SQL Date and Time and CTE

Quiz

- ”29-08-2019”, what is the right parsing string to convert this into a date format
  - “%Y-%m-%d”
  - “%d:%m:%Y”
  - “%m-%d-%Y”
  - “%d-%m-%Y” - correct
- Given data for 1 year, To find which month each customer made their first purchase and show on each row of the customer?
  - First_value(extract (month from date)) over (partition by customer_id order by Date desc)
  - Nth_value(extract (month from date),1) over (partition by customer_id order by Date) - correct
  - Last_value(extract (month from date)) over (partition by customer_id order by Date asc)
- Lag and Lead can work even without the order by clause within the window function statement?
  - True
  - False – correct
- Assume we have 5 rows of date and sales. How many NULL values will be at the sales_lag column based on this query?

  lag(sales,7) over (order by date) as sales_lag
  - 2
  - 5 - correct
  - 1
  - No Null values
- Right function to get moving average of past 3 days(including today)
  - Avg(sales) over (order by date curr row and 3 rows following)
  - Avg(sales) over (order by date)
  - Avg(sales) over (order by date 3 rows pred and curr row) - correct
  - Avg(sales) over (order by date 2 rows pred and 1 rows following)

Find the number of days between the first and last market dates.

```
SELECT
x.first_market,
x.last_market,
DATE_DIFF(x.last_market, x.first_market) days_first_to_last
FROM (
    SELECT
    min(market_start_datetime) first_market,
    max(market_start_datetime) last_market
    FROM farmers_market.datetime_demo
)x
```

<u>What if we want to find the difference in hour and minute</u>

```
SELECT
    TIMESTAMP_DIFF(max(market_end_datetime), min(market_start_datetime) ,HOUR)
      AS market_duration_hours,
    TIMESTAMP_DIFF(max(market_end_datetime), min(market_start_datetime), MINUTE)
      AS market_duration_mins
FROM farmers_market.datetime_demo
```

Note: We can also use the Date_diff to find hours too

Question: Let’s say we wanted to get a profile of each farmer’s market customer’s habits over time.

- **Customer’s first purchase**
- **Customer’s last purchase**
- **Total days they made a purchase**
- **how long they are a customer.**
- **Days since their last purchase**

```sql
SELECT customer_id,
      min(market_date) as first_transaction,
      max(market_date) as last_transaction,
      count(*) as total_ transactions,
      count(distinct market_date) as total_market_visits,
      datediff(max(market_date), min(market_date)) as customer_duration,
      datediff(curdate(), max(market_date)) as days_since_1st_purchase
FROM farmers_market.customer_purchases_date
group by customer_id
```

Question: Write a query that gives us the days between each visit a customer makes.

```
SELECT
customer_id,
market_date,
lag(market_date, 1) over (partition by customer_id
order by market_date) as last_purchase,
datediff(market_date, lag(market_date, 1) over
(partition by customer_id order by market_date)) as
days_from_prev_purchase
FROM farmers_market.customer_purchases_date
```

Let’s Extend the query: Find the avg. days it take for the customer between 2 purchases or how long it takes on an avg for a custome to comeback to the market.

```
select customer_id, avg(days_from_prev_purchase) from
(
        SELECT
        customer_id,
        market_date,
        lag(market_date, 1) over (partition by customer_id
        order by market_date) as last_purchase,
        datediff(market_date, lag(market_date, 1) over
        (partition by customer_id order by market_date)) as
        days_from_prev_purchase
        FROM farmers_market.customer_purchases_date
)x
group by x.customer_id
```

Question: Assume today's date is May 31, 2019, and the marketing director of the farmer's market wants to give infrequent customers(made less than 2 day of purchases) in the past 30 days an incentive to return to the market in June

```
SELECT x.customer_id,
COUNT(DISTINCT x.market_date) AS total_visits_in_30_days FROM
(
    SELECT
    DISTINCT customer_id,
    market_date,
    DATEDIFF('2019-05-31', market_date) as days_before_curr_date
    FROM farmers_market.customer_purchases
    WHERE DATEDIFF('2019-05-31', market_date) between 0 and 30
    ) x

    GROUP BY x.customer_id
    having total_visits_in_30_days <=2
```

Note: Why have we included days between 0 and 30 in the where clause?
The table has data even after May 31st so any days after May 31st will be -1, -2 and so on.
If the condition was only<30 then all those records after May 31st will also be included which is incorrect.

CTE

Find the avg. days it take for the customer between 2 purchases or how long it takes on an avg for a customer to comeback to the market.

```
with table_1_distinct as (select distinct
                                    customer_id, market_date
                                    from `farmers_market.customer_purchases`),

table_2_last_pur as (select customer_id, market_date as curr_purchase,
                  lag(market_date) over (partition by customer_id order by market_date) as prev_purchase,
                  date_diff(market_date, lag(market_date) over (partition by customer_id order by
                           market_date), day) as days_since_last_purchase
                  from table_1_distinct
),

table_3_avg_days as (select customer_id, avg(days_since_last_purchase) as avg_days_2_return_2market
         from table_2_last_pur
         group by customer_id order by avg_days_2_return_2market
)

select * from table_3_avg_days
```

Find the overall sales of each customer and get their names as well(Use CTE)

```
with cus_pur as (select customer_id,
                sum(quantity*cost_to_customer_per_qty) as total_purchase
                 from `farmers_market.customer_purchases`
                 group by customer_id),
cus_pur_with_name as (select cp.customer_id,
                          cp.total_purchase ,
                          c.customer_first_name
                          from cus_pur cp left join
                          `farmers_market.customer` c
                          on cp.customer_id = c.customer_id)
select * from cus_pur_with_name
```