Contents

Tips, Tricks, and Pitfalls

Level 2

I: Aliases

Just a quick tip here. Aliases are fantastic and you should use them often for both tables and columns. There are two ways you can write an alias:

```
SELECT
    COUNTRY CTRY,
```

Or

```
SELECT
    COUNTRY AS CTRY,
```

Personally, using AS is more clear and easier to read. I don't do it a lot out of habit but I encourage you to do so.

II: Case When

Here is another example on what a good CASE WHEN query might look like. I'm pretending "T" countries have some significance (they don't) just so I can illustrate the power of CASE WHEN better:

```
SELECT
        TICKER,
        COUNTRY,
        CASE WHEN
                COUNTRY = "NA" THEN "DOMESTIC"
                WHEN COUNTRY LIKE "T%" THEN "T COUNTRY"
                ELSE "OTHER INTL"
                END AS DOM_INTL_T
FROM
        DBO.SECURITY_INFO
ORDER BY
        CASE WHEN
                COUNTRY = "NA" THEN "DOMESTIC"
                WHEN COUNTRY LIKE "T%" THEN "T COUNTRY"
                ELSE "OTHER INTL"
                END
```

Notice a few things here:

1. You can use "in", "like", and "between" in CASE WHEN statements as well
2. You should always end with an ALIAS when you write a CASE WHEN
3. You can use your CASE WHEN in your ORDER BY, but do not include the alias (ordering doesn't care what a column's visual name is, just how to order it)

III: Group By and Aggregated Functions

Look at this query and guess what the numbers will look like:

```
SELECT
        TDH.COB_DATE,
        TDH.CUSIP,
        SUM(TDH.QUANTITY) QTY
FROM
        DBO.TRADE_DATA_HIST TDH
GROUP BY
        TDH.COB_DATE
```

Well, guess what. In most cases the CUSIP column will be completely useless here. Why? Look at the output:

COB_DATE	CUSIP	QTY
20180102		3731000
20180103		4349000
20180104		3512000
20180105		4113000
20180106		3349000
20180107		3713000

MySQL is just going to display the first CUSIP is sees but then continue to aggregate the data by the GROUP BY column. Many SQL languages won't even let you write this query because it is so counter-intuitive.

Therefore, I advise you to NEVER use a group by unless every column you are selecting is either:

- Being aggregated
- In the GROUP BY

In the above case the GROUP BY is saying “Just sum up all QTY once for each day”. CUSIP means nothing here.

If you truly want to sum up CUSIP and COB_DATE you would add CUSIP to the GROUP BY:

GROUP BY

TDH.COB_DATE, TDH.CUSIP

Here the results will look like this:

COB_DATE	CUSIP	QTY
20180102		0
20180102	C113609857	28000
20180102	C146500095	78000
20180102	C168746412	15000
20180102	C175210911	96000
20180102	C182368642	171000

Now this is better! This is displaying each CUSIP only once per day and summing up all the quantity.

For example, the first CUSIP, C113609857 might have had three trades, one for trader T1 and two for trader T2:

10,000

10,000

8,000

This GROUP BY will ignore all of the trader information and just sum up that CUSIP once on each day. So this day would equal 28,000.

IV: Decimal Precision and Round

Exactly how and why CAST works is beyond the scope of this course. Just make sure you understand how to cast since not only will you be casting as decimal but also other types later on as you use MySQL more.

Here is a simple cast:

```
SELECT CAST(122 AS DECIMAL(6,3))
```

6 = Max length of the output

3 = how many decimals

In this case you would get:

122.000

You can mess around here. You'll notice if you don't give enough characters a result will get unstable. Try a bunch of different number combinations.

For using this with real data just replace the "122" above with your column:

```
SELECT
    CAST(QUANTITY AS DECIMAL(6,3))
FROM
…
```

Round is much more straightforward. It just rounds the number.

```
SELECT ROUND(122.222,1)
```

122.2

The "1" indicates "only one decimal place". Here we use "5" instead:

ROUND(122.222,5)
122.22200

As with cast, you can simply replace our number (122.222) with a column name to achieve the same ROUND result.