BigQuery metadata analysis

Following document aims to facilitate SQream’s sales team to approach potential customers that currently use BigQuery analytics solution.

**The reasoning as explained by** [Brian Garback](mailto:briang@sqreamtech.com)

Customer engineers will understand what's possible and can drive innovation.

Customer executives will get a top-line view of the acceleration/savings.

SQream sales will have an idea of current spend and opportunity size for the land and expand.

SQream sales engineering can dictate which workloads we want to target, rather than a customer-driven POC.

SQream product and R&D will get a view of a target customer to better understand our strengths, coverage, and gaps.

If there's not a fit, we say so and everyone saves cycles now with an opportunity to revisit.

Below you will find formatted SQL queries. “--” is used for comments and is not part of the code to run. Please pay attention to:

1. The {projectId} parameter in the queries should be replaced with the actual Google project ID before running the query.
2. We use ‘region-us’ in all the queries. If the project is in a different region, this part should be changed accordingly.
3. We assume that the user running these queries has all the necessary permissions.
4. We assume the user has only one Google project. If there are multiple projects, each query should be run once per project.

### TABLES COUNT AND SIZES

This query helps in understanding the storage and data distribution across different datasets within a BigQuery project. It provides insights into:

* The number of tables per dataset.
* Data size in gigabytes of the tables in dataset - physical and logical size in Gigabytes - min, max, avg, median per dataset.
* The number of records in tables - min, max, avg, median per dataset

Relevant google documentation related to the main datasource INFORMATION\_SCHEMA.TABLE\_STORAGE : <https://cloud.google.com/bigquery/docs/information-schema-table-storage>

-----------------TABLES COUNT AND SIZES-----------------START

WITH dataset\_table\_info AS (

SELECT

TABLE\_CATALOG AS project\_id,

TABLE\_SCHEMA AS dataset\_id,

TABLE\_NAME AS table\_name,

TOTAL\_ROWS AS row\_count,

TOTAL\_PHYSICAL\_BYTES / (1024 \* 1024 \* 1024) AS physical\_size\_gb,

TOTAL\_LOGICAL\_BYTES / (1024 \* 1024 \* 1024) AS logical\_size\_gb

FROM

`{projectId}.region-us.INFORMATION\_SCHEMA.TABLE\_STORAGE`

),

physical\_size\_stats AS (

SELECT

dataset\_id,

physical\_size\_gb,

ROW\_NUMBER() OVER (PARTITION BY dataset\_id ORDER BY physical\_size\_gb) AS rn,

COUNT(\*) OVER (PARTITION BY dataset\_id) AS cnt

FROM

dataset\_table\_info

),

logical\_size\_stats AS (

SELECT

dataset\_id,

logical\_size\_gb,

ROW\_NUMBER() OVER (PARTITION BY dataset\_id ORDER BY logical\_size\_gb) AS rn,

COUNT(\*) OVER (PARTITION BY dataset\_id) AS cnt

FROM

dataset\_table\_info

),

row\_count\_stats AS (

SELECT

dataset\_id,

row\_count,

ROW\_NUMBER() OVER (PARTITION BY dataset\_id ORDER BY row\_count) AS rn,

COUNT(\*) OVER (PARTITION BY dataset\_id) AS cnt

FROM

dataset\_table\_info

),

table\_stats AS (

SELECT

dataset\_id,

COUNT(table\_name) AS table\_count,

MIN(physical\_size\_gb) AS min\_physical\_size\_gb,

MAX(physical\_size\_gb) AS max\_physical\_size\_gb,

AVG(physical\_size\_gb) AS avg\_physical\_size\_gb,

SUM(physical\_size\_gb) AS total\_physical\_size\_gb,

MIN(logical\_size\_gb) AS min\_logical\_size\_gb,

MAX(logical\_size\_gb) AS max\_logical\_size\_gb,

AVG(logical\_size\_gb) AS avg\_logical\_size\_gb,

SUM(logical\_size\_gb) AS total\_logical\_size\_gb,

MIN(row\_count) AS min\_rows,

MAX(row\_count) AS max\_rows,

AVG(row\_count) AS avg\_rows,

SUM(row\_count) AS total\_rows

FROM

dataset\_table\_info

GROUP BY

dataset\_id

),

physical\_median AS (

SELECT

dataset\_id,

APPROX\_QUANTILES(physical\_size\_gb, 2)[OFFSET(1)] AS median\_physical\_size\_gb

FROM

dataset\_table\_info

GROUP BY

dataset\_id

),

logical\_median AS (

SELECT

dataset\_id,

APPROX\_QUANTILES(logical\_size\_gb, 2)[OFFSET(1)] AS median\_logical\_size\_gb

FROM

dataset\_table\_info

GROUP BY

dataset\_id

),

row\_count\_median AS (

SELECT

dataset\_id,

APPROX\_QUANTILES(row\_count, 2)[OFFSET(1)] AS median\_rows

FROM

dataset\_table\_info

GROUP BY

dataset\_id

)

SELECT

ts.dataset\_id,

ts.table\_count,

ts.min\_physical\_size\_gb,

ts.max\_physical\_size\_gb,

ts.avg\_physical\_size\_gb,

pm.median\_physical\_size\_gb,

ts.total\_physical\_size\_gb,

ts.min\_logical\_size\_gb,

ts.max\_logical\_size\_gb,

ts.avg\_logical\_size\_gb,

lm.median\_logical\_size\_gb,

ts.total\_logical\_size\_gb,

ts.min\_rows,

ts.max\_rows,

ts.avg\_rows,

rm.median\_rows,

ts.total\_rows

FROM

table\_stats ts

JOIN

physical\_median pm ON ts.dataset\_id = pm.dataset\_id

JOIN

logical\_median lm ON ts.dataset\_id = lm.dataset\_id

JOIN

row\_count\_median rm ON ts.dataset\_id = rm.dataset\_id

ORDER BY

ts.dataset\_id;

-----------------TABLES COUNT AND SIZES-----------------END

### JOBS PAST 30 DAYS

This query provides a summary of query statistics over the past 30 days per query type, including:

* Query count and user count per day and query type.
* Minimum, maximum, average, median, and total gigabytes processed.
* Minimum, maximum, average, median, and total gigabytes billed.
* Minimum, maximum, average, median, and total slot utilization in hours.

Relevant google documentation related to the main datasource INFORMATION\_SCHEMA.INFORMATION\_SCHEMA.JOBS : <https://cloud.google.com/bigquery/docs/information-schema-jobs>

-----------------JOBS PAST 30 DAYS-----------------START

WITH job\_stats AS (

SELECT

DATE(creation\_time) AS job\_date,

job\_type,

COUNT(\*) AS query\_count,

COUNT(DISTINCT user\_email) AS user\_count,

MIN(total\_bytes\_processed) / (1024 \* 1024 \* 1024) AS min\_gb\_processed,

MAX(total\_bytes\_processed) / (1024 \* 1024 \* 1024) AS max\_gb\_processed,

AVG(total\_bytes\_processed) / (1024 \* 1024 \* 1024) AS avg\_gb\_processed,

APPROX\_QUANTILES(total\_bytes\_processed / (1024 \* 1024 \* 1024), 2)[OFFSET(1)] AS median\_gb\_processed,

SUM(total\_bytes\_processed) / (1024 \* 1024 \* 1024) AS total\_gb\_processed,

MIN(total\_bytes\_billed) / (1024 \* 1024 \* 1024) AS min\_gb\_billed,

MAX(total\_bytes\_billed) / (1024 \* 1024 \* 1024) AS max\_gb\_billed,

AVG(total\_bytes\_billed) / (1024 \* 1024 \* 1024) AS avg\_gb\_billed,

APPROX\_QUANTILES(total\_bytes\_billed / (1024 \* 1024 \* 1024), 2)[OFFSET(1)] AS median\_gb\_billed,

SUM(total\_bytes\_billed) / (1024 \* 1024 \* 1024) AS total\_gb\_billed,

MIN(total\_slot\_ms) / (1000 \* 60 \* 60) AS min\_slot\_utilization\_hours,

MAX(total\_slot\_ms) / (1000 \* 60 \* 60) AS max\_slot\_utilization\_hours,

AVG(total\_slot\_ms) / (1000 \* 60 \* 60) AS avg\_slot\_utilization\_hours,

APPROX\_QUANTILES(total\_slot\_ms / (1000 \* 60 \* 60), 2)[OFFSET(1)] AS median\_slot\_utilization\_hours,

SUM(total\_slot\_ms) / (1000 \* 60 \* 60) AS total\_slot\_utilization\_hours

FROM

`panoply-880-0110627aced8.region-us.INFORMATION\_SCHEMA.JOBS`

WHERE

creation\_time >= TIMESTAMP\_SUB(CURRENT\_TIMESTAMP(), INTERVAL 30 DAY)

GROUP BY

job\_date, job\_type

)

SELECT

job\_date,

job\_type,

query\_count,

user\_count,

min\_gb\_processed,

max\_gb\_processed,

avg\_gb\_processed,

median\_gb\_processed,

total\_gb\_processed,

min\_gb\_billed,

max\_gb\_billed,

avg\_gb\_billed,

median\_gb\_billed,

total\_gb\_billed,

min\_slot\_utilization\_hours,

max\_slot\_utilization\_hours,

avg\_slot\_utilization\_hours,

median\_slot\_utilization\_hours,

total\_slot\_utilization\_hours

FROM

job\_stats

ORDER BY

job\_date, job\_type;

-----------------JOBS PAST 30 DAYS-----------------END

SLOTS RESERVATIONS

This query shows all existing reservations and editions.

Relevant google documentation related to the main datasource INFORMATION\_SCHEMA.RESERVATIONS : <https://cloud.google.com/bigquery/docs/information-schema-reservations>

-----------------SLOTS RESERVATIONS-----------------START

select \*

from `{projectId}`.`region-us`.INFORMATION\_SCHEMA.RESERVATIONS

-----------------SLOTS RESERVATIONS-----------------END

USER DEFINED FUNCTIONS

This query shows all the existing user defined functions.

Relevant google documentation related to the main datasource INFORMATION\_SCHEMA.ROUTINES :

<https://cloud.google.com/bigquery/docs/routines#sql_1>

-----------------USER DEFINED FUNCTIONS-----------------START

select \*

from `{projectId}.region-us.INFORMATION\_SCHEMA.ROUTINES`

where routine\_type = 'FUNCTION'

-----------------USER DEFINED FUNCTIONS-----------------END

USER DEFINED PROCEDURES

This query shows all the existing user defined functions.

Relevant google documentation related to the main datasource INFORMATION\_SCHEMA.ROUTINES :

<https://cloud.google.com/bigquery/docs/routines#sql_1>

-----------------USER DEFINED PROCEDURE-----------------START

select \*

from `{projectId}.region-us.INFORMATION\_SCHEMA.ROUTINES`

where routine\_type = 'PROCEDURE'

-----------------USER DEFINED PROCEDURE-----------------END

### COUNT USER DEFINED FUNCTIONS AND PROCEDURES

This query shows the number of user defined functions and procedures and uses additional dimension - routine\_type to show the type (language used) of the routine.

-----------------COUNT USER DEFINED FUNCTIONS AND PROCEDURES-----------------START

select

routine\_type,

routine\_body,

count(\*)

from `{projectId}.region-us.INFORMATION\_SCHEMA.ROUTINES`

group by routine\_type, routine\_body

-----------------COUNT USER DEFINED FUNCTIONS AND PROCEDURES-----------------END