Cloud-Based ETL Pipeline with AWS and Python

Set Up Amazon S3 Buckets for ETL Storage

Objective: Design a cloud-based data lake to store streaming sales transaction data.

Actions Taken:

- Created an S3 bucket named etl-version1.
- Structured the bucket with prefixes (raw/, athena-results/) to simulate raw zone and result zone.
- Managed permissions and IAM policies for secure access from scripts and tools (Athena, Tableau).
- Validated configuration by inspecting uploaded objects in the AWS Console.

Real-World Insight:

Companies use Amazon S3 as the central data lake to stage incoming data for analytics pipelines.

Built a Real-Time ETL Pipeline Using Python and Faker

Objective: Simulate a real-time ingestion system streaming sales data to the cloud.

Tools Used: Python, Pandas, Faker, Boto3, AWS CLI, S3

Step-by-Step Process:

Extract:

- Used Faker to generate synthetic fields: TransactionID, Date, Product, Price, Quantity, Customer.

Transform:

- Added Revenue column (Price × Quantity).

- Cleaned column names with Pandas.

Load:

- Saved each transaction as a .csv file.

- Uploaded to S3 folder etl-version1/raw/ using boto3.

- Automated with a while loop running every 5 seconds.

Output:

- Dozens of timestamped .csv files simulating continuous data ingestion.

- Ideal for micro-batch analytics in real-time platforms.

Real-World Use Cases:

- E-commerce: Order logs.

- Fintech: Payment transactions.

- Retail: Point-of-sale data.

- IoT: Sensor telemetry from smart devices.

Bonus Insights:

- These pipelines closely mimic how tools like Apache Kafka, AWS Kinesis, and Flink stream data.

- Automating uploads simulates event-driven data pipelines feeding dashboards and models.