Project: Animal Intake Data Management and Analysis

Dataset Overview:

The dataset contains the following columns:

Column Name	Data Type	Description
animal_id	String	Unique identifier for each animal
name	String	Name of the animal
datetime	Timestamp	Date and time when the animal was brought in
monthyear	Date	Month and year (e.g., "202310")
found_location	String	Location where the animal was found
intake_type	String	Type of intake (e.g., "Stray", "Owner Surrender")
intake_condition	String	Condition of the animal upon intake (e.g., "Healthy", "Injured")
animal_type	String	Type of animal (e.g., "Dog", "Cat")
sex_upon_intake	String	Sex of the animal upon intake (e.g., "Male", "Female")
age_upon_intake	Integer	Age of the animal upon intake (in years)
breed	String	Breed of the animal
color	String	Color of the animal

Phase 1: Data Collection, Loading, and Exploration

Task 1: Upload Dataset to S3

Objective: Upload the raw dataset to an AWS S3 bucket.
Tools/Services: S3, Python (`boto3`), EC2
Steps:

a) Create an S3 bucket in AWS.
b) Upload the dataset (CSV or JSON) into the S3 bucket using `boto3` in Python.
c) Set proper S3 bucket permissions using IAM roles to restrict access.

Task 2: Data Exploration using Python

Objective:Perform basic exploratory data analysis (EDA) on the dataset.
Tools/Services:Python, Pandas
Steps:

a) Load the dataset into a Pandas DataFrame.
b) Check for missing values and handle them (impute or drop).
c) Explore summary statistics and visualize the data using `matplotlib` or `seaborn`.
d) Clean and preprocess the data as needed (e.g., handle null values, format `datetime`).

Phase 2: Data Storage & Transformation

Task 3: Data Cleaning and Transformation Using SQL (RDS)

Objective:Clean and preprocess the data using SQL queries in Amazon RDS

Tools/Services: Amazon RDS, SQL, Python

Steps:

a) Create a new database and a table to store the data in RDS.

Use SQL queries to:

b) Clean the data (e.g., remove duplicates, handle null values).
c) Normalize the `datetime` column and extract the `monthyear`.
d) Create new columns like `age_category` (e.g., "Puppy/Kitten", "Adult", "Senior").
e) Insert the cleaned data into the RDS database.

Task 4: Data Transformation Using AWS Glue (ETL Process)

Objective: Use AWS Glue to perform ETL operations on the data stored in S3 and move the transformed data to RDS.

Tools/Services:AWS Glue, S3, RDS

Steps:

a) Set up a Glue crawler to crawl the dataset in the S3 bucket and create a catalog table.
b) Create a Glue ETL job to:
c) Extract the data from S3.
d) Perform transformations such as converting `datetime` to a standard format, categorizing `age_upon_intake`, etc.
e) Load the transformed data into RDS.
f) Test the ETL job and ensure that data flows correctly into RDS.

Phase 3: Data Processing & Analysis with PySpark

Task 5: Data Processing Using PySpark on EC2

Objective: Process the data using PySpark on an EC2 instance.

Tools/Services:PySpark, EC2

Steps:

a) Launch an EC2 instance (e.g., t2.medium) and install PySpark.
b) Read the dataset from S3 using PySpark’s `spark.read` API.
c) Perform data transformations using PySpark DataFrames (e.g., filter data, aggregate statistics).
d) Save the processed data back to S3 in Parquet or JSON format for optimized querying.

Task 6: Data Aggregation and Analysis

Objective:Perform data aggregation and analysis using PySpark.

Tools/Services:PySpark, S3

Steps:

a) Aggregate the data based on intake type, monthyear, and animal type (e.g., count animals per month, intake type, and condition).
b) Perform additional analysis such as identifying trends in animal intakes over time or by location.
c) Export the aggregated results to S3 in a readable format (CSV or Parquet).

Phase 4: Automating the Data Pipeline Using AWS Lambda & EventBridge

Task 7: Automating Data Ingestion with AWS Lambda

Objective:Create a serverless function to automate the process of cleaning and processing new data as it arrives.

Tools/Services: AWS Lambda, S3, Python

Steps:

a) Create a Lambda function that triggers every time a new dataset is uploaded to S3.
   In the Lambda function, use Python and the `boto3` SDK to:
b) Trigger an AWS Glue job or RDS data import.
c) Process new data (e.g., update tables, perform necessary transformations).
d) Ensure the Lambda function is set with appropriate IAM permissions to access S3 and Glue.

Task 8: Set Up EventDriven Architecture with AWS EventBridge

Objective: Create an eventdriven architecture to automate data processing.

Tools/Services: AWS EventBridge, Lambda

Steps:

a) Set up an EventBridge rule that triggers on certain events, such as when new data is uploaded to S3 or when an EC2 instance finishes processing.
b) Configure the EventBridge rule to invoke the Lambda function created in Task 7 to process the new data.

Phase 5: Monitoring & Alerts with CloudWatch, SNS, and SES

Task 9: Monitoring with AWS CloudWatch

Objective: Monitor the AWS services (Lambda, Glue, etc.) using CloudWatch.

Tools/Services: AWS CloudWatch, Lambda, Glue

Steps:

a) Set up CloudWatch metrics and logs for monitoring Lambda function executions and Glue ETL jobs.
b) Create CloudWatch Alarms for failures or execution times that exceed thresholds.

Task 10: Notifications Using SNS and SES

Objective: Set up notifications for failures and job completions.

Tools/Services: AWS SNS, SES

Steps:

a) Configure AWS Simple Notification Service (SNS) to send alerts via email (using SES) when specific events occur (e.g., data processing failure, job completion).

b) Set up an SNS topic and subscribe the team's email addresses for notifications.

Phase 6: Reporting and Visualization

Task 11: Data Visualization Using Python

Objective: Visualize the processed data to gain insights.

Tools/Services: Python, Matplotlib, Seaborn.

Steps:

a) Retrieve the aggregated data from S3 or RDS and use Python's visualization libraries to create charts.

b) Create visualizations like:

c) Monthly animal intake trends.

d) Distribution of intake types and conditions.

e) Breakdown of intake data by breed or location.

Task 12: Build a Data Dashboard

Objective: Create an interactive dashboard for stakeholders.

Tools/Services: Python, Matplotlib,Seaborn

Steps:

a) Create a simple dashboard that displays key metrics:

b) Number of animals by `intake_type` and `monthyear`.

c) Average age by `animal_type`.

d) Intake distribution by `found_location`.

e) Host the dashboard on an EC2 instance or use AWS Amplify for deployment.

Deliverables:

i. Code Repository: A GitHub repository containing all the Python scripts, SQL queries, and PySpark code.

ii. AWS Setup Documentation: Documentation for setting up and configuring the AWS infrastructure.

iii. SQL Scripts and Glue Jobs: SQL scripts used in RDS and Glue jobs for ETL processing.

iv. Lambda Functions: Code for the AWS Lambda functions and eventdriven setup.

v. Visualizations: Data visualizations and interactive dashboard hosted on EC2 or AWS Amplify.