**Muhammad Fathan Muttaqy**

**BATCH 24**

**DATA PREPARATION**

BEGIN;

CREATE TABLE IF NOT EXISTS public.customers\_dataset

(

customer\_id character varying COLLATE pg\_catalog."default",

customer\_unique\_id character varying COLLATE pg\_catalog."default",

customer\_zip\_code\_prefix character varying COLLATE pg\_catalog."default",

customer\_city character varying COLLATE pg\_catalog."default",

customer\_state character varying COLLATE pg\_catalog."default"

);

CREATE TABLE IF NOT EXISTS public.geolocation\_dataset

(

geolocation\_zip\_code\_prefix character varying COLLATE pg\_catalog."default",

geolocation\_lat character varying COLLATE pg\_catalog."default",

geolocation\_lng character varying COLLATE pg\_catalog."default",

geolocation\_city character varying COLLATE pg\_catalog."default",

geolocation\_state character varying COLLATE pg\_catalog."default"

);

CREATE TABLE IF NOT EXISTS public.order\_items\_dataset

(

order\_id character varying COLLATE pg\_catalog."default",

order\_item\_id integer,

product\_id character varying COLLATE pg\_catalog."default",

seller\_id character varying COLLATE pg\_catalog."default",

shipping\_limit\_date timestamp without time zone,

price double precision,

freight\_value double precision

);

CREATE TABLE IF NOT EXISTS public.order\_payments\_dataset

(

order\_id character varying COLLATE pg\_catalog."default",

payment\_sequential integer,

payment\_type character varying COLLATE pg\_catalog."default",

payment\_installments integer,

payment\_value double precision

);

CREATE TABLE IF NOT EXISTS public.order\_reviews\_dataset

(

review\_id character varying COLLATE pg\_catalog."default",

order\_id character varying COLLATE pg\_catalog."default",

review\_score integer,

review\_comment\_title character varying COLLATE pg\_catalog."default",

review\_comment\_message character varying COLLATE pg\_catalog."default",

review\_creation\_date timestamp without time zone,

review\_answer\_timestamp timestamp without time zone

);

CREATE TABLE IF NOT EXISTS public.orders\_dataset

(

order\_id character varying COLLATE pg\_catalog."default",

customer\_id character varying COLLATE pg\_catalog."default",

order\_status character varying COLLATE pg\_catalog."default",

order\_purchase\_timestamp timestamp without time zone,

order\_approved\_at timestamp without time zone,

order\_delivered\_carrier\_date timestamp without time zone,

order\_delivered\_customer\_date timestamp without time zone,

order\_estimated\_delivery\_date timestamp without time zone

);

CREATE TABLE IF NOT EXISTS public.product\_dataset

(

no integer,

product\_id character varying COLLATE pg\_catalog."default",

product\_category\_name character varying COLLATE pg\_catalog."default",

product\_name\_lenght integer,

product\_description\_lenght integer,

product\_photos\_qty integer,

product\_weight\_g integer,

product\_length\_cm integer,

product\_height\_cm integer,

product\_width\_cm integer

);

CREATE TABLE IF NOT EXISTS public.sellers\_dataset

(

seller\_id character varying COLLATE pg\_catalog."default",

seller\_zip\_code\_prefix integer,

seller\_city character varying COLLATE pg\_catalog."default",

seller\_state character varying COLLATE pg\_catalog."default"

);

ALTER TABLE IF EXISTS public.customers\_dataset

ADD FOREIGN KEY (customer\_id)

REFERENCES public.orders\_dataset (customer\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.geolocation\_dataset

ADD FOREIGN KEY (geolocation\_zip\_code\_prefix)

REFERENCES public.sellers\_dataset (seller\_zip\_code\_prefix) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.geolocation\_dataset

ADD FOREIGN KEY (geolocation\_zip\_code\_prefix)

REFERENCES public.customers\_dataset (customer\_zip\_code\_prefix) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.order\_payments\_dataset

ADD FOREIGN KEY (order\_id)

REFERENCES public.orders\_dataset (order\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.order\_reviews\_dataset

ADD FOREIGN KEY (order\_id)

REFERENCES public.orders\_dataset (order\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.orders\_dataset

ADD FOREIGN KEY (order\_id)

REFERENCES public.order\_items\_dataset (order\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

ALTER TABLE IF EXISTS public.product\_dataset

ADD FOREIGN KEY (product\_id)

REFERENCES public.order\_items\_dataset (product\_id) MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

NOT VALID;

END;

**Annual Customer Activity Growth Analysis**

**Menampilkan rata-rata jumlah customer aktif bulanan (monthly active user) untuk setiap tahun**

WITH mau AS(SELECT

DATE\_PART('month', o.order\_purchase\_timestamp) AS month

DATE\_PART('year', o.order\_purchase\_timestamp) AS year

COUNT(DISTINCT c.customer\_unique\_id) AS monthly\_active\_user

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 1, 2)

SELECT year,

ROUND(AVG(monthly\_active\_user), 2) AS average\_mau

FROM mau

GROUP BY 1

ORDER BY 1 ASC

**--------------------------------------------------------------------------------------------------------------**

**Menampilkan jumlah customer baru pada masing-masing tahun**

**WITH new\_customers AS**

(SELECT

MIN(o.order\_purchase\_timestamp) AS first\_order, c.customer\_unique\_id

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 2)

SELECT

DATE\_PART('year', first\_order) AS year,

COUNT(1) AS new\_customers

FROM new\_customers

GROUP BY 1

ORDER BY 1 ASC

**--------------------------------------------------------------------------------------------------------------**

**Menampilkan jumlah customer yang melakukan pembelian lebih dari satu kali (repeat order) pada masing-masing tahun**

WITH repeat\_order AS

(SELECT

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

c.customer\_unique\_id AS customer\_repeat,

COUNT(o.order\_id) AS total\_order

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 1, 2

HAVING COUNT(o.order\_id) > 1)

SELECT

year,

COUNT(DISTINCT customer\_repeat) AS repeat\_customers

FROM repeat\_order

GROUP BY 1

ORDER BY 1 ASC

**--------------------------------------------------------------------------------------------------------------**

**Menampilkan rata-rata jumlah order yang dilakukan customer untuk masing-masing tahun**

**WITH orders AS**

(SELECT

c.customer\_unique\_id AS customer,

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

COUNT(1) AS frequency\_purchase

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 1, 2)

SELECT

year,

ROUND(AVG(frequency\_purchase), 3) AS average\_orders

FROM orders

GROUP BY 1

ORDER BY 1 ASC

**--------------------------------------------------------------------------------------------------------------**

**Menggabungkan ketiga metrik yang telah berhasil ditampilkan menjadi satu tampilan tabel**

WITH mau AS

(SELECT

year,

ROUND(AVG(monthly\_active\_user), 1) AS average\_mau

FROM(SELECT

DATE\_PART('month', o.order\_purchase\_timestamp) AS month,

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

COUNT(DISTINCT c.customer\_unique\_id) AS monthly\_active\_user

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 1, 2) AS subq

GROUP BY 1),

new\_customers AS

(SELECT

year,

COUNT(new\_customers) AS new\_customers

FROM(SELECT

MIN(DATE\_PART('year', o.order\_purchase\_timestamp)) AS year,

c.customer\_unique\_id AS new\_customers

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 2) AS subq

GROUP BY 1),

repeat\_order AS

(SELECT

year,

COUNT(DISTINCT customer\_repeat) AS repeat\_customers

FROM(SELECT

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

c.customer\_unique\_id AS customer\_repeat,

COUNT(o.order\_id) AS total\_order

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 1, 2

HAVING COUNT(o.order\_id) > 1) AS subq

GROUP BY 1),

avg\_orders AS

(SELECT

year,

AVG(total\_order) AS average\_orders

FROM(SELECT

DISTINCT c.customer\_unique\_id AS customer,

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

COUNT(DISTINCT o.order\_id) AS total\_order

FROM orders\_dataset AS o

JOIN customers\_dataset AS c ON c.customer\_id = o.customer\_id

GROUP BY 1, 2) AS subq

GROUP BY 1)

SELECT

m.year AS year,

average\_mau,

new\_customers,

repeat\_customers,

average\_orders

FROM mau AS m

JOIN new\_customers AS nc ON nc.year = m.year

JOIN repeat\_order AS ro ON ro.year = m.year

JOIN avg\_orders AS ao ON ao.year = m.year

GROUP BY 1, 2, 3, 4, 5

**Annual Product Category Quality Analysis**

**Membuat tabel yang berisi informasi pendapatan/revenue perusahaan total untuk masing-masing tahun**

CREATE TABLE revenue\_each\_year AS

SELECT

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

SUM(oi.price + oi.freight\_value) AS revenue

FROM orders\_dataset AS o

JOIN order\_items\_dataset AS oi ON oi.order\_id = o.order\_id

WHERE o.order\_status = 'delivered'

GROUP BY 1

ORDER BY year

select \* from revenue\_each\_year

**--------------------------------------------------------------------------------------------------------------**

**Membuat tabel yang berisi informasi jumlah cancel order total untuk masing-masing tahun**

CREATE TABLE cancel\_per\_year AS

SELECT

DATE\_PART('year', order\_purchase\_timestamp) AS year,

COUNT(order\_id) AS canceled\_order

FROM orders\_dataset

WHERE order\_status = 'canceled'

GROUP BY 1

ORDER BY year ASC

select \* from cancel\_per\_year

**--------------------------------------------------------------------------------------------------------------**

**Membuat tabel yang berisi nama kategori produk yang memberikan pendapatan total tertinggi untuk masing-masing tahun**

CREATE TABLE top\_product\_category\_by\_revenue\_per\_year AS

SELECT

year,

top\_product\_category\_by\_revenue,

product\_category\_revenue

FROM(SELECT

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

p.product\_category\_name AS top\_product\_category\_by\_revenue,

SUM(price + freight\_value) AS product\_category\_revenue,

RANK() OVER(PARTITION BY DATE\_PART('year', o.order\_purchase\_timestamp)

ORDER BY SUM(oi.price + oi.freight\_value) DESC

) AS rank

FROM orders\_dataset AS o

JOIN order\_items\_dataset AS oi ON oi.order\_id = o.order\_id

JOIN product\_dataset AS p ON p.product\_id = oi.product\_id

WHERE order\_status = 'delivered'

GROUP BY 1, 2

) AS subq

WHERE rank = 1

select \* from top\_product\_category\_by\_revenue\_per\_year

**--------------------------------------------------------------------------------------------------------------**

**Membuat tabel yang berisi nama kategori produk yang memiliki jumlah cancel order terbanyak untuk masing-masing tahun**

CREATE TABLE top\_canceled\_product\_category\_by\_per\_year AS

SELECT

year,

top\_canceled\_product\_category,

canceled\_product\_category

FROM(SELECT

DATE\_PART('year', o.order\_purchase\_timestamp) AS year,

p.product\_category\_name AS top\_canceled\_product\_category,

COUNT(o.order\_id) AS canceled\_product\_category,

RANK() OVER(PARTITION BY DATE\_PART('year', order\_purchase\_timestamp)

ORDER BY COUNT(o.order\_id) DESC

) AS rank

FROM orders\_dataset AS o

JOIN order\_items\_dataset AS oi ON oi.order\_id = o.order\_id

JOIN product\_dataset AS p ON p.product\_id = oi.product\_id

WHERE order\_status = 'canceled'

GROUP BY 1, 2

) AS subq

WHERE rank = 1

select \* from top\_canceled\_product\_category\_by\_per\_year

**--------------------------------------------------------------------------------------------------------------**

**Menggabungkan informasi-informasi yang telah didapatkan ke dalam satu tampilan tabel**

SELECT

r.year,

tp.top\_product\_category\_by\_revenue,

tp.product\_category\_revenue,

r.revenue AS total\_revenue,

tc.top\_canceled\_product\_category,

tc.canceled\_product\_category,

co.canceled\_order AS total\_canceled\_order

FROM revenue\_each\_year AS r

JOIN cancel\_per\_year AS co ON co.year = r.year

JOIN top\_product\_category\_by\_revenue\_per\_year AS tp ON tp.year = r.year

JOIN top\_canceled\_product\_category\_by\_per\_year AS tc ON tc.year = co.year

**Analysis of Annual Payment Type Usage**

**Menampilkan jumlah penggunaan masing-masing tipe pembayaran secara all time diurutkan dari yang terfavorit (Hint: Perhatikan struktur (kolom-kolom apa saja) dari tabel akhir yang ingin didapatkan)**

SELECT

payment\_type,

COUNT(order\_id) AS payment\_type\_usage

FROM order\_payments\_dataset

GROUP BY 1

ORDER BY 2 DESC

**--------------------------------------------------------------------------------------------------------------**

**Menampilkan detail informasi jumlah penggunaan masing-masing tipe pembayaran untuk setiap tahun (Hint: Perhatikan struktur (kolom-kolom apa saja) dari tabel akhir yang ingin didapatkan)**

SELECT

payment\_type,

SUM(CASE WHEN year = 2016 THEN payment\_type\_usage ELSE 0 END) AS "2016",

SUM(CASE WHEN year = 2017 THEN payment\_type\_usage ELSE 0 END) AS "2017",

SUM(CASE WHEN year = 2018 THEN payment\_type\_usage ELSE 0 END) AS "2018",

SUM(payment\_type\_usage) AS sum\_payment\_type\_usage

FROM (SELECT

DATE\_PART('year', order\_purchase\_timestamp) AS year,

payment\_type,

COUNT(payment\_type) AS payment\_type\_usage

FROM orders\_dataset AS o

JOIN order\_payments\_dataset AS op ON op.order\_id = o.order\_id

GROUP BY 1, 2) AS subq

GROUP BY 1

ORDER BY 2 DESC