Data Cleaning with BigQuery

Sebagai seorang yang berprofesi sebagai *Data Scientist*, pembersihan data atau *cleaning data* merupakan hal yang tidak asing, karena untuk menganalisis suatu data dibutuhkan data yang konsisten dan terstruktur. Data yang diberikan pun biasanya merupakan data yang kotor atau *raw.* Oleh karena itu, seorang *Data Scientist* harus melakukan *data cleaning.* Tidak hanya menggunakan python, Cleaning Data pun bisa dilakukan dengan *BigQuery.* Meskipun biasanya *BigQuery* seringkali digunakan untuk mengambil dan mengelola data, *BigQuery* juga merupakan alat yang sangat efektif untuk membersihkan dan menyiapkan data untuk analisis. Berikut adalah alur kerja (workflow) langkah demi langkah yang bisa Anda ikuti untuk membersihkan data menggunakan *BigQuery*:

1. Identifikasi masalah dan Understanding Data
   langkah awal adalah memahami data apa yang diberikan dan pahami data yang diberikan terlebih dahulu. Sebagai contoh, disini kita menggunakan data "New York City Taxi and Limousine Comission(TLC)". Data ini terdiri dari 20 kolom.

2. Cek Duplikasi
   Dalam data ini dalam semua kolom memang terdapat duplikasi, namun duplikat tersebut merupakan duplikat yang dapat diterima atau redudansi yang sah. Tetapi jika anda ingin mengecek berikut caranya :

```
SELECT passenger_count(bisa diganti dengan nama kolom anda), COUNT(nama kolom anda) AS freq
FROM `dataset_nyc.raw_taxi`
GROUP BY passenger_count
HAVING COUNT(*) > 1
```

   lalu outputnya akan seperti :

Row	passenger_count	freq
1	1.0	54299
2	*null*	0
3	3.0	958
4	4.0	364
5	2.0	5096

3. Mencari Missing Value (data yang hilang/berisi null)
   Langkah selanjutnya kita mencari missing value dari data tersebut dan skalanya berapa persen jumlah missing value yang ada pada kolom tersebut . Biasanya jika data tidak memiliki banyak kolom kita bisa mencari dengan query sql seperti :

```
“SELECT
  COUNT(*) AS total_rows,
  COUNTIF((isi dengan nama kolom) IS NULL) AS jumlah_null
    ROUND(100 * COUNTIF(VendorID IS NULL) / COUNT(*), 2) AS jumlah_null_pct
  From FROM `fluid-outcome-468109-u6.dataset_nyc.raw_taxi`; “
```

Dari *query* tersebut memang kita bisa melihat *missing valuenya* ada berapa dan skalanya berapa persen, tetapi jika kita mempunyai banyak kolom *query* tersebut kurang efektif. Oleh karena itu, ini merupakan “*query shortcut”* nya yaitu dengan cara **Dynamic SQL + EXECUTE IMMEDIATE** seperti :

```
DECLARE query STRING;

-- Buat query untuk semua kolom secara dinamis
SET query = (
  SELECT STRING_AGG(
    'SELECT "' || column_name || '" AS column_name, ' ||
    'COUNTIF(' || column_name || ' IS NULL) AS null_count, ' ||
    'ROUND(100 * COUNTIF(' || column_name || ' IS NULL) / COUNT(*), 2) AS null_pct ' ||
    'FROM `fluid-outcome-468109-u6.dataset_nyc.raw_taxi`',
    ' UNION ALL '
  )
  FROM `fluid-outcome-468109-u6.dataset_nyc`.INFORMATION_SCHEMA.COLUMNS
  WHERE table_name = 'raw_taxi'
);

-- Jalankan hasil query tersebut
EXECUTE IMMEDIATE query;
```

penjelasan query :

- “Declare query String” : Kita buat variabel bernama *query* yang akan berisi SQL string dinamis (dibuat berdasarkan nama-nama kolom di tabel)
- “string_agg’ : Menggabungkan semua *sub-query* (satu per kolom) jadi satu string panjang
- “ || “ : tanda tersebut adalah operator konkatenasi string
- “countif(... is null)” : Menghitung baris NULL
- “round(../ count(*), 2 : Menghitung persentase NULL dengan 2 desimal
- “FROM INFORMATION_SCHEMA.COLUMNS’ : Ambil daftar kolom dari metadata tabel BigQuery
- “union all” digunakan untuk **menggabungkan hasil pengecekan NULL dari tiap kolom menjadi satu tabel vertikal** (1 kolom per baris), bukan menyebar ke kanan

Lalu outputnya akan seperti :

Row	column_name	null_count	null_pct
1	improvement_surcharge	0	0.0
2	tip_amount	0	0.0
3	payment_type	4324	6.34
4	PULocationID	0	0.0
5	passenger_count	4324	6.34
6	ehail_fee	68211	100.0
7	congestion_surcharge	4324	6.34
8	tolls_amount	0	0.0
9	trip_type	4334	6.35
10	fare_amount	0	0.0

11	lpep_dropoff_datetime	0	0.0
12	extra	0	0.0
13	total_amount	0	0.0
14	VendorID	0	0.0
15	mta_tax	0	0.0
16	trip_distance	0	0.0
17	store_and_fwd_flag	4324	6.34
18	lpep_pickup_datetime	0	0.0
19	RatecodeID	4324	6.34
20	DOLocationID	0	0.0

4. *Handling Missing Values*

Dari kolom diatas kita bisa lihat jika terdapat *missing value* pada kolom `payment type`, `passenger_count`,`congestion_surcharge`,`trip_type`,store_and_fwd_flag` dan `ratecodeid`. Kolom-kolom tersebut merupakan kolom yang memiliki missing value yang sedikit atau sekitar 6% sedangkan kolom `ehailfee` memiliki missing values yang besar yakni 100%. Oleh karena itu saya memutuskan untuk drop kolom ehail fee. Sedangkan untuk kolom yang missing valuenya 6% saya isi dengan median berikut adalah querynya :

```
CREATE OR REPLACE TABLE `fluid-outcome-468109-u6.dataset_nyc.raw_taxi_cleaned` AS

WITH medians AS (
  SELECT
    APPROX_QUANTILES(passenger_count, 100)[OFFSET(50)] AS med_passenger_count,
    APPROX_QUANTILES(congestion_surcharge, 100)[OFFSET(50)] AS med_congestion,
    APPROX_QUANTILES(trip_type, 100)[OFFSET(50)] AS med_trip_type,
    APPROX_QUANTILES(RatecodeID, 100)[OFFSET(50)] AS med_ratecode
  FROM `fluid-outcome-468109-u6.dataset_nyc.raw_taxi`
)

SELECT
  -- Kolom yang diisi/imputasi
  COALESCE(t.payment_type, 5) AS payment_type,  -- angka 5 = Unknown
  COALESCE(t.passenger_count, m.med_passenger_count) AS passenger_count,
  COALESCE(t.congestion_surcharge, m.med_congestion) AS congestion_surcharge,
  COALESCE(t.trip_type, m.med_trip_type) AS trip_type,
  COALESCE(CAST(t.store_and_fwd_flag AS STRING), 'Unknown') AS store_and_fwd_flag,
  COALESCE(t.RatecodeID, m.med_ratecode) AS RatecodeID,

  -- Kolom lain tetap disertakan
  t.VendorID,
  t.lpep_pickup_datetime,
  t.lpep_dropoff_datetime,
  t.trip_distance,
  t.PULocationID,
  t.DOLocationID,
  t.fare_amount,
  t.extra,
  t.mta_tax,
  t.tip_amount,
  t.tolls_amount,
  t.improvement_surcharge,
  t.total_amount

FROM `fluid-outcome-468109-u6.dataset_nyc.raw_taxi` t
CROSS JOIN medians m;
```

penjelasan query :

- Membuat (atau menimpa) tabel baru bernama raw_taxi_cleaned
- Sub-query bernama “median” yang menghitung **median (kuantil ke-50)** untuk 4 kolom numerik menggunakan APPROX_QUANTILES(..., 100)[OFFSET(50)]
- payment_type NULL, diisi dengan **5** (kode untuk 'Unknown ').
- “coalesce” untuk mengembalikan nilai null dengan step berikutnya seperti menjadi median nya
- “cast” menggunakan cast agar bisa mengubah type data tersebut menjadi string
- cross join dengan sub query dan menjadi file baru raw_taxi_cleaned
- kolom “ehail” tidak dimasukan ke query agar tabel yang baru tidak ada lagi kolom ehail

Saat di run, outputnya akan menghasilkan tabel baru dan tidak memiliki missing values lagi. Sekian penjelasan saya mengenai data cleaning menggunakan BigQuery terimakasih!