02 BigQuery Arrays -KirkYagami 👨‍💻🕵️

Introduction

Arrays in BigQuery are powerful data structures that allow you to store multiple values of the same data type in a single field. They are particularly useful when dealing with repeated fields in nested data structures, which are common in many datasets, including web analytics data.

Basic Array Concepts

Creating Arrays

In BigQuery, you can create arrays using the ARRAY function or array literals:

```
SELECT ARRAY[1, 2, 3] AS number_array,
       ARRAY['a', 'b', 'c'] AS string_array;
```

Accessing Array Elements

You can access individual elements of an array using zero-based indexing:

```
SELECT ARRAY[1, 2, 3][OFFSET(0)] AS first_element,
       ARRAY['a', 'b', 'c'][OFFSET(1)] AS second_element;
```

Working with Arrays in BigQuery

UNNEST Function

- First Preview the table to understand the table data.
- Then execute the below query

```
SELECT
visitId, date,
device, geoNetwork, hits
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`
  LIMIT 10;
```

The UNNEST function is crucial when working with arrays. It flattens an array into individual rows.

```
SELECT *
FROM table_name,
```

```
    UNNEST(array_column) AS element
```

- `table_name`: The name of the table you're querying.
- `array_column`: The name of the array column you want to unnest.
- `AS element`: This renames the individual elements from the array to a new column name (in this case, "element").

Example using the Google Analytics sample dataset:

```
SELECT
  date,
  hits.page.pagePath
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`,
  UNNEST(hits) AS hits
LIMIT 5
```

This query unnests the `hits` array, allowing us to access the `pagePath` for each hit.

Array Functions

https://cloud.google.com/bigquery/docs/reference/standard-sql/array_functions

ARRAY_LENGTH

Returns the number of elements in an array.

```
SELECT
  fullVisitorId,
  ARRAY_LENGTH(hits) AS num_hits
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`
LIMIT 5
```

ARRAY_AGG

Aggregates values into an array.

```
SELECT
  date,
  ARRAY_AGG(DISTINCT device.browser) AS browsers_used
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`
GROUP BY
```

```
  date
LIMIT 5
```

Filtering Arrays

Checks if an array contains a specific value.

```
SELECT
  fullVisitorId,
  hits.page.pagePath
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`,
  UNNEST(hits) AS hits
WHERE
  '/home' IN (hits.page.pagePath)
LIMIT 5;
```

Filtering with Subqueries

You can use subqueries to filter based on array conditions:

```
SELECT
  fullVisitorId,
  (SELECT COUNT(*)
   FROM UNNEST(hits) AS h
   WHERE h.eCommerceAction.action_type = '6') AS num_purchases
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`
WHERE
  (SELECT COUNT(*)
   FROM UNNEST(hits) AS h
   WHERE h.eCommerceAction.action_type = '6') > 0
LIMIT 5
```

This query counts the number of purchases for each visitor who made at least one purchase.

Combining Arrays

ARRAY_CONCAT

Concatenates two or more arrays.

```
SELECT
  fullVisitorId,
  ARRAY_CONCAT(
    ARRAY_AGG(DISTINCT device.browser),
```

```
    ARRAY_AGG(DISTINCT device.operatingSystem)
  ) AS user_tech
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`
GROUP BY
  fullVisitorId
LIMIT 5
```

Advanced Array Techniques

Working with Nested Arrays

Sometimes, you might encounter arrays within arrays. You can use multiple UNNEST statements to handle these:

```
SELECT
  date,
  product.v2ProductName,
  product.productRevenue
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`,
  UNNEST(hits) AS hits,
  UNNEST(hits.product) AS product
WHERE
  product.productRevenue IS NOT NULL
LIMIT 5
```

This query unnests both the `hits` array and the `product` array within each hit to access product revenue information.

Generating Arrays

You can generate arrays based on certain conditions:

```
SELECT
  fullVisitorId,
  ARRAY(
    SELECT AS STRUCT
      h.page.pagePath,
      h.time
    FROM
      UNNEST(hits) AS h
    WHERE
      h.type = 'PAGE'
    ORDER BY
      h.time
```

```
  ) AS pageview_sequence
FROM
  `bigquery-public-data.google_analytics_sample.ga_sessions_*`
LIMIT 5
```

This query creates an array of page views for each session, ordered by time.

Conclusion

Arrays in BigQuery provide a powerful way to work with repeated and nested data structures. By mastering array functions and techniques, you can efficiently analyze complex datasets like web analytics data. Practice these concepts to become proficient in using arrays in your BigQuery queries.