What is the average number of purchases made by loyalty program members compared to non-members?

```
--calculate the average of count of orders.id and group by loyalty_program

WITH orders_by_customer_loyalty AS (
SELECT COUNT(orders.id) AS count_orders,
 orders.customer_id,
 customers.loyalty_program
FROM core.orders
LEFT JOIN core.customers
 ON orders.customer_id = customers.id
GROUP BY customers.loyalty_program,
 orders.customer_id
HAVING count_orders >= 2)

SELECT AVG(count_orders) as avg_order_count,
loyalty_program,
FROM orders_by_customer_loyalty
GROUP BY loyalty_program;
```

How does the time between purchases differ between loyalty customers vs. non-loyalty customers?

```
WITH LastPurchase AS (
 SELECT customer_id,
 purchase_ts,
 DATE_DIFF(purchase_ts, LAG(purchase_ts) OVER(PARTITION BY customer_id ORDER BY
purchase_ts), DAY) AS time_since_last_purchase
FROM core.orders
GROUP BY 1,2
)
SELECT AVG(LastPurchase.time_since_last_purchase) AS avg_time_to_buy_again,
 customers.loyalty_program
FROM LastPurchase
LEFT JOIN core.customers
 ON LastPurchase.customer_id = customers.id
GROUP BY loyalty_program;
```

```sql
-- 1. What were the order counts and total sales in USD for Macbooks each year?
SELECT EXTRACT(year from purchase_ts) as purchase_year,
 COUNT(id) as order_counts,
 ROUND (SUM(usd_price),2) as total_sales
FROM core.orders
WHERE LOWER(product_name) LIKE '%macbook%'
GROUP BY 1
ORDER BY 1;

SELECT DATE_TRUNC(purchase_ts, year) as purchase_year,
 COUNT(id) as order_counts,
 ROUND (SUM(usd_price),2) as total_sales
FROM core.orders
WHERE LOWER(product_name) LIKE '%macbook%'
GROUP BY 1
ORDER BY 1;

SELECT EXTRACT(year from purchase_ts) as purchase_year,
     EXTRACT(month from purchase_ts) as purchase_month,
 COUNT(id) as order_counts,
 ROUND (SUM(usd_price),2) as total_sales
FROM core.orders
WHERE LOWER(product_name) LIKE '%macbook%'
GROUP BY 1,2
ORDER BY 1,2;



-- 2. Throughout 2019 to 2020, how many Macbooks were ordered in USD?
SELECT COUNT(id) as macbook_count,
FROM `core.orders`
WHERE LOWER(product_name) LIKE "%macbook%"
AND extract(year from purchase_ts) BETWEEN 2019 AND 2020
AND currency = 'USD';

--
-- 3. Return the unique combinations of product IDs and product names of all Apple
and Bose products.
SELECT DISTINCT product_name,
 product_id,
FROM core.orders
WHERE LOWER(product_name) LIKE "%macbook%"
 OR LOWER(product_name) LIKE "%apple%"
 OR LOWER(product_name) LIKE "%bose%"
ORDER BY 1 DESC;


WITH ProductIDs AS (
   SELECT DISTINCT product_name, product_id
   FROM core.orders
   WHERE LOWER(product_name) LIKE "%macbook%"
       OR LOWER(product_name) LIKE "%apple%"
       OR LOWER(product_name) LIKE "%bose%"
),
ProductCounts AS (
 SELECT product_name, COUNT(product_id) AS distinct_product_count
 FROM ProductIDs
 GROUP BY product_name
)
SELECT p.product_name, p.product_id, pc.distinct_product_count
FROM ProductIDs p
JOIN ProductCounts pc ON p.product_name = pc.product_name
ORDER BY p.product_name;

-- 1.What were the order counts, sales, and AOV for Macbooks sold in North America
for each quarter across all years?
--count of orders, sum of usd_sales, average of usd_sales, filtered by Macbooks and
region = NA. ALso filter by date_trunk (quarters)
SELECT date_trunc(orders.purchase_ts, quarter) as quarter,
 count (orders.id) as order_count,
 ROUND(SUM (orders.usd_price),2) as total_sales,
 round(AVG(orders.usd_price),2) as AOV
From core.orders
Left join core.customers on orders.customer_id = customers.id
Left join core.geo_lookup on geo_lookup.country = customers.country_code
 WHERE LOWER (orders.product_name) like '%macbook%'
 AND geo_lookup.region = 'NA'
GROUP BY 1
ORDER BY 1;
```

```
-- 1.1 What is the average quarterly order count and total sales for Macbooks sold
in North America? (i.e. “For North America Macbooks, average of X units sold per
quarter and Y in dollar sales per quarter”)
-- The average of count of orders.id and average of total sales, filtered by
Macbooks and North America.

with quarterly_metrics as (  -----CTE with quarterly metrics
   SELECT date_trunc(orders.purchase_ts, quarter) as purchase_quarter,
   count (orders.id) as order_count,
   ROUND(SUM (orders.usd_price),2) as total_sales
 From core.orders
 Left join core.customers on orders.customer_id = customers.id
 Left join core.geo_lookup on geo_lookup.country = customers.country_code
   WHERE LOWER (orders.product_name) like '%macbook%'
   AND geo_lookup.region = 'NA'
 GROUP BY 1
 ORDER BY 1
 )

SELECT ROUND(avg (order_count),2) as avg_order_count,
 ROUND(avg (total_sales),2) as avg_total_sales
From quarterly_metrics;



--2.For products purchased in 2022 on the website or products purchased on mobile
in any year, which region has the average highest time to deliver?
-- time to deliver by region. time to deliver = date diff (delivery_ts and
purchase_ts), days. Filter products where purchase_platform = mobile OR
purchase_platform = website AND purchase_ts = 2022; for the last part I need to
extract the year from purchase_ts. Final output has to retrieve avg of time to
deliver, filtered by distinct regions. Order time to deliver DESC.

 SELECT avg(datetime_diff(order_status.delivery_ts, order_status.purchase_ts, day))
as time_to_deliver,
   geo_lookup.region
 FROM core.order_status
 LEFT JOIN core.orders
```

```
   on orders.id = order_status.order_id
 LEFT JOIN core.customers
   on orders.customer_id = customers.id
 LEFT JOIN core.geo_lookup
   on customers.country_code = geo_lookup.country
 WHERE LOWER (orders.purchase_platform) LIKE '%mobile%' OR
   (LOWER (orders.purchase_platform) LIKE '%website%' AND extract(year from
orders.purchase_ts) = 2022)
 GROUP BY 2
 ORDER BY 1 DESC;

--2.2 For products purchased in 2022 on the website or Samsung products purchased
in 2021, which region has the average highest time to delive, in weeks?
-- I need to have an output with region and the average time to deliver, in weeks,
being time to deliver = diff between purchase and deliver, ranked DESC. Filtered by
any products purchase in 2022 on the purchase platform = website OR Samsung
products purchased in any platform in 2021

SELECT geo_lookup.region,
 ROUND(avg(date_diff(order_status.delivery_ts, order_status.purchase_ts, week)),2)
as average_weeks_to_deliver
FROM core.order_status
LEFT JOIN core.orders
 ON order_status.order_id = orders.id
LEFT JOIN core.customers
 ON customers.id = orders.customer_id
LEFT JOIN core.geo_lookup
 ON customers.country_code = geo_lookup.country
WHERE (LOWER (orders.purchase_platform) = 'website' AND extract (year from
order_status.purchase_ts) = 2022) OR
 (LOWER (orders.product_name) LIKE '%samsung%' AND extract (year from
order_status.purchase_ts) = 2021)
GROUP BY 1
ORDER BY 2;




-- 3.What was the refund rate and refund count for each product overall
```

```
--output: product_name, refund rate and refund count. Refund count = count of
is_refunded. Refund rate = create a helper column named is_refunded where refund_ts
not null = 1 else 0; then the avg of is_refunded
SELECT (CASE WHEN orders.product_name = '27in"" 4k gaming monitor' THEN '27in 4K
gaming monitor' ELSE orders.product_name END) as product_clean,
 COUNT(order_status.refund_ts) as refund_count,
 ROUND(AVG(CASE WHEN order_status.refund_ts IS NOT NULL THEN 1 ELSE 0 END),3) as
refund_rate
FROM core.orders
LEFT JOIN core.order_status
 ON orders.id = order_status.order_id
GROUP BY 1
ORDER BY 3;




-- 4.Within each region, what is the most popular product?
--The most popular product is the product with highest number of orders. 1)
calculate count of orders grouped by product name. 2) filtered by region and ranked
by number of orders.
--My solution:
WITH best_selling_products AS (
 SELECT (CASE WHEN orders.product_name = '27in"" 4k gaming monitor' THEN '27in 4K
gaming monitor' ELSE orders.product_name END) as product_clean,
   COUNT (orders.id) as total_orders,
   geo_lookup.region,
   ROW_NUMBER() over (PARTITION BY region ORDER BY count (orders.id) DESC) as
ranking
 FROM core.orders
 LEFT JOIN core.customers
   ON orders.customer_id = customers.id
 LEFT JOIN core.geo_lookup
   ON customers.country_code = geo_lookup.country
 GROUP BY 1, 3
 ORDER BY 3, 2 DESC)
SELECT *
FROM best_selling_products
WHERE ranking = 1;
```

```sql
--Christine's:
with sales_by_product as (
 select region,
   case when product_name = '27in"" 4k gaming monitor' then '27in 4K gaming
monitor' else product_name end as product_clean,
   count(distinct orders.id) as total_orders
 from core.orders
 left join core.customers
   on orders.customer_id = customers.id
 left join core.geo_lookup
   on geo_lookup.country = customers.country_code
 group by 1,2),

ranked_orders as (
 select *,
   row_number() over (partition by region order by total_orders desc) as
order_ranking
 from sales_by_product
 order by 4 asc)

select *
from ranked_orders
where order_ranking = 1;

--5.How does the time to make a purchase differ between loyalty customers vs.
non-loyalty customers?
--Need to define time to make a purchase = date difference between
customers.created_on and orders.purchase_ts. Pull average of this metric for
loyalty and non=loyalty.

SELECT ROUND(AVG(DATE_DIFF(orders.purchase_ts, customers.created_on, day)),2) as
avg_days_to_purchase,
 ROUND(AVG(DATE_DIFF(orders.purchase_ts, customers.created_on, week)),2) as
avg_weeks_to_purchase,
 customers.loyalty_program
FROM core.orders
LEFT JOIN core.customers
 ON orders.customer_id = customers.id
```

```
GROUP BY loyalty_program;

--6.Update this query to split the time to purchase per loyalty program, per
purchase platform. Return the number of records to benchmark the severity of nulls
SELECT ROUND(AVG(DATE_DIFF(orders.purchase_ts, customers.created_on, day)),2) as
avg_days_to_purchase,
 ROUND(AVG(DATE_DIFF(orders.purchase_ts, customers.created_on, week)),2) as
avg_weeks_to_purchase,
 customers.loyalty_program,
 orders.purchase_platform,
 COUNT(*) as row_count
FROM core.orders
LEFT JOIN core.customers
 ON orders.customer_id = customers.id
GROUP BY customers.loyalty_program, orders.purchase_platform;
```