Contents

Tips, Tricks, and Pitfalls

Level 1

I: General Style

As with most programming languages, when in doubt, indent. Also, CAPS always.

Here is an excellent example of well written SQL code:

```
SELECT
    TDH.POSITION_ID,
    TDH.CUSIP,
    TDH.QUANTITY
FROM
    DBO.TRADE_DATA_HIST TDH
WHERE
    TDH.QUANTITY > 100
    AND TDH.COB_DATE > 20180103
ORDER BY
    TDH.POSITION_ID
```

Some people choose to place “AND” at the end of the line rather than at the beginning (example below). Feel free to use either method.

```
WHERE
    TDH.QUANTITY > 100 AND
    TDH.COB_DATE > 20180103
```

I know we didn’t cover aliases (“TDH.”) just yet, but I wanted to demonstrate proper style for those as well.

Writing code this way may seem tedious as it does take up a lot more space, but we’re not printing the code and it makes it WAY easier to follow as a code-reviewer.

II: Commenting

1. A If you would like to test something new you have added to your code I strongly suggest you take an existing query you have written, add your new code, and comment out all the existing filters using the number symbol (#).

   For example, let’s say you want to add a filter to our above query which only shows cases where EMPLOYEE_ID = “T1”:

```
SELECT
    TDH.POSITION_ID,
    TDH.CUSIP,
    TDH.QUANTITY
FROM
    DBO.TRADE_DATA_HIST TDH
WHERE
    #TDH.QUANTITY > 100
    #AND TDH.COB_DATE > 20180103
    #AND
    TDH.EMPLOYEE_ID = “T1”
#ORDER BY
    #TDH.POSITION_ID
```

You’ll notice everything you comment out turns blue.

Another way to do this could be bulk commenting:

```
/*
TDH.QUANTITY > 100
AND TDH.COB_DATE > 20180103
AND
*/
```

As with most programming languages, in between the /*s everything gets commented out.

2. Aside from the above example, you might also want to comment out lines if you run into some problematic SQL code that is failing. I do this a lot when grading homework assignments. I'll comment out nearly everything and slowly start uncommenting lines until I figure out which line was the culprit.

3. Comment to explain your code. There are usually two places you should do this.
   a. At the very top of your code, stored procedure, or function you would have a comment block explaining what your code is attempting to do. Here is an example, I would love to see this in homework submissions on the larger queries for homeworks 4-7, but it is overkill for the earlier assignments.

```
/*
*    PROJECT: Sum by Quantity                    *
*    USE: Used by trading desk to see total shares  *
*           traded for each cusip on a daliy basis  *
*    AUTHOR: Mark Ross                           *
*    CREATED: March 01 2021                      *
*/

SELECT
        TDH.POSITION_ID,
        TDH.CUSIP,
```

…continue query

4. Comment above an unusual filter or something that might not be intuitive to someone reading your code. Here's an example:

```
SELECT
    FT.ITEM_ID
FROM
    DBO.FAKE_TABLE_NAME FT
WHERE
    #In this "FAKE_TABLE", quantity less than 100 should not exist. There are some cases where they do but those are technical error and we would like to omit them.
    FT.QUANTITY > 100
```

5. Finally, comment above any line that is complicated at all. Something like this does **NOT** require commenting:

```
SELECT
    SUM(FT.QUANTITY),
```

But a line like this certainly should:

```
SELECT
    SUM(FT.QUANTITY * FT.RATE) / SUM(FT.QUANTITY),
```

Here is how I would do it:

```
SELECT
    #This gives us the weighted average rate based on quantity
    SUM(FT.QUANTITY * FT.RATE) / SUM(FT.QUANTITY),
```

III: Additional Tips

1. If you would like to filter on a single item best use **=** instead of **in** . I've seen students do

```
WHERE
    TDH.QUANTITY IN (100)
```

This is excessive and confusing (and slower). If it is only a single value, use **=**

```
WHERE
    TDH.QUANTITY = 100
```