Case Study New York City Taxi QuestionsDue

Note: There are two tables in this database; trips_sep & trips_oct.

1)

a) Find the third-most expensive trip (Total Amount column) in September.

```
select total_amount as third_most_expensive_trip_sep from trips_sep
order by total_amount desc
limit 1
offset 2;
```

I solved it by using order by desc clause which will order the total amount in max to low order and I used offset and limit to retrieve the 3rd value.

b) Find the most expensive trip per mile (Total Amount/Mile) in October.

```
select
case
    when trip_distance > 1 then total_amount/trip_distance
    else total_amount
end as most_expensive_trip_per_mile, total_amount, trip_distance
from trips_oct
order by most_expensive_trip_per_mile desc
    limit 1;
```

Here I have used case statement to calculate amount per mile value as we have some zero value in trip_distance column which will create division by zero error.
So if the trip distance is less than 1, this will result the total amount as it is.
And I have used order by desc clause to order the calculated amount in max to low order and limit to retrieve the first value.

c) Find the most generous trip in September (highest tip).

```
select max(tip_amount) as most_generous_trip from trips_sep;
```

For this query I have used max aggregate function to retrieve the highest tip amount.

d) Find the longest trip duration in September.

```
select max(age(lpep_dropoff_datetime, lpep_pickup_datetime)) as
longest_trip_duration
from trips_sep;
```

Here I have used age function to retrieve the difference between drop off time and pickup time which will result trip duration and max aggregate function to retrieve the longest trip distance.

e) Find the average tip amount by the hour in September.

```
select avg(tip_amount) as average_tip_amount from trips_oct;
```

I have used avg aggregate function to retrieve the average tip amount

2) Find the number of trips by day of the week in October. (Create a "Day of Week" column,e.g., Monday, Tuesday, ..., Sunday).

```
select count(*) as number_of_trips_by_day, to_char(lpep_pickup_datetime,

'Day') as day_of_week from trips_oct


group by day_of_week;
```

In this I have used to_char function to retrieve the day of a week and count function to count the number of trips and group by to group the count based on day of the week.

3) Find the average trip amount by the hour in October.

```
select round(avg(total_amount)::numeric,2) as average_trip_amount_by_hour,
date_part('hour',lpep_dropoff_datetime) as hour
from trips_oct
group by hour
order by hour;
```

Here I have used date-part function to extract the hour of the trip. For this I have drop off time because the trip amount will be decide when the meter is disengaged. Then I have used avg aggregate function and round function to retrieve average amount. I have casted the value to numeric as round function only accepts numeric value. Then I have grouped by hour.

4) Determine which airport welcomes more customers: JFK or EWR. Note: Use a CASEexpression to retrieve the names as "JFK," "Group Ride," or "Newark" from the ***ratecodeid column***. Refer to the Data Dictionary file.

```
select count(*),
case
        when ratecodeid = 1 then 'Standard rate'
        when ratecodeid = 2 then 'JFK'
        when ratecodeid = 3 then 'Newark'
        when ratecodeid = 4 then 'Nassau or Westchester'
        when ratecodeid = 5 then 'Negotiated fare'
        when ratecodeid = 6 then 'Group ride'
end     as rate_code
from trips_sep
group by rate_code
order by count(*) desc;
```

From the output of this query we can say that JFK welcomes more customer than EWR.

In this query I have used case expression to name each and every entity of ratecodeid

And used count and group by to calculate the number trips to each and every entity.

5) Create buckets or price ranges for the total amount and find the number of trips in eachprice range for each driver in September.

Use the following price ranges:

- 0 <= Total Amount < 10
- 10 <= Total Amount < 20
- 20 <= Total Amount < 30
- 30 <= Total Amount < 40
- Total Amount >= 40; ELSE

```
select count(*),
case
        when total_amount >= 0 and total_amount < 10 then 'Very Low'
        when total_amount >= 10 and total_amount < 20 then 'Low'
        when total_amount >= 20 and total_amount < 30 then 'Medium'
        when total_amount >= 30 and total_amount < 40 then 'High'
        when total_amount >=40 then 'Very High'
        else 'other'
end as price_range,
driver_id
from trips_sep
group by price_range,driver_id
order by count(*) desc;
```

Here I have used the same case expression to divide each and every price range and then grouped the count by price range and driver.

6) Write a query to find the top three highest total amounts for each driver in October.

```
select total_amount, driver_id from(
select total_amount, driver_id,
row_number() over(partition by driver_id order by total_amount desc) as rank
from trips_oct)
where rank <= 3;
```

Here I have used a window function to give row numbers to total amount partitioned by drive and sorted in max to low in the subquery.
Then I have filtered the first 3 values to get the top three highest total amount of each drivers in the main query.

7) Find the 10 lowest total amounts for Driver 1 in October.

select total_amount, driver_id from trips_oct
where driver_id = 1
order by total_amount
limit 10;

For this I have used simple filter condition to filter driver 1 and sorted it by low to max.

8) Write a query to track the cumulative earnings of Driver 1 after each trip in

October.(Hint: Running total, Window functions).

select trip_id,

sum(total_amount) over (order by trip_id) as running_total,

row_number() over(order by trip_id) as trip_number

from trips_oct

where driver_id = 1;

For this query I have used aggregate function as a window function to

calculate the running total. And also I have given row numbers for each

trip to know the trip number. Finally filtered the output for driver id 1.

9) Is there any new driver in October? (Hint: Find drivers who exist in the October table

butnot in the September table)

Note: Return unique driver IDs.

select distinct(driver_id) as new_drivers from trips_oct
where driver_id not in (select driver_id from trips_sep);

Here I have used the main query to retrieve the driver id of oct and filter the one which are not in sep through the sub query.

10) Find the total amount difference between September and October. In the output, display the total amount for September, the total amount for October, and the difference betweenthe two.

```
with sep_total as (
select round(sum(total_amount)::numeric, 2) as sep_amount from trips_sep
),
oct_total as (
select round(sum(total_amount)::numeric, 2) as oct_amount from trips_oct
)
select sep_amount, oct_amount, (sep_amount-oct_amount) as difference
from sep_total, oct_total;
```

Here I have used cte's to retrieve total amount from sep and oct table and then used them to find the difference. Used round function with casting the double precision to numeric to show the output in 2 decimals.

11) Find the total revenue (Total Amount) for each driver in both September and October. Ensure the output displays the total revenue for September, the total revenue for October,and the difference between the two. Sort the total revenue in descending order.

```
with sep_total as (
select round(sum(total_amount)::numeric, 2)as sep_amount, driver_id  from
trips_sep
group by driver_id
),
oct_total as (
select round(sum(total_amount)::numeric, 2)as oct_amount, driver_id  from
trips_oct
group by driver_id
)
SELECT
   o.driver_id,
   s.sep_amount,
   o.oct_amount,
case
        when sep_amount is null then oct_amount
   else (o.oct_amount - s.sep_amount)
end AS revenue_difference
FROM oct_total o
left JOIN sep_total s ON o.driver_id = s.driver_id
order by oct_amount desc;
```

Here I have used cte’s to retrieve total amount of each month for respective drivers.

Then I have left joined the cte’s to get all the driver’s from oct. Finally I have used case expression to deal with the null values of new driver revenue in sep

month.

12) Find the total revenue (Total Amount) by day of the week (Monday, Tuesday, ..., Sunday)for both September and October. Ensure the output displays the total revenue for each day of the week in September, the total revenue for each day of the week in October, andthe difference between the two. Sort the total revenue in descending order.

```
with sep_day as(
select to_char(lpep_pickup_datetime, 'Day')  as day, round(sum(total_amount)::numeric,
2) as sep_amount
from trips_sep
group by day
),
oct_day as(
select to_char(lpep_pickup_datetime, 'Day')  as day, round(sum(total_amount)::numeric,
2) as oct_amount
from trips_oct
group by day
)
select o.day, sep_amount, oct_amount, (sep_amount - oct_amount) as diference
from sep_day s
join oct_day o
```

using(day)

order by oct_amount desc;

Same as the previous question I have used cte's to retrieve the total amount of months and group by day of week. Finally using join I arrived at the desired output.