SQL for Data Science – Week 2:

Filtering, Sorting, and Calculating Data

Objectives

- Compare analytics tool and CPU time performance between a filtered and unfiltered dataset.
- Given a dataset analysis requirement, use WHERE, IN, NOT, AND, and OR alone or in combination to filter the dataset.
- Determine whether or not to use wildcards in a data filter or search situation.
- Use wildcards to search or filter data based on requirements. Use regular expressions for text processing
- Use ORDER BY to sort data according to requirements for number of columns in the sort, sort direction, and sort position.
- Create common math operation calculated fields and aliases for calculated fields.
- Use AVG, COUNT, MAX, MIN, SUM to profile data.
- Summarize data according to one or more criterion using GROUP BY and HAVING clauses.

Basics of Filtering

- Reduce number of records you retrieve
- Reduce strain on the client application

Where Clause

- SELECT * FROM WHERE;
- Common Operators
  - =
  - <>
    - not equal
  - > / < / >= / <=
  - BETWEEN
  - IS NULL
    - Where no information for column
    - WHERE ProductName IS NULL
      - Is there some type of information for every record

IN/OR/NOT

- IN
  - Specify a range of conditions
  - Comma delimited list of values
  - WHERE SupplierID IN (9,10,11);
  - Looking for specific values

- OR
    - Will not evaluate the second condition in a where clause if the first condition is met
    - Where ProductName = ‘Tofu’ OR ‘Konbu’
- OR WITH AND
    - WHERE (SUPPLIERID = 9 OR SUPPLIERID = 11) AND (whatever)
    - SQL processes AND Before OR
- IN vs OR
    - Benefit of IN
        - Long list of options
        - Faster than OR
        - Don’t have to think about order with IN
        - Can contain another select
- NOT
    - WHERE NOT City = ‘London’ AND Not City= ‘Seattle’;

LIKE Operator

- Uses LIKE
- Search pattern made from literal text
- Can only be used strings
- Uses
    - %Pizza
        - Anything ending with pizza
    - Pizza%
        - Anything starting with pizza
    - %Pizza%
        - Anything before and after word pizza
    - S%E
        - Anything that starts with S and ends with E
    - T%@gmail.com

        Anything that starts t and ends with the gmail address
- Underscores can also be used instead of %
- Downsides
    - Takes long to run
    - Better to use = , < , >
    - Placement of wildcard is v important

ORDER BY

- Sorts data
- Usually not return in any specific way otherwise
- SELECT * FROM database ORDER BY Characteristic
- Can order by more than one column
- Column sorted doesn’t have to be retrieved

- Must be the last clause in the select statement
- Can sort by column position
    - ORDER BY 2,3
- Sort by direction
    - Desc, Asc

Math Operations

- UnitsOnOrder * UnitsPrice AS Total_Cost
- Use parantheses

Aggregate Functions

- AVG()
- COUNT()
- MIN()
- MAX()
- SUM()
- SELECT AVG(UnitPrice) AS avg_price FROM Products
- NULL Values ignored by min and max functions
- DISTINCT is helpful
    - COUNT(Distinct customer_id)
    - Cannot use count(distinct *)

GROUP BY/ HAVING

- GROUP BY
    - SELECT FROM GROUP BY Region
    - Nulls will be grouped together
    - Will need to be summarized by all the columns
- HAVING
    - SELECT FROM GROUP BY HAVING COUNT(*) >=2;
- WHERE before the data is grouped
- HAVING after the data is grouped
- SELECT FROM WHERE GROUP BY HAVING COUNT
- Group by does not sort data
- Order by does sort data

ORDER

SELECT
FROM
WHERE
GROUP BY
HAVING
ORDER BY