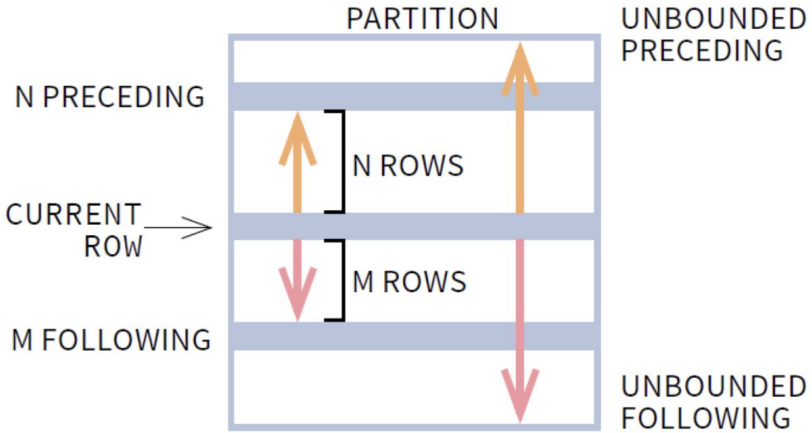
SQL-06 | Window contd. Date and Time Functions

Lecture Queries

Resources

- MySQL Window Functions (recommended)
- PostgreSQL Window functions.
- Mode Window functions.



Question: Let’s say you want to find out if the total sales on each market date are higher or lower than they were on the previous market date.

```
SELECT
    market_date,
    SUM(quantity * cost_to_customer_per_qty) AS market_date_total_sales,
    LAG(SUM(quantity * cost_to_customer_per_qty), 1) OVER (ORDER BY
market_date) AS previous_market_date_total_sales
FROM farmers_market.customer_purchases
GROUP BY market_date
```

Question - Calculate the moving average on a window frame of 1 preceding and 1 following.

```
SELECT MONTH(date), SUM(sale),
    AVG(SUM(sale)) OVER (ORDER BY MONTH(date)
                RANGE BETWEEN 1 PRECEDING
AND 1 FOLLOWING) AS sliding_avg
    FROM sales GROUP BY MONTH(date);
```

5- DAY moving average

```
SELECT MONTH(date), SUM(sale),
    AVG(SUM(sale)) OVER (ORDER BY MONTH(date)
                RANGE 4 PRECEDING) AS
sliding_avg
    FROM sales GROUP BY MONTH(date);
```

Question: Find the employee with the second highest salary in each department.

```
SELECT
        employee_name,
        department,
        salary,
        NTH_VALUE(employee_name, 2) OVER  (
                PARTITION BY department
                ORDER BY salary DESC
    RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING
        ) second_highest_salary
FROM
        farmers_market.employee;
```

Date & Time Functions

Creation of datetime_demo table

```
CREATE TABLE farmers_market.datetime_demo AS
        (
SELECT market_date,
market_start_time,
market_end_time,
STR_TO_DATE(CONCAT(market_date, ' ', market_start_time), '%Y-%m-%d
%h:%i %p')
                    AS market_start_datetime,
                STR_TO_DATE(CONCAT(market_date, ' ', market_end_time), '%Y-%m-%d
%h:%i %p')
                    AS market_end_datetime
            FROM farmers_market.market_date_info
```

Question: From each market_start_datetime, extract the following:

- day of week,
- month of year,
- year,
- hour and
- minute from the timestamp

```
SELECT
        market_start_datetime,
    EXTRACT(DAY FROM
market_start_datetime) AS start_day,
    EXTRACT(YEAR FROM
market_start_datetime) AS date_year,
    EXTRACT(MONTH FROM
market_start_datetime) AS month_of_year,
    EXTRACT(HOUR FROM
market_start_datetime) AS hour_of_day,
    EXTRACT(MINUTE FROM
market_start_datetime) AS minute_of_time
FROM farmers_market.datetime_demo;
```

Question: Let's say you want to calculate how many sales occurred within the first 30 minutes after the farmer's market opened, how would you dynamically determine what cutoff time to use?

```
SELECT market_start_datetime,
    DATE_ADD(market_start_datetime, INTERVAL 30 MINUTE) AS mktstrt_date_
 plus_30min
 FROM farmers_market.datetime_demo
```

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

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

1. First purchase - date
2. Last purchase - date
3. Count of distinct purchases

```
SELECT customer_id,
    MIN(market_date) AS first_purchase,
    MAX(market_date) AS last_purchase,
    COUNT(DISTINCT market_date) AS count_of_purchase_dates,
    DATEDIFF(MAX(market_date), MIN(market_date)) AS days_between_first_
 last_purchase,
    DATEDIFF(CURDATE(), MAX(market_date)) AS days_since_last_purchase
 FROM farmers_market.customer_purchases
 GROUP BY customer_id
```

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

```
SELECT
        customer_id,
    market_date,
    LAG(market_date, 1) OVER (PARTITION BY
customer_id ORDER BY market_date) AS last_purchase,
        DATEDIFFmarket_date, (LAG(market_date, 1) OVER
(PARTITION BY customer_id ORDER BY market_date)) AS
count_bw_prchs
FROM farmers_market.customer_purchases;
```

Question: today's date is May 31, 2019, and the marketing director of the farmer's market wants to give infrequent customers an incentive to return to the market in June.

```
SELECT DISTINCT
        customer_id,
    market_date
FROM
farmers_market.customer_purchases
WHERE DATEDIFF("2019-06-31",
market_date) <= 31
```

Question: Today’s date is May 31, 2019, and the marketing director of the farmer’s market wants to give infrequent customers(with only 1 purchase) an incentive to return to the market in April.

```
SELECT x.customer_id,
    COUNT(DISTINCT x.market_date) AS market_count
FROM (
    SELECT DISTINCT customer_id, market_date
    FROM farmers_market.customer_purchases
    WHERE DATEDIFF(market_date, '2019-05-31') <= 31
)x
GROUP BY x.customer_id
HAVING COUNT(DISTINCT market_date) = 1
```

Reference DateTime Functions

- https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html