Veo muchas veces este cálculo

DATEDIFF(SECOND, ..., ...) / 3600.0

Para facilitar mantenimiento y evitar errores, podrías definir una CTE base con la duración ya calculada, o crear una función escalar dbo.HorasEntre(inicio, fin)

TRY\_CAST repetido varias veces

DATEADD(DAY, -2, TRY\_CAST(Inicio AS DATETIME))

Aparece varias veces por campo. Podés ahorrar parsing con un WITH inicial:

WITH DatosParseados AS (

SELECT \*,

TRY\_CAST(Inicio AS DATETIME) AS InicioDT,

TRY\_CAST(Fin AS DATETIME) AS FinDT

FROM ConCubo

)

Así luego usás directamente:

DATEADD(DAY, -2, InicioDT) AS Inicio\_Corregido

LEFT JOIN TablaVinculadaUNION VU

ON ISNUMERIC(VU.OP) = 1

AND TRY\_CAST(VU.OP AS INT) = s.ID\_Limpio

Esto anula el uso de índices. Si el campo VU.OP ya es numérico o se puede transformar en una columna adicional OP\_Int, sería mejor hacer:

-- En preparación

ALTER TABLE TablaVinculadaUNION ADD OP\_Int AS TRY\_CAST(OP AS INT) PERSISTED;

-- Y luego

ON VU.OP\_Int = s.ID\_Limpio

Cuidado con duplicación en joins si OP no es única

Asegurate de que OP en TablaVinculadaUNION tenga una única coincidencia por ID\_Limpio, o bien agregá un TOP 1, ROW\_NUMBER() o DISTINCT

esto en castellano

ISNUMERIC() y TRY\_CAST() desactivan índices (Un índice es una estructura auxiliar que crea el motor de base de datos para acelerar la búsqueda de datos en una tabla)

Cuando aplicás funciones en los campos del JOIN, el motor de SQL Server no puede usar índices. Eso significa escaneos completos de tabla (table scan), lo cual es lento si TablaVinculadaUNION tiene muchas filas.

Recalculás el cast en cada fila del JOIN

Si la tabla tiene 1 millón de filas, estás haciendo 1 millón de TRY\_CAST() durante el JOIN. Esto se puede evitar.