SQL CHALLENGE

Please note, further details have been included in the sql file itself that outline notable query design decisions.

A brief overview is presented below.

Query 1

This query required the use of JOIN to combine tables, as the query needed to provide data from columns in both table.

WHERE was used to filter as no aggregate functions were needed. Aliases were used to improve presentation. The order of the tables could

have been switched in the INNER JOIN statement with no relevant benefit..

Query 2

This query required the use of aggregation functions so GROUP BY was used as a filter.

The aggregation functions included nested CASE statements to clean the data in the relevant columns. This was to handle an

edge case, where an employer receiving 0 commission may be counted as receiving commission, if COUNT is naively used,

due to a quirk in how the function works. Aliases were used.

Query 3

This query was very similar to query 1, except extra care was given to the choice of column in the filter, to make uses of indexes.

Aliases were used as normal.

Query 4

This query required using an aggregation function as a filter, so GROUP BY and HAVING were needed. COUNT was used to check how many employees were in each department. An Alias was used too. An outer JOIN was required to see data entires in the DEPT table that did not directly correspond

to data in the EMP table.

Query 5

This query required GROUP BY to use an aggregation function once again, and ROUND was used to format the data to two decimal places,

which is typical in finance. An inner join was all that was required as NULL values were irrelavent. It's worth noting that the AVG

function ignores null values in it's calculation.