## User Testing

//Dokumen yang berisikan tentang bagaimana untuk melakukan test berdasarkan scenario / business requirement yang ditentukan

**Part 2**

1. Berapa banyak customer yang registrasi di platform setiap harinya?
2. Expected Output

| registration\_date | jumlah\_customer |
| --- | --- |
| 2023-11-01 | 1 |
| 2023-11-02 | 1 |
| 2023-11-03 | 1 |
| 2023-11-04 | 1 |
| 2023-11-05 | 1 |

1. Output

Kami membuat perintah SELECT pada tabel Customer dengan function Count untuk menghitung jumlah customer, berikut adalah query yang kami buat:

SELECT

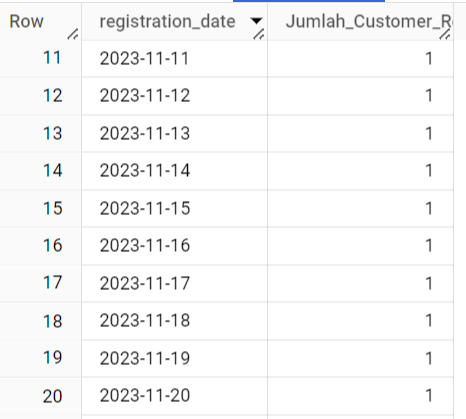
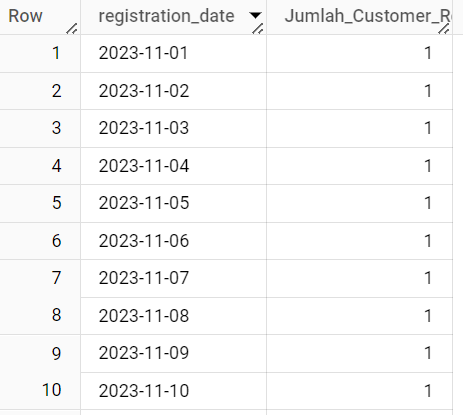
Registration\_date,

COUNT(id\_customer) AS Jumlah\_Customer\_Regristrasi

FROM aqueous-freedom-398208.OFD\_Team\_1.Customer

GROUP BY registration\_date

ORDER BY registration\_date ASC;



1. Status : **pass**
2. Berapa banyak jumlah order yang terbentuk setiap harinya?
   1. Expected Output

| order\_date | Jumlah\_order |
| --- | --- |
| 2023-11-01 | 3 |
| 2023-11-02 | 1 |
| 2023-11-03 | 1 |
| 2023-11-04 | 1 |

* 1. Output

Kami membuat perintah SELECT pada tabel order dengan function COUNT untuk menghitung jumlah order, berikut query yang kami buat :

SELECT

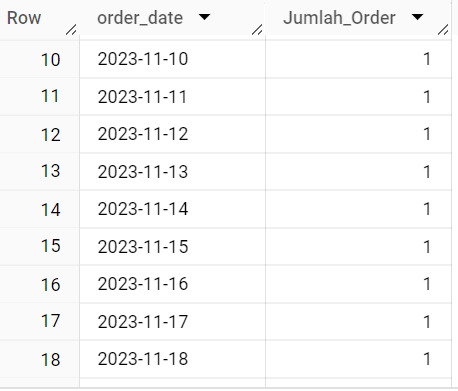
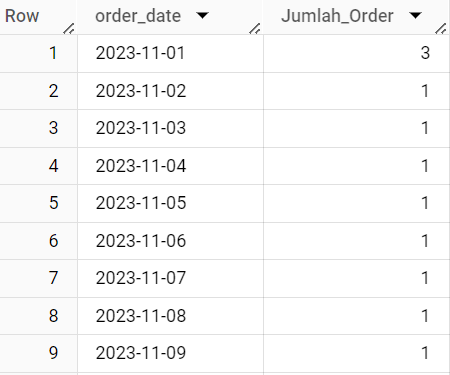
Order\_date,

COUNT(id\_order) AS Jumlah\_Order

FROM aqueous-freedom-398208.OFD\_Team\_1.Order

GROUP BY order\_date

ORDER BY order\_date ASC;



* 1. Status: **Pass**

1. Berapa keuntungan yang didapat dari OK! Food Delivery setiap harinya?
   1. Expected Output

| Tanggal | Keuntungan |
| --- | --- |
| 2023-11-01 | 144000 |
| 2023-11-02 | 36000 |
| 2023-11-03 | 105000 |
| 2023-11-04 | 36000 |
| 2023-11-05 | 70000 |

* 1. Output

Kami membuat perintah SELECT dengan menjumlahkan Quantity dikali Price untuk menghasilkan keuntungan, berikut query yang kami buat:

SELECT

o.order\_date AS Tanggal,

SUM(o.quantity \* f.price) AS Keuntungan

FROM aqueous-freedom-398208.OFD\_Team\_1.Order o

INNER JOIN aqueous-freedom-398208.OFD\_Team\_1.Food f ON o.fk\_food\_id = f.id\_food

GROUP BY o.order\_date

ORDER BY o.order\_date ASC;



* 1. Status: **Pass**

1. Berapa *average spending* 1 orang customer setiap harinya dalam 1 bulan?
2. Expected Output

| nama\_customer | Bulan | Average\_daily\_spend |
| --- | --- | --- |
| Badu | 2023-11-01 | 18000 |
| Budi | 2023-12-01 | 43000 |
| Coki | 2024-01-01 | 80000 |
| Danu | 2024-02-01 | 18000 |

1. Output

Kami membuat perintah SELECT dengan metode subquery untuk mempermudah menghitung average\_daily\_spend, berikut query yang kami buat:

WITH MonthlyTransactions AS (

SELECT

DATE\_TRUNC(o.order\_date, MONTH) AS bulan,

o.fk\_customer\_id,

COUNT(\*) AS jumlah\_transaksi

FROM `aqueous-freedom-398208.OFD\_Team\_1.Order` o

GROUP BY bulan, o.fk\_customer\_id

),

MonthlyExpenditures AS (

SELECT

DATE\_TRUNC(o.order\_date, MONTH) AS bulan,

o.fk\_customer\_id,

SUM(f.price \* o.quantity) AS total\_pengeluaran

FROM `aqueous-freedom-398208.OFD\_Team\_1.Order` o

JOIN `aqueous-freedom-398208.OFD\_Team\_1.Food` f ON o.fk\_food\_id = f.id\_food

GROUP BY

bulan, o.fk\_customer\_id

),

CustomerData AS (

SELECT

c.id\_customer,

c.name AS nama\_customer

FROM

`aqueous-freedom-398208.OFD\_Team\_1.Customer` c

)

SELECT

cd.nama\_customer,

mt.bulan AS Bulan,

COALESCE(ROUND(me.total\_pengeluaran / mt.jumlah\_transaksi, 2), 0) AS rata\_rata\_pengeluaran\_per\_transaksi

FROM

MonthlyTransactions mt

LEFT JOIN

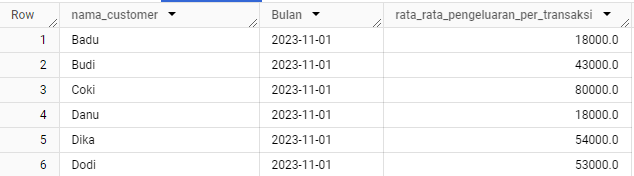
MonthlyExpenditures me ON mt.bulan = me.bulan AND mt.fk\_customer\_id = me.fk\_customer\_id

JOIN

CustomerData cd ON mt.fk\_customer\_id = cd.id\_customer

ORDER BY

Bulan ASC, cd.nama\_customer ASC;



1. Status: **Pass**

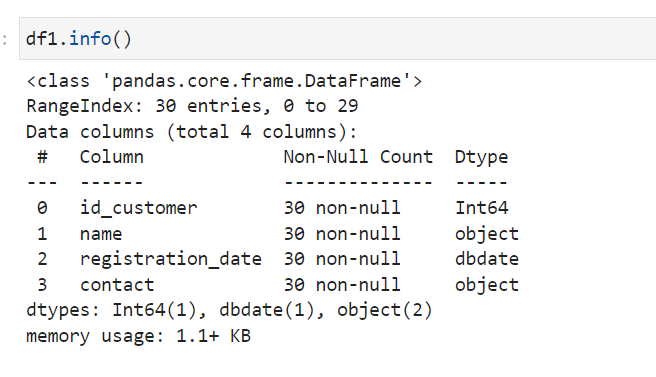
**Part 3**

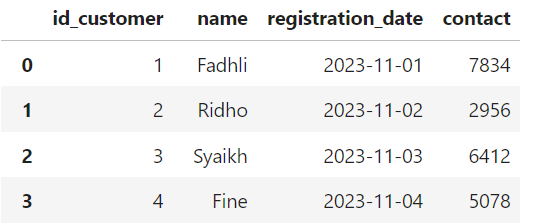
1. Apakah semua data yang masuk komplit? Semua column/row terisi ?  
     
   **A. Table Customer**

a. Expected Output

| Column | Data Type |
| --- | --- |
| id\_customer | int64 |
| name | object |
| registration\_date | dbdate |
| contact | object |

b. output   
 Pada table Customer kami membuat dataset mandiri (dummy) kemudian kami melakukan pengecekan data menggunakan jupyter lab dengan sintax  
df2.info()

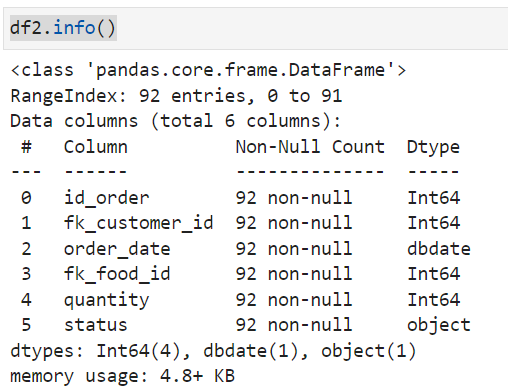




B. **Table Order** a. Expected Output

| Column | Data Type |
| --- | --- |
| id\_order | int64 |
| fk\_customer\_id | int64 |
| order\_date | dbdate |
| fk\_food\_id | Int64 |
| quantity | Int64 |
| status | Object |

b. output   
 Pada table Order kami membuat dataset mandiri (dummy) kemudian kami melakukan pengecekan data menggunakan jupyter lab dengan sintax  
df2.info()

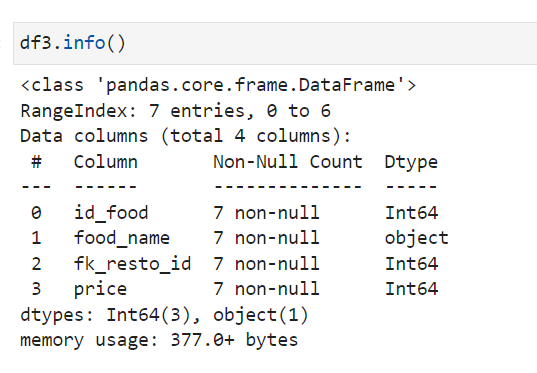


****

C. **Table Food**  
 a. Expected Output

| Column | Data Type |
| --- | --- |
| id\_food | int64 |
| food\_name | Object |
| fk\_resto\_id | Int64 |
| Price | Int64 |

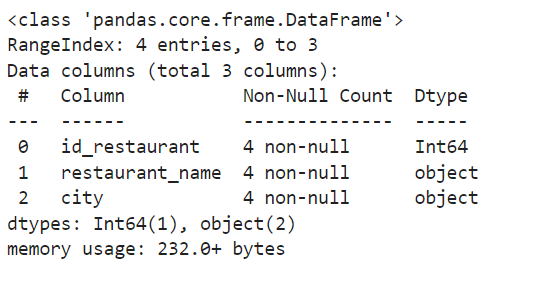
b. output   
 Pada table Food kami membuat dataset mandiri (dummy) kemudian kami melakukan pengecekan data menggunakan jupyter lab dengan sintax  
df3.info()



  
  
D. **Table Restaurant**  
 a. Expected Output

| Column | Data Type |
| --- | --- |
| id\_restaurant | int64 |
| restaurant\_name | Object |
| City | Object |

b. output   
 Pada table Restaurant kami membuat dataset mandiri (dummy) kemudian kami melakukan pengecekan data menggunakan jupyter lab dengan sintax  
df4.info()



  
Status : **PASS**

1. Apakah ada data type yang tidak sesuai..?

##### **Jawab :**

##### Berdasarkan penjelasan pada soal nomor 1, dapat disimpulkan bahwa tidak ada data yang tidak sesuai dengan kondisi yang direncanakan. Semua data telah sesuai dengan persyaratan yang ditetapkan

## Apakah ada pertanyaan utama tim business development sebelumnya masih bisa terjawab? Jawab : Terkait pertanyaan-pertanyaan tim pengembangan bisnis, semuanya telah dijawab dan tidak ada lagi yang tertinggal.

## 

**Part 4**

1. Segmentasi customer berdasarkan jumlah order dan uang yang sudah dikeluarkan di platform OK! Food Delivery?
   1. Expected output

| name | total\_order | total\_spend | level |
| --- | --- | --- | --- |
| Badu | 2 | 108000 | Supreme |
| Beri | 2 | 50000 | Grandmaster |
| Budi | 2 | 36000 | Master |
| Coki | 2 | 80000 | Supreme |

* 1. Output

Pada tabel berikut kami menghitung total\_spend untuk setiap costumer dan memberikan level sesuai dengan total spend masing masing customer

SELECT

Kategori.name,

kategori.total\_order,

kategori.total\_spend,

CASE

WHEN kategori.total\_order > 3 AND kategori.total\_spend >= 80000 THEN 'Big Spender'

WHEN kategori.total\_spend < 10000 THEN 'Beginner'

WHEN kategori.total\_spend < 40000 THEN 'Master'

WHEN kategori.total\_spend < 80000 THEN 'Grandmaster'

ELSE 'Supreme' END level

FROM

(

SELECT

c.name,

COUNT(\*) AS total\_order,

SUM(f.price\*o.quantity) AS total\_spend

FROM aqueous-freedom-398208.OFD\_Team\_1.Order o

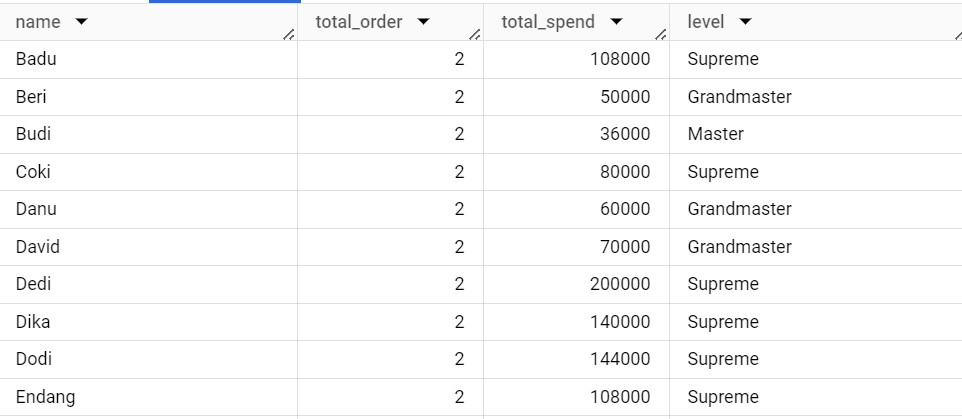
INNER JOIN aqueous-freedom-398208.OFD\_Team\_1.Food f ON f.id\_food = o.fk\_food\_id

INNER JOIN aqueous-freedom-398208.OFD\_Team\_1.Customer c ON c.id\_customer = o.id\_order

GROUP BY 1

) AS kategori

ORDER BY 1;



* 1. Status: **Pass**

1. Semua transaksi yang masih ‘menggantung’ dari D-1 agar bisa di*follow up* oleh tim operational / *customer service*
2. Expected Output

| Name | order\_date | status | contact |
| --- | --- | --- | --- |
| Fine | 2023-11-04 | pending | 5078 |
| Beri | 2023-11-06 | pending | 8645 |
| Budi | 2023-11-07 | pending | 3091 |
| Danu | 2023-11-11 | pending | 4276 |

1. Ouput

Kami membuat data dummy yang kemudian kami manipulasi menggunakan perintah SELECT pada tabel customer dengan menggunakan function WHERE guna mempermudah siapa saja yang berstatus pending

SELECT

c.name,

o.order\_date,

o.status,

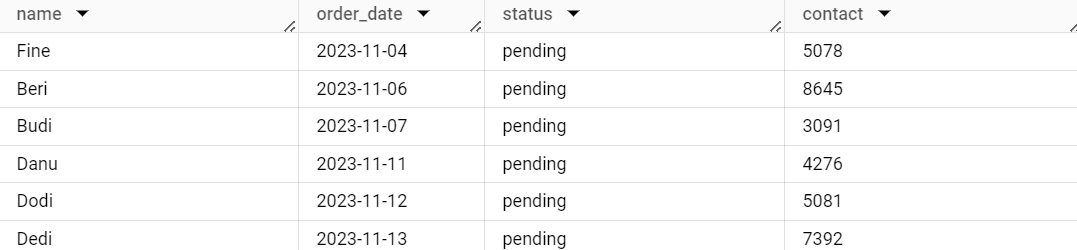
c.contact

FROM aqueous-freedom-398208.OFD\_Team\_1.Customer c

JOIN aqueous-freedom-398208.OFD\_Team\_1.Order o ON c.id\_customer = o.id\_order

WHERE o.status = 'pending'

ORDER BY o.order\_date ASC;



1. Status : **PASS**

**Part 5 (Dataset Tim-1)**

1. *Revenue harian per kota*
2. Expected output

| order\_date | city | daily\_revenue |
| --- | --- | --- |
| 2023-11-01 | Surabaya | 36000 |
| 2023-11-02 | Bandung | 36000 |

1. Output

SELECT

o.order\_date,

r.city,

SUM(f.price \* o.quantity) AS daily\_revenue

FROM `aqueous-freedom-398208.OFD\_Team\_1.Order` o

INNER JOIN`aqueous-freedom-398208.OFD\_Team\_1.Food` f ON o.fk\_food\_id = f.id\_food

INNER JOIN `aqueous-freedom-398208.OFD\_Team\_1.Restaurant` r ON f.fk\_resto\_id = r.id\_restaurant

GROUP BY order\_date,city

ORDER BY order\_date,city;



1. Status: **Pass**
2. *Customer segmentation report harian*
3. Expected output

| id\_customer | name | today\_date | last\_order | total\_transaction | total\_spending | segmentasi |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Arif | 2024-02-13 | 2024-01-14 | 1 | 75000 | New |
| 2 | Budi | 2024-02-13 | 2024-01-14 | 1 | 70000 | New |

1. Output

drop table if exists`aqueous-freedom-398208.OFD\_Team\_1.costumer\_segmentation`;

create table `aqueous-freedom-398208.OFD\_Team\_1.costumer\_segmentation` AS

WITH tabel AS (

SELECT

id\_customer,

c.name,

DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY) AS last\_order\_date,

DATE\_DIFF(o.order\_date, c.registration\_date, DAY) AS days\_since\_registration,

COUNT(\*) OVER (PARTITION BY c.name) AS total\_transaction,

SUM(f.price \* o.quantity) OVER (PARTITION BY c.name) AS total\_spending,

ROW\_NUMBER() OVER (PARTITION BY c.name ORDER BY o.order\_date DESC) AS row\_num,

LAG(o.order\_date) OVER (PARTITION BY c.name ORDER BY o.order\_date) AS prev\_order\_date

FROM aqueous-freedom-398208.OFD\_Team\_1.Customer c

LEFT JOIN aqueous-freedom-398208.OFD\_Team\_1.Order o ON c.id\_customer = o.fk\_customer\_id

LEFT JOIN aqueous-freedom-398208.OFD\_Team\_1.Food f ON f.id\_food = o.fk\_food\_id AND o.order\_date >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY)

WHERE c.registration\_date <= DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY)

AND o.order\_date >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY) -- Tambahkan kondisi untuk memfilter tanggal pesanan

)

SELECT

id\_customer,

name,

DATE\_SUB(CURRENT\_DATE(), INTERVAL 0 DAY) AS today\_date,

DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY) AS last\_order,

total\_transaction,

total\_spending,

CASE

WHEN total\_spending >= 500001 THEN 'Big Spender'

WHEN total\_transaction <= 1 THEN 'New'

WHEN total\_transaction <= 5 THEN 'Good'

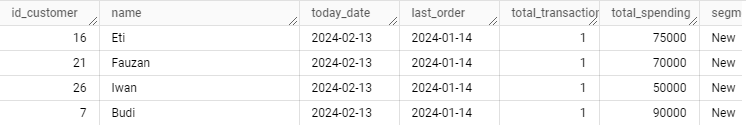
WHEN total\_transaction <= 10 THEN 'Potential'

ELSE 'Loyalist'

END AS segmentation

FROM tabel

ORDER BY last\_order\_date DESC;



1. Status: **Pass**
2. *Growth harian dan kontribusi revenue perkota*
3. Expected output

**Growth**

| Kota | Tanggal | Revenue\_hari\_kemarin | Revenue | Growth\_Rate\_Precentage |
| --- | --- | --- | --- | --- |
| Bandung | 2023-11-02 | 0 | 36000 | null |
| Bandung | 2023-11-04 | 36000 | 36000 | 0% |

**Kontribusi**

| date | city | total\_revenue | contribution |
| --- | --- | --- | --- |
| 2023-11-01 | Surabaya | 36000 | 100 |
| 2023-11-02 | Bandung | 36000 | 100 |

1. Output

**Growth**

drop table if exists `aqueous-freedom-398208.OFD\_Team\_1.growth\_harian\_perkota`;

create table `aqueous-freedom-398208.OFD\_Team\_1.growth\_harian\_perkota` AS

WITH DailyRevenue AS (

SELECT

r.city AS Kota,

o.order\_date AS Tanggal,

SUM(o.quantity \* f.price) AS Revenue,

LAG(SUM(o.quantity \* f.price), 1) OVER (PARTITION BY r.city ORDER BY o.order\_date) AS Revenue\_Hari\_Kemarin,

LAG(SUM(o.quantity \* f.price), 2) OVER (PARTITION BY r.city ORDER BY o.order\_date) AS Revenue\_Dua\_Hari\_Sebelumnya

FROM

`aqueous-freedom-398208.OFD\_Team\_1.Order` o

INNER JOIN

`aqueous-freedom-398208.OFD\_Team\_1.Food` f ON o.fk\_food\_id = f.id\_food

INNER JOIN

`aqueous-freedom-398208.OFD\_Team\_1.Customer` c ON o.fk\_customer\_id = c.id\_customer

INNER JOIN

`aqueous-freedom-398208.OFD\_Team\_1.Restaurant` r ON f.fk\_resto\_id = r.id\_restaurant

GROUP BY

r.city, o.order\_date

)

SELECT

Kota,

Tanggal,

COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya, 0) AS Revenue\_Hari\_Kemarin,

Revenue,

CASE

WHEN COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya) IS NULL THEN NULL

ELSE CONCAT(ROUND(((Revenue - COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya)) / NULLIF(COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya), 0)) \* 100, 2), '%')

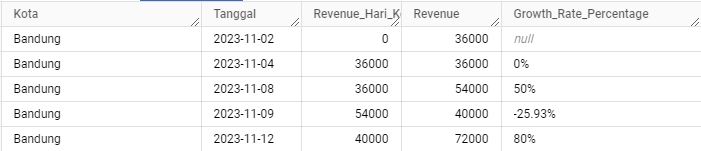
END AS Growth\_Rate\_Percentage

FROM

DailyRevenue

ORDER BY

Kota, Tanggal;



**Kontribusi**

drop table if exists `aqueous-freedom-398208.OFD\_Team\_1.kontribusi\_harian\_perkota`;

CREATE TABLE `aqueous-freedom-398208.OFD\_Team\_1.kontribusi\_harian\_perkota` AS

WITH daily\_total AS (

SELECT

order\_date,

SUM(f.price \* o.quantity) AS total\_revenue\_all\_cities

FROM `aqueous-freedom-398208.OFD\_Team\_1.Order` o

JOIN `aqueous-freedom-398208.OFD\_Team\_1.Food` f ON o.fk\_food\_id = f.id\_food

JOIN `aqueous-freedom-398208.OFD\_Team\_1.Restaurant` r ON f.fk\_resto\_id = r.id\_restaurant

GROUP BY

order\_date

)

SELECT

o.order\_date AS date,

r.city,

SUM(f.price \* o.quantity) AS total\_revenue,

ROUND((SUM(f.price \* o.quantity) / dt.total\_revenue\_all\_cities) \* 100, 2) AS contribution

FROM `aqueous-freedom-398208.OFD\_Team\_1.Order` o

JOIN `aqueous-freedom-398208.OFD\_Team\_1.Food` f ON o.fk\_food\_id = f.id\_food

JOIN `aqueous-freedom-398208.OFD\_Team\_1.Restaurant` r ON f.fk\_resto\_id = r.id\_restaurant

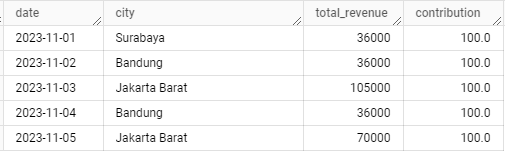
JOIN daily\_total dt ON o.order\_date = dt.order\_date

GROUP BY

r.city, o.order\_date, dt.total\_revenue\_all\_cities

ORDER BY

o.order\_date, r.city;



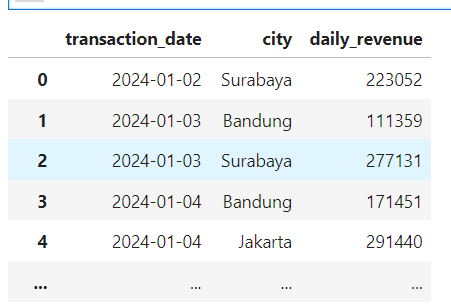
1. Status: **Pass**

**Part 5 (Dataset CodingID)**

1. *Revenue harian per kota*
2. Expected output

| transaction\_date | city | daily\_revenue |
| --- | --- | --- |
| 2024-01-02 | surabaya | 223052 |
| 2024-01-03 | bandung | 111359 |
| 2024-01-04 | surabaya | 277131 |
| 2024-01-05 | bandung | 171451 |
| 2024-01-06 | jakarta | 291440 |

1. Output



SELECT

t.transaction\_date,

r.kota AS city,

SUM(t.price) AS daily\_revenue

FROM`aqueous-freedom-398208.ok\_food\_dataset.transaction\_2024` t

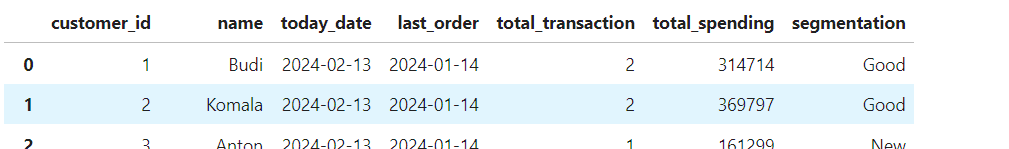
INNER JOIN`aqueous-freedom-398208.ok\_food\_dataset.resto` r ON t.resto\_id = r.id

GROUP BY t.transaction\_date, r.kota

ORDER BY t.transaction\_date, r.kota;

1. Status: **Pass**
2. *Customer segmentation report harian*
3. Expected output

| customer \_id | name | today \_date | last\_order | total\_transaction | total\_spending | segmentation |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | budi | 2024-02-13 | 2024-01-14 | 2 | 314714 | Good |

1. Output  
   

drop table if exists `aqueous-freedom-398208.OFD\_Team\_1.coding\_id\_customer\_segmentasi`;

CREATE TABLE `aqueous-freedom-398208.OFD\_Team\_1.coding\_id\_customer\_segmentasi`as

WITH tabel AS (

SELECT

c.id AS customer\_id,

c.name,

DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY) AS last\_order\_date,

COUNT(t.id) AS total\_transaction,

COALESCE(SUM(t.price), 0) AS total\_spending

FROM aqueous-freedom-398208.ok\_food\_dataset.customer c

LEFT JOIN aqueous-freedom-398208.ok\_food\_dataset.transaction\_2024 t ON c.id = t.customer\_id

WHERE t.transaction\_date >= DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY)

GROUP BY c.id, c.name

)

SELECT

customer\_id,

name,

DATE\_SUB(CURRENT\_DATE(), INTERVAL 0 DAY) AS today\_date,

DATE\_SUB(CURRENT\_DATE(), INTERVAL 30 DAY) AS last\_order,

total\_transaction,

total\_spending,

CASE

WHEN total\_spending >= 500001 THEN 'Big Spender'

WHEN total\_transaction <= 1 THEN 'New'

WHEN total\_transaction <= 5 THEN 'Good'

WHEN total\_transaction <= 10 THEN 'Potential'

ELSE 'Loyalist'

END AS segmentation

FROM tabel

ORDER BY last\_order\_date DESC;

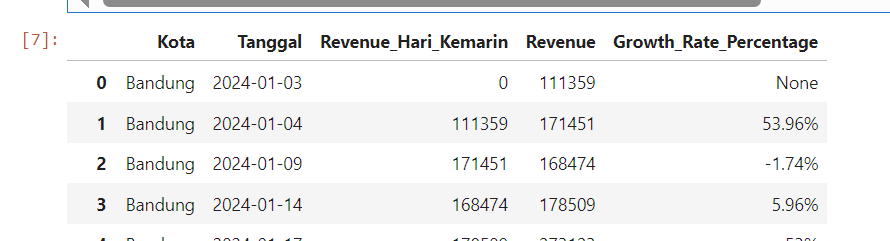
1. Status: **Pass**
2. *Growth harian dan kontribusi revenue perkota*
3. Expected output
4. *Growth harian*

| kota | tanggal | *revenue\_hari\_kemarin* | *revenue* | *Growt\_pescentage* |
| --- | --- | --- | --- | --- |
| *bandung* | *2024-01-03* | *0* | *111359* | *none* |

1. *kontribusi revenue perkota*

| *date* | *city* | *total\_revenue* | *contribution* |
| --- | --- | --- | --- |
| *2024-01-02* | *Surabaya* | *223052* | *100.0* |
| *2024-01-03* | *Bandung* | *111359* | *29.0* |
| *2024-01-03* | *Surabaya* | *277131* | *71.0* |

1. Output
2. ***Growth harian***



drop table if exists `aqueous-freedom-398208.OFD\_Team\_1.coding\_id\_\_growth\_harian`;

CREATE TABLE `aqueous-freedom-398208.OFD\_Team\_1.coding\_id\_\_growth\_harian`as

WITH DailyRevenue AS (

SELECT

r.kota AS Kota,

t.transaction\_date AS Tanggal,

SUM(t.price) AS Revenue,

LAG(SUM(t.price), 1) OVER (PARTITION BY r.kota ORDER BY t.transaction\_date) AS Revenue\_Hari\_Kemarin,

LAG(SUM(t.price), 2) OVER (PARTITION BY r.kota ORDER BY t.transaction\_date) AS Revenue\_Dua\_Hari\_Sebelumnya

FROM

`aqueous-freedom-398208.ok\_food\_dataset.transaction\_2024` t

INNER JOIN

`aqueous-freedom-398208.ok\_food\_dataset.resto` r ON t.resto\_id = r.id

GROUP BY

r.kota, t.transaction\_date

)

SELECT

Kota,

Tanggal,

COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya, 0) AS Revenue\_Hari\_Kemarin,

Revenue,

CASE

WHEN COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya) IS NULL THEN NULL

ELSE CONCAT(ROUND(((Revenue - COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya)) / NULLIF(COALESCE(Revenue\_Hari\_Kemarin, Revenue\_Dua\_Hari\_Sebelumnya), 0)) \* 100, 2), '%')

END AS Growth\_Rate\_Percentage

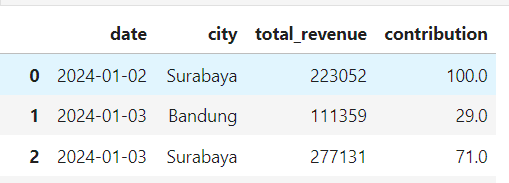
FROM

DailyRevenue

ORDER BY

Kota, Tanggal;

1. **Kontribusi harian**



drop table if exists `aqueous-freedom-398208.OFD\_Team\_1.coding\_id\_kontribusi\_harian\_perkota`;

CREATE TABLE `aqueous-freedom-398208.OFD\_Team\_1.coding\_id\_kontribusi\_harian\_perkota`as

WITH city\_daily\_revenue AS (

SELECT

t.transaction\_date AS date,

r.kota AS city,

SUM(t.price) AS total\_revenue

FROM

`aqueous-freedom-398208.ok\_food\_dataset.transaction\_2024` t

INNER JOIN

`aqueous-freedom-398208.ok\_food\_dataset.resto` r ON t.resto\_id = r.id

GROUP BY

t.transaction\_date, r.kota

),

daily\_total AS (

SELECT

transaction\_date AS order\_date,

SUM(price) AS total\_revenue\_all\_cities

FROM

`aqueous-freedom-398208.ok\_food\_dataset.transaction\_2024`

GROUP BY

transaction\_date

)

SELECT

cdr.date,

cdr.city,

cdr.total\_revenue,

ROUND((cdr.total\_revenue / dt.total\_revenue\_all\_cities) \* 100, 0) AS contribution

FROM

city\_daily\_revenue AS cdr

JOIN

daily\_total AS dt ON cdr.date = dt.order\_date

ORDER BY

cdr.date, cdr.city;

1. Status: **Pass**