ADVANCED SQL

Window Functions

Window Functions

- SQL window functions are a form of aggregate functions but use values from rows in a window to calculate return values.
- Window functions are not entirely standard SQL, and are not supported by all platforms
- Window functions use a new clause, the OVER() clause which has the following capabilities:
  - Defines window partitions to from groups of rows
  - Orders rows within a partition (ORDER BY)

https://drill.apache.org/docs/sql-window-functions-introduction/

Window Functions: Example

```
SELECT emp_name,
dealer_id, sales,
AVG(sales)OVER (PARTITION BY dealer_id)
AS avgsales
FROM q1_sales
```

https://drill.apache.org/docs/sql-window-functions-introduction/

Example Aggregate Window Functions

- AVG()
- COUNT()
- MAX()
- MIN()
- SUM()

https://drill.apache.org/docs/aggregate-window-functions/

Example Ranking Window Functions

- CUME_DIST(): Relative rank of a current row
- DENSE_RANK(): Rank of a value in a group of values
- NTILE(): Requires ORDER BY, divides the window into a number of ranked groups
- PERCENT_RANK(): Calculates the percent rank.
- RANK(): Ranking within a window
- ROW_NUMBER(): Returns a row number

https://drill.apache.org/docs/ranking-window-functions/

Example Value Window Functions

- LAG(): Returns the value before the current row
- LEAD(): Returns the value after the current row
- FIRST_VALUE(): The first value in a window
- LAST_VALUE(): Returns the last value in a given window.

https://drill.apache.org/docs/value-window-functions/

In Class Exercise:

- Using the sample data, create a report comparing a customer's last order with their most recent order.

TIME SERIES ANALYSIS

Time Series Analysis

- Time series analysis often involves aggregations by time elements.
- The first step in any time series analysis is making sure your time stamp is in fact a time stamp.
- These functions vary between databases but most have comparable functionality

Time Series Analysis

- You can extract pieces of a timestamp using the DATE_PART() function.
- This works for common pieces, however, there are certain artifacts which you will not be able to extract using this function.

Time Series Analysis

```
SELECT DATE_PART('YEAR', <date_field>) AS
data_year,
COUNT(*)
FROM <data>
GROUP BY data_year
```

Time Series Analysis

- The nearest_date() function allows you to bucket data to the nearest date part, such as week, quarter hour, or quarter year.
- The time_bucket() function is similar but accepts a timestamp and allows you to aggregate your data by arbitrary increments. (In milliseconds or nanoseconds)

https://drill.apache.org/docs/time-series-analysis-functions/

Time Series Analysis

```
SELECT
nearestDate(`eventDate`, 'QUARTER_HOUR' )
AS eventDate,
COUNT(*) AS hitCount
FROM dfs.`log.httpd`
GROUP BY eventDate
```

SQL Challenge

- You want to identify customers who are buying less merchandise from your store. Using the store database, create a report which compares a customers purchases by quarter.