
- Authoring & Running T-SQL in Notebooks

9) Fabric Data Engineering – Apache Spark
- Spark Architecture (Deep-dive)
- Working with Data in Spark DataFrames
- Internal Execution of DataFrames
- Spark SQL for Data Processing
- Transformations via DataFrame API
- Actions via DataFrame API
- User-Defined Functions (UDFs)
- Visualizing Data in Spark Notebooks
- Hands-On Spark Data Analysis

10) Fabric Data Engineering – Delta Lake Tables

- Introduction to Delta Lake
- Understanding Delta Lake
- Creating Delta Tables
- Working with Delta Tables in Spark
- Using Delta Tables for Streaming Data
- Slowly Changing Dimensions (SCD Type 1 & 2)
- Time Travel & Versioning
- Medallion Architecture Implementation
- Understanding V-Order in Fabric

11) Fabric Data Warehouse

- Introduction to Data Warehousing
- Data Warehouse Fundamentals
- Data Warehousing in Microsoft Fabric
- Querying & Transforming Data
- Using SQL Endpoints
- Preparing Data for Reporting
- Securing & Monitoring Data Warehouses
- Copying Data from Lakehouse to Warehouse

12) Fabric Real-time Intelligence

- Getting & Processing Data in Event Streams
- Ingesting Data into KQL Databases
- Storing Data in Eventhouse & KQL Databases
- Understanding KQL Tables
- Writing Queries in Kusto Query Language (KQL)

13) Activators in Fabric

- What are Activators?
- Tracking Data Using Activators
- Creating Activator Objects & Rules
- Using Activators with Event Stream

14) Power BI in Fabric

- Connecting Fabric with Power BI
- Creating Dashboards & Reports
- Power BI Desktop vs Cloud
- Integrating Power BI with Fabric
- OneLake Connections
- Power BI Desktop with Fabric
- Power BI Desktop with OneLake
- Power BI Desktop with Lakehouse
- Power BI Desktop with Synapse
- Power BI Cloud with OneLake

Resources:

https://www.youtube.com/watch?v=q2LlvqQ-QH8&list=PLWf6TEjiiuIAXoxSa2ZZCCbf4XEUXPC3M&pp=0gcJCbAEOCosWNin

Module 18: Cosmos-DB

MODULE 1: Introduction to Azure Cosmos DB

- Azure Cosmos DB is a NoSQL, fully managed, globally distributed database.
- Offers millisecond latency, 99.999% availability, and automatic scaling.
- Ideal for high-velocity operational workloads, IoT, real-time apps, and microservices.

MODULE 2: Cosmos DB Architecture

- Globally distributed architecture with multi-region replication.
- Supports multi-master writes and automatic failover.
- Five consistency levels: Strong → Eventual.
- Secure using VNETs, RBAC, and managed identities.

MODULE 3: Data Models & APIs

- Supports multiple APIs:
  - NoSQL (Core SQL API) – most used
  - MongoDB
  - Cassandra
  - Gremlin (Graph)
  - Table API
  - PostgreSQL Distributed API
- Uses containers (tables) and items (JSON documents).

MODULE 4: Partitioning

- Critical for performance & scalability.
- Partition Key determines how data is distributed.
- Good keys have high cardinality and uniform access.
- Avoid hot partitions by selecting the right key.
- Physical vs Logical partitions.

MODULE 5: Request Units (RUs)

- RU = cost of CPU + memory + IO per operation.
- Reads cost fewer RUs; queries & writes cost more.
- Provisioning modes:
  - Manual throughput
  - Auto scale
  - Serverless
- Optimize by using point reads, partitioning well, and indexing correctly.

MODULE 6: Indexing

- Cosmos DB indexes all data by default.
- You can customize indexing:
- Included/Excluded paths
- Composite indexes
- Spatial indexes
- Proper indexing reduces RU consumption and improves query performance.

MODULE 7: Querying Cosmos DB

- Queries use SQL-like syntax for JSON documents.
- Supports JOIN, ORDER BY, array functions, user-defined functions.
- Uses continuation tokens for pagination.
- Best practice: Query using partition key for low RU cost.

MODULE 8: Change Feed

- Logs inserts & updates in order.
- Enables real-time data processing.
- Can be consumed by:
- Azure Functions
- Event Hubs
- ADF
- Stream Analytics
- Best for event-driven architectures, CDC pipelines, real-time analytics.

MODULE 9: Integration with Azure Services

- ADF: Copy Activity to/from Cosmos, incremental loads using _ts.
- Synapse Link: Query analytical store without ETL.
- Event Grid / Functions: Event-driven automation.
- Stream Analytics: Real-time ingestion.

MODULE 10: Performance & Cost Optimization

- Optimize RUs by:
- Selecting correct partition key
- Using Auto scale
- Avoiding cross-partition queries
- Reducing indexing paths
- Use TTL to reduce storage cost.
- Avoid large documents; prefer embedding or referencing correctly.

MODULE 11: Monitoring & Troubleshooting

- Tools: Azure Monitor, Diagnostic logs, Metrics.
- Key metrics:
- RU consumption
- Throttling (429 errors)
- Data usage
- Latency
- Common issues: high RU usage, hot partitions, indexing misconfigurations.

MODULE 12: Real-World Use Cases

- IoT telemetry ingestion.
- E-commerce product catalogs.
- User profile stores.
- Real-time order tracking.
- Session management.
- Fraud detection.
- Real-time analytics using Synapse Link.

MODULE 13: Hands-On Labs

- Create Cosmos DB account + containers.
- Insert/read/query JSON documents.
- Configure partitioning & indexing.
- Build ADF pipelines with Cosmos DB.
- Implement Change Feed with Azure Functions.
- Enable Synapse Link & run Spark queries.
- Monitor RU consumption & optimize.

Resources:

https://www.youtube.com/watch?v=FimrsNEJ83c&list=PLmamF3YkHLoIg_l-dZo1yD26YE3LpkxMp

Module 19: Power BI(Optional)

1) Introduction
- Installation
- Interface of Power BI
- Introduction to our Source Data

2) Introduction to SQL
- Data Definition Language
- Data Manipulation Language
- String Functions
- Numeric Functions
- Date Functions
- Conversion Functions
- Joins
- ER Modeling
- Normalization
- SQL Select Variants
- Sorting
- Conditional Functions
- Sub Queries
- Set Operators
- Manipulating large Data Sets

3) Different Data Sources in Power BI
- Type of Connectors
- Loading Data in Power BI
- Ways to Create Visual
- Understanding Fields Box
- Understanding Aggregation
- Power BI Architecture

4) Drilling into your data
- Drill up and Down your data
- Advanced drill down option

5) Formatting visual

6) Structuring your data
- Wide Table Vs Tall Table
- Transforming your Data Structure (Unpivot Data)
- Fill & Replace Data
- Splitting and formatting your data
- Date & Time Transformations

7) Power BI Dashboard
- Introduction to Dashboard Data
- Loading Data & Knowing about Relationships
- Ways to combine your data
- Cardinality
- Absence of Common Columns
- Adding Tables & Matrix
- Adding Bar & Column Charts
- Adding Pie & Donut Charts
- Adding Line Charts
- Adding Tree Maps
- Adding Area Charts
- Adding Combo Charts
- Adding Bubble Maps
- Adding Filled Maps
- Adding Cards and Multi Row Cards
- Inserting Elements

8) Filters
- Adding Slicers
- Understanding the Filters Pane
- Drill through Filters
- Month's vs Calendar Months

9) Making your visuals Interactive
- Using Bookmarks and Buttons to present insights
- Refresh data and data source settings
- Assignment instructions and introduction to dataset
- Building Dashboard

10) Parameters and Functions

11) Data Analysis Expressions (DAX)

12) Introduction to DAX
- Importance of DAX
- Data Types in DAX
- Steps to Create Calculated Columns
- Measures in DAX
- DAX Syntax DAX Functions
- Operations DAX Tables and Filtering

Resources:

https://www.youtube.com/watch?v=h9hgkh-iraI&list=PLjNd3r1KLjQt0xN_y8F6BSIOVNvdQmq4d

Module 20: DBT

1) Introduction

What is dbt and Why should you use it?
- How can you take the most out of this course?
- ESSENTIAL READ ME - Course Resources
- Use-case and Input Data Model Overview
- Snowflake Registration
- Snowflake's Authentication types and the Key-Pair Authentication
- Automatic Snowflake Data Import
- ONLY FOR REFERENCE - Snowflake Behind the Scenes
- dbt Roles, Users and Database Tables overview
- Setting Up Course Folder and Virtualenv
- Dec 2 README: Note on a bug in dbt-snowflake v1.10
- dbt Installation
- dbt Project Setup
- Say Hello to our dbt Project Folder
- Datasets and Data Flow Overview

Models Section

- Learning Objectives - Models
- Models Overview
- Theory: CTE - Common Table Expressions
- Creating our first model: Airbnb listings
- Models Quiz
- Create the src_hosts model

Materializations

- Learning Objectives - Materializations
- Materializations Overview
- Model Dependencies and dbt's ref tag
- Table type materialization & Project-level Materialization config
- Incremental materialization
- Ephemeral materialization

Seeds & Sources

- Learning Objectives - Seeds and Sources
- Seeds and Sources Overview
- Seeds
- Sources
- Source Freshness Checks

Snapshots

- Learning Objectives - Snapshots
- Snapshots Overview
- Create a Snapshot

Tests

- Learning objectives - Tests
- Tests Overview
- Generic Tests
- Debugging dbt Tests
- Saving Test Failures to the Data Warehouse
- Saving Test Failures into a Custom DB Schema
- Singular Tests
- Unit Tests
- Create your own singular test
- Tests Quiz
- Data Contracts
- Custom Generic Tests
- Custom Tests with Parameters
- Setting the Tests' Severity: Warning vs Error

Jinja & Macros

- Jinja Basics
- Let's take Jinja for a Drive
- dbt-Specific Jinja Features
- Create Your Own Macros
- Advanced Jinja
- Advanced Macros in Action
- Assignment: Macros Output Formatting
- Installing Third-Party Package

Documentation

- Learning Objectives - Documentation
- Documentation Overview
- Writing and Exploring Basic Documentation
- Markdown-based Docs, Custom Overview Page and Assets
- The Lineage Graph (Data Flow DAG)
- Document the dim_hosts_cleansed table

Analyses, Hooks & Exposures

• Learning Objectives - Analyses, Hook and Exposures
• Analyses
• Hooks
• Grants - Managing Permissions in dbt the Modern Way
• Nov 2025 - Snowflake Authentication Update
• USERNAME/PASSWORD VERSION - Setting up a BI Dashboard in Snowflake and Preset
• KEYPAIR - Say Hi to our BI Tool - preset.io
• KEYPAIR - Set up a BI Dashboard with our Final Tables
• READ ME - Exposures naming convention changes in recent dbt releases
• Exposures

dbt Fusion

• dbt Fusion - Introduction
• Exploring the Technical Differences Between dbt Fusion and dbt Core
• dbt Fusion Feature Matrix
• Licence Changes
• dbt Fusion and the VSCode Extension Installation and Walkthrough
• dbt Fusion Installation for Production Systems
• Running dbt Core and dbt Fusion side by side
• Upgrade your Projects to Fusion and dbt 1.10+ using dbt-autofix
• Building our Project with Fusion
• Official VSCode Extension - Development Workflow & Execution
• Official VSCode Extension - Navigation & Code Intelligence Features
• Official VSCode Extension - Project and Column-Level Lineage
• Upgrading Fusion with the dbtf System Utility

Hero Section

• Welcome to Hero
• Have your say in the course's roadmap
• A note on the dbt-expectations setup
• Annoucement for dbt 1.10+ users

Great Expectations

• Great Expectations Overview
• Comparing row counts between models
• Looking for outliers in your data
• Implementing test warnings for extremal items
• Validating column types
• Monitoring categorical variables in the source data
• Debugging dbt tests and Working with regular expressions
• dbt-expectations and test debugging quiz

Variables, Logging & Date Ranges

• Course Feedback and Moving on
• Logging to the dbt Log File
• Logging to the Screen
• Disabling Log Messages
• A short knowledge check
• Working with Jinja Variables
• A note for Windows users using cmd
• Working with dbt Variables
• Setting Default Values
• Using Date Ranges to Make Incremental Models Production-Ready

Orchestration & Dagster

• Overview of the Popular dbt Orchestration Tools and how to Choose the Right Tool
• Dagster Installation
• Creating a Dagster project and connecting it to our dbt Project
• Dagster dbt files and UI Overview
• Manage, Orchestrate and Debug your dbt Project with Dagster
• A Note on the Advanced Dagster Section
• Advanced Dagster: Using Partitions with Incremental Models

VSCode Power User (dbt Core Extension)

• Free VSCode Extension - Power User for dbt Core
• Introduction to the Power User for dbt Core VSCode Extension
• Install and Configure Power User for dbt Core
• dbt Power User - Working with Models, Autocomplete and Query Results
• dbt Power User - Lineage and Documentation
• How to Get an API Key for the Advanced Features
• Use AI to Generate Documentation
• Generate dbt Model from Source Definition or SQL
• Working with Column-Level Lineage
• Generate and edit dbt Tests
• Find Problems in your dbt Project with Health Check
• Use AI to Interpret Queries via Query Explanations
• dbt Project Governance
• Query Translation (SQL dialects)

Resources:

https://www.youtube.com/watch?v=zZVQluYDwYY&t=3719s&pp=ygURc25vd2ZsYWtlIHByb2plY3Q%3D

Module 21: CI/CD & Azure DevOps

What is CI/CD
- Continuous Integration explained
- Continuous Deployment/Delivery explained
- Importance of CI/CD in DevOps
- Benefits of automated delivery pipelines

Prerequisites
- Required Azure & DevOps knowledge
- Tools needed before starting
- Basic understanding of YAML

New YAML Experience
- Overview of Azure DevOps YAML editing
- Features of new YAML pipeline editor
- Advantages of YAML-first DevOps

Azure DevOps Architecture
- Azure DevOps core components
- Repos, Boards, Artifacts, Pipelines
- Hosted agent's vs self-hosted agents
- Agent pools & pipeline execution

Build Minutes Availability
- Microsoft-hosted agents build runtime limits
- Usage considerations

Stages, Jobs, and Steps
- Stage structure in YAML
- Jobs and how they run in parallel or sequentially
- Steps and task-level details

Demo: Stages, Jobs, and Steps
- Implementation of a clear YAML pipeline
- How to structure stages in real-world pipelines

Adding Dependencies Between Stages and Jobs
- Sequential vs parallel stages
- Using depends On
- Conditional execution patterns

Demo: Adding Dependencies
- Linking multi-stage pipelines
- Real example of stage-to-stage flow

Azure DevOps Pipelines Dependencies
- Understanding pipeline dependency flow
- Managing build → deploy workflows
- Handling multi-repo dependency scenarios

Introduction to Container Apps Task
- Azure Container Apps overview
- Using tasks for deployment
- CI/CD flow for Container Apps

Setting Up the Demo Environment
- Required Azure resources
- Preparing project structure
- Configuring pipelines

YAML: Creating the CI/CD Pipeline
- Writing YAML from scratch
- Adding triggers, variables, and agents
- Build → deploy structure

Demo: Running the Pipeline
- Executing the YAML pipeline
- Observing logs & outputs
- Troubleshooting build failures

YAML: Deploy Bicep Using YAML Pipelines
- Deploying Bicep templates
- Using AzureCLI task in YAML
- Infrastructure-as-Code automation

YAML: Multistage Bicep Template Deployment
- Multi-stage Infra deployment
- Using environments (Dev, Test, Prod)
- Secure deployments with approvals

🔹 Basic YAML CI Pipeline for Web App
- Simple YAML pipeline for building a web app
- Restore → Build → Test steps

🔹 Demo: Basic YAML CI Pipeline for Web App
- Running the basic CI pipeline
- Reviewing job output

🔹 Complete YAML CI Pipeline for Web App & Database
- Adding database migration tasks
- Multi-job pipeline structure
- Including backend + DB steps

🔹 Demo: Running the Complete CI Pipeline
- Observing CI execution for both app & DB

🔹 Complete YAML CI/CD Pipeline for Web App, Database & Infra
- Full lifecycle automation
- Integrating CI + CD + IaC
- Best practices for production pipelines

🔹 Demo: Running Complete CI/CD Pipeline
- Running end-to-end deployment
- Validating infrastructure, DB, and app deployment

🔹 Best Practices
- YAML naming conventions
- Using templates for reusability
- Secrets & security best practices
- Pipeline optimization

🔹 Basic YAML Pipelines for Kubernetes
- Kubernetes build & deploy workflow
- Container image creation
- Simple K8s deployment YAML

🔹 Demo: Basic Kubernetes Pipeline
- Build → push image → deploy
- Review of K8s manifests

🔹 Complete Pipelines for Kubernetes
- Multi-stage deployment strategy
- Canary & blue-green approaches
- Production-ready k8s CI/CD

🔹 Presentation: Complete Kubernetes Pipelines
- Architectural overview
- Best practices for containerized workloads

🔹 Demo: Complete Kubernetes Pipelines
- Full end-to-end CI/CD demo
- Scaling, rolling updates & validation

🔹 Introduction to IaC with Terraform
- What is Terraform
- IaC workflow in Azure DevOps
- Terraform vs Bicep comparison

🔹 Deploy Terraform Configuration Manually
- init, plan, apply workflow
- Authenticating Terraform with Azure
- Demo of manual provisioning

🔹 Complete CI/CD Pipelines for Terraform Using YAML & TF State
- Remote state backend setup
- Pipeline for validate → plan → apply
- Safe Terraform deployment design

🔹 Static Analysis for Terraform
- TFLint, Checkov, TFsec usage
- Enforcing security & standards
- Best practices for analysis

🔹 Introduction to Templates & YAML Modularization
- Splitting YAML into templates
- Reusable stage, job, and step templates
- Pipeline modularity benefits

🔹 Demo with YAML Files

- Implementing template-based pipelines
- Importing variable templates
- Modular structure in practice

🔹 YAML Pipelines with Templates

- Best structure for templates
- Parameterized deployments
- Template libraries

🔹 Variable Groups

- Centralized variable management
- Linking variable groups to pipelines
- Secret variables

🔹 Demo: Variable Groups

- Using Key Vault-integrated variables
- Secure pipeline configurations

🔹 Trigger Pipeline from Another Pipeline

- Pipeline completion triggers
- Multi-repo pipeline triggering
- Example YAML structure

Resources:

https://www.youtube.com/watch?v=A_N5oHwwmTQ&list=PLl4APkPHzsUXseJO1a03CtfRDzr2hivbD

Azure Data Engineering Certifications For 2025-26

1. Foundation Level

🔹 AZ-900 — Microsoft Azure Fundamentals

- Covers cloud concepts, Azure services, security, governance, cost management.
- Recommended but optional for data engineers.

2. Core Data Engineer Certification

🔹 DP-203 — Microsoft Azure Data Engineer Associate

- This is the primary and official certification for Azure Data Engineers.
- Skills measured:
- Design & implement data storage
- Develop data processing solutions
- Manage and optimize data workflows & pipelines
- Work with Azure Data Lake Storage, ADF, Databricks, Synapse, Stream Analytics
- Technologies covered:
- ADF, Databricks, Spark, Azure Stream Analytics, Event Hubs, Cosmos DB, Azure SQL, Synapse, Kusto, Delta Lake.

3. Optional / Complementary Certifications

- These are not mandatory but highly relevant.

◆ DP-500 — Azure Enterprise Data Analyst Associate

- Focus on Power BI, data modeling, performance tuning, reporting with big datasets.
- Good if you also work with analytics/BI.

◆ AZ-305 — Azure Solutions Architect Expert

- Covers designing cloud solutions, networking, compute, identity, security.
- Useful for senior data engineers or architects.

◆ AI-102 — Azure AI Engineer Associate

- Good if your data engineering role touches ML pipeline integration.
- Covers Cognitive Services, Azure ML, vector databases.

◆ DP-100 — Azure Data Scientist Associate

- ML-focused, not required for data engineers but relevant for hybrid DE/DS roles.

4. Real-Time & Big Data Focus (Optional)

🔹 AZ-320 — Azure Support Engineer for Connectivity Specialty

- Useful for Event Hubs, networking, and troubleshooting.

🔹 Microsoft Fabric Certifications (coming soon)

- Microsoft Fabric/Data Engineering certifications are rolling out gradually.
- For now, relevant: Fabric Analytics Engineer Associate (DP-600)

⭐ Recommended Path for YOU (Azure Data Engineer with 5+ years experience)

- Since you already have Azure + ADF + Databricks experience:
- DP-203 (mandatory)
- DP-600 (if working with Fabric)
- AZ-305 (to transition into Data Architect roles)
- AI-102 or DP-100 (if ML pipelines are in your work)

Module 22: Building Real-time Projects (Final)

This is the most important and final step to become an Azure Data Engineer. Doing it is the best way to learn it. If you want to become a Data Engineer, start building Data Engineering projects. I can totally understand if you are an absolute beginner; it might be challenging to grasp the end-to- end functionality of a project. That's the main issue I am trying to solve using my YouTube channel. I want to help people, mostly beginners, by uploading real-time projects. This will greatly help them understand how Data Engineering projects are built in real-time scenarios. I have already uploaded two videos that cover the end-to-end functionality of an Azure Data Engineering Project. Start building the project by watching the below two videos.

After watching and building the resources with different areas sing the above module, you will have a clear understanding of how different Azure stacks are used in real-world projects. This will also help you answer questions asked in interviews for the Azure Data Engineering role easily. There are also some Azure Project videos available on YouTube uploaded by other YouTubers. I would strongly recommend watching as many videos as possible and trying to implement them in your subscription. This will help you get hands-on experience with different types of projects and receive guidance from different Data Engineer experts. I have provided links to some of the project videos available on YouTube.

1. https://www.youtube.com/watch?v=IaA9YNlg5hM
2. https://www.youtube.com/watch?v=pMqnvXgPKlI&list=PLOlK8ytA0MghGmAAT8W2u7VY mICdzeU5t
3. https://www.youtube.com/watch?v=pTpAKIJH9BM&t=537s
4. youtube.com/watch?v=buIkP-Al2mU
5. https://youtu.be/DzxtCxi4YaA?si=SVEAkvJ6Ti2KxdvN
6. https://youtu.be/cSOTxxvqwyE?si=KzXoMUsYtObw9dfa
7. https://youtu.be/JCDrvXwh4BQ?si=FUNPhFYnGJxyJK_2
8. https://www.youtube.com/watch?v=4fWUuqsUAPQ&list=PLrG_BXEk3kXw7NlUqxf8IE7Uc3kKwAI2p
9. https://www.youtube.com/watch?v=okrKwdn9Z34&list=PLrG_BXEk3kXyEV0dzmAN-49tLrQsM0jUa
10. https://www.youtube.com/watch?=mECDWTYiKp4&list=PLrG_BXEk3kXx6KE4nBmhf6QwSHMbznP2W
11. https://www.youtube.com/watch?v=8_Tr2vbTyw4
12. https://www.youtube.com/watch?v=hUTDVTuZO60&t=3038s
13. https://www.youtube.com/watch?v=XYQU5ZMuMuE&t=12273s
14. https://www.youtube.com/watch?v=P6dbfkSSmH4
15. https://www.youtube.com/watch?v=uHiyZitmIS0&t=1652s
16. https://www.youtube.com/watch?v=H9cK7ELbjLg&t=2723s
17. https://www.youtube.com/watch?v=01LVHch-1x0
18. https://www.youtube.com/watch?v=V-s8c6jMRN0&list=PLO9LeSU_vHCWUvkE1FrGeNxSve7YtJrYl
19. https://www.youtube.com/watch?v=B8uwFmVt4sU&t=401s&pp=ygULREJUIFByb2plY3Q%3D
20. https://www.youtube.com/watch?v=vJM0wDrTRxI&t=792s&pp=ygUeYXp1cmUgZGF0YSBlbmdpbmVlcmluZyBwcm9qZWN00gcJCSIKAYcqIYzv
21. https://www.youtube.com/watch?v=0GTZ-12hYtU&t=15674s&pp=ygUeYXp1cmUgZGF0YSBlbmdpbmVlcmluZyBwcm9qZWN00gcJCSIKAYcqIYzv
22. https://www.youtube.com/watch?v=d3Vw3VtKDnc&t=325s&pp=ygUeYXp1cmUgZGF0YSBlbmdpbmVlcmluZyBwcm9qZWN0
23. https://www.youtube.com/watch?v=Pg1GQZe9x5A&t=11314s&pp=ygUeYXp1cmUgZGF0YSBlbmdpbmVlcmluZyBwcm9qZWN0
24. https://www.youtube.com/watch?v=SeQEGZzdXr8&t=11s&pp=ygUeYXp1cmUgZGF0YSBlbmdpbmVlcmluZyBwcm9qZWN0

(Watch the Other Parts from this YT channel)

If you complete all the stages, then you can consider yourself an Intermediate Azure Data Engineer. You can apply for any Junior to Intermediate level Azure Data Engineering role. The only final thing you need to concentrate on is to build your resume/CV in a proper way by including all the required technologies that you learned in the above stages. If you are not a beginner, it would not take a full 1 Year to complete all the projects; however, a beginner would need at least 2 Years to prepare.

Projects:

Project 1

Data Lake Integration and Optimization with PySpark

Load data into a data lake and use PySpark for integrating, transforming, and optimizing data. Develop a system to uphold a structured data storage within the data lake for analytics support.

Project 2

Leverage Snowflake for Retail Sales Data Warehousing

Develop a strong data warehousing system using Snowflake for a retail business. Gather and modify sales data from different origins to support advanced analysis for managing inventory and predicting sales.

Project 3

Apache Airflow automation for ETL workflow orchestration

Develop a thorough ETL (Extract, Transform, Load) pipeline to automate the extraction, transformation, and loading of data into a data warehouse. Incorporate scheduling, error management, and monitoring to ensure a reliable ETL process.

Project 4

Exploring and transforming data using Azure Databricks

Perform standard DataFrame methods to explore and transform data. Key Points: Create a lab environment, Azure Databricks cluster.

Project 5

Ingest and load data into the Data Warehouse

The project involves transferring data into Synapse dedicated SQL pools with Poly Base and COPY through T-SQL. Employ workload management and Copy activity within an Azure Synapse pipeline for ingesting petabyte-scale data.

Project 6

Leverage Azure Databricks Workflows or Azure Data Factory for data transformation

The project focuses on constructing data integration pipelines to collect data from various sources, modifying the data through mapping data flows and notebooks, and transferring the data into one or more data destinations.

THANK YOU

FOR DOWNLOADING

Follow

Save

If you find this document helpful, I'd appreciate it if you could like, share, and follow me for more updates and insights!

Like

Share

Ganesh R

Senior Azure Data Engineer

@rg_data_talks

@rganesh0203