Airlines Real-Time ETL Pipeline

Problem:

Airline companies receive large volumes of flight-related data daily, including delays, schedules, and airport information. Manually processing this data for analysis is inefficient and error-prone. A scalable, automated solution is needed to:

- Handle daily flight data ingestion,
- Join it with static airport information,
- Store it in a structured format for reporting and insights.

Dataset:

The project uses two datasets in csv format stored in Amazon S3:

- **Airports** : Contains airport codes, names, cities, and states. Static and rarely changes.
- **Flights** : Contains daily flight records with carrier, airport codes, departure and arrival delays. Arrives in a date-partitioned structure.

Tools:

- **Amazon S3** – Stores both static (dimension) and dynamic (fact) datasets.
- **AWS Glue Crawler** – Crawls S3 data to infer schema and update the Data Catalog.
- **AWS Glue Job** – Performs data transformations and joins.
- **Amazon EventBridge** – Triggers the ETL process when new flight data lands in S3.
- **AWS Step Functions** – Orchestrates the flow: crawler → wait → Glue job.
- **Amazon Redshift** – Serves as the centralized data warehouse for analytics.
- **Amazon SNS** – Sends notifications on ETL job success or failure.
- **AWS IAM** – Manages access and roles for Glue, Redshift, and other services.

Implementation:

Step 1 – Create S3 Bucket and Upload Source Files

- Navigate to the Amazon S3 console.
- Choose General Purpose, Click Create bucket and name the bucket:
  nandu-s3-airline
- Inside the bucket, create two folders:
  - airline_dim/
  - flights_data/
- Upload the respective .csv files into each folder:
  - airports.csv → airline_dim/
  - flights.csv → flights_data/

Step 2 - Create Amazon Redshift Serverless Cluster

- Navigate to the Amazon Redshift console.
- Select "Redshift Serverless" and click Create.
- Create a new Namespace named: namespace-1.
- Within this namespace, create a Workgroup named: workgroup-1.
- After creation, click "Query data" to open the Redshift Query Editor.
- Assign admin credentials and set the default database name as: dev.

Step 3 – Load Dimension Data into Redshift

- Create a new schema:
  CREATE SCHEMA airlines;
- Create the dimension table for airport details:
  ```
  CREATE TABLE airlines.airports_dim (
     airport_id BIGINT,
     city VARCHAR(100),
     state VARCHAR(100),
     name VARCHAR(200)
  );
  ```
- Load the data from S3 into the Redshift table using the COPY command:
  a. The S3 URI can be found in the Properties tab of the uploaded airports.csv file in S3.
  b. The IAM role used here (AmazonRedshift-CommandsAccessRole-20250714T193240) must be assigned to the Redshift **namespace.**
  ```
  COPY airlines.airports_dim
  FROM 's3://nandu-s3-airline/airline_dim/airports.csv'
  IAM_ROLE 'arn:aws:iam::196219932948:role/service-role/AmazonRedshift-CommandsAccessRole-20250714T193240'
  DELIMITER ','
  IGNOREHEADER 1
  REGION 'us-east-1';
  ```
- Verify loaded data:
  SELECT * FROM airlines.airports_dim;
- Create an empty fact table for storing processed flight data:
  ```
  CREATE TABLE airlines.daily_flights_fact (
     carrier VARCHAR(10),
     dep_airport VARCHAR(200),
     arr_airport VARCHAR(200),
     dep_city VARCHAR(100),
     arr_city VARCHAR(100),
     dep_state VARCHAR(100),
  ```

```
    arr_state VARCHAR(100),
    dep_delay BIGINT,
    arr_delay BIGINT
);
```

- Verify table creation:
  SELECT * FROM airlines.daily_flights_fact;

Step 4 - Configure VPC and Security Group Settings

- Go to Amazon Redshift → Workgroup-1 → Data Access
- Security Groups → [Your SG ID]
  - Click Edit Inbound Rules → Add:
  - Type: Redshift
  - Protocol: TCP
  - Port Range: 5439
  - Source: Custom → Select the subnet CIDR from Redshift VPC

  This allows secure Redshift access within your VPC network.

- Create S3 Gateway VPC Endpoint
  - Go to VPC → Endpoints → Create Endpoint
  - Select service: com.amazonaws.<region>.s3
  - Choose your VPC ID
  - Set Endpoint Type: Gateway
  - Select Route Tables associated with Redshift's subnet
  - Click Create

  This allows Redshift to access S3 internally, avoiding public internet routes.

Step 5 - Create Database in AWS Glue

- Navigate to the AWS Glue Console → Databases section.
- Click "Add database".
- Enter the Database name: "airlines_datamart" and click Create database.

Step 6 - Create IAM Role for Glue, S3, and Redshift Access

- Go to the IAM Console → Roles and click Create role.
- Choose Trusted entity type:

    - Select AWS service
    - Use case: Glue
- Attach the following AWS managed policies:
    - AmazonS3FullAccess
    - AmazonRedshiftFullAccess
    - AWSGlueServiceRole
    - AWSGlueConsoleFullAccess
    - SecretsManagerReadWrite *(optional – for future use with credentials)*
    - AWSKeyManagementServicePowerUser *(optional – for KMS encrypted buckets)*
- Name the role like "aws_connector_s3_redshift_glue_role" and Click Create Role.

Step 7 - Create JDBC Connection in AWS Glue

- Navigate to AWS Glue Console → Connections → Create connection.
- Choose Connection Type: Select JDBC
- In the JDBC configuration:
- Paste the JDBC URL (found in Redshift → Workgroup)
- Use your Redshift credentials (username & password)
- Select the same VPC, Subnet, and Security Group used by your Redshift workgroup
- Click Create to finish the connection setup
- After creation, click Actions → Test connection
    - Assign the IAM role: "aws_connector_s3_redshift_glue_role"
    - Confirm the connection is successful

Step 8 - Create Crawlers in AWS Glue

1. Create Crawler: "airlines_dim_crawler"
    - Go to AWS Glue Console → Crawlers → Create Crawler
    - Name: airlines_dim_crawler
    - Click "Add a JDBC data source"
    - Data source: Choose JDBC
    - Select the previously created JDBC connection
    - Include path: dev/airlines/airports_dim
    - IAM Role: Choose "aws_connector_s3_redshift_glue_role"(created earlier)
    - Target database: Select "airlines_datamart"
    - Click Create Crawler
    - Repeat the same steps for crawler 2 and 3
2. Crawler 2
    - Name: "airlines_fact_crawler"
    - JDBC path: "dev/airlines/daily_flights_fact"

3. Crawler 3
    - Name: “flights_data_crawler”
    - Data source: S3
    - Include path: “s3://nandu-s3-airline/flights_data/”

Step 9 - Run Crawlers to Create Tables in Glue Catalog

- Go to AWS Glue → Crawlers and run each of the following:
    - airlines_dim_crawler
    - airlines_fact_crawler
    - flights_data_crawler
- After each crawler completes successfully, they create schema-based tables in the airlines_data_mart Glue database.Tables like:
    - dev_airlines_airports_dim
    - dev_airlines_daily_flights_fact
    - flights_data

Step 10 - Create IAM Role for Redshift and Assign It

- Ensure an IAM role named “temp_role” is created with the following permissions:
    - AmazonRedshiftFullAccess
    - AmazonS3FullAccess
    - Custom policy: S3AccessForGlue
- This role allows Redshift to access AWS services like S3 and Glue.
- Go to Amazon Redshift → Namespace → [namespace-1] → Edit
    - Under IAM Roles, click Add IAM Role
    - Select the role: “arn:aws:iam::196219932948:role/temp_role”
    - Save the changes.

Step 11 - Temporary S3 Structure

- Bucket: “redshift-temp-bucket123"
- Path: temp_data/
- Subfolders inside:
    - airline_dim/
    - airline_fact/

Step 12 - Create Glue Job Using Visual ETL

- Go to AWS Glue Studio → Visual ETL → Jobs and click Create Job.
- Set Job name: "flight_data_ingestion"

1. In the source node Choose Glue Data Catalog.
    a. Set Name: flight_data
    b. Database: airlines_datamart
    c. Table: flights_data

   Choose the IAM role: "aws_connector_s3_redshift_glue_role" then Start a session
2. Drag a Filter node onto the canvas.
    a. Set Name: filter
    b. Set Parent Node: flight_data
    c. Configure the Filter Condition:
        - Key: depdelay
        - Condition: >=
        - Value: 60
3. Add a new Source node Type: Glue Data Catalog
    a. Name: airport_dim
    b. Database: airlines_data_mart
    c. Table: dev_airlines_airports_dim
    d. Temporary directory in S3: s3://redshift-temp-bucket123/temp_data/airline_dim/
    e. IAM Role: temp_role (arn:aws:iam::196219932948:role/temp_role)
4. Drag Join node onto canvas.
    a. Name: join_dep
    b. Node parents: filter, airport_dim
    c. Join type: Left Join
    d. Join condition: originairportid = airport_id
5. Drag Change schema onto canvas.
    a. Name: modify_dep
    b. Node Parents: join_dep
    c. remove airport_id and originairportid and then modify names and datatype according to redshift "daily_flights_fact" table
6. Drag Join node onto canvas.
    a. Name: join_arr
    b. Node parents: modify_dep, airport_dim
    c. Join type: Left Join
    d. Join condition: destairportid = airport_id
7. Drag Change schema onto canvas.
    a. Name: modify_arr
    b. Node Parents: join_arr
    c. remove airport_id and destairportid and then modify names according to redshift "daily_flights_fact" table

8. Drag Target Glue Data Catalog.
    a. Set Name: target_redshift
    b. Node Parent: modify_arr
    c. Database: airlines_datamart
    d. Table: dev_airlines_airports_fact
    e. Temporary directory in S3: s3://redshift-temp-bucket123/temp_data/airline_fact/
    f. IAM Role: temp_role (arn:aws:iam::196219932948:role/temp_role)
9. In the Job Details tab of your Glue job:
    a. Set Requested number of workers: 2
    b. Enable Job Bookmark to allow incremental loads
    c. Add job parameter: --JOB_NAME and beside tab job name "flight_data_ingestion".
10. Since Visual Glue Job doesn't capture temp path for airport_dim,
    a. redshift_tmp_dir = "s3://redshift-temp-bucket123/temp_data/airline_dim/"
    b. manually add this in airports_dim script between table_name and transformation_ctx.
11. Save it.

Step 13 - Enable EventBridge for S3 Bucket

- Go to S3 → nandu-s3-airline bucket → Properties
- Scroll to Event notifications
- Enable EventBridge by toggling "On"

Step 14 - Create SNS topic

1. Go to Amazon SNS → Topics → Create topic
2. Choose Standard as the type
3. Set Name: snstopic1
4. Click Create topic

Step 14 - Create Step Function for Orchestration

1. Go to AWS Step Functions → Create State Machine and Name it: airline_data_pipeline
2. Add Start Crawler
    a. Name: StartCrawler
    b. Arguments: { "Name": "flights_data_crawler" }
    c. Next state: GetCrawler
3. Add Get Crawler
    a. Name: GetCrawler
    b. Arguments: { "Name": "flights_data_crawler" }
    c. Next state: is_running?

4. Add Choice
   a. Name: is_running?
   b. Rule 1:
      a. Expression: $.states.input.Crawler.State
      - Operator: matches String
      - Value: RUNNING
      - Next state: wait for running

   Default: Next state: StartGlueJob

5. Add Wait State
   a. Name: wait for running
   b. Wait type: Fixed interval
   c. Seconds: 5
   d. Next state: GetCrawler
6. Add Start Glue Job
   a. Name: StartGlueJob
   b. Arguments: { "JobName": "flight_data_ingestion" }
   c. Wait for task to complete: enable
   d. Catch error:
      a. Errors: States.TaskFailed
      b. Fallback state: fail_sns

   e. Next state: is_succeeded?

7. Add Choice
   a. Name: is_succeeded?
   b. Rule 1:
      a. Expression: $states.input.JobRunState
      b. Operator: matches string
      c. Value: SUCCEEDED
      d. Next state: success_sns

   c. Default: Next state: fail_sns

8. Add SNS Publish
   a. Name: success_sns
   b. Assign SNS topic
   c. Message: "success"
   d. Next state: End

9. Add SNS Publish – Fail
    a. Name: fail_sns
    b. Assign SNS topic
    c. Message: "failed"
    d. Next state: End
10. Save it

Step 16 - Create EventBridge Rule to Trigger Step Function

- Go to Amazon EventBridge → Rules → Create rule
    - Name: flight_data_ingestion_rule
    - Event bus: default
- Choose Rule with an event pattern
    - Event source: AWS services
    - Service: Simple Storage Service (S3)
    - Event type: Amazon S3 Event Notification
- Specify event:
    - Select: Object Created
    - Bucket name: nandu-s3-airline
- Edit event pattern (JSON) to match Hive-style S3 structure if needed.
- Set Target:
    - Type: Step Function state machine
    - Choose: airline_data_pipeline
- Click Create and Enable the rule.

Step 17 - Create VPC Endpoints for Orchestration

- CloudWatch Endpoint
    - Go to VPC → Endpoints → Create Endpoint
    - Name: cloudwatch-endpoint
    - Service: com.amazonaws.<region>.monitoring
    - Subnets: same as used by Glue jobs
    - Security group and vpc same as Redshift
    - Click Create
- VPC-Glue Endpoint
    - Name: glue-endpoint
    - Service: com.amazonaws.<region>.glue
    - Repeat same steps as above

Step 18 – Automated ETL Triggered on New File Upload

- When a new file is uploaded to the daily_flights folder in S3,

- EventBridge Rule triggers the Step Function,
- Which runs the Glue Crawler → Glue Job,
- The job processes and loads data into Redshift automatically.