Zen do SQL

Porto Alegre,
25 de Novembro de 2016

Indo Visitar o mestre Zen

Muitas pessoas conhecem a história de Nan-in, um mestre zen chinês que viveu na era Meiji. Um dia, um professor foi visitá-lo. Ele estava intrigado com a influência que esse mestre exercia nos jovens.

Neste momento, o mestre ofereceu-lhe um chá e o serviu com toda calma desse mundo.

E mesmo após a xícara estar cheia, o mestre continuou derramando o chá sobre a xícara. O professor falou:

“Por acaso, não percebeu que a xícara está completamente cheia e que já não cabe mais nenhuma gota?”

O mestre então, parou de derramar o chá sobre a xícara e disse calmamente:
“Assim como esta xícara, o senhor está cheio de opiniões e conceitos pré-estabelecidos. Desta forma, como poderia entrar um novo ensinamento? Como poderei dar-lhe novas ideias e perspectivas, se você não tem espaço pra elas?”

Em seguida, o mestre fez uma pausa por um breve momento e disse-lhe com olhar compreensivo, porém firme: “Se você realmente busca ter conhecimento constante, então tem que esvaziar sempre a sua xícara”.
O aluno olhou o mestre perplexo e só então percebeu a veracidade que havia naquelas sábias palavras.

Vocês vêem SQL...

```
USE NEOREAL_ESPELHO

DECLARE @CARTEIRA TABLE (CARTEIRA NVARCHAR(150))
INSERT INTO @CARTEIRA
SELECT CARTEIRA1 FROM MIS_sp.DBO.CARTEIRA_RECOVERY WITH(NOLOCK) WHERE CARTEIRA2 = 'RECOVERY MASSIFICADOs'

select * from (
SELECT distinct BASE.*,TEL.QTDE_TEL,TEL.QTDE_CEL,TEL.QTDE_CEL_BOM,TEL.QTDE_CEL_NOVO,TEL.QTDE_COM,TEL.QTDE_RES,TEL.QTDE_OUTROS,
    UPPER(ISNULL(AC.DESCRICAORESPOSTA,'SEM RESPOSTA'))ULTIMA_RESPOSTA,
    COALESCE(CONVERT(CHAR,AC.DATAHORAANDAMENTO,120),'-') ULTIMO_ACIONAMENTO_HUMANO,
    COALESCE(upper(AC.NOMEUSUARIO),'-')ULTIMO_USUARIO,
    COALESCE(AC.INTENSIDADE_ACIONAMENTO,0) INTENSIDADE_ACIONAMENTO,
    COALESCE(AC.FAIXA_DE_INTENSIDADE,'SEM ACIONAMENTO')FAIXA_DE_INTENSIDADE,
    HC.FILADECOBRANCA,
    CONVERT(CHAR,HC.DATAHORACOBRANCA,120)DATAHORACOBRANCA,
    ENDV.CIDADE,
    ENDV.UF,
    COALESCE(CONVERT(CHAR,DATA_ULTIMO_ACORDO,120),'-')DATA_ULTIMO_ACORDO,
    CASE
        WHEN SALDO < 500 THEN '< 500'
        WHEN SEGMENTO  = 'CBV' AND SALDO BETWEEN 500 AND 7000 THEN 'CBV Demais 500,01 À 7.000,00'
        WHEN SEGMENTO  = 'CBV' AND SALDO > 7000 THEN 'CBV Demais > 7.000,00'
        WHEN SEGMENTO  = 'CMV' AND SALDO BETWEEN 500 AND 7000 THEN 'CBV Demais 500,01 À 7.000,00'
        WHEN SEGMENTO  = 'CMV' AND SALDO BETWEEN 7000 AND 10000 THEN 'CMV Demais 7.000,00 À 10.000,00'
        WHEN SEGMENTO  = 'CMV' AND SALDO BETWEEN 10001 AND 30000 THEN 'CMV Demais 10.000,00 À 30.000,00'
        WHEN SEGMENTO  = 'CMV' AND SALDO BETWEEN 30001 AND 50000 THEN 'CMV Demais > 30.000,00'
        WHEN SEGMENTO  = 'CMV' AND SALDO > 50000 THEN 'CMV Demais > 50.000,00'
    END AS RANGE_VALOR,
        TIT as TITULOS/*,
        PRODUTO*/
FROM(
    (
 ---------------- BASE PRINCIPAL ------------------------
    SELECT
        C.NUMEROCONTRATO,
        C.CARTEIRA,
        DV.CPF,
        MAX(C.DATARECEBIMENTOCONTRATO)DATARECEBIMENTOCONTRATO,
        DV.NOME,
        DV.TIPOPESSOA,
        DATEDIFF(DAY,VENCDEBITO,GETDATE()) ATRASO,
        CASE
            WHEN DATEDIFF(DAY,VENCDEBITO,GETDATE()) < 181 THEN '0-180'
            WHEN DATEDIFF(DAY,VENCDEBITO,GETDATE()) BETWEEN 181 AND 360 THEN '181-360'
            WHEN DATEDIFF(DAY,VENCDEBITO,GETDATE()) BETWEEN 361 AND 720 THEN '361-720'
            WHEN DATEDIFF(DAY,VENCDEBITO,GETDATE()) BETWEEN 721 AND 1080 THEN '721-1080'
            WHEN DATEDIFF(DAY,VENCDEBITO,GETDATE()) BETWEEN 1081 AND 1800 THEN '1081-1800'
            WHEN DATEDIFF(DAY,VENCDEBITO,GETDATE()) > 1800 THEN 'ACIMA DE 1800'
            END AS FAIXA_ATRASO,
        --(DATEDIFF(DAY,VENCDEBITO,GETDATE())-DATEDIFF(DAY,DATARECEBIMENTOCONTRATO,GETDATE())) AS ATRASO_GRUPO,
        C.NOMEFUNCIONARIO,
        C.STATUSCONTRATO,
        T.VALOR AS SALDO,
        A.DESCRICAO AS SEGMENTO,
        b.DESCRICAO AS GRUPO,
        DAC.DESCRICAO AS PEDRA,
        EMAIL
    FROM CONTRATOS C WITH(NOLOCK)

    LEFT JOIN (SELECT NUMEROCONTRATO,NUMEROCGCCPF CPF,NOME,TIPOPESSOA,ISNULL(len(EMail)-LEN(replace(email,'.',''))+1,0) AS EMAIL FROM DEVEDORES WITH(NOLOCK) WHERE NUMEROCONTRATO IN (
                                        WHERE STATUSCONTRATO NOT IN ('DEVOLVIDO','LIQUIDADO')AND
                                            CARTEIRA IN (select * from @carteira))
                                            /*GROUP BY NUMEROCONTRATO,NUMEROCGCCPF,NOME,TIPOPESSOA*/)  DV ON C.NUMEROCONTRATO = DV.NUMEROCONTRATO

    LEFT JOIN (SELECT NUMEROCONTRATO,DESCRICAO  FROM DADOS_ADICIONAIS_DO_CONTRATO WITH(NOLOCK) WHERE TIPODADO IN ('GRUPO RELACIONADO AO ID CONTATO')) DAC ON C.NUMEROCONTRATO = DAC.NU
```

Eu vejo o Nó Górdio

Vamos desatar o Nó Górdio

A Solução de Alexandre o Grande

Conta-se que o rei da Frígia morreu sem deixar herdeiro e que, ao ser consultado, o Oráculo anunciou que o sucessor chegaria à cidade num carro de bois. A profecia foi cumprida por um camponês, de nome Górdio, que foi coroado. Para não esquecer de seu passado humilde ele colocou a carroça, com a qual ganhou a coroa, no templo de Zeus. E a amarrou com um enorme nó a uma coluna. O nó era, na prática, impossível de desatar e por isso ficou famoso.

Górdio reinou por muito tempo e quando morreu, seu filho Midas assumiu o trono. Midas expandiu o império mas não deixou herdeiros. O Oráculo foi ouvido novamente e declarou que quem desatasse o nó de Górdio dominaria todo o mundo.

Quinhentos anos se passaram sem ninguém conseguir realizar esse feito, até que em 334 a.C Alexandre, o Grande, ouviu essa lenda ao passar pela Frígia. Intrigado com a questão, foi até o templo de Zeus observar o feito de Górdio. Após muito analisar, desembainhou sua espada e cortou o nó. Lenda ou não o fato é que Alexandre se tornou senhor de toda a Ásia Menor poucos anos depois.

Voltando ao Inicio: Bancos Relacionais

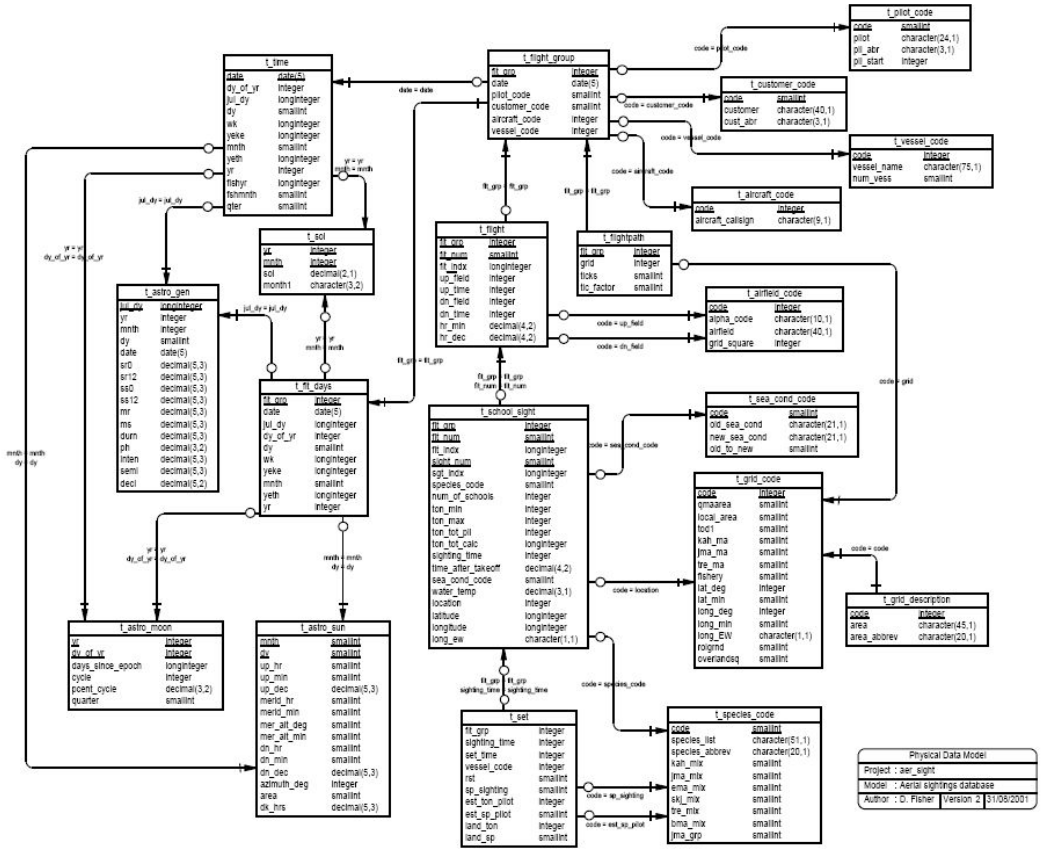
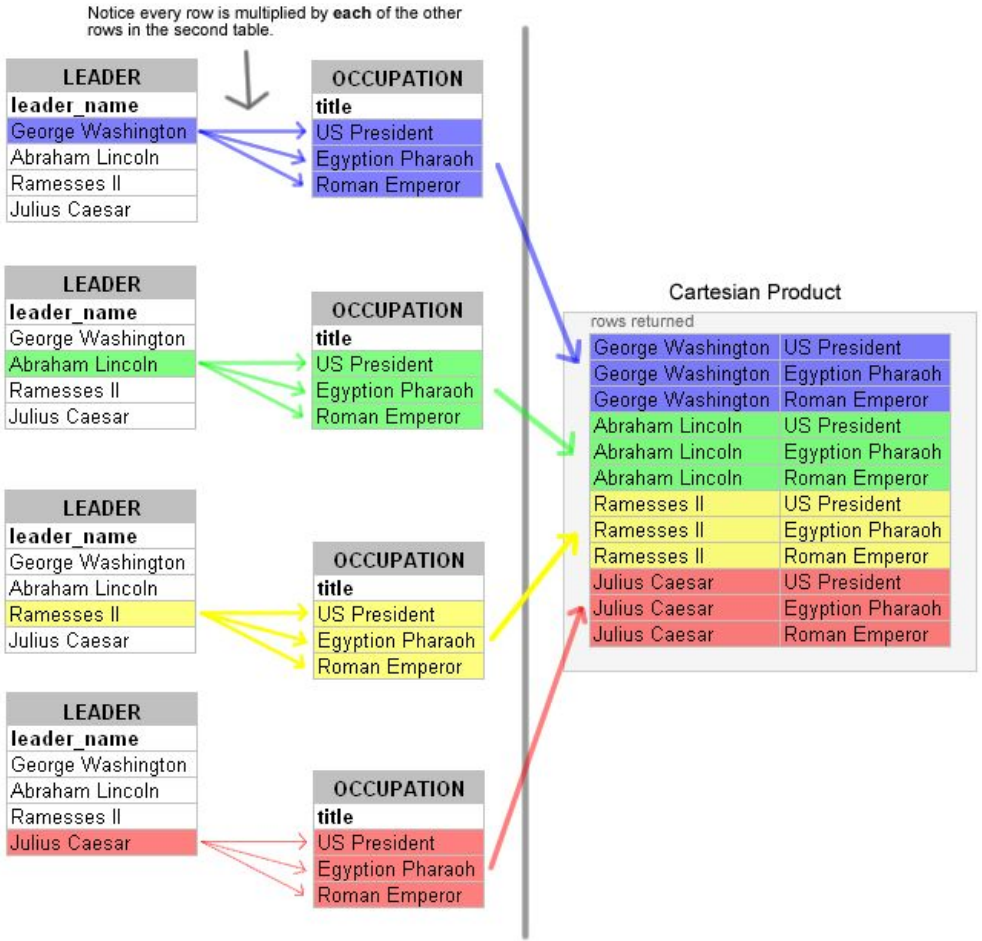


Figure 1: Entity Relationship Diagram (ERD) for the aer_sight database

Um banco relacional é um banco em que as tabelas se relacionam entre si a fim de poderem compor dados complexos como na imagem ao lado:

Porém, como se relacionam?



Eles se relacionam através de produtos cartesianos que são implementados via comandos de Join, isso cria uma tabela “virtual” que contém o produto do total de linhas de todas as tabelas.

Ou Seja,

Se temos 1 tabela de 10000 registros, outra com 1000000 e outra com 1000000 de registros, o servidor deve loopar no minimo: **10000000000000000 (10 quadrilhoes de vezes)** para produzir um resultado.

Apesar de utilizarmos cláusulas nos nossos joins, de qualquer forma ele tera q loopar várias vezes para produzir o resultado

Concluindo

De todos os fatores que veremos nesse curso, de longe é o tamanho do produto cartesiano da consulta oque mais afeta a performance dela.

E sob o ângulo da Engenharia de Software?

Existem diversos conceitos e boas práticas na engenharia de software, mas, uma das mais importantes é o principio de Single-Purpose, ou seja,

SINGLE RESPONSIBILITY PRINCIPLE

Just Because You Can, Doesn't Mean You Should

Cada Artefato de codigo, podendo ser: Consulta, Programa, Módulo, Arquivo, deve ter apenas um e apenas um proposito e uso

Ou seja, cada consulta SQL deve retornar apenas os dados sobre apenas um aspecto da realidade fotografada no Banco de Dados, ou seja:

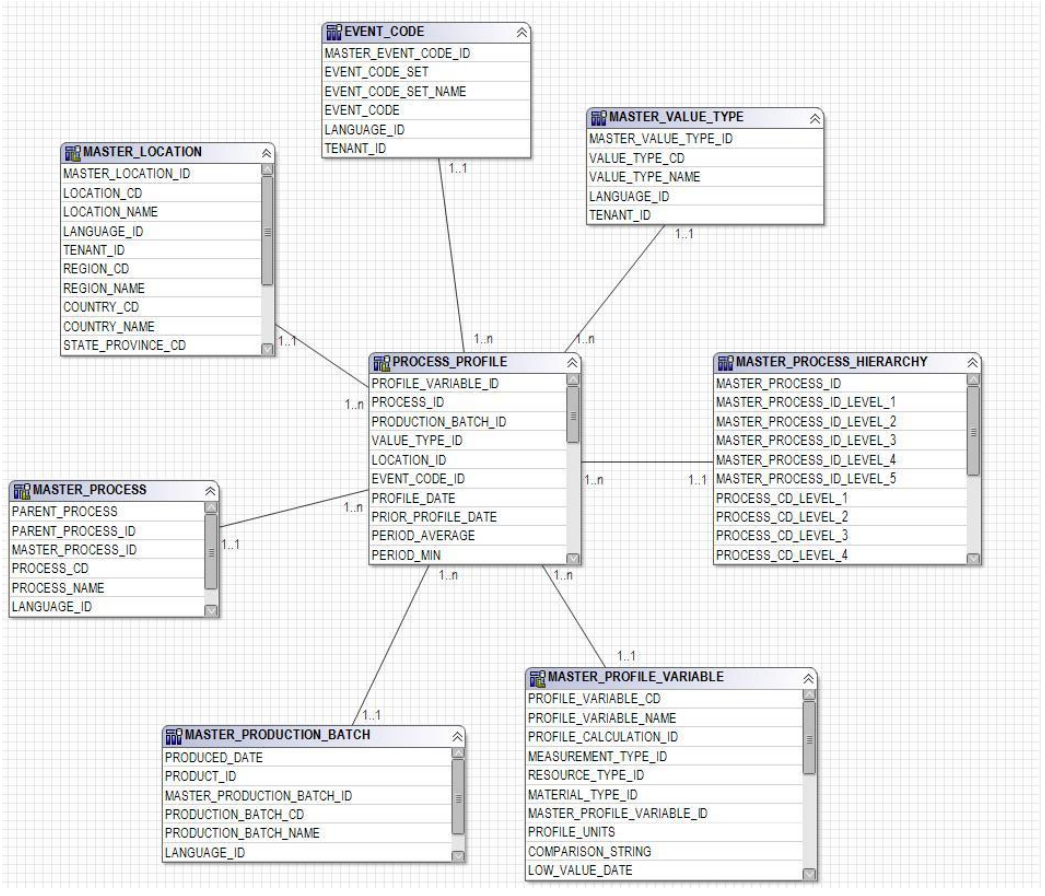
Acionamentos OU Pagamentos OU Cliente OU Mailing OU Valor Aberto

Tabelas Fato e Dimensão

Basicamente, as tabelas de um banco de dados se dividem em 2 categorias:

- **Tabelas Fato**: Elas contém os dados que realmente aconteceram e que são a realidade do banco de dados como: Clientes, Pagamentos, Acionamentos
- **Tabelas Dimensão**: Elas contém as propriedades dos fatos, e são relacionadas a elas, como: Status_Pagamento, Convenios, Células, Tipos_Resultados. São as caracteristicas do fat.

Então……



Temos que ter apenas uma consulta por tabela Fato, as tabelas fatos geralmente formam um esquema chamado de estrela, porque elas geralmente são centrais a uma série de pequenas tabelas relacionadas...

Ou seja, cada consulta SQL deve retornar apenas os dados sobre apenas um aspecto da realidade fotografada no Banco de Dados, ou seja:

Acionamentos OU Pagamentos OU Cliente OU Mailing OU Valor Aberto

Mas, na prática….

Significa que:

- Temos que ter apenas uma tabela fato por consulta
- Mas, algumas tabela fato são compostas de mais de uma tabela, por exemplo: Acordo e Parcela são duas tabelas
    - Nesse caso, eu sugiro ter dados duplicados entre as tabelas para facilitar a consulta **(Se performance nessa tabela for critica).**
- Em todas as oportunidades em que possamos restringir a consulta (clausula where) na tabela fato, devemos ter um índice.
- Os dados das tabelas de dimensões que precisamos, nós fazemos depois da consulta principal da tabela fato rodar, pois ai o produto cartesiano é bem menor.

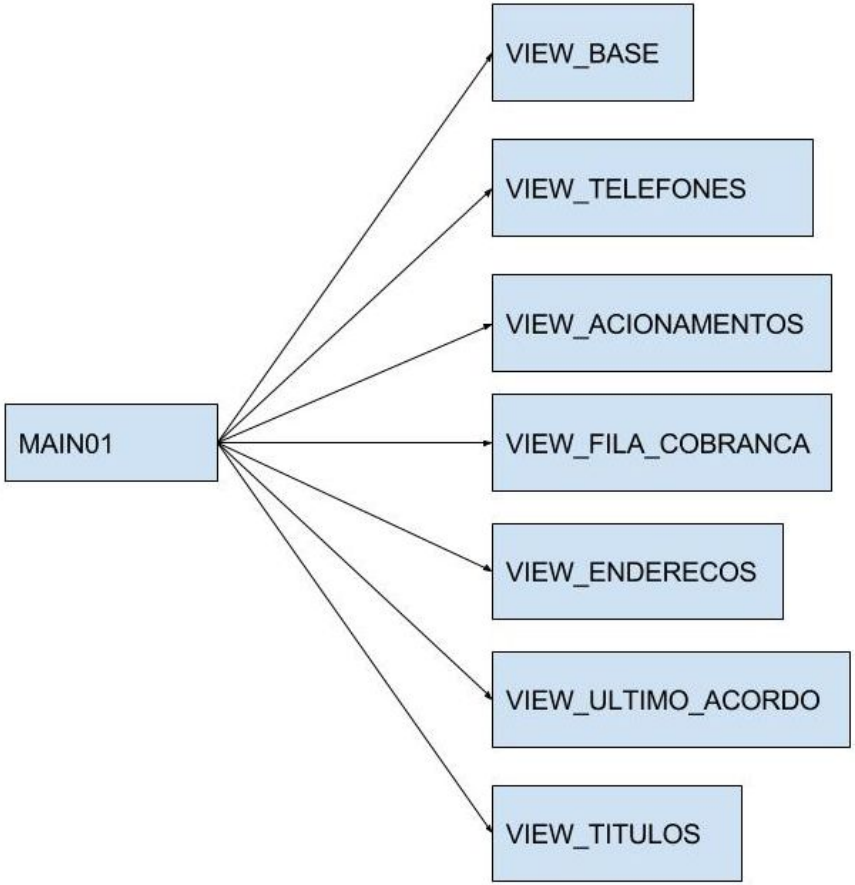
Mas, na prática….

```
SELECT CONVENIOS.NOME, FATO.TOTAL_DIVIDA FROM
(   SELECT COD_CONVENIO, SUM(CLIENTES.TOTAL_DIVIDA)
    FROM CLIENTES) AS FATO,
CONVENIOS
WHERE
CONVENIOS.COD_CONVENIO = FATO.COD_CONVENIO
GROUP BY COD_CONVENIO
```

Produto Cartesiano = 10 registros

Produto Cartesiano = 100000000 de registros

Picotando o SQL como Alexandre o Grande (1)



Pegamos a consulta grande do inicio do curso que tinha 450 linhas e picotamos elas em diversas views q contem os dados que precisamos fazer os produtos cartesianos.

Dessa forma, podemos isolar os dados rodar as consultas individualmente para verificar os problemas de falta de indice e tal...

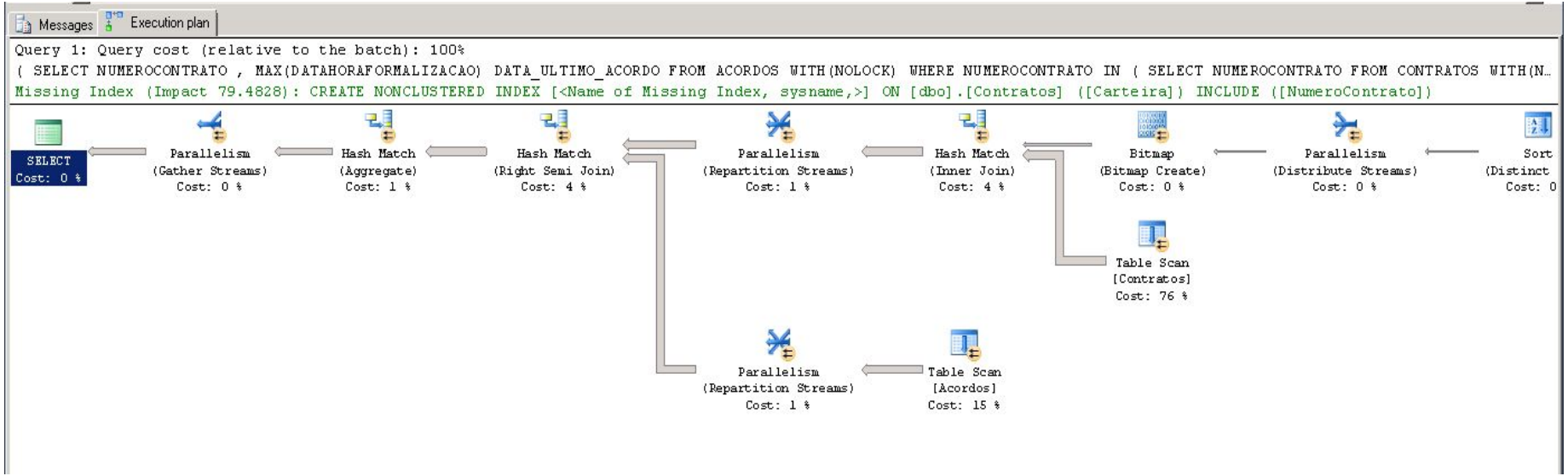
```sql
SELECT * FROM
    --------------- BASE PRINCIPAL -----------------------
    VIEW_BASE  as BASE
    --------------- BASE PRINCIPAL -----------------------
    --------------- TELEFONES ----------------------
    LEFT JOIN
    VIEW_TELEFONES AS TEL
    ON
    BASE.NUMEROCONTRATO = TEL.NUMEROCONTRATO
    --------------- TELEFONES ----------------------
    --------------- ACIONAMENTOS -----------------------
    LEFT JOIN
    VIEW_ACIONAMENTOS AS AC
    ON
    BASE.NUMEROCONTRATO = AC.NUMEROCONTRATO
    --------------- ACIONAMENTOS -----------------------
    --------------- FILA DE COBRANÇA -----------------------
    LEFT JOIN
    VIEW_FILA_COBRANCA AS HC
    ON
    BASE.NUMEROCONTRATO = HC.NUMEROCONTRATO
    --------------- FILA DE COBRANÇA -----------------------
    --------------- ENDEREÇO -----------------------
    LEFT JOIN
    VIEW_ENDERECOS AS ENDV
    ON
    BASE.NUMEROCONTRATO = ENDV.NUMEROCONTRATO
    --------------- ENDEREÇO -----------------------
    --------------- ULTIMO ACORDO -----------------------
    LEFT JOIN
    VIEW_ULTIMO_ACORDO AS ACO
    ON
    BASE.NUMEROCONTRATO = ACO.NUMEROCONTRATO
    --------------- ULTIMO ACORDO -----------------------
    --------------- TITULOS -----------------------
    LEFT JOIN
    VIEW_TITULOS AS TIT
    ON
    BASE.NUMEROCONTRATO = TIT.NUMEROCONTRATO
    --------------- TITULOS -----------------------
```

Picotando o SQL como Alexandre o Grande (2)

Conseguimos reduzir o SQL de 450 para 40 linhas, incluindo os comentários.

E depois...

Utilizando as ferramentas de analise do SQL Server podemos abrir cada uma das consultas filhas e verificar a falta de indice ou o ponto ofensor da performance:



Podemos ver acima q esta faltando um indice….

Resultado: Dados retornados em 2:37 minutos

Results | Messages

NUMEROCONTRATO	CARTEIRA	CPF	DATARECEBIMENTOCONTRATO	NOME	TIPOPESSOA	ATRASO	FAIXA_ATRASO	NOMEFUNCIONARIO	STATUSCONTRATO	SA
100059320159001	CAIXA FEV-15	36127025800	2016-07-04 00:00:00	JEFFERSON APARECIDO DA SILVA	F	1859	ACIMA DE 1800	ZANC CBV OURO - PR	Cobrança	63
100125190159001	CAIXA FEV-15	22424861870	2015-10-05 00:00:00	SUSANA DE SA FRANCO DONEGA	F	1353	1081-1800	ZANC CBV DIAMANTE	Acordo Cancelado	10
100162640159001	CAIXA FEV-15	05808437695	2016-07-04 00:00:00	JANAINA THEROZENE DE LOURDES	F	1677	1081-1800	ZANC CBV OURO	Cobrança	54
100226210159001	CAIXA FEV-15	83684069000	2015-10-05 00:00:00	JOAO CARLOS RAMOS DA SILVA JUNIOR	F	1621	1081-1800	ZANC CBV OURO	Cobrança	51
100509330159001	CAIXA FEV-15	40874745268	2015-10-05 00:00:00	JOAQUIM FERREIRA DA SILVA	F	1747	1081-1800	ZANC CBV OURO	Cobrança	38
100071620159001	CAIXA FEV-15	13631534710	2015-10-08 00:00:00	SUELANE CARLOS DA SILVA	F	1974	ACIMA DE 1800	ZANC CBV OURO - PR	Cobrança	55
100095180159001	CAIXA FEV-15	76953840200	2015-10-05 00:00:00	SOLONAIDE ALVES CARVALHO	F	1718	1081-1800	ZANC CBV OURO	Cobrança	98
100178580159001	CAIXA FEV-15	04692757663	2015-10-08 00:00:00	JEFERSON ALVES DOS SANTOS SILVA	F	1880	ACIMA DE 1800	ZANC CBV OURO - PR	Cobrança	11
100133750159001	CAIXA FEV-15	06034393817	2016-10-05 00:00:00	SILVANA DE CARVALHO CAUNER FONTINHAS	F	1473	1081-1800	ZANC CBV DIAMANTE	Cobrança	19
100238320159001	CAIXA FEV-15	06144207929	2016-09-02 00:00:00	TIAGO JOSE MENDES	F	1103	1081-1800	ZANC CBV DIAMANTE	Cobrança	55
100244020159001	CAIXA FEV-15	24240028353	2015-10-05 00:00:00	VALDECY COSTA	F	1767	1081-1800	ZANC CBV OURO	Acordo Cancelado	25
100240020159001	CAIXA FEV-15	01059679558	2016-10-10 00:00:00	UIDE GOMES DOS SANTOS	F	1767	1081-1800	ZANC CBV OURO	Cobrança	79
100500150159001	CAIXA FEV-15	32052146334	2016-07-04 00:00:00	JOAO MANUEL TEIXEIRA DE OLIVEIRA	F	1828	ACIMA DE 1800	MILANY VITORIA SABINO DE MORAIS	Acordo Ativo	12
100916440159001	CAIXA FEV-15	80699200415	2016-10-10 00:00:00	JOSE WELLINGTON RAMOS	F	1852	ACIMA DE 1800	ZANC CBV OURO - OR	Cobrança	82
100985340159001	CAIXA FEV-15	09981774740	2016-07-04 00:00:00	JUCELIA DO ROSARIO	F	1796	1081-1800	ZANC CBV OURO	Cobrança	11
100987250159001	CAIXA FEV-15	83652752104	2015-10-08 00:00:00	JUCIVALDO GOMES DA SILVA	F	1865	ACIMA DE 1800	ZANC CBV OURO - PR	Cobrança	99
100049470159001	CAIXA FEV-15	77862481534	2016-10-10 00:00:00	JAQUELINE DE JESUS SANTANA	F	1810	ACIMA DE 1800	ZANC CBV OURO	Cobrança	52
100025520159001	CAIXA FEV-15	16759540549	2016-09-02 00:00:00	DIONISIO BISPO DE BARROS	F	1100	1081-1800	ZANC CBV DIAMANTE	Cobrança	12

Query executed successfully. | SRV02STM065 (11.0 RTM) | sa (58) | NeoReal_Espelho | 00:02:37 | 181779 rows

Antes havia sido 25 minutos

Próximas melhorias:

- Salvar os dados em vez de usar views, criando tabelas temporárias para ter os dados, essas tabelas temporárias podem ter índices, oque aumenta também a performance.
- Usar identadores online de SQL, oque ajuda a formatar de forma fácil o arquivo texto do comando SQL, um exemplo é o https://sqlformat.org/

Conclusão

- O Objetivo desse curso é mostrar que na realidade o grande ofensor a performance do SQL é o tamanho do produto cartesiano, ou seja, o excesso de tabelas na cláusula FROM de uma Consulta
- É muito mais fácil, eficiente e rápido fazer pequenas consultas em cima das tabelas fato principais e depois consolidar elas.
- Essa prática está mais de acordo com as práticas de engenharia de software que pregam q um software é apenas um conjunto de pequenos softwares que se agregam.