Important links

<https://jsonplaceholder.typicode.com/posts>

[JSONPlaceholder - Guide (typicode.com)](https://jsonplaceholder.typicode.com/guide/)

<https://jsonplaceholder.typicode.com/posts>

[Format data to json](http://jsonviewer.stack.hu/)

Learning links

[Execution model (apache.org)](https://beam.apache.org/documentation/runtime/model/)

[Apache Jdbc Connection](https://beam.apache.org/releases/pydoc/2.24.0/apache_beam.io.jdbc.html#:~:text=PTransforms%20for%20supporting%20Jdbc%20in,portable%20Flink%20and%20Spark%20runners.&text=Transforms%20provided%20in%20this%20module,in%20the%20Beam%20Java%20SDK)

[Quotas and limits  |  BigQuery  |  Google Cloud](https://cloud.google.com/bigquery/quotas)

[Use query queues  |  BigQuery  |  Google Cloud](https://cloud.google.com/bigquery/docs/query-queues)

Bigquery

**Adding"" in notepadd++**

Find:(.+)

Replace: "\1"

**Get bigquery Schema**

bq show \

--schema \

--format=prettyjson \

dbce-c360-dqops-prod-3ad4:data\_quality\_monitoring.results\_raw\_20200713 > /tmp/myschema1.json

**Schema Auto Detect:**

SCHEMA\_AUTODETECT

**Stackdriver search:**

resource.type="dataflow\_step"

resource.labels.job\_id="2021-06-13\_02\_25\_26-3294092084461434224"

jsonPayload.message =~ "#######LyticsPOST####"

resource.type="dataflow\_step"

resource.labels.job\_id="2021-10-12\_03\_58\_14-2724845405637396435"

jsonPayload.message=~"lytics mp490"

**Copy table with partition:**

CREATE TABLE (new?)dataset.new\_table PARTITION BY DATE(date\_column) AS SELECT \* FROM dataset.table\_to\_copy

**BigQuery:**

SELECT count(\*) FROM `dbce-c360-isl-dev-f03b.data\_consistency\_monitoring.results\_raw` where use\_case='registration\_JANRAIN'

and timeStamp in ('2021-07-21 10:19:19 UTC','2021-07-21 10:19:20 UTC','2021-07-21 10:19:21 UTC')

SELECT distinct r.application FROM `dbce-c360-isl-dev-f03b.data\_consistency\_monitoring.results\_raw`,

UNNEST(responses) r

where use\_case='registration\_JANRAIN'

and timeStamp in ('2021-07-21 10:19:19 UTC','2021-07-21 10:19:20 UTC','2021-07-21 10:19:21 UTC')

SELECT r.application,r.response\_code FROM `dbce-c360-isl-dev-f03b.data\_consistency\_monitoring.raw\_test`,

UNNEST(responses) r

where r.response\_code!=200 and r.application='segment'

and use\_case='registration\_JANRAIN'

and timeStamp in ('2021-07-21 10:19:19 UTC','2021-07-21 10:19:20 UTC','2021-07-21 10:19:21 UTC')

SELECT eve.message, eve.application FROM `dbce-c360-isl-dev-f03b.data\_consistency\_monitoring.results\_raw`,

UNNEST(evaluation.validation) ev,

UNNEST(ev.errors) eve

where use\_case='optout\_KOT'

and timeStamp in ('2021-07-18 09:32:33 UTC', '2021-07-18 09:32:33 UTC')

bq load --source\_format=NEWLINE\_DELIMITED\_JSON data\_consistency\_monitoring.resilient\_pipeline\_results\_raw

gs://data\_consistency\_monitor\_dqops\_prod/tmp/bq\_load/f8a467dd97a946ab999a475b8d0ef69d/dbce-c360-dqops-prod-3ad4.data\_consistency\_monitoring.resilient\_pipeline\_results\_raw/9ea432d0-ad01-444f-ad27-e143caa69075

bq load --source\_format=NEWLINE\_DELIMITED\_JSON data\_quality\_monitoring.results\_raw

gs://data\_consistency\_monitor\_dqops\_prod/tmp/bq\_load/8820f627f4c84eedaccf5ceadfdad519/dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw/f5e5c6eb-33c3-483d-9a4e-07c0a89efd6

bq load --autodetect --source\_format=NEWLINE\_DELIMITED\_JSON ops\_data\_quality\_monitoring.result\_raw\_opt\_temp gs://data\_consistency\_monitoring\_data/temp/bq\_load/cac89ab227fc471fa8878609cab15bf4/dbce-c360-dqops-prod-3ad4.ops\_data\_quality\_monitoring.result\_raw\_opt\_temp/4a5e0f98-e10d-438f-93b2-1dbf616336f2

Create TABLE `dbce-c360-dqops-prod-3ad4.data\_consistency\_monitoring.resilient\_pipeline\_results\_raw\_cluster` CLUSTER BY timestamp AS(

SELECT \* FROM `dbce-c360-dqops-prod-3ad4.data\_consistency\_monitoring.resilient\_pipeline\_results\_raw` WHERE DATE(timestamp) > "2022-06-19")

|  |  |
| --- | --- |
| Description | Query |
| Query to select records that failed for a given day | SELECT \* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw`  WHERE evaluation.status = "Failed" and date(timestamp) = '2020-08-13'; |
| Query to select failed records for a specific validation type | SELECT \* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw`, unnest(evaluation.validation) as e  WHERE e.status = "Failed" and e.type = "correctness" and date(timestamp) = '2020-08-13'; |
| Query to select records that failed for a specific validation type and application | SELECT b.\* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw` b, unnest(evaluation.validation) as e, unnest(e.errors) as a  WHERE a.application = "segment" and e.type = "completeness" and date(timestamp) = '2020-08-13'; |
| Query to select records that failed a specific error code in a given marketing program number | SELECT b.\* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw` b, unnest(evaluation.validation) as e, unnest(e.errors) as a  WHERE a.application = "segment" and a.type = "EMAILONDEMAND\_OPT\_CHOICE\_DATE\_MISMATCH" and date(timestamp) = '2020-08-13' and b.marketing\_program\_number = 119; |
| Query to select all records with having a given external ID | SELECT b.\* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw` b, unnest(external\_id) ex  WHERE ex.value = "use\_REOqGMAC7mN8VP1xuD51NBmUasw" and date(timestamp) = '2020-08-13' and b.marketing\_program\_number = 119; |
| Query to select all records having a given email | SELECT b.\* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw` b, unnest( responses ) resp  WHERE resp.profile\_metadata.email like "%monitoring%" and date(timestamp) = '2020-08-13' and b.marketing\_program\_number = 119 |

*Check resci remediation:*

WITH

segmentStatus AS (

SELECT

\*

FROM

`dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw`,

UNNEST(evaluation.appncorretness) e

WHERE

e.status = "Success"

AND DATE(timestamp) >= '2022-03-01'

AND DATE(timestamp) <= '2022-03-24'

AND e.application='segment' ),

resciStatus AS (

SELECT

\*

FROM

segmentStatus,

UNNEST(evaluation.appncorretness) e

WHERE

e.status = "Failed"

AND e.application='retention science' )

SELECT

\*

FROM

resciStatus,

UNNEST(external\_id) z

WHERE

z.type = "segment\_id" and issue\_type is null

Update records in table with required error type and application:

--Create table with particular error and application that needs to be updated:

Create table `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.error\_update` as

(

SELECT `hash`,marketing\_program\_name ,legal\_entity ,issue\_type ,remediation\_status FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw\_test` ,

unnest(evaluation.validation) as e, unnest(e.errors) as a

WHERE region='NALA'and mp\_status= 'Live' and issue\_type ='C360 Issue' and remediation\_status is null and a.application = "segment" and a.type = "RECORD\_NOT\_FOUND" and date(timestamp)>'2022-03-01'

)

--Create table with distinct values

Create table `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.error\_update1` as

(

select distinct \* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.error\_update`

)

--Verify the count

select count(\*)

from `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw\_test` a1 Join `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.error\_update1` b2

on a1.marketing\_program\_name = b2.marketing\_program\_name

and a1.hash=b2.hash

and a1.legal\_entity=b2.legal\_entity

and a1.issue\_type=b2.issue\_type

and a1.remediation\_status is null

--Update the main table

update `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw\_test` a

set a.remediation\_status='True'

from `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.error\_update1` b

where a.marketing\_program\_name = b.marketing\_program\_name

and a.`hash`=b.`hash`

and a.legal\_entity=b.legal\_entity

and a.issue\_type=b.issue\_type

and a.remediation\_status is null

In GCP Console:-

bq cp data\_quality\_monitoring.results\_raw@1648645200000 data\_quality\_monitoring.results\_raw (to override same table)

[How to instantly recover a table in Google BigQuery | by Davor Hafnar | Medium](https://medium.com/@dhafnar/how-to-instantly-recover-a-table-in-google-bigquery-544a9b7e7a8d)

bq cp segment\_members\_daily.segment\_members\_expanded@1706736600000segment\_members\_daily.segment\_members\_expanded\_test\_2024-02-01

Where 1648645200000= Date and time in Unix Timestamp format for the date time to which you want to restore the table

SELECT

  json\_extract\_scalar(dt,

    "$['type']")

FROM (

  SELECT

    dt

  FROM

    `dbce-c360-segamerws-prod-35ac.personas\_usa\_olay.tracks`,

    UNNEST(json\_extract\_array(context\_external\_ids)) dt

  WHERE

    user\_id='use\_6ykzzSi1JlxtmSgStUT1YttDrC5'

    AND event LIKE 'email\_%')

  AND event LIKE 'email\_%')

select \* from

(select userId,  context, replaced, if(json\_extract(replaced, '$') is null, false, true) valid\_json from  `dbce-c360-119lake-prod-36f0.r\_data\_validation.formatted\_track\_context\_traits\_phoneNumber\_1`

) where valid\_json is false

select distinct dt from

    (

        select JSON\_EXTRACT(traits, "$['traits.territoryName']") dt from

        `dbce-c360-119lake-prod-36f0.consumer\_raw.identity\_complete\_tbl`

        where marketingProgramNumber = 146 and DATE(ingestTimeStamp) >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 7 DAY)

    )

select distinct dt from

    (

        select JSON\_EXTRACT(properties, "$.subjectLine.0.name") dt from

        `dbce-c360-119lake-prod-36f0.consumer\_raw.track\_complete\_tbl`

        where marketingProgramNumber = 414 and DATE(ingestTimeStamp) >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 7 DAY)

    )

where dt is not null

 select properties,userId  from

        `dbce-c360-119lake-prod-36f0.consumer\_raw.track\_complete\_tbl`

        where marketingProgramNumber = 292 and DATE(ingestTimeStamp) >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 7 DAY) and regexp\_contains(properties, 'questions') and regexp\_contains(properties, 'Όλα έτοιμα!Τα στοιχεία σου έχουν καταχωρηθεί προκειμένου να λάβεις τα κουπόνια σου Fairy μέσω email')

SELECT

  user\_id,

  id AS marketingProgramNumber

FROM (

  SELECT

    JSON\_EXTRACT\_SCALAR(dt, "$['type']") type,

    JSON\_EXTRACT\_SCALAR(dt, "$['id']") id,

    user\_id

  FROM (

    SELECT

      dt,

      user\_id

    FROM

      `usa\_downy\_simple\_leads\_capture\_jebbit\_14760.tracks`,

      UNNEST(JSON\_EXTRACT\_ARRAY(context\_external\_ids)) dt

    WHERE

      user\_id='eeb15dda6462fdefeb3232c5dbb0392f45964e3b3431c904df1dc4784ac86526'

      AND event LIKE '%survey\_question%'))

WHERE

  type ='marketingProgramNumber'

Reference- MP 383 batch 3 EU

249 completed check

Clustered table:

Create or Replace table `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw\_clustered` PARTITION BY DATE(timestamp) CLUSTER BY timestamp,use\_case,region,marketing\_program\_number AS (SELECT \* FROM `dbce-c360-dqops-prod-3ad4.data\_quality\_monitoring.results\_raw` where date(timestamp)>='2022-08-01')

INSERT INTO `dbce-c360-isl-dev-f03b.rtdf\_cc\_test\_automation.Users` VALUES (1,'test.user@pg.com','Test','User','2023-02-02 05:42:39 UTC')

Select 1 as testcase\_id, "Verify 'Identify' event in Segment Source with 'email','emailSha256Hash', 'sourceId', 'marketingProgramNumber' and 'registrationDate' traits" testcase\_name, '' testcase\_desc, 'Jebbit' integration, '2023-02-02 05:42:39 UTC' created\_at

CREATE TEMP FUNCTION

  jsonObjectKeys(input STRING)

  RETURNS ARRAY<String>

  LANGUAGE js AS """ return Object.keys(JSON.parse(input)); """;

WITH

  keys AS (

  SELECT

    jsonObjectKeys(properties) AS keys,

    userId,

    sourceId

  FROM

    `dbce-c360-119lake-prod-36f0.consumer\_raw.track\_complete\_tbl`

  WHERE

    properties IS NOT NULL

    AND marketingProgramNumber = 519

    AND DATE(ingestTimestamp)='2023-02-16')

SELECT

  DISTINCT k,

  userId,

  sourceId

FROM

  keys

CROSS JOIN

  UNNEST(keys.keys) AS k

WHERE

  k LIKE '%@%'

ORDER BY

  k

Select a.\* from EXTERNAL\_QUERY

("dbce-c360-segamaws-prod-9669.us-east4.cdp-staging-traces\_cdp-staging-traces","SELECT \* FROM view\_profiles limit 5") a;

Get Spanner Schema:

SELECT \* FROM EXTERNAL\_QUERY("dbce-c360-segglblws-prod-b902.us.cdp-traits\_cdp-traits\_pii", "select column\_name,SPANNER\_TYPE from information\_schema.columns where table\_name='traits';")

mvn clean

mvn build

mvn package

gsutil cp target/spanner-read-write-bundled-0.0.1.jar gs://us-east4-isl-composer-2-7238166d-bucket/dags/