Week 4: Data Aggregation in SQL

LittleDabase, Employee Self Join, and Parch and Posey Database

GROUP BY

SQL aggregate functions like COUNT, AVG, and SUM have something in common: they all aggregate across the entire table. But what if you want to aggregate only part of a table? For example, you might want to count the number of orders for each account.

In situations like this, you'd need to use the GROUP BY clause. GROUP BY allows you to separate data into groups, which can be aggregated independently of one another.

Notice that you can group by multiple columns, but you have to separate column names with commas just as with ORDER BY.

1. Find the total sales in usd for each account. You should include two columns: the total sales for each company's orders in usd and the company name.
2. Find the total number of times each type of channel from the web_events was used. Your final table should have two columns - the channel and the number of times the channel was used.
3. **YOUR TURN** What was the largest order placed by each account in terms of total usd. Provide only two columns - the account name and the total usd. Order the data by dollar amount.
4. **YOUR TURN** Find the number of sales reps in each region. Your final table should have two columns - the region and the number of sales_reps. Order from fewest reps to most reps.
5. **YOUR TURN** For each account, determine the average amount of each type of paper they purchased across their orders. Your result should have four columns - one for the account name and one for the average spent on each of the paper types.
6. Determine the number of times a particular channel was used in the web_events table for each sales rep. Your final table should have three columns - the name of the sales rep, the channel, and the number of occurrences. Order your table with the highest number of occurrences first.

HAVING

The WHERE clause doesn't allow you to filter on aggregate columns, that's where the HAVING clause comes in. The HAVING clause comes after GROUP BY and before ORDER BY.

7. How many of the sales reps have more than 5 accounts that they manage?

Technically, we can get this using a SUBQUERY as shown below:

8. ** YOUR TURN** How many accounts have more than 20 orders?
9. ** YOUR TURN** Which account has the most orders?
10. **YOUR TURN** How many accounts spent more than $30,000 total with Parch and Posey throughout the years?
11. **YOUR TURN** Which account has spent the most with us?
12. **YOUR TURN** Which account used facebook most as a channel?
13. **YOUR TURN** Which accounts used facebook as a channel to contact customers more than 6 times?
14. **YOUR TURN** Which channel was the most frequently used by different accounts?

Figure 1: DATE_TRUNC

DATE Function with GROUP BY

GROUPing BY a date column is not usually very useful in SQL, as these columns tend to have transaction data down to a second. Keeping date information at such a granular data is both a blessing and a curse, as it gives really precise information (a blessing), but it makes grouping information together directly difficult (a curse).

Lucky for us, there are a number of built in SQL functions that are aimed at helping us improve our experience in working with dates:

- **DATE_TRUNC** allows you to truncate your date to a particular part of your date-time column. Common trunctions are day, month, and year.
- **DATE_PART** can be useful for pulling a specific portion of a date, but notice pulling month or day of the week (dow) means that you are no longer keeping the years in order. Rather you are grouping for certain components regardless of which year they belonged in.

15. Find the sales ($) for all orders in each year, ordered from largest to smallest. Do you notice any trends in the yearly sales totals?
16. **YOUR TURN** Which month did Parch & Posey have the largest sales ($)?
17. **YOUR TURN** In which year and month did Walmart spend($) the most on gloss paper?

CASE

The CASE statement is SQL's way of handling if/then logic. The CASE statement is followed by at least one pair of WHEN and THEN statements—SQL's equivalent of IF/THEN in Excel. Because of this pairing, you might be tempted to call this SQL CASE WHEN, but CASE is the accepted term.

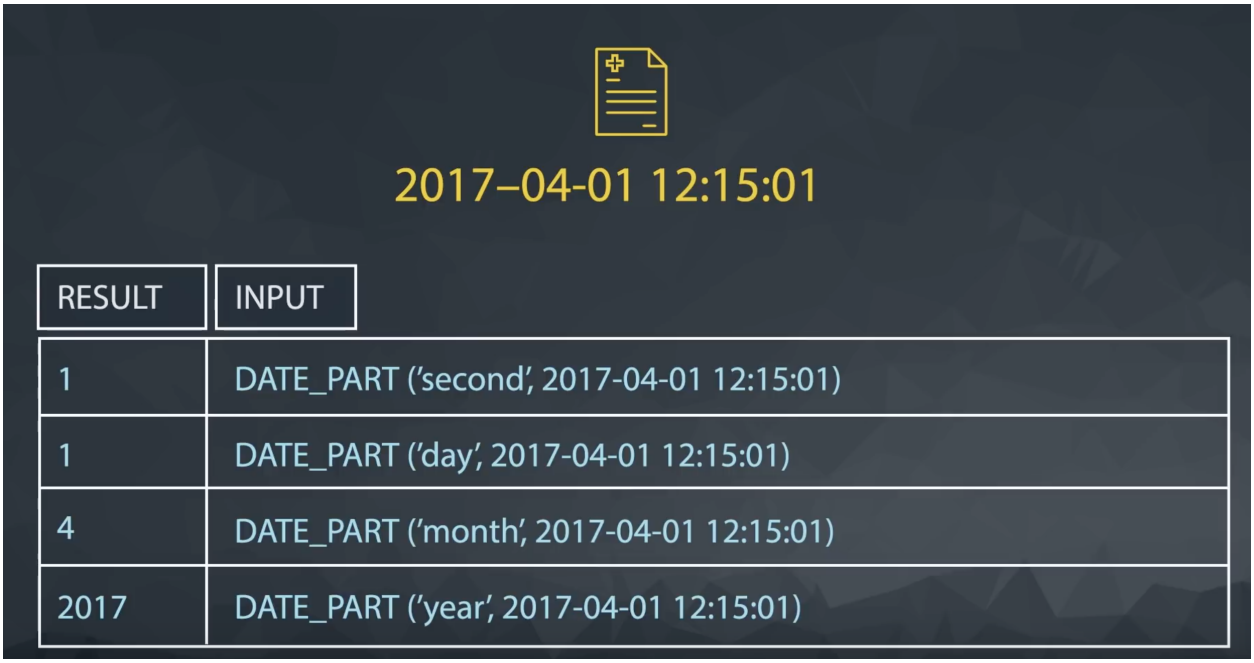

Figure 2: DATE_PART

Every CASE statement must end with the END statement. The ELSE statement is optional, and provides a way to capture values not specified in the WHEN/THEN statements.

- The CASE statement always goes in the SELECT clause
- CASE must include the following components: WHEN, THEN, and END. ELSE is an optional component.
- You can make any conditional statement using any conditional operator (like WHERE) between WHEN and THEN. This includes stringing together multiple conditional statements using AND and OR.
- You can include multiple WHEN statements, as well as an ELSE statement to deal with any unaddressed conditions.

18. We would like to understand 3 different branches of customers based on the amount associated with their purchases. The top branch includes anyone with a Lifetime Value (total sales of all orders) greater than 200,000 usd. The second branch is between 200,000 and 100,000 usd. The lowest branch is anyone under 100,000 usd. Provide a table that includes the level associated with each account. You should provide the account name, the total sales of all orders for the customer, and the level. Order with the top spending customers listed first.

19. Restrict the results of the preivous question to the orders occurred only in 2016 and 2017.

20. **YOUR TURN** We would like to identify top performing sales reps, which are sales reps associated with more than 200 orders. Create a table with the sales rep name, the total number of orders, and a column with top or not depending on if they have more than 200 orders. Place the top sales people first in your final table.

21. The previous question didn't account for the middle, nor the dollar amount associated with the sales. Management decides they want to see these characteristics represented as well. We would like to identify top performing sales reps, which are sales reps associated with more than 200 orders or more than 750000 in total sales. The middle group has any rep with more than 150 orders or 500000 in sales. Create a table with the sales rep name, the total number of orders, total sales across all orders, and a column with top, middle, or low depending on this criteria. Place the top sales people based on dollar

amount of sales first in your final table.