/\*

----XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

----How would you clean up this case Statement

----XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

\*/

CASE WHEN

(CASE WHEN Phase LIKE 'D%' THEN 'O'

ELSE CASE WHEN PayType = 'P1' THEN 'O'

ELSE CASE WHEN PayType = 'EU' THEN 'E'

ELSE 'L'

END

END

END

) = 'O' AND PayType <> 'P1' THEN LEFT(Department, 5) + 'C'

ELSE CASE WHEN Job LIKE '%A' OR Job LIKE '%B' THEN Job

ELSE Department

END

END AS Department,

/\*

/\* Response: \*/

Collapse the CASE conditional logic significantly. There are CASE statements here that do nothing meaningful and conditions which conflict with one another.

The outcome is a determination of a result for PayType=’P1’ because the result set of having both have PayType = ‘P1’ and <> ‘P1’ will always be NULL. The outcome is AS Department.

Therefore the only thing that matters is the result of either LEFT(Department, 5)+C, Job, or Department. The conditionals being matched here are doing way too much work.

Simplify as follows (This was tested in MySQL and results in the same output except that implicit concatenation is not allowed and so the + operator was replaced during the test with CONCAT() function):

CASE WHEN (Phase LIKE 'D%' AND PayType<>'P1') THEN LEFT(Department, 5)+'C'

ELSE

CASE WHEN (Job LIKE '%A' OR Job LIKE '%B') then Job

ELSE Department

END

END

AS Department

----XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

----What might be the unintended Consequence of this where clause

----XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

\*/

SELECT Time\_PerspectiveKey

,VersionsKey

,MonthKey

,AccountsKey

,OrganizationKey

,CASE WHEN AccountsKey IN ('46998', '46999') THEN - 1 \* [VALUE] ELSE Value END AS Value\_Adjusted

FROM dbo.FactTableFinancial\_Cube

WHERE AccountsKey >= '40000'

AND VersionsKey = 'ACT'

AND AccountsKey <= '41999'

AND NOT (OrganizationKey LIKE 'B%')

AND NOT (OrganizationKey LIKE 'C%')

AND NOT (OrganizationKey LIKE 'G%')

OR

AccountsKey IN ('46998', '46999')

AND NOT (OrganizationKey LIKE 'B%')

AND NOT (OrganizationKey LIKE 'C%')

AND NOT (OrganizationKey LIKE 'G%')

/\* Answer \*/

The same record will appear multiple times in the result set because a single record could match both condition subparts. For example, a record with AccountsKey ‘46998’ AND OrganizationKey ‘Z’ will be listed twice.

/\*

----XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

----How would you re-write these case statements

----XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

\*/

,CASE WHEN (CASE WHEN Transaction\_Type = 'I' AND - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount)

ELSE Retention\_Amount END)) > 0 THEN - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount)

ELSE Retention\_Amount

END))

ELSE CASE WHEN Transaction\_Type = 'C' AND - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount)

ELSE Retention\_Amount END)) < 0 THEN - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount)

ELSE Retention\_Amount

END))

ELSE 0

END

END) <> 0 THEN 0

ELSE (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Invoice\_Extension) ELSE Invoice\_Extension END) + - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount)

ELSE Retention\_Amount

END))

END AS Invoice\_Amount

,CASE WHEN Transaction\_Type = 'I' AND - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount)

ELSE Retention\_Amount

END)) > 0 THEN - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount) ELSE Retention\_Amount END))

ELSE CASE WHEN Transaction\_Type = 'C' AND - (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount) ELSE Retention\_Amount END)) < 0 THEN

- (1 \* (CASE WHEN Transaction\_Type = 'C' THEN - 1 \* (Retention\_Amount) ELSE Retention\_Amount END))

ELSE 0

END

END AS Retention,

First the formatting is very hard to read. That would need to be fixed. Secondly, in my experience CASE is way overused in most queries because people try to apply human deductive reasoning to data parsing routines. It’s a mistake. Computers are not human and do not care about information. They match data to data and operations. So, the result below may look nothing like the original but is designed to deliver the same result with a lot less work on the part of the CPU and yet to be human readable.