□□□

□Now

□Dream Small

□Life as an Album

□Thoughts from Work

□Links

□ A Lightweight Data Quality Framework

#Looker #DataQuality #DataEngineering

Monitor Table Health in Looker & Get Slack Alerts

□ Who Should Read This?

Anyone who:

- Cares about data quality
- Wants to trust the table and be confident to use the numbers from it

□ TL;DR

A lightweight framework to monitor data quality at the result level, not pipeline level.
It collects table statistics over time in BigQuery*, visualizes trends in Looker, and alerts anomalies in Slack.

> ***sample logic will be provided, which you can implement in other SQL based DB or pipelines.**

So you can answer:
“Is this Explore/dashboard still using healthy data?”

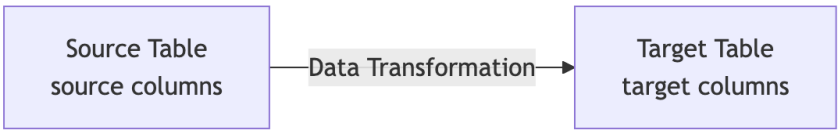
> **□ This NOT a framework about Data Job□ or Pipelines□, but about Result□ □, the confidence of usability.**

- We’re not watching job runtimes or DAG failures.
- We’re watching for something more subtle:

“The job succeeded, but the result looks… off.”

Example:
A daily job loaded zero rows due to an upstream issue. The job didn’t fail. No alert fired. But your dashboard now shows… nothing □

□ Table Statistics You Should Track between Source Table and Target Tables



- **source table**: could be a staging table directly read from API
- **target table**: could be the aggregation, business tables for reporting, dashboard, modeling which consumed by actual data persons like Data Analyst, Scientists et al.

These metrics give you a historical pulse on the health of your tables:

Metric	Why It Matters
Empty / Null Count	Missing values are a pain in time-series and ML. COALESCE(column, 'Other') can skew distributions.
Cardinality	Number of unique values (e.g. status types, channels). Reflects business dynamics.
Selectivity	Ratio of cardinality to total rows. Measures data uniqueness.
Density	Non-null values / total rows. Primary keys should be near 100%, i.e. amount valid records/rows you will get
Min/Max	Useful for numeric columns like transaction_amount, invoice_amount or invoice_date. Watch for variance.

□ Build & Deploy in BigQuery

the following code sample is a internal bigquery schedule tool called BQ runner in .yaml language, the core logic is this the .sql part with a little help from .jinja template to

Set up a YAML config like this:

```
type: query
table_description: |
{policy: { accessTier: BOARD }, description: '{{bq_runner_info}}'}
field_descriptions:
query_args:
source_table_full_name: gcp_project.gcp_dataset.table_to_be_monitored
source_table_type: source
source_column_names:
- column_name_key
- column_name_categorical
- column_name_numeric
- column_others_to_be_monitored
/* ignore below if you don't have a target table to compare - start */ target
target_table_full_name: gcp_project.gcp_dataset.table_to_be_compared
target_table_type:
target target_column_name:
- column_name_key
- column_name_categorical
- column_name_numeric
- column_others_to_be_compared
/* ignore below if you don't have a target table to compare - end */

query: |
{% for col in query_args.source_column_names -%}

UNION ALL

SELECT
"{{ partition. format( 'YYYY-MM-DD') }}" AS partition_date
, {{ query_args.source_table_full_name }}" AS table_full_name
, 'source' AS table_type
,  COUNT(*) AS num_rows
, "{{ col }}" AS column_name
, SAFE_CAST(MIN(IF({{ col }}="", NULL, {{ col }})) AS STRING) AS min_value
```

```
, SAFE_CAST(MAX(IF({{ col }}='', NULL, {{ col }})) AS STRING) AS max_value
, COUNT(CASE WHEN IF({{ col }}='', NULL, {{ col }}) IS NULL THEN 1 END) AS num_empty_values
, COUNT(DISTINCT IF({{ col }}='', NULL, {{ col }})) AS num_unique_values
, SAFE_DIVIDE(COUNT(DISTINCT {{ col }}), COUNT(*)) AS selectivity
, SAFE_DIVIDE( (COUNT(*) - COUNT(CASE WHEN IF({{ col }}='', NULL, {{ col }}) IS NULL THEN 1 EN
FROM `{{ query_args.source_table_full_name }}`
WHERE 1=1
AND DATE(_PARTITIONTIME) = "{{ partition.format( 'YYYY-MM-DD') }}"
GROUP BY 1, 2
{% endfor -%}

{% for tcn in query_args.target_column_name -%}
UNION ALL

SELECT
"{{ partition. format( 'YYYY-MM-DD') }}" AS partition_date
, {{ query_args.source_table_full_name }}" AS table_full_name
, 'source' AS table_type
,  COUNT(*) AS num_rows
, "{{ col }}" AS column_name
, SAFE_CAST(MIN(IF({{ col }}="'', NULL, {{ col }})) AS STRING) AS min_value
, SAFE_CAST(MAX(IF({{ col }}='', NULL, {{ col }})) AS STRING) AS max_value
, COUNT(CASE WHEN IF({{ col }}='', NULL, {{ col }}) IS NULL THEN 1 END) AS num_empty_values
, COUNT(DISTINCT IF({{ col }}='', NULL, {{ col }})) AS num_unique_values
, SAFE_DIVIDE(COUNT(DISTINCT {{ col }}), COUNT(*)) AS selectivity
, SAFE_DIVIDE( (COUNT(*) - COUNT(CASE WHEN IF({{ col }}='', NULL, {{ col }}) IS NULL THEN 1 EN
FROM `{{ query_args.source_table_full_name }}`
WHERE 1=1
AND DATE(_PARTITIONTIME) = "{{ partition.format( 'YYYY-MM-DD') }}"
GROUP BY 1, 2
{% endfor -%}
```

The code is a configuration for a BigQuery data quality monitoring framework that:

- Takes source table information as input parameters
- Iterates through specified columns in the source table

For each column, generates SQL that calculates key data quality metrics:

- Row count
- Min/max values
- Empty value count
- Unique value count
- Selectivity (unique values / total rows)
- Density (non-null values / total rows)

> It then attempts to do the same for target table columns, though there's a bug in the target table section where it references col instead of tcn and doesn't use the target table name. The UNION ALL combines all these metrics into a single result set, creating a daily snapshot of table health that can be stored, visualized in Looker, and used for anomaly detection.

Below is a sample output:

table_full_name	column_name	min_value	max_value	num_empty_values	cardinality	selectivity	density
customer_stage_d	channel	Audiowatch	WhatsApp	0	10	2.2601423889705052E-4	1.0
customer_stage_d	country	AR	Zimbabwe	29565	135	0.003073793648999887	0.33178890270087014
customer_stage_d	topic	Abuse	personal-image	34403	13	3.164199344558707E-4	0.22244321392247712
customer_stage_d	uniqueCaseId	00090af7c44570c49ca11893cab7dd4f	ffffc0c9ff257dfc5da8d9c4fc03908c	0	21913	0.4952650016951068	1.0
customer_stage_d	userId	000bc0bd424c49aa85f7f134e54838fc	fff9872230244ef28136bf422ff50864	29690	9248	0.20904056955588202	0.328963724714657
_customer_support_case_user_entity_d	channel	Chat	Messaging	0	3	6.780427166911516E-5	1.0
_customer_support_case_user_entity_d	countryName	Albania	Zimbabwe	29696	112	0.002531359475646965	0.3288281161713188
_customer_support_case_user_entity_d	topic	Abuse	personal-image	34403	13	3.164199344558707E-4	0.22244321392247712
_customer_support_case_user_entity_d	uniqueCaseId	00090af7c44570c49ca11893cab7dd4f	ffffc0c9ff257dfc5da8d9c4fc03908c	0	21913	0.4952650016951068	1.0
_customer_support_case_user_entity_d	userId	000bc0bd424c49aa85f7f134e54838fc	fff9872230244ef28136bf422ff50864	29690	9248	0.2090179681319923	0.328963724714657
case_history_snapshot_*	case_number	00045980-40dc-490f-893f-3b335c4d2d18	fff9543f-e4c0-4a10-ab37-6447000cefb8	0	21933	0.9999544086805873	1.0
case_history_snapshot_*	channel	Chat	Social	0	4	1.8236527765113522E-4	1.0
case_history_snapshot_*	country	AR	Zimbabwe	7168	131	0.005972462843074678	0.6732014224491657
case_history_snapshot_*	topic	Abuse	personal-image	8221	24	0.0010941916659068112	0.6251937631075043
case_history_snapshot_*	user_id	000bc0bd424c49aa85f7f134e54838fc	fff9872230244ef28136bf422ff50864	6892	9682	0.4414607458739856	0.685784626607094

The framework will:

• Run daily queries to compute stats

• Store results in an audit table

• Store results in an audit table
• Power Looker dashboards with this metadata or you can plug into any dashboard tools

□ Visualize in Looker

Use Looker dashboards to track table trends with intuitive visualizations:
• □ Spider Chart: Compare each table to a “target” healthy shape
• □ Row Count: Overall row changes over time
• □ Column Cardinality: Expect stability unless business logic changed
• □ Selectivity & Density: Should remain steady for trusted columns

> primary_key = 1, 100% is good

Rule of Thumb:

- □ Stable = Good. □
- Sudden change = Investigate. □□

□ Alert in Slack

Simple Threshold Alerts*:
• Row Count drops by >20,000
 Might indicate upstream data loss
• Cardinality of 'channel' < 4
 Might indicate tracking error or missing dimension

> you need familiar with your business, know the generic trend

Set alerts via a scheduled query + Slack webhook. Example use cases:
• Sudden dip in data volume (e.g. user signups, transactions)
• Business dimension suddenly missing (e.g. acquisition channels)

Why This Works

You don't need a giant platform to catch most data issues. Just track the right signals, visualize the trends, and alert on anomalies.

> Because what really matters isn't "Did the job run?" **It's: "Can I trust this Explore result?**"

Previous Next

2024-08-01
Express Lane vs Express checkout

2023-04-15
Museum of Failure - MO