SQL Test —> Coveo

💫 By Laura Manolache

On importing data (.csv files) provided to Snowflake:

Step 1: Log in to the account.

Step 2: Installhttps://developers.snowflake.com/snowsql/ Log in & Select Database. i.e. : “use database coveo”

Step 3: Create File Format.

```
CREATE
OR
replace FILE format my_csv_format
type = csv field_delimiter = '|'
skip_header = 1 null_if = ('NULL', 'null')
empty_field_as_null = true compression = gzip;
```

Step 4: Create Table in Snowflake using Create Statement.

Step 5: Load CSV file.

```
put file://D:\searches.csv @coveo.PUBLIC.%searches;
```

Step 6: Copy the data into Target Table.

```
copy into searches from @%searches
file_format = (format_name = 'my_csv_format' ,
error_on_column_count_mismatch=false) pattern = '.*searches.c
on_error = 'skip_file';
```

SQL

1.1 List all visits with at least one clicks

```
SELECT
       visitid
FROM   (
         SELECT
```

```
            try_cast(visitid as varchar) AS visitid,
             count(DISTINCT clickid) AS total_clicks
         FROM  coveo.PUBLIC.clicks
         GROUP BY 1) a
WHERE  total_clicks >=1
```

1.2.

The percentage of search having clicks per day, over the last 7 days, including overall summary value (using a single SQL query, adding a column to produce the summary value for the overall period)

```
WITH all_searches AS (
         SELECT
                DATE(datetime) AS date,
                COUNT(id)      AS total_searches
         FROM   coveo.PUBLIC.searches
         GROUP  BY DATE(datetime)),

     search_w_clicks AS (
         SELECT DATE(a.datetime) AS date,
                COUNT(clickid)   AS counts_search_w_click
         FROM   coveo.PUBLIC.searches a
                JOIN coveo.PUBLIC.clicks c
                  ON a.id = c.searchid
         GROUP  BY 1),

     perentage AS (
         SELECT
           a.date,
           counts_search_w_click / total_searches * 100 AS
           percentage_search
         FROM   all_searches a
                JOIN search_w_clicks b
                  ON a.date = b.date)
SELECT
   date,
   percentage_search,
```

```
   Avg(percentage_search)
   OVER (
   ORDER BY DATE ROWS BETWEEN 6 preceding AND CURRENT ROW) AS
   mavg_perc_7days
FROM   perentage
```

1.3 List the 10 most active users (by number of searches) having at least 1 click on a document coming from a source starting with “Confluence”.

```
SELECT
a.userid,
Count(DISTINCT id)  AS total_searches,
Dense_rank() OVER (ORDER BY Count(DISTINCT id) DESC)
AS ranked_users
FROM     coveo.PUBLIC.searches a
JOIN     coveo.PUBLIC.clicks b
ON       a.id = b.searchid
WHERE    CONTAINS(b.sourcename, 'Confluence')
         --and b.documentcategory= 'page'
GROUP BY 1
ORDER BY 2 DESC limit 10
```

OR

```
SELECT userid,
       total_searches,
       ranked_users
FROM   (
        SELECT
        a.userid,
        Count(DISTINCT id)AS total_searches,
        Dense_rank()
        OVER (
        ORDER BY Count(DISTINCT id) DESC) AS ranked_users
        FROM   coveo.PUBLIC.searches a
               JOIN coveo.PUBLIC.clicks b
                 ON a.id = b.searchid
```

```
         WHERE  CONTAINS(b.sourcename, 'Confluence')
         --and b.documentcategory= 'page'
         GROUP  BY 1) a
WHERE  ranked_users <= 10
```

1.4 List the top 10 most popular items and display their average click rank.

```
SELECT product,
       avg_rank
FROM   (SELECT
          c_product  AS product,
          Dense_rank()
          OVER (
          partition BY c_product
          ORDER BY Count(DISTINCT id) DESC) AS ranked_items,
          Round(Avg(clickrank), 2)          AS avg_rank
         FROM   coveo.PUBLIC.searches a
                JOIN coveo.PUBLIC.clicks b
                  ON a.id = b.searchid
         GROUP  BY 1)
WHERE  ranked_items <= 10
ORDER  BY 2 DESC
```

1.5

List all events related to visits, in order of date, adding the following information
o The sequence number (1 to N) of the event over the visit
o The time difference (in milliseconds) between the event and the previous one.
The first event having 0 or null since no previous event

```
SELECT id    AS searchid,
       visitid,
       datetimeAS date_search,
       Rank()
         OVER (
           partition BY visitid
           ORDER BY date_search ASC) AS search_rank,
       Datediff(millisecond, Lag(a.datetime)
                      OVER (
```

```
                        partition BY a.visitid
              ORDER BY a.datetime), a.datetime) AS time_diff
FROM   coveo.PUBLIC.searches a
GROUP  BY 1,
          2,
          3
ORDER  BY a.datetime
```