String Functions and Views in BigQuery

Optimizing SQL for Analytics in the Cloud

Objectives

1. Compare and contrast: relate BigQuery's SQL syntax for string functions and views to your existing PostgreSQL knowledge
2. Manipulate strings: identify and apply the most common BigQuery string functions (CONCAT, SUBSTR, REGEXP_EXTRACT) for data cleaning.
3. Abstract data logic: understand the syntax and use cases for creating, querying, and managing standard views in BigQuery
4. Master reference: navigate the official Google Cloud documentation to find definitive syntax and examples for any BigQuery function or feature

BigQuery String Functions

- BigQuery uses GoogleSQL (standard SQL dialect for Google Cloud). Syntax is often similar to Postgres, but function names and specifics may vary
- Case sensitivity: BigQuery is generally case-insensitive except for string matching
- Some essential string functions
    - CONCAT
    - SUBSTR
    - LOWER and UPPER
    - LIKE
    - REGEXP_*

GoogleSQL Documentation

- Single source of truth for function names, arguments, and examples
- Documentation for String Functions

Regular Expressions (REGEXP_...)

BigQuery datasets often contain raw, unstructured text fields (eg URLs, log entries). Regular expressions are essential for extracting standardized data from these sources.

Key REGEXP functions:

- REGEXP_CONTAINS: returns true if the regex pattern is found in the string
- REGEXP_EXTRACT: returns the first captured group (or null if no match)
- REGEXP_REPLACE: replaces matched substrings with a specified replacement string

Note: BigQuery's GoogleSQL uses the re2 regular expression library

re2 library information

Standard Views

A named query that you can treat like a virtual table. Logical views don't store data, they store the definition of the query.

Use cases

- Abstraction: hiding complex JOIN logic and calculations from end users
- Security/access: exposing only certain columns or subsets of data to different user groups
- Code reuse: standardizing clean, transformed data sets for multiple reports

BigQuery View Syntax

- Very similar to postgres
- Use CREATE OR REPLACE VIEW if view already exists

```
CREATE VIEW
dataset_name.view_name AS
SELECT
        user_id,
        REGEXP_EXTRACT(raw_url,
        r'\/(\d+)\/') AS product_id,
        TRIM(LOWER(email_address)) AS
        standardized_email,
        timestamp
FROM raw_logs.web_traffic
WHERE timestamp >= '2024-01-01' ;
```

Materialized Views

- Materialized views in BigQuery are similar to standard views, but the query results are pre-calculated and stored

Core benefits:

- Cost reduction: queries against a materialized view only scan the pre-calculated result set, dramatically reducing the amount of data processed and therefore the cost
- Performance boost: reports of dashboards that query a materialized view are faster because the heavy lifting (joins, aggregations) has already been done
- Automatic refresh: BigQuery automatically maintains and refreshes the materialized view when the underlying base tables are updated

Materialized View Syntax

- Materialized view must be deleted before being redefined
- There are restrictions to what SQL functions can be used (no window functions, non-deterministic functions, etc)

```
CREATE MATERIALIZED VIEW
dataset_name.mv_daily_sales_summary
AS
SELECT
        DATE(transaction_time) AS
sale_date, product_id,
SUM(sale_amount) AS total_sales_usd
FROM raw_logs.sales_transactions
GROUP BY 1, 2 ;
```

Additional Resources

- BigQuery string functions
- BigQuery Logical Views
- BigQuery Materialized Views