**Advanced SQL Queries**

**Author:- Azarudhin Usman  Reference: Youtube- @AnalystAdithya**

**Introduction:**

**This document covers SQL topics like Aggregate functions, Joins, Windows Functions and joins.**

**-- Display the palindrome which has least character**
**-- Query**

```
with a as (SELECT word
FROM (
  VALUES
    ('Madam'),
    ('Bob'),
    ('Wow'),
    ('deed'),
      ('rotator'),
      ('Tech')
) AS word_table(word))

select word, reverse(word),
case when lower(word) = lower(reverse(word)) then 'Palindrome' else 'No'
end as Palin
from a
where length(word) = (select min(length(word)) from a )
```

**Output:**

```
with a as (SELECT word
FROM (
  VALUES
    ('Madam'),    ('Bob'),    ('Wow'),    ('deed'), ('rotator'),    ('Tech')
) AS word_table(word))

select word, reverse(word),
case when lower(word) = lower(reverse(word)) then 'Palindrome' else 'No' end as Palin,
count(word) from a
group by word
having length(word) = (select min(length(word)) from a )
```

Data Output  Messages  Notifications

| | word text | reverse text | palin text | count bigint |
|---|---|---|---|---|
| 1 | Bob | boB | Palindrome | 1 |
| 2 | Wow | woW | Palindrome | 1 |

**-- Display the word which are not palindrome without creating more than 1 cte**

**--Query**

```
with a as (SELECT word
FROM (
  VALUES
    ('Madam'),
    ('Bob'),
    ('Wow'),
    ('deed'),
      ('rotator'),
      ('Tech')
) AS word_table(word))

select word, reverse(word),
case when lower(word) = lower(reverse(word)) then 'Palindrome' else 'No'
end as Palin
from a
where substring(lower(word),1,1)!= right(lower(word),1)
```

**Output:**

Query  Query History

```
1  with a as (SELECT word
2  FROM (
3    VALUES
4      ('Madam'),    ('Bob'),    ('Wow'),    ('deed'), ('rotator'),    ('Tech')
5  ) AS word_table(word))
6
7  select word, reverse(word),
8  case when lower(word) = lower(reverse(word)) then 'Palindrome' else 'No' end as Palin
9  from a
10 where substring(lower(word),1,1)!= right(lower(word),1)
11
```

Data Output  Messages  Notifications

| | word text | reverse text | palin text |
|---|---|---|---|
| 1 | Tech | hceT | No |

**********

**-- Find the relative propotion of winning money**

```
with a as (SELECT name,prize
FROM (
  VALUES
    ('Allen',200),
    ('Bob',300),
    ('Mike',100),
```

```
    ('Blake',400)
) AS word_table(name, prize))
```

**--Query**

```
select name, prize, round(prize/totalsum*100,2) as Winning_Propotion
from (select name, prize, sum(prize)over()::decimal as totalsum from a)b
```

```
with a as (SELECT name,prize
FROM (
  VALUES
    ('Allen',200),
    ('Bob',300),
    ('Mike',100),
    ('Blake',400)
) AS word_table(name, prize))

select name, prize, round(prize/totalsum*100,2) as Winning_Propotion
from (select name, prize, sum(prize)over()::decimal as totalsum from a)b
```

Data Output  Messages  Notifications

| | name text | prize integer | winning_propotion numeric |
|---|---|---|---|
| 1 | Allen | 200 | 20.00 |
| 2 | Bob | 300 | 30.00 |
| 3 | Mike | 100 | 10.00 |
| 4 | Blake | 400 | 40.00 |

**********

**-- Leet code problem**

**Find the latest price change of a product on or before a particular date. (example - 2020-08-23)**

**Sample Output:**

| | customer_id integer | new_value integer |
|---|---|---|
| 1 | 1 | 35 |
| 2 | 2 | 50 |
| 3 | 3 | 10 |

**If no price change, then give *10 for that product**

```
create table orderdate (customer_id integer, prize integer, changed_date
date)


Insert into orderdate
  VALUES
    (1,20,'2020-08-21'),
    (2,50,'2020-08-21'),
    (1,30,'2020-08-22'),
    (1,35,'2020-08-23'),
      (2,65,'2020-08-24'),
      (3,20,'2020-08-24')
```

**-- Query**

```
select customer_id, new_value from (
with a as(
select * from orderdate where changed_Date <= '2020-08-23'),
b as (select distinct customer_id from orderdate)

select b.customer_id, changed_Date, case when changed_Date <= '2020-08-23'
then prize else 10 end as new_value,
rank() over(partition by b.customer_id order by a.changed_date desc) as
ranking
from b left join a on b.customer_id = a.customer_id) c where ranking =1
```

**Output:**

```
select customer_id, new_value from (
with a as(
select * from orderdate where changed_Date <= '2020-08-23'),
b as (select distinct customer_id from orderdate)

select b.customer_id, changed_Date,
case when changed_Date <= '2020-08-23' then prize else 10 end as new_value,
rank() over(partition by b.customer_id order by a.changed_date desc) as ranking
from b left join a on b.customer_id = a.customer_id) c where ranking =1
```

Data Output | Messages | Notifications

| | customer_id integer | new_value integer |
|---|---|---|
| 1 | 1 | 35 |
| 2 | 2 | 50 |
| 3 | 3 | 10 |

**-- Find the total non-mac revenue made from customers who ordered the MAC**

**Sample Output:**

| customer_id integer | nonrevenue bigint |
|---|---|
| 1 | 140000 |
| 3 | 50000 |
| 4 | 0 |

**-- Query**

```
with a as (select * from (
values
      (1,'iphone',50000),
      (1,'pc',90000),
      (1,'mac',80000),
      (2,'iphone',50000),
      (3,'iphone',50000),
      (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000)
) as apple_store(customer_id, products, rate))

select customer_id, sum(case when products = 'mac' then 0 else rate end)
as nonrevenue
from a where customer_id in (select distinct customer_id from a where
products='mac')
group by customer_id
order by customer_id
```

**Output:**

```
with a as (select * from (
values
    (1,'iphone',50000),
    (1,'pc',90000),
    (1,'mac',80000),
    (2,'iphone',50000),
    (3,'iphone',50000),
    (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000)
) as apple_store(customer_id, products, rate))

select customer_id, sum(case when products = 'mac' then 0 else rate end) as nonrevenue
from a where customer_id in (select distinct customer_id from a where products='mac')
group by customer_id
order by customer_id
```

Data Output  Messages  Notifications

| | customer_id integer | nonrevenue bigint |
|---|---|---|
| 1 | 1 | 140000 |
| 2 | 3 | 50000 |
| 3 | 4 | 0 |

**-- Display the customer who purchased only Mac**

**Sample Output:**

| customer_id integer | products text | rate integer |
|---|---|---|
| 4 | mac | 80000 |

**-- Query**

```
with a as (select * from (
values
      (1,'iphone',50000),
      (1,'pc',90000),
      (1,'mac',80000),
      (2,'iphone',50000),
      (3,'iphone',50000),
      (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000),
(5,'airpods',2000)
) as apple_store(customer_id, products, rate))

select * from a where customer_id not in (select customer_id from a where
products !='mac')
```

**Output:**

```
1  with a as (select * from (
2  values
3      (1,'iphone',50000),
4      (1,'pc',90000),
5      (1,'mac',80000),
6      (2,'iphone',50000),
7      (3,'iphone',50000),
8      (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000), (5,'airpods',2000)
9  ) as apple_store(customer_id, products, rate))
10
11 select * from a where customer_id not in (select customer_id from a where products !='mac')
```

Data Output  Messages  Notifications

| | customer_id integer | products text | rate integer |
|---|---|---|---|
| 1 | 4 | mac | 80000 |

**-- Display the customers who purchased more than 2 orders**

**Sample Output:**

| | customer_id integer | count bigint |
|---|---|---|
| 1 | 1 | 3 |
| 2 | 2 | 2 |
| 3 | 3 | 2 |

**-- Query**

```
with a as (select * from (
values
      (1,'iphone',50000),
      (1,'pc',90000),
      (1,'mac',80000),
      (2,'iphone',50000),
      (3,'iphone',50000),
      (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000),
(5,'airpods',2000)
) as apple_store(customer_id, products, rate))

select customer_id, count(*) from a group by customer_id having count(*) >=2
```

**Output:**

```
with a as (select * from (
values
    (1,'iphone',50000),
    (1,'pc',90000),
    (1,'mac',80000),
    (2,'iphone',50000),
    (3,'iphone',50000),
    (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000), (5,'airpods',2000)
) as apple_store(customer_id, products, rate))

select customer_id, count(*) from a group by customer_id having count(*) >=2
```

Data Output Messages Notifications

| | customer_id integer | count bigint |
|---|---|---|
| 1 | 1 | 3 |
| 2 | 2 | 2 |
| 3 | 3 | 2 |

**-- Display the customer who order same product more than once.**

**Sample Output:**

| customer_id integer | count bigint |
|---|---|
| 2 | 2 |
| 5 | 2 |

**-- Query**

```
with a as (select * from (
values
      (1,'iphone',50000),
      (1,'pc',90000),
      (1,'mac',80000),
      (2,'iphone',50000),
      (3,'iphone',50000),
      (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000),
(5,'airpods',2000),(5,'airpods',2000)
) as apple_store(customer_id, products, rate))

select customer_id, count(*) from a group by products, customer_id having
count(*) =2
```

**Output:**

```
with a as (select * from (
values
    (1,'iphone',50000),
    (1,'pc',90000),
    (1,'mac',80000),
    (2,'iphone',50000),
    (3,'iphone',50000),
    (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000), (5,'airpods',2000),(5,'airpods',2000)
) as apple_store(customer_id, products, rate))

select customer_id,products, count(*) from a group by products, customer_id having count(*) =2
```

Data Output Messages Notifications

| | customer_id integer | products text | count bigint |
|---|---|---|---|
| 1 | 2 | iphone | 2 |
| 2 | 5 | airpods | 2 |

**-- Find the customer who purchased in the apple store which is >=25% of that revenue.**

**Sample Output:**

| | customer_id integer | revernuepercent numeric |
|---|---|---|
| 1 | 3 | 26.97 |
| 2 | 1 | 45.64 |

**-- Query**

```
with a as (select * from (
values
      (1,'iphone',50000),
      (1,'pc',90000),
      (1,'mac',80000),
      (2,'iphone',50000),
      (3,'iphone',50000),
      (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000),
(5,'airpods',2000),(5,'airpods',2000)
) as apple_store(customer_id, products, rate)),
b as (
select distinct customer_id, (sum(rate) over(partition by
customer_id)::decimal) as amountpercustomer,
      sum(rate) over()::decimal as totalrevenue from a
group by customer_id, rate) --select * from b

select b.customer_id, round((b.amountpercustomer/b.totalrevenue)*100,2)
from b
where round((b.amountpercustomer/b.totalrevenue)*100,2) > 25;
```

**Output:**

```
with a as (select * from (
values
    (1,'iphone',50000), (1,'pc',90000), (1,'mac',80000),(2,'iphone',50000), (3,'iphone',50000),
    (3,'mac',80000),(2,'iphone',50000), (4,'mac',80000), (5,'airpods',2000),(5,'airpods',2000)
) as apple_store(customer_id, products, rate)),
b as (
select distinct customer_id, (sum(rate) over(partition by customer_id)::decimal) as amountpercustomer,
    sum(rate) over()::decimal as totalrevenue from a
group by customer_id, rate) --select * from b

select b.customer_id, round((b.amountpercustomer/b.totalrevenue)*100,2) revernuepercent from b
where round((b.amountpercustomer/b.totalrevenue)*100,2) > 25;
```

Data Output  Messages  Notifications

| | customer_id integer | revernuepercent numeric |
|---|---|---|
| 1 | 3 | 26.97 |
| 2 | 1 | 45.64 |

**********

**-- Find the customer who took longest time between the first order and second order.**

**Sample output:**

| customerid integer | datedifference integer |
|---|---|
| 103 | 8 |

**-- Query**

```
with c as (
with b as (
with a as (select * from (
values (1,101,'2023-12-11'),(2,102,'2023-12-12'),(3,103,'2023-12-13'),
      (4,101,'2023-12-19'),(5,102,'2023-12-20'),(6,103,'2023-12-21'),
      (7,101,'2023-12-12'),(8,102,'2023-12-18')
) as datedetails(id,customerid, orderdate))

select *, rank() over(partition by customerid order by orderdate) as ranking
from a)
select customerid, orderdate, lag(orderdate)over(partition by customerid)
as lag_Details,
(orderdate::date-lag(orderdate)over(partition by customerid)::date) as
datedifference
from b where ranking <=2 )

select customerid, datedifference from c
where datedifference is not null
order by datedifference desc limit 1
```

**Output:**

```
with c as (
with b as (
with a as (select * from (
values (1,101,'2023-12-11'),(2,102,'2023-12-12'),(3,103,'2023-12-13'),
    (4,101,'2023-12-19'),(5,102,'2023-12-20'),(6,103,'2023-12-21'),
    (7,101,'2023-12-12'),(8,102,'2023-12-18')
) as datedetails(id,customerid, orderdate))
select *, rank() over(partition by customerid order by orderdate) as ranking from a)
select customerid, orderdate, lag(orderdate)over(partition by customerid) as lag_Details,
(orderdate::date-lag(orderdate)over(partition by customerid)::date) as datedifference
from b where ranking <=2 )

select customerid, datedifference from c
where datedifference is not null
order by datedifference desc limit 1
```

Data Output  Messages  Notifications

| | customerid integer | datedifference integer |
|---|---|---|
| 1 | 103 | 8 |

**-- Display the customer who ordered only once.**

**Sample output:**

| customerid integer | count bigint |
|---|---|
| 104 | 1 |

**-- Query**

```
with a as (select * from (
values (1,101,'2023-12-11'),(2,102,'2023-12-12'),(3,103,'2023-12-13'),
      (4,101,'2023-12-19'),(5,102,'2023-12-20'),(6,103,'2023-12-21'),
      (7,101,'2023-12-12'),(8,102,'2023-12-18'), (9,104,'2023-12-01')
) as datedetails(id,customerid, orderdate))

      select customerid, count(*) from a group by customerid
      having count(*) = 1
```

**Output:**

```
with a as (select * from (
values (1,101,'2023-12-11'),(2,102,'2023-12-12'),(3,103,'2023-12-13'),
    (4,101,'2023-12-19'),(5,102,'2023-12-20'),(6,103,'2023-12-21'),
    (7,101,'2023-12-12'),(8,102,'2023-12-18'), (9,104,'2023-12-01')
) as datedetails(id,customerid, orderdate))

    select customerid, count(*) from a group by customerid
    having count(*) = 1
```

Data Output Messages Notifications

| | customerid integer | count bigint |
|---|---|---|
| 1 | 104 | 1 |

**-- Find the last product purchased by each customer without using rank function**

**Sample Output:**

| | customer_id integer | last_orderid integer |
|---|---|---|
| 1 | 1 | 3 |
| 2 | 5 | 10 |
| 3 | 4 | 8 |
| 4 | 2 | 7 |
| 5 | 3 | 6 |

**-- query**

```
with a as (select * from (
values
      (1,1,'iphone',50000),
      (2,1,'pc',90000),
      (3,1,'mac',80000),
      (4,2,'iphone',50000),
      (5,3,'iphone',50000),
      (6,3,'mac',80000),(7,2,'iphone',50000), (8,4,'mac',80000),
(9,5,'airpods',2000),(10,5,'airpods',2000)
) as apple_store(orderid, customer_id, products, rate)),
b as (
select customer_id, max(orderid) as last_orderid from a
group by customer_id)
select * from b;
```

**Output:**

```
with a as (select * from (
values
    (1,1,'iphone',50000),
    (2,1,'pc',90000),
    (3,1,'mac',80000),
    (4,2,'iphone',50000),
    (5,3,'iphone',50000),
    (6,3,'mac',80000),(7,2,'iphone',50000), (8,4,'mac',80000), (9,5,'airpods',2000),(10,5,'airpods',2000)
) as apple_store(orderid, customer_id, products, rate)),
b as (
select customer_id, max(orderid) as last_orderid from a
group by customer_id)
select * from b;
```

Data Output Messages Notifications

| | customer_id integer | last_orderid integer |
|---|---|---|
| 1 | 1 | 3 |
| 2 | 5 | 10 |
| 3 | 4 | 8 |
| 4 | 2 | 7 |

**-- Display along with the last ordered products**

**Sample Output:**

| | customer_id integer | reverse text |
|---|---|---|
| 1 | 1 | pc |
| 2 | 2 | iphone |
| 3 | 3 | iphone |
| 4 | 4 | mac |
| 5 | 5 | airpods drone |

**-- Query**

```
with a as (select * from (
values
      (1,1,'iphone',50000),
      (2,1,'pc',90000),
      (3,1,'mac',80000),
      (4,2,'iphone',50000),
      (5,3,'iphone',50000),
      (6,3,'mac',80000),(7,2,'iphone',50000), (8,4,'mac',80000),
(9,5,'airpods',2000)
) as apple_store(orderid, customer_id, products, rate)),
b as (
select customer_id,  string_agg(products, ',') as concatenated_order from
a
group by customer_id)
select customer_id,case
when length(reverse(split_part(reverse(concatenated_order),',',1)))>0
then  reverse(split_part(reverse(concatenated_order),',',1))
else concatenated_order end as lastorder
from b
order by customer_id
```

**Output:**

```sql
with a as (select * from (
values
    (1,1,'iphone',50000),
    (2,1,'pc',90000),
    (3,1,'mac',80000),
    (4,2,'iphone',50000),
    (5,3,'iphone',50000),
    (6,3,'mac',80000),(7,2,'iphone',50000), (8,4,'mac',80000), (9,5,'airpods',2000),(10,5,'airpods drone',2000
) as apple_store(orderid, customer_id, products, rate)),
b as (select customer_id,  string_agg(products, ',') as concatenated_order from a
group by customer_id)
select customer_id,reverse(split_part(reverse(concatenated_order),',',1)) from b order by customer_id
```

Data Output  Messages  Notifications

| | customer_id integer | reverse text |
|---|---|---|
| 1 | 1 | pc |
| 2 | 2 | iphone |
| 3 | 3 | iphone |
| 4 | 4 | mac |
| 5 | 5 | airpods drone |

**********