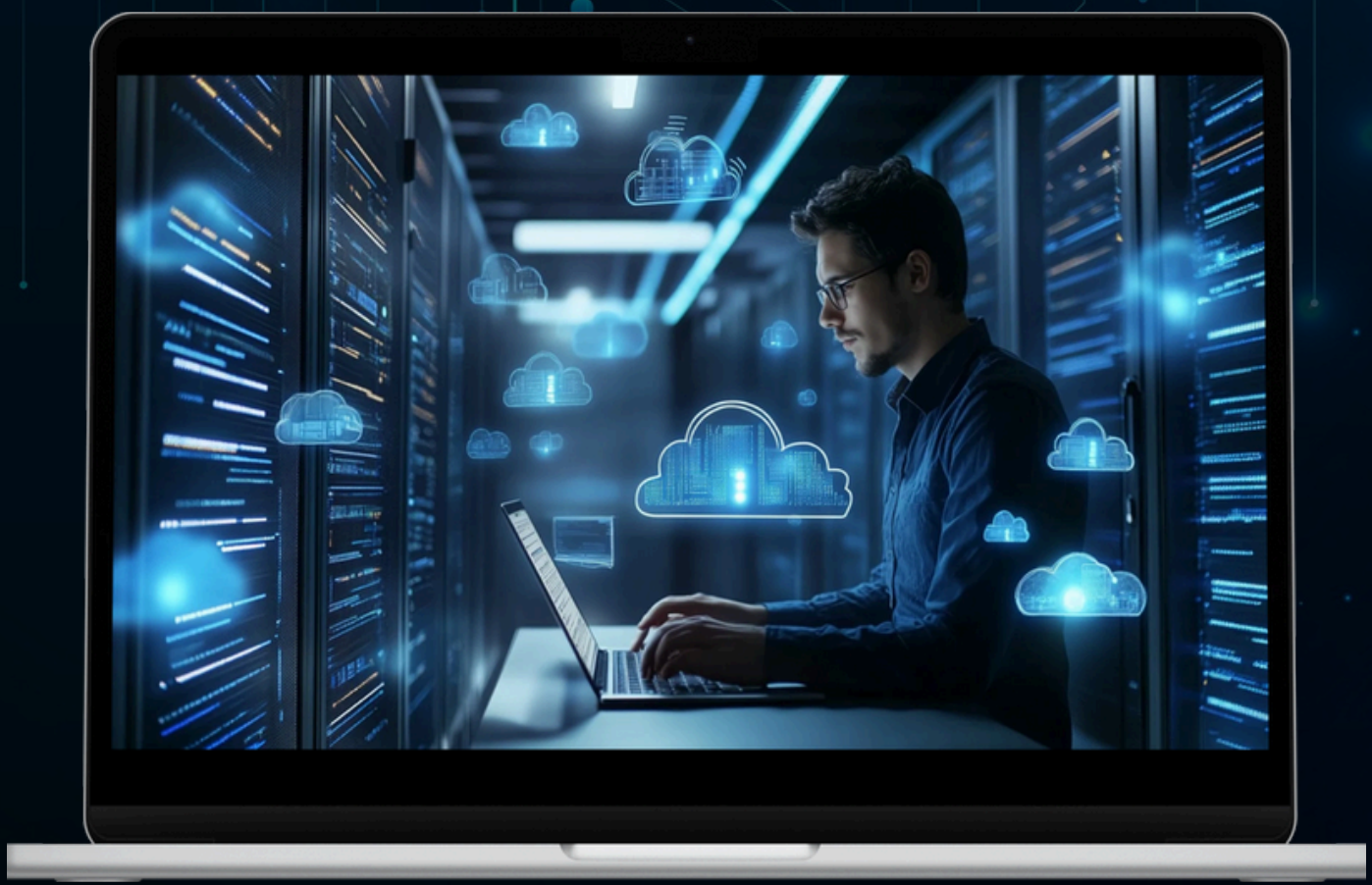
Cloud Upskill

Azure Data Engineering

Course Content



+91 8019953358 Info@cloudupskills.com www.cloudupskills.com

What is ADE (Azure Data Engineer)?

The Azure Data Engineer (ADE) course equips learners with the skills to design and implement data solutions using Microsoft Azure. It focuses on building pipelines, integrating data from various sources, transforming it for analytics, and ensuring secure, scalable, and high- performance data processing systems in the cloud.

Why Azure Data Engineering Matters?

IIn a world driven by data, organizations rely on cloud platforms like Azure to make smarter, faster decisions. Azure Data Engineers are pivotal in handling massive volumes of data, ensuring it is reliable, clean, and ready for insights. With businesses moving to the cloud at an accelerating pace, the demand for professionals who can manage andorchestrate data pipelines is higher than ever.

Career Opportunities After this Course

- Cloud Data Engineer
- Big Data Engineer
- Data Engineer
- ETL Developer
- Azure Data Analyst
- Business Intelligence Engineer

+91 8019953358 Info@cloudupskills.com www.cloudupskills.com

Who Should attend this Course?

- ✅ Fresh Graduates interested in Cloud & Dataroles
- ✅ IT Professionals transitioning to Cloud Data Engineering
- ✅ Data Analysts & BI Developersupskilling for Azure roles
- ✅ Any one preparing for Microsoft DP-203 Certification

+91 8019953358 Info@cloudupskills.com www.cloudupskills.com

What You'll Learn in the ADE Program

You will:

- ✅ Design and build scalable data pipelines using Azure Data Factory
- ✅ Transform and clean data with Azure Data bricks and SQL
- ✅ Implement data storage using Azure Data Lake and Synapse Analytics
- ✅ Monitor and optimize data workflows in production
- ✅ Apply best practices in data governance, security, and compliance

Tools You'll Master

- **Azure Data Factory**
  Data pipeline orchestration
- **Azure Synapse**
  Analytics and data warehousing
- **Azure Monitor**
  Pipeline monitoring and diagnostics
- **Azure Databricks**
  Scalable big data processing
- **Azure Data Lake**
  Bigdata storage
- **Azure KeyVault**
  Secure credential management

Career & Project Readiness

- 🎓 Real-world projects aligned with industry scenarios
- 📘 Prepared for Microsoft DP-203 certification exam
- 🤝 Contribute to cloud data teams from day one
- 📁 Build a portfolio with labs and case studies

+91 8019953358 Info@cloudupskills.com www.cloudupskills.com

Cloud Upskill

Azure Data Engineering Training

End to End Data Engineering on Microsot Azure

SQL

Introduction To DBMS

- File Management System And Its Drawbacks
- Database Management System (DBMS) and Data Models
    - Physical Data Models
    - Logical Data Models
        - Hierarchical Data Model (HDBMS)
        - Network Data Model (NDBMS)
        - Relational Data Model (RDBMS)
        - Object Data Model (ODBMS)
        - Object Relational Data Model (ORDBMS)
- Conceptual Data Models
    - Entity – Relationship (E-R) Model

Introduction To SQL Server

- Advantages and Drawbacks Of SQL Server Compared To Oracle And DB2
    - Connecting To Server
    - Server Type
    - Server Name
    - Authentication Modes
        - Sql Server Authentication Mode
        - Windows Authentication Mode

- Login and Password
- Sql Server Management Studio and Tools In Management Studio
    - Object Explorer
    - Object Explorer Details
    - Query Editor

TSQL (Transact-Structured Query Language)

Introduction To TSQL

- History and Features of TSQL
- Types Of TSQL Commands
    - Data Definition Language (DDL)
    - Data Manipulation Language (DML)
    - Data Query Language (DQL)
    - Data Control Language (DCL)
    - Transaction Control Language (TCL)
    - Database
        - Creating Database
        - Altering Database
        - Deleting Database
        - Constrains
            - Procedural Integrity Constraints
            - Declarative Integrity Constraints
                - Not Null
                - Unique
                - Default
                - Check constraints
                - Primary Key
                - Referential Integrity
                - foreign key constraints
                - Data Types In TSQL
                - Table

- Creating Table
- Altering Table
- Deleting Table

Data Manipulation Language

- Insert
    - Identity
    - Creating A Table From Another Table
    - Inserting Rows From One Table To Another
    - Update
        - Computed Columns
        - Delete
        - Truncate
        - Differences Between Delete and Truncate

Data Query Language (DQL)

- Select
- Where clause
- Order By Clause
- Distinct Keyword
- Isnull() function
- Column aliases
- Predicates
    - Between … And
    - In
    - Like
    - Is Null

Built In Functions

- Scalar Functions
    - Numeric Functions
    - Character Functions
    - Conversion Functions
    - Date Functions

- Aggregate Functions
    - COUNT
    - SUM
    - AVG
    - MIN,
    - MAX
- Convenient Aggregate Functions
- Statistical Aggregate Functions
- Group By and Having Clauses
- Super Aggregates
- Over(partition by ...) Clause
- Ranking Functions
- Common Table Expressions (CTE)

Set Operators

- Union
- Intersect
- Except

Joins

- Inner Join
    - Equi Join
    - Natural Join
    - Non-Equi Join
    - Self Join
    - Outer Join
        - Left Outer Join
        - Right Outer Join
        - Full Outer Join
        - Cross Join

Sub Queries

- Single Row Sub Queries
- Multi Row Sub Queries
    - Any or Some
    - ALL
    - Nested Sub Queries
    - Co-Related Sub Queries
        - Exists and Not Exists

Indexes

- Clustered Index
- NonClustered Index
- Create
- Alter
- Drop Indexes
- Using Indexes

Security

- Login Creation
    - SQL Server Authenticated Login
    - Windows Authenticated Login
    - User Creation
    - Granting Permissions
    - Revoking Permissions
    - Roles

Views

- Purpose Of Views
- Creating
- Altering
- Dropping Indexes
- Simple and Complex Views
- Encryption
- Schema Binding Options in creating views

Transaction Management

- Introduction
- Begin Transaction
- Commit Transaction
- Rollback Transaction
- Save Transaction
- Role Of Log File In Transaction Management
- Implicit Transactions

TSQL Programming

- Drawbacks Of TSQL that leads to TSQL Programming
- Introduction To TSQL Programming
- Control statements In TSQL Programming
    - Conditional Control Statements
        - If
        - Case
    - Looping Control Statements
        - While

Cursors

- Working With Cursors
- Types Of Cursors
    - Forward_Only and Scroll Cursors
    - Static Cursors
    - Dynamic Cursors
    - Keyset Cursors
    - Local Cursors
    - Global Cursors

Stored Sub Programs

- Advantages Of Stored Sub Programs compared to Independent SQL Statements

- Stored Procedures
    - Creating
    - Altering
    - Dropping
    - Optional Parameters
    - Input Parameters
    - Output Parameters
    - Permissions on Stored Procedures

User Defined Functions

- Creating, Altering and Dropping
- Types Of User Defined Functions
    - Scalar Functions
    - Table Valued Functions
        - Inline Table Valued Functions
        - Multi Statement Table Valued Functions
- Permissions On User Defined Functions

Triggers

- Purpose of Triggers
- Differences Between Stored Procedures and User Defined Functions and Triggers
- Creating
- Altering
- Dropping Triggers
- Magic Tables
- Instead Of Triggers

Exception Handling

- Implementing Exception Handling
- Adding and removing User Defined Error Messages To And From SQL Server Error Messages List
- Raising Exceptions Manual

Working With XML Data Type
Attach and Detach of Database

Python

Introduction to Python

- What is Python?
- WHY PYTHON?
- History
- Features
    - Dynamic
    - Interpreted
- Object oriented
- Embeddable
- Extensible
- Large standard libraries
- Free Source
- Open source
- Why Python is General Language?
- Limitations of Python
- What is PSF?
- Python implementations
- Python applications
- Python versions
- Python in Real Time Industry
- Software Development Architectures

Python Software's

- Python Distributions
- Download & Python Installation Process
    - Windows
    - Unix
    - Linux
    - Mac
- Online Python IDLE
- Python Real-time IDEs
    - Spyder
    - Jupyter Note Book
    - PyCharm
    - Rodeo,
    - Visual Studio Code
    - ATOM
    - PyDevetc

Python Language Fundamentals

- Python Implementation Alternatives/Flavors
- Keywords
- Identifiers
- Constants / Literals
- Data types
- Python VS JAVA
- Python Syntax

Different Modes of Python

- Interactive Mode
- Scripting Mode
- Programming Elements
- Structure of Python program
- First Python Application
- Comments in Python
- Python file extensions
- Setting Path in Windows
- Edit and Run python program without IDE
- Edit and Run python program using IDEs
- INSIDE PYTHON
- Programmers View of Interpreter
- Inside INTERPRETER
- What is Byte Code in PYTHON?
- Python Debugger

Python Variables

- bytes Data Type
- byte array
- String Formatting in Python
- Math, Random, Secrets Modules
- Introduction
- Initialization of variables
- Local variables
- Global variables
- 'global' keyword

- Input operations
- Output operations
- Data conversion functions
    - int()
    - float()
    - complex()
    - str()
    - chr()
    - ord()

Operators

- Arithmetic Operators
- Comparison Operators
- Python Assignment Operators
- Logical Operators
- Bitwise Operators
- Shift operators
- Membership Operators
- Identity Operators
- Ternary Operator
- Operator precedence
- Difference between “is” vs “==”

Input & Output Operators

- Print
- Input
- Command-line arguments

Control Statements

- Conditional control statements
- If
- If-else
- If-elif-else
- Nested-if
- Loop control statements
- for
- while

- Nested loops
- Branching statements
- Break
- Continue
- Pass
- Return
- Case studies

Data Structures or Collections

- Introduction
- Importance of Data structures
- Applications of Data structures
- Types of Collections
- Sequence
- Strings
- List
- Tuple
- range
- Non sequence
- Set
- Frozen set
- Dictionary

Data Structures or Collections

- What is string
- Representation of Strings
- Processing elements using indexing
- Processing elements using Iterators
- Manipulation of String using Indexing and Slicing
- String operators
- Methods of String object
- String Formatting
- String functions
- String Immutability
- Case studies

- Nested loops
- Branching statements
- Break
- Continue
- Pass
- Return
- Case studies

Data Structures or Collections

- Introduction
- Importance of Data structures
- Applications of Data structures
- Types of Collections
- Sequence
- Strings
- List
- Tuple
- range
- Non sequence
- Set
- Frozen set
- Dictionary

Data Structures or Collections

- What is string
- Representation of Strings
- Processing elements using indexing
- Processing elements using Iterators
- Manipulation of String using Indexing and Slicing
- String operators
- Methods of String object
- String Formatting
- String functions
- String Immutability
- Case studies

List Collection

- What is List
- Need of List collection
- Different ways of creating List
- List comprehension
- List indices
- Processing elements of List through Indexing and Slicing
- List object methods
- List is Mutable
- Mutable and Immutable elements of List
- Nested Lists
- List_of_lists
- Hardcopy
- shallowCopy
- DeepCopy
- zip() in Python
- How to unzip?
- Python Arrays
- Case studies

Tuple Collection

- What is tuple?
- Different ways of creating Tuple
- Method of Tuple object
- Tuple is Immutable
- Mutable and Immutable elements of Tuple
- Process tuple through Indexing and Slicing
- List v/s Tuple
- Case studies

Set Collection

- What is set?
- Different ways of creating set
- Difference between list and set
- Iteration Over Sets
- Accessing elements of set
- Python Set Methods

- Python Set Operations
- Union of sets
- functions and methods of set
- Python Frozen set
- Difference between set and frozenset ?
- Case study

Dictionary Collection

- What is dictionary?
- Difference between list, set and dictionary
- How to create a dictionary?
- Python Hashing?
- Accessing values of dictionary
- Python Dictionary Methods
- Copying dictionary
- Updating Dictionary
- Reading keys from Dictionary
- Reading values from Dictionary
- Reading items from Dictionary
- Delete Keys from the dictionary
- Sorting the Dictionary
- Python Dictionary Functions and methods
- Dictionary comprehension

Functions

- What is Function?
- Advantages of functions
- Syntax and Writing function
- Calling or Invoking function
- Classification of Functions
    - No arguments and No return values
    - With arguments and No return values
    - With arguments and With return values
    - No arguments and With return values
    - Recursion

- Python argument type functions :
    - Default argument functions
    - Required(Positional) arguments function
    - Keyword arguments function
    - Variable arguments functions
- pass' keyword in functions
- Lambda functions/Anonymous functions
    - map()
    - filter()
    - reduce()
- Nested functions
- Non local variables
- global variables
- Closures
- Decorators
- Generators
- Iterators
- Monkey patching

Python Modules

- Importance of modular programming
- What is module
- Types of Modules.
    - Pre defined
    - User defined
- User defined modules creation
- Functions based modules
- Class based modules
- Connecting modules
- Import module
- From … import
- Module alias / Renaming module
- Built In properties of module

Packages

- Organizing python project into packages
- Types of packages
  - pre defined
  - user defined.
- Package v/s Folder
- py file
- Importing package
- PIP
- Introduction to PIP
- Installing PIP
- Installing Python packages
- Un installing Python packages

OOPs

- Procedural v/s Object oriented programming
- Principles of OOP
  - Encapsulation
  - Abstraction (Data Hiding)
- Classes and Objects
- How to define class in python
- Types of variables
  - instance variables
  - class variables.
- Types of methods
  - instance methods
  - class method
  - static method
- Object initialization
- 'self' reference variable
- 'cls' reference variable
- Access modifiers
  - private(__)
  - protected(_)
  - public

- AT property class
- Property() object
- Creating object properties using setaltr, getaltr functions
- Encapsulation(Data Binding)
- What is polymorphism?
- Overriding
    - Method overriding
    - Constructor overriding
- Overloading
    - Method Overloading
    - Constructor Overloading
- Operator Overloading
    - Class re-usability
    - Composition
    - Aggregation
    - Inheritance
        - single
        - multi-level
        - multiple
        - hierarchical
        - hybrid inheritance
        - Diamond inheritance
- Constructors in inheritance
- Object class
- super()
- Runtime polymorphism
- Method overriding
- Method resolution order(MRO)
- Method overriding in Multiple inheritance
- Hybrid Inheritance
- Duck typing
- Concrete Methods in Abstract Base Classes
- Difference between Abstraction & Encapsulation

- Inner classes
- Introduction
- Writing inner class
- Accessing class level members of inner class
- Accessing object level members of inner class
- Local inner classes
- Complex inner classes
- Case studies

Exception Handling & Types of Errors

- What is Exception?
- Why exception handling?
- Syntax error v/s Runtime error
- Exception codes
    - AttributeError
    - ValueError
    - IndexError
    - TypeError
- What is Exception?
- Why exception handling?
- Syntax error v/s Runtime error
- Exception codes
- Handling exception – try except block
- Try with multi except
- Handling multiple exceptions with single except block
- Finally block
    - Try-except-finally
    - Try with finally
    - Case study of finally block
- Raise keyword
    - Custom exceptions / User defined exceptions
    - Need to Custom exceptions

Regular expressions

- Understanding regular expressions
- String v/s Regular expression string
- “re” module functions
- Match()
- Search()
- Split()
- Findall()
- Compile()
- Sub()
- Subn()
- Expressions using operators and symbols
- Simple character matches
- Special characters
- Character classes
- Mobile number extraction
- Mail extraction
- Different Mail ID patterns
- Data extraction
- Password extraction
- URL extraction
- Vehicle number extraction

File &Directory handling

- Introduction to files
- Opening file
- File modes
- Reading data from file
- Writing data into file
- Appending data into file
- Line count in File
- CSV module
- Creating CSV file
- Reading from CSV file
- Writing into CSV file
- Object serialization – pickle module
- XML parsing
- JSON parsing

Python Logging

- Logging Levels
- implement Logging
- Configure Log File in over writing Mode
- Timestamp in the Log Messages
- Python Program Exceptions to the Log File
- Requirement of Our Own Customized Logger
- Features of Customized Logger

Date & Time module

- How to use Date & Date Time class
- How to use Time Delta object
- Formatting Date and Time
- Calendar module
- Text calendar
- HTML calendar

OS module

- Shell script commands
- Various OS operations in Python
- Python file system shell methods
- Creating files and directories
- Removing files and directories
- Shutdown and Restart system
- Renaming files and directories
- Executing system commands

Multi-threading & Multi Processing

- Introduction
- Multi tasking v/s Multi threading
- Threading module
- Creating thread – inheriting Thread class
- Using callable object
- Life cycle of thread
- Single threaded application
- Multi threaded application

- Can we call run() directly?
- Need to start() method
- Sleep()
- Join()
- Synchronization
    - Lock class
    - acquire()
    - release() functions

Garbage collection

- Introduction
- Importance of Manual garbage collection
- Self reference objects garbage collection
- 'gc' module
- Collect() method
- Threshold function

Python Data Base Communications(PDBC)

- Introduction to DBMS applications
- File system v/s DBMS
- Communicating with MySQL
- Python – MySQL connector
- connector module
- connect() method
- Oracle Database
- Install cx_Oracle
- Cursor Object method
- execute() method
- executeMany() method
- fetchone()
- fetchmany()
- fetchall()
- Static queries v/s Dynamic queries
- Transaction management

Python – Network Programming

- What is Sockets?
- What is Socket Programming?
- The socket Module
- Server Socket Methods
- Connecting to a server
- A simple server-client program
- Server
- Client

Tkinter & Turtle

- Introduction to GUI programming
- Tkinter module
- Tk class
- Components / Widgets
- Label
- Entry
- Button
- Combo
- Radio
- Types of Layouts
- Handling events
- Widgets properties

Data analytics modules

- Numpy
- Introduction
- Scipy
- Introduction
- Arrays
- Datatypes
- Matrices
- N dimension arrays
- Indexing and Slicing
- Pandas
- Data Frames

- Merge
- Join
- Concat
- MatPlotLib introduction
- Drawing plots

DJANGO

- Introduction to PYTHON Django
- What is Web framework?
- Why Frameworks?
- Define MVT Design Pattern
- Difference between MVC and MVT

PANDAS

- Introduction to Pandas
- Environment Setup Pandas
- Introduction to Data Structures
    - Dimension & Description
    - Series
    - DataFrame
    - Data Type of Columns
    - Panel

Pandas — Series

- Series
- Create an Empty Series
- Create a Series f
- rom ndarray
- rom dict
- rom Scalar
- Accessing Data from Series with Position
- Retrieve Data Using Label (Index)

- Merge
- Join
- Concat
- MatPlotLib introduction
- Drawing plots

DJANGO

- Introduction to PYTHON Django
- What is Web framework?
- Why Frameworks?
- Define MVT Design Pattern
- Difference between MVC and MVT

PANDAS

- Introduction to Pandas
- Environment Setup Pandas
- Introduction to Data Structures
    - Dimension & Description
    - Series
    - DataFrame
    - Data Type of Columns
    - Panel

Pandas — Series

- Series
- Create an Empty Series
- Create a Series f
- rom ndarray
- rom dict
- rom Scalar
- Accessing Data from Series with Position
- Retrieve Data Using Label (Index)

- Merge
- Join
- Concat
- MatPlotLib introduction
- Drawing plots

DJANGO

- Introduction to PYTHON Django
- What is Web framework?
- Why Frameworks?
- Define MVT Design Pattern
- Difference between MVC and MVT

PANDAS

- Introduction to Pandas
- Environment Setup Pandas
- Introduction to Data Structures
    - Dimension & Description
    - Series
    - DataFrame
    - Data Type of Columns
    - Panel

Pandas — Series

- Series
- Create an Empty Series
- Create a Series f
- rom ndarray
- rom dict
- rom Scalar
- Accessing Data from Series with Position
- Retrieve Data Using Label (Index)

Pandas – Iteration

- Iterating a DataFrame
- iteritems()
- iterrows()
- itertuples()

Pandas – Sorting

- By Label
- Sorting Algorithm

Pandas – Working with Text Data
Pandas – Options and Customization

- get_option(param)
- set_option(param,value)
- reset_option(param)
- describe_option(param)
- option_context()

Pandas – Indexing and Selecting Data

- .loc()
- .iloc()
- .ix()
- Use of Notations

Pandas – Statistical Functions

- Percent_change
- Covariance
- Correlation
- Data Ranking

Pandas – Window Functions

- .rolling() Function
- .expanding() Function
- .ewm() Function

Pandas – Aggregations

- Applying Aggregations on DataFrame

Pandas – Missing Data

- Cleaning / Filling Missing Data
- Replace NaN with a Scalar Value
- Fill NA Forward and Backward
- Drop Missing Values
- Replace Missing (or) Generic Values

Pandas – GroupBy

- Split Data into Groups
- View Groups
- Iterating through Groups
- Select a Group
- Aggregations
- Transformations
- Filtration

Pandas – Merging/Joining

- Merge Using 'how' Argument

Pandas – Concatenation

- Concatenating Objects
- Time Series

Pandas – Date Functionality
Pandas – Timedelta
Pandas – Categorical Data

- Object Creation

Pandas – Visualization

- Bar Plot
- Histograms
- Box Plots

- Area Plot
- Scatter Plot
- Pie Chart

Pandas – IO Tools

- Pandas – IO Tools

Pandas – Sparse Data
Pandas – Caveats & Gotchas
Pandas – Comparison with SQL

NUMPY

- NUMPY - INTRODUCTION
- NUMPY - ENVIRONMENT
- NUMPY - NDARRAY OBJECT
- NUMPY - DATA TYPES
    - Data Type Objects (dtype)
- NUMPY - ARRAY ATTRIBUTES
    - shape
    - ndim
    - itemsize
    - flags
- NUMPY - ARRAY CREATION ROUTINES
    - empty
    - zeros
    - ones
- NUMPY - ARRAY FROM EXISTING DATA
    - asarray
    - frombuffer
    - fromiter

- NUMPY - ARRAY FROM NUMERICAL RANGES
    - arange
    - linspace
    - logspace
- NUMPY - INDEXING & SLICING
- NUMPY - ADVANCED INDEXING
    - Integer Indexing
    - Boolean Array Indexing
- NUMPY - BROADCASTING
- NUMPY - ITERATING OVER ARRAY
    - Iteration
    - Order
    - Modifying Array Values
    - External Loop
    - Broadcasting Iteration
- NUMPY – ARRAY MANIPULATION
    - reshape
    - ndarray.flat
    - ndarray.flatten
    - ravel
    - transpose
    - ndarray.T
    - swapaxes
    - rollaxis
    - broadcast
    - broadcast_to
    - expand_dims
    - squeeze
    - concatenate
    - stack
    - hstack and numpy.vstack
    - split

- hsplit and numpy.vsplit
- resize
- append
- insert
- delete
- unique

- NUMPY – BINARY OPERATORS
  - bitwise_and
  - bitwise_or
  - invert()
  - left_shift
  - right_shift

- NUMPY - STRING FUNCTIONS
- NUMPY - MATHEMATICAL FUNCTIONS
  - Trigonometric Functions
  - Functions for Rounding

- NUMPY - ARITHMETIC OPERATIONS
  - reciprocal()
  - power()
  - mod()

- NUMPY - STATISTICAL FUNCTIONS
  - amin() and numpy.amax()
  - ptp()
  - percentile()
  - median()
  - mean()
  - average()
  - Standard Deviation
  - Variance

- NUMPY – SORT, SEARCH & COUNTING FUNCTIONS
    - sort()
    - argsort()
    - lexsort()
    - argmax() and numpy.argmin()
    - nonzero()
    - where()
    - extract()
- NUMPY – BYTE SWAPPING
    - ndarray.byteswap()
- NUMPY – COPIES & VIEWS
    - No Copy
    - View or Shallow Copy
    - Deep Copy
- NUMPY – MATRIX LIBRARY
    - empty()
    - matlib.zeros()
    - matlib.ones()
    - matlib.eye()
    - matlib.identity()
    - matlib.rand()
- NUMPY – LINEAR ALGEBRA
    - dot()
    - vdot()
    - inner()
    - matmul()
    - Determinant
    - linalg.solve()

- NUMPY - MATPLOTLIB
  - Sine Wave Plot
  - subplot()
  - bar()
- NUMPY – HISTOGRAM USING MATPLOTLIB
  - histogram()
  - plt()
- NUMPY - I/O WITH NUMPY
  - save()
  - savetxt()

Cloud Upskill

Pyspark

Introduction to Big Data & Apache Spark

- Limitations of traditional data processing systems
- Big Data characteristics
    - Volume
    - Velocity
    - Variety
    - Veracity
    - Value
- Hadoop vs Spark comparison
- Why Spark is faster than MapReduce
- Spark use cases in real-world industries
- Spark ecosystem overview
    - Spark SQL
    - Streaming,
    - MLlib
    - GraphX
- PySpark vs Scala Spark
- Spark deployment modes
    - Local
    - Standalone
    - YARN
    - Kubernetes

Spark Architecture & Core Concepts

- Spark Architecture
    - Master–Worker architecture
    - Driver program responsibilities
    - Executors and task execution lifecycle
    - Cluster managers
        - YARN
        - Mesos
        - Kubernetes
    - Application vs Job vs Stage vs Task

Execution Model

- Lazy evaluation
- Actions vs Transformations
- Directed Acyclic Graph (DAG)
- Logical plan vs Physical plan
- Catalyst optimizer overview
- Tungsten execution engine

Data Movement Concepts

- Narrow transformations
- Wide transformations
- Shuffle operations and cost
- Fault tolerance and lineage

PySpark Environment Setup

- Installing PySpark (Windows / Linux / Mac)
- Running Spark in local mode
- Spark shell vs PySpark shell
- Jupyter Notebook with PySpark
- SparkSession and SparkContext
- Spark configuration parameters
- Understanding Spark UI

PySpark RDD Fundamentals

- What is an RDD
- RDD characteristics
    - immutable
    - distributed
- Creating RDDs
    - parallelize
    - external sources
- RDD transformations
    - map
    - flatMap
    - filter

    - distinct
    - union
    - intersection

- RDD actions

    - collect
    - count
    - take
    - reduce

- Pair RDD operations
- RDD persistence and caching
- When to use RDD vs DataFrame

DataFrame & Dataset API

- DataFrame Basics

    - What is a DataFrame
    - Schema and metadata
    - Immutability and distributed nature

- Creating DataFrames

    - From CSV, JSON, Parquet, Avro
    - From databases (JDBC)
    - From Python collections
    - Reading nested and semi-structured data

- Schema Management

    - Schema inference
    - Explicit schema definition
    - StructType and StructField
    - Handling schema evolution

DataFrame Transformations & Column Operations

- select
- selectExpr
- withColumn
- withColumnRenamed
- drop
- dropDuplicates
- filter vs where
- when / otherwise (conditional logic)
- cast
- type conversion
- explode
- posexplode
- array
- map functions
- String functions
    - split
    - regexp_replace
    - trim
- Date
- timestamp functions
- Null handling strategies
    - dropna
    - fillna
    - isNull / isNotNull

Joins, Aggregations & Window Functions

- Aggregations
    - groupBy fundamentals
    - agg() usage
    - Built-in aggregation functions
    - Custom aggregations

- Joins
    - Inner join
    - Left join
    - Right join
    - Full join
    - Cross join
    - Broadcast join
    - Join hints
    - Handling join skew
    - Salting technique
- Window Functions
    - Window specification
    - partitionBy
    - orderBy
    - row_number
    - rank, dense_rank
    - lag lead
    - Running totals
    - moving averages
    - Real-world analytical use cases

Performance Optimization Techniques

- Partitioning & Parallelism
    - Understanding partitions
    - repartition vs coalesce
    - Optimal partition sizing
- Caching & Persistence
    - cache vs persist
    - Storage levels
    - Memory vs disk trade-offs

- Query Optimization
  - Predicate pushdown
  - Column pruning
  - File format optimization (Parquet vs ORC)
  - Compression techniques

- Shuffle Optimization
  - Reducing shuffle operations
  - Adaptive Query Execution (AQE)
  - Skew handling techniques

Incremental & Batch Processing Design

- ncremental data processing concepts
- Handling late-arriving data
- Merge (UPSERT) logic
- Deduplication strategies
- Idempotent pipeline design
- Slowly Changing Dimensions (SCD Type 1 & 2)
- Watermarking concepts

Error Handling, Logging & Debugging

- Try–except handling in PySpark
- Bad records handling
- Corrupt file handling
- Logging using Log4j
- Spark UI job analysis
- Debugging failed jobs
- Handling out-of-memory issues

Working with File Formats & Storage

- CSV, JSON limitations
- Parquet internals
- ORC advantages
- Partitioned data layout
- Writing optimized output files
- Handling small file problem
- Cloud storage integration (ADLS / S3 / GCS)

Spark SQL

- Writing SQL queries on DataFrames
- Temporary and global views
- SQL vs DataFrame API
- UDFs vs SQL functions
- Performance considerations

PySpark with Cloud & Data Engineering Tools

- PySpark with Azure Data Factory
- PySpark with Databricks
- Orchestration concepts
- Parameterization
- CI/CD considerations

Real-World Use Cases & Projects

- End-to-end ETL pipeline
- Incremental ingestion use case
- Data cleaning and enrichment project
- Analytics reporting use case
- Performance tuning case study

Azure Data Factory

Introduction to Data Integration & Azure Data Factory

- What is data integration
- ETL vs ELT concepts
- Role of ADF in modern data platforms
- ADF vs SSIS vs Informatica vs Talend
- When to use ADF vs Databricks vs Synapse
- Azure Data Factory pricing overview
- Typical enterprise use cases
- ADF limits and constraints

Azure Data Factory Architecture & Core Concepts

- ADF Architecture
    - Azure Data Factory overview
    - Control plane vs Data plane
    - ADF as orchestration layer
    - Integration with Azure ecosystem
- Core Components
    - Pipelines
    - Activities
    - Datasets
    - Linked Services
    - Integration Runtime (IR)
- Control Flow vs Data Flow
    - Control flow execution model
    - Data flow Spark-based execution
    - When to use Data Flows vs Databricks

Integration Runtime Deep Dive

- Types of Integration Runtime
    - Azure Integration Runtime
    - Self-hosted Integration Runtime
    - Azure SSIS Integration Runtime
- IR Concepts
    - Compute provisioning
    - Scaling and performance
    - Network and firewall considerations
    - High availability setup
    - Security best practices

Linked Services & Datasets

- Linked service configuration
- Authentication methods
    - Managed Identity
    - Service Principal
    - Key-based authentication
- Connecting to:
    - Azure Blob Storage
    - ADLS Gen2
    - Azure SQL Database
    - On-prem databases
- Dataset properties
- Parameterized datasets
- Schema drift handling

Pipeline Development Fundamentals

- Creating pipelines
- Adding and configuring activities
- Pipeline parameters and variables
- Expressions and dynamic content
- System variables
- Debug vs Triggered runs
- Publishing pipelines

Copy Activity – Deep Dive

- Copy activity architecture
- Source configuration
- Sink configuration
- Mapping and schema handling
- Data consistency verification
- Parallel copy
- Partition options
- Fault tolerance
- Performance tuning strategies
- Copy activity limitations

Control Flow Activities

- Lookup activity
- Get Metadata activity
- If Condition activity
- Switch activity
- ForEach activity
- Until activity
- Execute Pipeline activity
- Wait activity
- Validation activity
- Delete activity
- Filter activity
- Fail activity
- Web activity (REST API integration)
- Stored Procedure activity

Mapping Data Flows

- What are Mapping Data Flows
- Spark-based execution model
- Data Flow clusters
- Transformations:
    - Source & Sink
    - Select, Filter, Derived Column
    - Join, Lookup, Exists
    - Aggregate
    - Window
    - Conditional Split
    - Flatten
- Handling schema drift
- Debugging data flows
- Performance optimization
- When NOT to use data flows

Incremental Loading Strategies

- Full load vs Incremental load
- Watermark pattern
- Last modified date approach
- Delta column strategy
- Change Data Capture (CDC)
- Handling late-arriving data
- Incremental loads with Copy Activity
- Incremental loads with Data Flows

Triggers & Pipeline Orchestration

- Trigger Types
    - Schedule trigger
    - Tumbling window trigger
    - Event-based trigger

- Orchestration Patterns
    - Pipeline dependency chaining
    - Fan-in and fan-out pattern
    - Metadata-driven pipelines
    - Reusable pipeline frameworks

Error Handling & Monitoring

- Retry policies
- Failure paths
- Custom error messages
- Pipeline-level error handling
- Monitoring pipeline runs
- Activity run diagnostics
- Alerts and notifications
- Integration with Azure Monitor & Log Analytics

Performance Optimization

- Parallel copy settings
- Data partitioning techniques
- Optimizing Integration Runtime
- Cost vs performance trade-offs
- Throttling considerations
- Debug vs production performance

Security, Governance & Networking

- Managed Identity
- Azure Key Vault integration
- Role-Based Access Control (RBAC)
- Private endpoints
- Firewall rules
- Data encryption (at rest & in transit)
- Auditing and compliance

CI/CD & DevOps for ADF

- Git integration
- Branching strategies
- ARM templates
- Environment promotion (Dev → Test → Prod)
- Parameterization for environments
- Release pipelines
- Best practices for enterprise deployments

Real-World ADF Use Cases & Projects

- End-to-end ETL pipeline
- Incremental ingestion project
- Metadata-driven ingestion framework
- REST API ingestion pipeline
- Hybrid (on-prem to cloud) ingestion
- Cost-optimized ingestion pipeline

Azure Data Bricks

Introduction to Azure Databricks & Lakehouse

- Evolution from Data Warehouse to Data Lake to Lakehouse
- What is Azure Databricks
- Databricks vs Synapse vs HDInsight
- Databricks use cases in analytics, ML, and AI
- Azure Databricks pricing and cost model
- Databricks editions (Standard, Premium)
- When to use Databricks vs ADF vs Spark on VM

Azure Databricks Architecture

- Workspace Architecture
    - Azure Databricks workspace components
    - Notebooks, Jobs, Clusters, Repos
    - Workspace organization best practices
- Control Plane vs Data Plane
    - Control plane responsibilities
    - Data plane responsibilities
    - Customer-managed vs Microsoft-managed VNET
    - Security isolation model
- Databricks Runtime
    - Optimized Spark runtime
    - Runtime versions (LTS vs ML)
    - Photon engine overview
    - Security and performance enhancements

Databricks Workspace & Development Tools

- Notebooks overview
- Supported languages (PySpark, SQL, Scala, R)
- Notebook execution model
- Magic commands
- DBFS and workspace storage
- Databricks Repos (Git integration)
- Notebook versioning and collaboration

Clusters & Compute Management

- Cluster Types
    - Interactive clusters
    - Job clusters
    - Single-node clusters
- Cluster Configuration
    - Node types
    - Driver vs Worker nodes
    - Autoscaling
    - Auto-termination
- Cost Optimization
    - Spot vs On-Demand instances
    - Cluster policies
    - Right-sizing compute
- Advanced Configuration
    - Init scripts
    - Environment libraries
    - High concurrency clusters

Working with Data in Databricks

- Reading data from ADLS Gen2
- Supported file formats (CSV, JSON, Parquet, ORC)
- Mounting storage vs direct access
- Schema inference vs explicit schema
- Handling semi-structured data
- Writing optimized outputs

Spark & PySpark in Azure Databricks

- SparkSession in Databricks
- DataFrames and transformations
- Actions and lazy evaluation
- Joins and aggregations
- Window functions
- Performance considerations in Databricks
- Comparing Databricks vs open-source Spark

Delta Lake Fundamentals

- Delta Lake Architecture
    - Transaction log (_delta_log)
    - How Delta ensures ACID
    - Delta tables vs Parquet tables
- Delta Table Operations
    - Creating Delta tables
    - Managed vs external tables
    - Insert
    - update
    - delete

- Delta Features
    - Time Travel
    - Schema enforcement
    - Schema evolution
    - Constraints and data quality

Delta Lake Performance Optimization

- Optimize command
- File compaction
- Z-Ordering
- Data skipping
- Vacuum
- Retention policies
- Photon acceleration

Incremental Processing & SCD Patterns

- Merge Into deep dive
- Upsert patterns
- Delete patterns
- Deduplication strategies
- Slowly Changing Dimension Type 1
- Slowly Changing Dimension Type 2
- Handling late-arriving data
- Idempotent processing

Lakehouse Architecture Patterns

- Medallion Architecture
    - Bronze layer (raw ingestion)
    - Silver layer (cleansed & conformed)
    - Gold layer (business aggregates)

- Architectural Considerations
    - Schema evolution across layers
    - Data quality checks
    - Performance vs cost trade-offs
    - Analytics consumption patterns

Databricks Jobs & Orchestration

- Databricks Jobs overview
- Job scheduling
- Multi-task jobs
- Job dependencies
- Retry logic and alerts
- Parameterized notebooks
- Job vs workflow orchestration
- Integration with Azure Data Factory

Governance with Unity Catalog

- Unity Catalog overview
- Metastore concepts
- Catalogs, schemas, tables
- Data lineage
- Fine-grained access control
- Row-level security
- Column-level security
- Audit logging

Security & Secret Management

- Databricks security model
- Secret scopes
- Azure Key Vault-backed secrets
- Managed Identity
- Network security (private endpoints)
- Encryption at rest and in transit

CI/CD & DevOps for Databricks

- Git integration with Repos
- Branching strategies
- Notebook promotion
- Databricks CLI
- Infrastructure as Code (ARM / Terraform)
- Environment separation (Dev/Test/Prod)

Monitoring, Troubleshooting & Optimization

- Spark UI in Databricks
- Job monitoring
- Cluster logs
- Performance bottleneck analysis
- Memory and shuffle optimization
- Cost monitoring strategies

Learning Journey & Certification Path

Understand Cloud & Data Fundamentals

- Complete AZ-900: Microsoft Azure Fundamentals to grasp basic
- cloud concepts. Proceed to DP-900: Microsoft Azure Data Fundamentals to learn about core data services.

Dive into Data Engineering

- Enroll in the DP-203: Data Engineering on Microsoft Azure
- course. Gain hands -on experience with Azure services like Data Factory, Synapse Analytics, and Databricks.

Practice and Prepare

- Utilize Microsoft Learn's self-paced modules and labs.
- Engage in practice assessments to test your knowledge.

Get Certified

- Schedule and pass the DP-203 certification exam to earn the Azure Data Engineer Associate credential.

Advance Your Career

- Leverage your certification to pursue roles such as Data Engineer, Data Analyst, or Solutions Architect.

+91 8019953358 Info@cloudupskills.com www.cloudupskills.com

About Cloud Upskill:

Cloud Upskill is one of the leading training institutes in Hyderabad, offering certified, job-oriented training with live projects, workshops, and hands-on practice. With over 10+ years of experience, the institute provides online training delivered by certified, industry-expert trainers.

High Industry Demand
Our courses are designed with current market requirements, so our students stay in demand across top companies.

100% Student Satisfaction
Our learners consistently rate our training 5-star for teaching quality, support, and real-world applicability.

Expert Trainers & Practical Learning
Every program is led by experienced professionals and includes hands-on projects, assignments, and mentoring to build real skills.

Proven Track Record
Delivering industry-focused training and successful careers for nearly 10 years with strong placement outcomes.

Key Highlights:

1. Free Tablet
2. Lifetime Recordings
3. Course Materials
4. Weekly Assessment Tests
5. Mock Interviews
6. Resume Building Support
7. 100% Placement Assistance
8. Hands-on Real Projects
9. Course Completion Certificate

+91 8019953358 Info@cloudupskills.com www.cloudupskills.com

Cloud Upskill

Contact Us :

Reach out to our dedicated team for any inquiries, assistance, or information you need.

info@cloudupskills@gmail.com

Prestige Sky Tech, Financial District, Nanakramguda, Hyderabad, Telangana 500032

www.cloudupskills.com

+91 - 8019953358