Automated Data Quality Assessment and Monitoring System

1. Solution Summary

This project demonstrates a complete end-to-end Data Quality (DQ) framework -
from data profiling, cleaning, and metric computation to dashboarding and ongoing monitoring
The goal was to ensure data reliability, completeness, and accuracy while simulating a real-

Although the assignment mentioned SQL + Dashboard integration, all tasks were implemented us
A conceptual SQL schema was also added in Markdown to demonstrate how the same metrics could

2 Environment & Tools

- Language: Python 3.x

- Environment: Jupyter Notebook

- Libraries: pandas, numpy, plotly, dash, os, datetime, json, smtplib

- Output Files:

    1. news_events_cleaned.csv

    2. data_profile.csv

    3. data_quality_summary.csv

    4. data_quality_log.csv

    5. data_cleaning_log.csv

3 Data Cleaning & Preprocessing

Goal: Ensure data is complete, consistent, valid, and formatted correctly.

Steps:

1. Data Loading:

   Load raw dataset using pandas.

   Combined multiple JSONL files into one and converted it into a dataframe.

2. Standardization:

   - Trim whitespace

   - Convert dates to consistent formats

   - Normalize categorical columns (e.g., lowercase)

- Remove duplicates

3. Missing Value Treatment:

   - Drop high-missing columns

   - Fill moderate-missing columns using median/mode

4. Data Type Conversion:

   Ensure columns like found_at, id, and count have appropriate data types.

5. Output:

   Save the cleaned dataset as news_events_cleaned.csv.

4 Data Profiling (Exploratory Data Analysis)

Goal: Generate statistical summaries and understand data characteristics.

Steps:

1. Implemented a safe_nunique() function to compute unique values even for JSON-like ob

2. Computed:

   - Data types (dtype)

   - Missing value percentage (missing_%)

   - Number of unique values (unique_values)

3. Saved the profile as data_profile.csv for record-keeping.

Visual Analysis:

Plotted missing value distribution and datatype frequency using Plotly bar charts for bette

5 Data Quality Metrics Calculation

****Metrics Computed:****

Metric Description:

```
Completeness (%)    -->     % of non-missing values
Uniqueness (%)      -->     % of unique records
Consistency (%)     -->     % of consistent values across key fields
Validity (%)        -->     % of records matching valid formats
Timeliness (%)      -->     % of recent records
Accuracy (%)        -->     % of logically correct records
```

****Final Output Example:****

```
Metric  Value
Completeness (%)    56.28
Uniqueness (%)      98.73
Consistency (%)     100.00
Validity (%)        100.00
Timeliness (%)      17.71
Accuracy (%)        100.00
Invalid_Records     8,955,872
```

6 Data Quality Dashboard

Context:

```
The assignment required SQL + Dashboard integration.
However, since SQL and BI tool access was unavailable, the entire workflow was executed in P
A conceptual SQL schema (in Markdown) was added to show how results could be stored in a rel
```

Dashboard Components:

```
1. Overall metric summary cards

2. Distribution charts for missing values, data types, and outliers

3. Interactive Plotly visuals for profiling and trend analysis
```

Files:

```
    data_quality_dashboard.py
```

7 Conceptual SQL Schema

```
-- Table: data_quality_metrics
CREATE TABLE data_quality_metrics (
run_id SERIAL PRIMARY KEY,
timestamp TIMESTAMP,
completeness NUMERIC,
uniqueness NUMERIC,
consistency NUMERIC,
validity NUMERIC,
timeliness NUMERIC,
accuracy NUMERIC,
invalid_records INT
);

-- Table: data_profile
CREATE TABLE data_profile (
column_name VARCHAR(255),
dtype VARCHAR(50),
missing_percent NUMERIC,
unique_values INT
```

```
);
```

8 Ongoing Data Quality Monitoring

Goal: Automate continuous monitoring and alerting for future data loads.

Approach:

```
1. Implemented a Python script (data_quality_monitor.py) to:

2. Compute key metrics (Completeness, Duplicate Rate, Timeliness)

3. Log results into a CSV (data_quality_log.csv)

4. Generate alerts when thresholds are breached
```

Example Log Entry:

```
timestamp   rows    completeness_%  duplicate_rate_%    timeliness_%
2025-11-11 09:00    50000   92.35   1.24    78.91
```

Alert Thresholds:

```
THRESHOLDS = {
"completeness_%": 90,
"duplicate_rate_%": 5
}
```

Alert Output Example:

```
 ALERT: Completeness dropped below 90%
```

9 Summary of Files

```
File Name   Description

01_data_cleaning_and_profiling.ipynb    -->     Cleaning, transformation and profiling steps
02_data_quality_dashboard.ipynb     -->     Dashboard visualization
03_data_quality_monitoring.ipynb    -->     Automated monitoring simulation
data_profile.csv    -->     Profiling summary
data_quality_log.csv    -->     Monitoring logs
data_cleaning_log.csv   -->     Ceaning logs
data_quality_summary.csv    -->     Summary of Data Quality
news_events_cleaned.csv     -->     Cleaned data
README.md   -->     High-level project summary
```