Automate Data Quality using AWS Glue DQ

Author: Praveen Kumar H B

August 2024

# Introduction

Data quality refers to the condition of data based on factors such as accuracy, completeness, reliability, relevance, and timeliness. High-quality data is essential for effective data analysis, decision-making, and operations. Generally, Data would be loaded to BigQuery as a Final stage after Extract, Data quality, transformations. But now the preference changed to ELT Process over ETL, it is expected to load the Qualified data before processing.

# AWS Glue Data Quality

AWS Glue Data Quality is a feature within AWS Glue, a fully managed extract, transform, and load (ETL) service provided by Amazon Web Services (AWS) this feature enables users to measure and monitor the quality of their data within their data lakes, data warehouses, and data pipelines. It helps ensure that the data being processed and analyzed is accurate, complete, and reliable.

## Key Features

1. **Data Quality Rules:** Users can define data quality rules that specify the criteria data must meet. These rules can be based on various data quality dimensions, such as accuracy, completeness, consistency, timeliness, and validity.
2. **Automated Validation:** AWS Glue Data Quality can automatically validate data against the defined rules. This helps identify data quality issues early in the data processing pipeline.
3. **Integration with AWS Glue ETL Jobs:** Data quality checks can be integrated into AWS Glue ETL jobs, allowing users to validate data quality as part of their data transformation processes.
4. **Metrics and Reporting:** The feature provides metrics and reports on data quality, enabling users to monitor the quality of their data over time and identify trends or patterns in data quality issues.
5. **Alerting and Notifications:** Users can set up alerts and notifications to be informed of data quality issues as they occur. This allows for prompt action to address any problems.
6. **Customizable Rules:** Users can create custom data quality rules tailored to their specific data quality requirements. These rules can be as simple or complex as needed.

## Benefits

* **Improved Data Integrity:** By ensuring that data meets predefined quality standards, AWS Glue Data Quality helps maintain high data integrity.
* **Enhanced Decision Making:** Reliable, high-quality data supports better analytics and decision-making processes.
* **Automated Quality Control:** Automating data quality checks reduces the need for manual intervention and minimizes the risk of human error.
* **Scalability:** AWS Glue Data Quality is designed to scale with your data, making it suitable for large and growing datasets.

## Set Up DQ

1. **Define Data Quality Rules:** Specify the rules and criteria for data quality based on your requirements.
2. **Integrate with ETL Jobs:** Incorporate data quality checks into your AWS Glue ETL jobs.
3. **Monitor and Report:** Use AWS Glue Data Quality metrics and reports to monitor data quality and identify issues.
4. **Set Up Alerts:** Configure alerts to notify stakeholders of data quality issues in real-time.

## Entry Points

### Data quality for the AWS Glue Data Catalog

AWS Glue Data Quality evaluates objects that are stored in the AWS Glue Data Catalog It offers non-coders an easy way to set up data quality rules. These personas include data stewards and business analysts.

### Data quality for AWS Glue ETL jobs

AWS Glue Data Quality for AWS Glue ETL jobs lets you perform proactive data quality tasks. Proactive tasks help you identify and filter out bad data before you load a data set into your data lake.

## Example Use Cases

* **Data Lake Quality Assurance:** Ensuring that data ingested into a data lake meets quality standards before it is made available for analysis.
* **ETL Pipeline Validation:** Validating data at various stages of an ETL pipeline to catch and correct issues before data is loaded into a data warehouse.
* **Compliance and Governance:** Monitoring data quality to ensure compliance with regulatory requirements and internal data governance policies.

# How AWS Glue ETL jobs works

A diagram of a software flow

Description automatically generated

## Creating an ETL job – Example

1. Below code to initialize spark context.

import sys

from awsglue.transforms import \*

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job

sc = SparkContext.getOrCreate()

glueContext = GlueContext(sc)

spark = glueContext.spark\_session

job = Job(glueContext)

1. Import the EvaluateDataQuality class that evaluatesAWS Glue Data Quality.

from awsgluedq.transforms import EvaluateDataQuality

1. Read in the source data by using the .csv file that's stored in the public Amazon S3 bucket.

medicare = spark.read.format("csv").option("header", "true").option("inferSchema", "true").load('s3://awsglue-datasets/examples/medicare/Medicare\_Hospital\_Provider.csv')

medicare.printSchema()

1. Convert the data to an AWS Glue DynamicFrame.

from awsglue.dynamicframe import DynamicFrame

medicare\_dyf = DynamicFrame.fromDF(medicare,glueContext,"medicare\_dyf")

1. Create the ruleset using Data Quality Definition Language (DQDL).

EvaluateDataQuality\_ruleset = """

Rules = [

ColumnExists "Provider Id",

IsComplete "Provider Id",

ColumnValues " Total Discharges " > 15

]

]

"""

1. Define the run in JSON Config file.

{

"dataset\_dictionary": {

"dataset\_name1": {

"file\_size\_threshold\_in\_mb": 30,

"dataset\_rule\_dictionary": {

"dataset\_name1\_rule1": {

"rule\_enabled":true,

"rule\_name": "Indicates a material needs to be mapped in enrich",

"rule1": "CustomSql \"select \* from primary where (not ((column1 is not null) and (column2 is null)))\"",

"filter\_records": false

},

"dataset\_name1\_rule2": {

"rule\_enabled":true,

"rule\_name": "Indicates a material needs to be mapped in enrich",

"rule2": "ColumnExists \"Provider Id\"",

"filter\_records": true

},

"dataset\_name1\_rule2": {

"rule\_enabled":true,

"rule\_name": "Indicates a material needs to be mapped in enrich",

"rule2": "IsComplete \"Provider Id",\"",

"filter\_records": true

},

"dataset\_name1\_rule2": {

"rule\_enabled":true,

"rule\_name": "Indicates a material needs to be mapped in enrich",

"rule2": "ColumnValues \" Total Discharges \" > 15",

"filter\_records": true

}

}

}

}

1. Validate the dataset against each rule defined in config file.

EvaluateDataQualityMultiframe = EvaluateDataQuality().process\_rows(

frame=medicare\_dyf,

ruleset=EvaluateDataQuality\_rule[‘rule1’],

publishing\_options={

"dataQualityEvaluationContext": "EvaluateDataQualityMultiframe",

"enableDataQualityCloudWatchMetrics": False,

"enableDataQualityResultsPublishing": False,

},

additional\_options={"performanceTuning.caching": "CACHE\_NOTHING"},

)

1. Review the results.

ruleOutcomes = SelectFromCollection.apply(

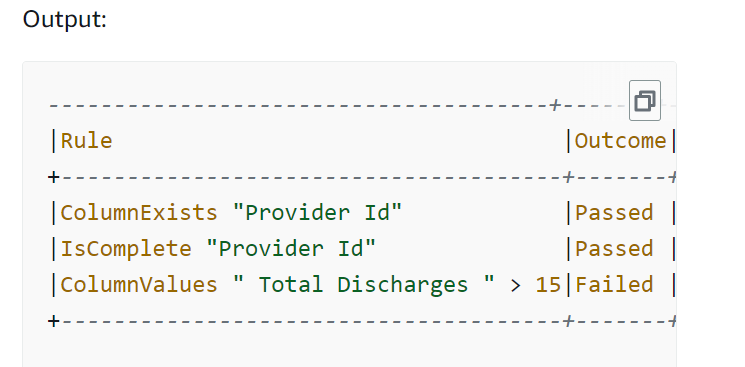
dfc=EvaluateDataQualityMultiframe,

key="ruleOutcomes",

transformation\_ctx="ruleOutcomes",

)

ruleOutcomes.toDF().show(truncate=False)



1. Filter passed rows and review the failed rows from the Data Quality row-level results.

rowLevelOutcomes = SelectFromCollection.apply(

dfc=EvaluateDataQualityMultiframe,

key="rowLevelOutcomes",

transformation\_ctx="rowLevelOutcomes",

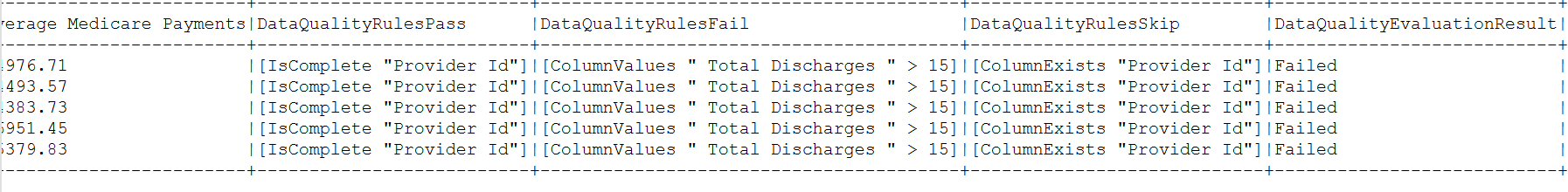
)

rowLevelOutcomes\_df = rowLevelOutcomes.toDF() # Convert Glue DynamicFrame to SparkSQL DataFrame

rowLevelOutcomes\_df\_passed = rowLevelOutcomes\_df.filter(rowLevelOutcomes\_df.DataQualityEvaluationResult == "Passed") # Filter only the Passed records.

rowLevelOutcomes\_df.filter(rowLevelOutcomes\_df.DataQualityEvaluationResult == "Failed").show(5, truncate=False) # Review the Failed records

Output:



**Note** that AWS Glue Data Quality added four new columns (DataQualityRulesPass, DataQualityRulesFail, DataQualityRulesSkip, and DataQualityEvaluationResult). This indicates the records that passed, the records that failed, rules skipped for row-level evaluation, and the overall row-level results.

1. Send email notification of failed rule.

#Send a email notification for failed rule attaching failed records.

Emailservice.Send(rowLevelOutcomes\_df\_failed)

# Summary

AWS Glue Data Quality is a powerful feature for ensuring the quality of data within AWS Glue environments, providing tools for defining, monitoring, and maintaining high data quality standards.

# Appendix

## DQDL

Data Quality Definition Language (DQDL) is a formal language used to define, enforce, and manage the quality of data within a system. It specifies rules, constraints, and policies to ensure that data meets certain standards and requirements. These rules can be related to various aspects of data quality, such as accuracy, completeness, consistency, timeliness, and validity.

### Key Concepts in DQDL

1. **Data Quality Rules:** These are specific conditions that data must satisfy. For example, a rule might state that all entries in a "date of birth" field must be valid dates.
2. **Constraints:** These are limitations or conditions placed on data to ensure its quality. Constraints might include range constraints (e.g., ages must be between 0 and 120), uniqueness constraints (e.g., no duplicate IDs), and mandatory fields (e.g., a phone number must be provided).
3. **Policies:** These are overarching guidelines or principles that govern how data quality is maintained. Policies might include procedures for data entry, data validation, and data correction.
4. **Validation and Enforcement:** DQDL includes mechanisms for checking whether data adheres to the defined quality rules and constraints. It can also enforce these rules by preventing invalid data from being entered into the system or by triggering corrective actions when violations are detected.

### Benefits of Using DQDL

* **Improved Data Integrity:** By defining and enforcing data quality rules, DQDL helps ensure that data remains accurate, consistent, and reliable.
* **Consistency Across Systems:** DQDL provides a standardized way to define data quality requirements, which can be applied uniformly across different systems and databases.
* **Automated Quality Control:** DQDL can automate the process of data validation and correction, reducing the need for manual checks and interventions.
* **Enhanced Decision Making:** High-quality data leads to better insights and more informed decision-making.

### CustomSQL

This rule type has been extended to support two use cases:

* Run a custom SQL statement against a dataset and checks the return value against a given expression.
* Run a custom SQL statement where you specify a column name in your SELECT statement against which you compare with some condition to get row-level results.

### Limitations

AWS Glue Data Quality service limits:

* You can have 2000 rules in a ruleset. If your rulesets are larger, we recommend splitting into multiple rulesets.
* The size of the ruleset is 65KB. If your rulesets are larger, we recommend splitting into multiple rulesets.

### Syntax

CustomSql *<SQL\_STATEMENT>* *<EXPRESSION>*

* **SQL\_STATEMENT** – A SQL statement that returns a single numeric value, surrounded by double quotes.
* **EXPRESSION** – An expression to run against the rule type response in order to produce a Boolean value. For more information, see [Expressions](https://docs.aws.amazon.com/glue/latest/dg/dqdl.html#dqdl-syntax-rule-expressions).

### Example of DQDL Usage

Consider a customer database where the following DQDL rules might be applied:

* **Rule 1:** All email addresses must match a standard email format.
* **Rule 2:** Customer IDs must be unique.
* **Rule 3:** The date of birth field must contain valid dates and cannot be left blank.
* **Rule 4:** The age of customers must be between 0 and 120 years.

## External References

* To learn more about DQDL and supported rule types, see [Data Quality Definition Language (DQDL) reference](https://docs.aws.amazon.com/glue/latest/dg/dqdl.html).
* [Data quality checks](https://docs.aws.amazon.com/prescriptive-guidance/latest/modern-data-centric-use-cases/data-quality-checks.html)