

# Relatório DWH



**Discentes:**

**2675 Daniel Marçal**

**2691 Marcelo Pereira**

**Docente: Norberto Albino**

# Índice

<b>Introdução .....</b>	<b>3</b>
<b>Matriz BUS .....</b>	<b>4</b>
<b>Matriz de prioridades .....</b>	<b>5</b>
<b>Definição do processo de negócio .....</b>	<b>6</b>
<b>Granularidade .....</b>	<b>7</b>
<b>Definição das tabelas de dimensão .....</b>	<b>8</b>
<b>Definição da tabela de facto .....</b>	<b>9</b>
<b>Star schema.....</b>	<b>10</b>
<b>SSIS .....</b>	<b>11</b>
<b>Data Warehouse .....</b>	<b>12</b>
<b>Preenchimento de tabelas de Dimensão .....</b>	<b>13</b>
<b>Preenchimento da tabela de Factos .....</b>	<b>15</b>
<b>Registo de Auditoria .....</b>	<b>18</b>
<b>Master Package .....</b>	<b>20</b>
<b>SSAS.....</b>	<b>21</b>
<b>Desenvolvimento do Cubo .....</b>	<b>22</b>
<b>Conclusão .....</b>	<b>25</b>
<b>Bibliografia .....</b>	<b>26</b>

## Introdução

Numa primeira parte do projeto que nos foi pedido no âmbito da U.C *Data Warehouse* tínhamos como objetivo construir uma matriz BUS dos principais processos de negócio efetuados numa instituição de ensino superior, bem como a matriz de prioridades da mesma.

Como seguimento desta tarefa, foi-nos atribuído um processo de negócio para analisarmos e construir a estrutura base do star schema, não antes de se construir as suas tabelas de dimensão e a tabela de factos.

Um dos passos seguintes foi também a necessidade de analisar a granularidade do negócio em questão uma vez que iríamos trasitar para o processo de ETL (Extract, Transform, Load) concebido e carregado pelo SSIS (SQL Server Integration Services). Este mesmo processo passaria pela criação para cada dimensão já pensada e criada anteriormente em SSMS (SQL Services Managment Studio) de uma package que, resumidamente, teria um data source onde se encontram os dados, uma fase de tratamento de dados quando necessário e um processo de carregamento para a nossa base de dados. Neste último cenário aprendemos a importância de existir uma auditoria feita ao longo de todo o processo e, como tal, foi também necessário desenvolver e implementar este processo no nosso projeto.

Por fim, faltava apenas mais uma passo que seria a realização de uma pequena análise usando a base de dados da Microsoft Adventure Works em SSAS (SQL Services Analysis Services), não antes de termos feito uma última package onde se inclui todas as outras anteriores e onde se pode correr todo o processo de uma só vez, a master package.

Matriz BUS

PROCESSOS DE NEGÓCIO	data	hora	estudante	Professor	Funcionário	formação acadêmica	funções de gestão	área científica	curso	U.C	Setor de Venda	Estado de Avaliação	Assiduidade	Sumário	Turma	Regime estatutário	Tipo de Inscrição	Regime de ingresso	Programa e carga horária	Tipo de aula	Informação de acesso	Produto	Fornecedor	Contrato	UO	Cliente	Pagamentos	Docs Carga Horária
Inscrição/matricula estudantes	X		X		X			X	X	X					X	X	X	X			X				X		X	
Contratação de funcionários	X			X	X	X	X	X																X	X			
Aquisição de bens e serviços	X	X			X						X											X	X	X	X		X	
Transação de produtos e serviços	X	X	X	X	X						X											X	X		X	X	X	
Distribuição de serviço docente	X			X	X	X	X	X	X	X			X	X	X					X	X				X			X
Lançamento de resultados	X	X	X	X					X	X		X			X					X					X			X
Candidaturas a cursos especiais	X	X	X		X			X	X	X							X		X		X				X			

Figura 1

## Matriz de prioridades

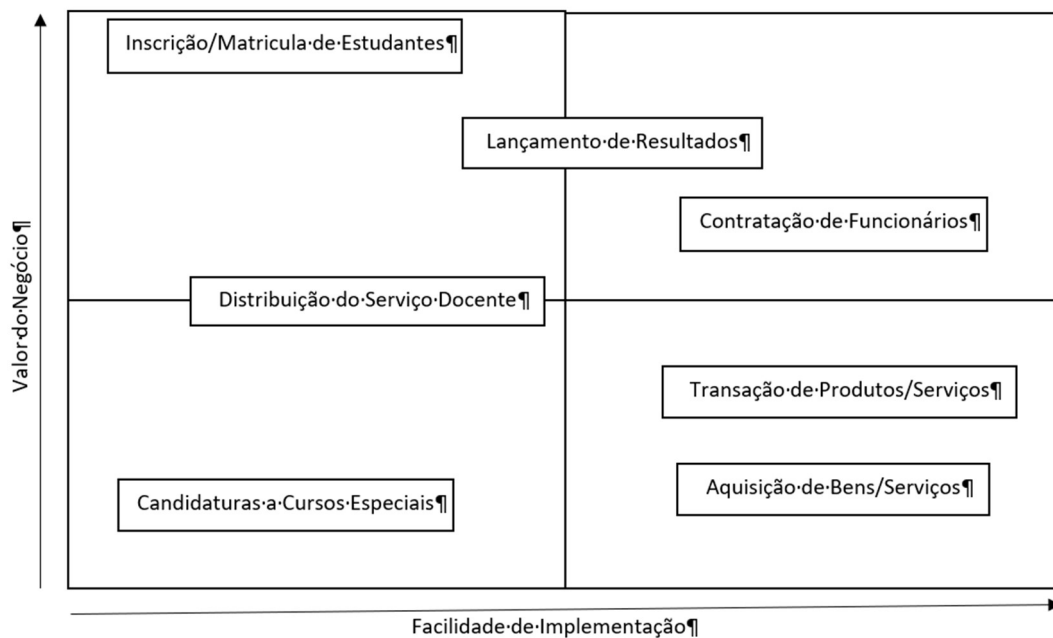


Figura 2

### NOTA:

A matriz BUS e a matriz de prioridades serão apenas apresentadas as próprias matrizes sendo que se deve ter em consideração que a segunda foi construída com base na facilidade de implementação vs dimensões da matriz BUS, e que o valor do negócio foi avaliado com base na importância por nós próprios atribuído.

## Definição do processo de negócio

O processo de negócio atribuído ao nosso grupo foi “o processo de vendas e requisição de produtos”. Este processo baseia-se no registo de qualquer venda que seja feita pela instituição bem como requisição de material. Exemplos disto são as vendas feitas na papelaria, no bar e inclusive a requisição de livros na biblioteca.

Na figura 3 pudemos observar os diferentes processos já mencionados e concluímos que o tema que nos foi atribuído tem uma alta facilidade de implementação e comparando aos restantes processos, o valor de negócio é inferior.

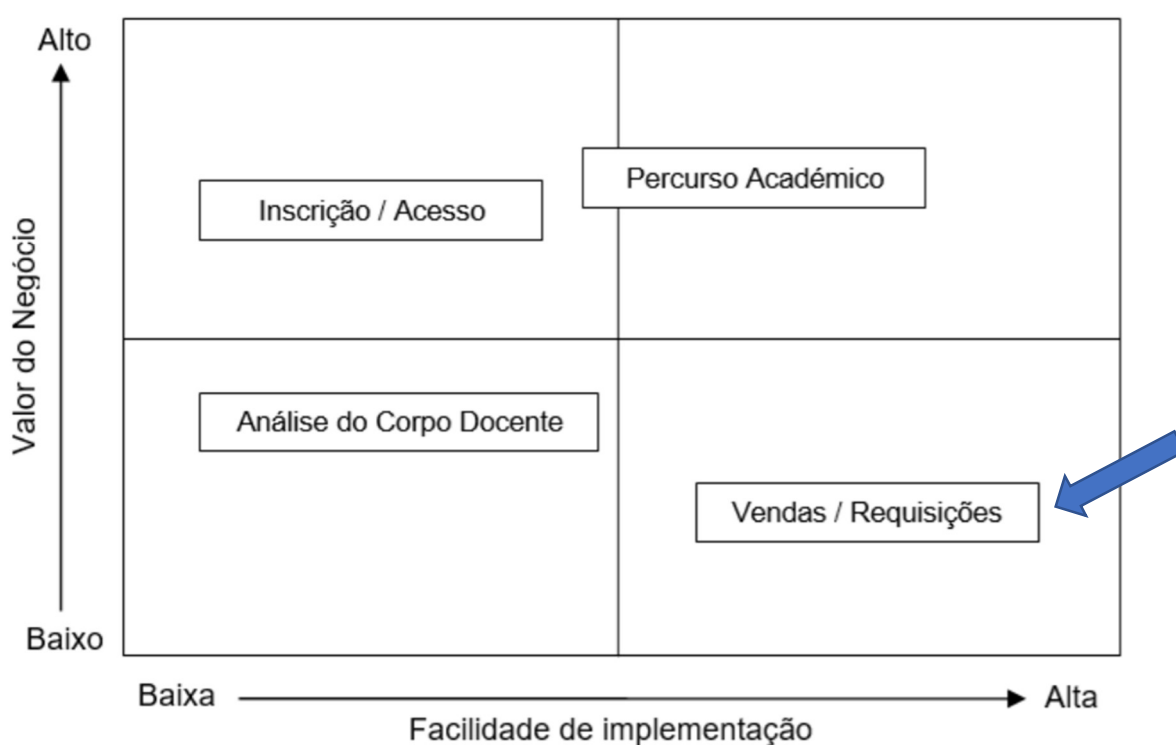


Figura 3

## Granularidade

A granularidade tem como objetivo especificar o que uma linha na tabela de facto representa. É determinado pelas necessidades do sistema operacional que captura eventos do processo de negócio.

Neste projeto uma linha deverá representar uma transação feita, ou seja, cada registo efetuado, por cada item vendido. Mesmo que o mesmo cliente, compre no mesmo momento, o mesmo item, com quantidade superior a 1, serão feitas as linhas necessárias para obter todas as transações efetuadas (ex: Se adquirir 3 itens iguais, obtemos 1 linha com quantidade 3). Mais a frente é apresentada a dimensão Produto, onde não existe o atributo quantidade, por estas razões referidas.

Um aspeto a salientar tem a ver com a escolha da dimensão Hora, onde optamos por fazer por fazer ao minuto, havendo 1440 linhas na respetiva dimensão.

A atomicidade dos dados tem de ser uma condição muito importante no desenvolver deste projeto, como iremos futuramente trabalhar e analisar linhas da nossa tabela de facto, teremos que fazer com que os dados sejam o mais atómicos possível, facilitando o trabalho de análise futuro.

## Definição das tabelas de dimensão

As tabelas de dimensão são tabelas que possuem keys que servem como identificadores únicos ou identificadores da dimensão noutra tabela para fazer a ligação da informação necessária, bem como os atributos precisos que contêm a informação anteriormente referida e, por fim, os valores de cada atributo para identificar o tipo de dados desse mesmo atributo.

Estas tabelas são utilizadas para descrever o negócio de forma clara e servem para atribuir o devido sentido à tabela de factos. São responsáveis por conter descritivos textuais que caracterizam as medidas existentes na tabela de factos.

Em todas as dimensões criadas, foram usadas “Surrugate Keys”, que constituem a chave primária de cada tabela. Este tipo de chaves apresenta várias vantagens:

- Independência que têm face às chaves do sistema transacional onde não cria influência sobre o DWH
- Facilidade de integração de fontes de dados heterogêneos
- Introdução de registos que não existam no sistema transacional, criando um registo dummy com uma chave fictícia
- Permite a alteração de registos
- Melhor performance

Dentro do processo de negócio em estudo e também com base num data-mart entregue pelo docente, definimos as seguintes dimensões:

- Dimensão Data
- Dimensão Hora
- Dimensão Produto
- Dimensão Setor de Venda
- Dimensão Cliente
- Dimensão Auditoria



## Definição da tabela de facto

Uma tabela de facto tem de conter características como: ser composta por várias medidas, crescimento elevado e rápido e a performance tem de ser um aspeto a ter em consideração.

Terá de ter componentes como a sua chave ou chaves primárias, as chaves estrangeiras e medidas do negócio necessárias.

Na tabela de facto encontramos as “Surrogate keys” das tabelas de dimensão como foreign keys e esta tabela está no centro de um esquema denominado star schema e nela está presente toda a informação necessária para completar o processo que, neste caso, é uma transação.

Observando a tabela podemos ver que esta é composta por facto do tipo Aditivo como a Quantidade, Preço unitário. Semi-Aditivos como o preço total. Os restantes são não aditivos (o imposto e o desconto vêm em percentagem).



TRANSAÇÃO_KEY
DATA_KEY
HORA_KEY
PRODUTO_KEY
CLIENTE_KEY
SETOR_KEY
FORNECEDOR_KEY
PREÇO_TOTAL
PREÇO_UNITÁRIO
TIPO_PAGAMENTO
QUANTIDADE
DESCONTO
IMPOSTO
ESTADO
DATA_INICIO
DATA_FIM

Figura 4

## Star schema

Por último, o nosso star schema que é o conjunto de todas as tabelas ligadas e que por norma assume a forma de uma estrela com a tabela de factos no centro rodeada pelas tabelas de dimensão.

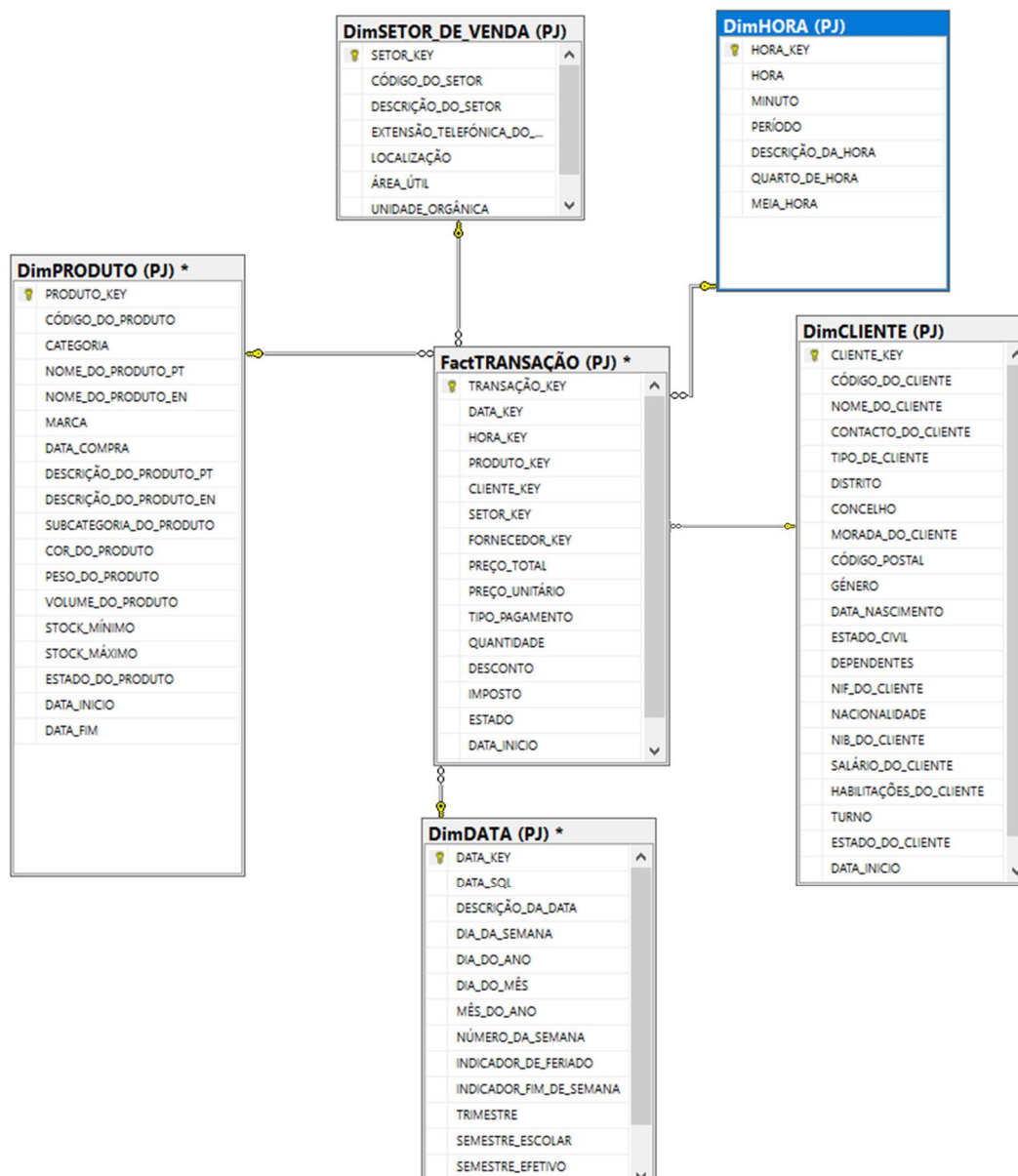


Figura 5

## SSIS

O Sql Server Integration Services mais conhecido por SSIS foi a ferramenta usada durante toda a fase de ETL.

É uma ferramenta que teve como base o Microsoft Visual Studio e que possui alguns conceitos importantes de compreender. Em primeiro lugar existem componentes com fins muito diferentes tais como a Solução onde podem ou não existir um ou mais projetos e que procura reunir todos os projetos do negócio no mesmo sítio (referente ao Visual Studio). Estes projetos sim, podem ter os mais variados fins sendo que realizámos um projeto de SSIS e um de SSAS. No caso do SSIS existe depois uma pequena hierarquia entre uma *master package* e as restantes packages de carregamento de dados onde se encontram primeiramente os *control flows* que nos dão uma visão de todo o processo desse package e os *data flows* que nos indicam o percurso dos dados e como eles são tratados.

## Data Warehouse

É uma ferramenta de suporte ao processo analítico da organização.

As suas principais características são:

- Histórico
- Consistência
- Agregação dos dados
- Ligação a SGBDR
- Forma de organização

Ao longo do semestre estudámos 2 tipos de arquiteturas, mas a que iremos implementar no nosso projeto é a de Ralph Kimball, em que o data warehouse é composto por data marts e é baseado numa arquitetura bus onde é feita uma modelação da organização com o conjunto de processos de negócio.

Num data warehouse podem ser identificados 4 componentes:

- Os Sistemas fonte
- A área de estágio (ETL)
- A área de apresentação
- A área de acesso aos dados

No desenrolar deste projeto, começámos por preparar o nosso próprio data mart, depois fomos buscar as fontes a tabelas relacionais (SI.EstBarreiro) e excel como a data para depois fazermos um processo ETL no Visual Studio com os seguintes objetivos:

- Registo de Auditoria- Onde fazíamos o controlo do preenchimento de tabelas e do número de linhas e também a obtenção do número de erros durante o processo
- Fazer uma SlowChanging Dimension
- Tratar do preenchimento da tabela de Factos
- Fazer uma Master Package Final

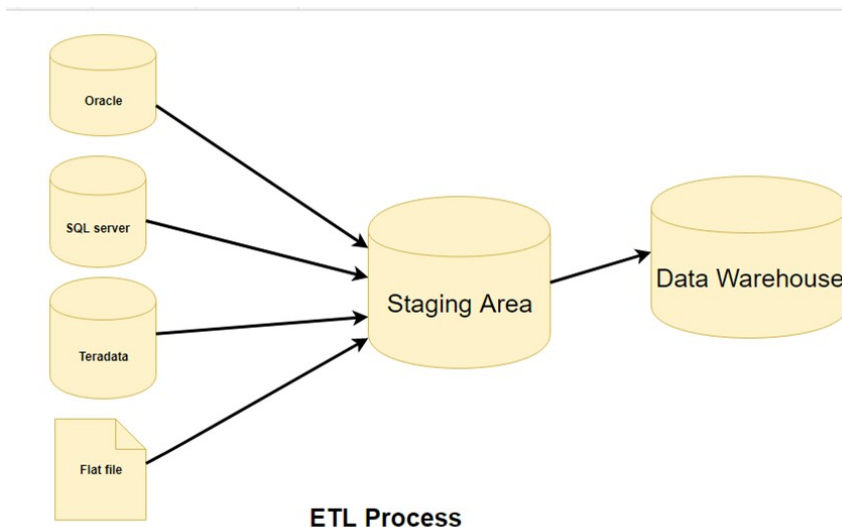


Figura 6

## Preenchimento de tabelas de Dimensão

Durante esta parte do projeto tivemos dois grandes desafios sendo eles o preenchimento das tabelas Produto e Cliente e mais tarde a constante mudança da tabela produto. Para as tabelas data e setor de venda foi mais simples fazer o processo, porque a source foi fornecida pelo docente e a tabela setor de venda tem poucos registos.

Começando pela tabela Cliente, quando olhámos para esta tabela reparámos imediatamente que fazia junção de duas tabelas (Funcionários e Alunos). Então durante o processo de tratamento dos dados no data flow tivemos que fazer este Query para obter uma Tabela Cliente com os registos pretendidos:

```
SQL command text:

select C.[Código do Cliente],
A.NomeAluno as Nome,
A.ConcelhoAulas as Concelho,
A.DistritoAulas as Distrito,
A.LocalidadeAulas as Morada_do_Cliente,
A.Sexo as Genero,
A.data_nasc as Data_Nascimento,
A.EstadoCivil as Estado_Civil
FROM gf.Cliente C
inner join sa.Aluno A
ON A.NumAluno = C.[Código do Cliente]
WHERE [Código do Tipo de Cliente]= 1

UNION

select C.[Código do Cliente],
F.Nome as Nome,
Con.Nome as Concelho,
D.Nome as Distrito,
F.Localidade as Morada_do_Cliente,
F.Sexo as Genero,
F.[Data de Nascimento] as Data_Nascimento,
F.[Estado Civil] as Estado_Civil
FROM gf.Cliente C
inner join rh.Funcionário F ON F.[Código do Funcionário] = C.[Código do Cliente]
left join Comum.Distritos D ON D.CodDistrito = F.CodigoDistrito
left join Comum.Concelhos Con on Con.Codigo = F.CodigoConcelho
WHERE [Código do Tipo de Cliente]= 2
```

Figura 7

Onde o código do tipo de Cliente (1 e 2) se refere ao Aluno e ao Funcionário. Com este “UNION” foi possível então juntar as duas tabelas e obter assim os registos.

Relativamente à tabela Produto, o problema que surgiu foi, sabendo que se trata de uma tabela com alguns produtos repetidos com certas alterações e sendo esta tabela, uma tabela vulnerável a várias mudanças e atualizações ao longo do tempo, decidimos fazer então, uma SlowChanging Dimension. Este tipo de tabelas sofre várias alterações ao longo do tempo, mas é possível tratar essas alterações da forma que a organização pretender, fazendo um novo registo ou simplesmente alterar o registo presente, não guardando histórico.

Então na Dimensão produto, começámos por definir a Source com o seguinte Query, conforme os atributos que queríamos obter:

SQL command text:

```
SELECT GF.Produto.[Código do Produto], GF.Produto.Designação,
GF.Produto.Descrição, GF.Produto.Peso, GF.[Unidade de
Medida].Designação AS [Unidade de medida], GF.Produto.[Valor Unitário
de Venda],
        GF.Produto.[Valor Unitário de Compra],
GF.Produto.Localização, GF.Produto.[Quantidade Mínima],
GF.Imposto.Designação AS Imposto, GF.Imposto.Valor, GF.[Família de
Produto].Designação AS Categoria,
        GF.Produto.[Data da Operação]
FROM GF.Produto INNER JOIN
        GF.[Família de Produto] ON GF.Produto.[Código da
Família] = GF.[Família de Produto].[Código da Família] INNER JOIN
        GF.[Unidade de Medida] ON GF.Produto.[Código da
Unidade] = GF.[Unidade de Medida].[Código da Unidade] INNER JOIN
        GF.Imposto ON GF.Produto.[Código do Imposto] =
GF.Imposto.[Código do Imposto] Figura 8
```

De seguida fizemos a SlowChanging definido três atributos diferentes , um para Changing Attribute, outro para Historical attribute e um último para Fixed Attribute. Unimos tudo e colocámos no nosso datamart (destination):

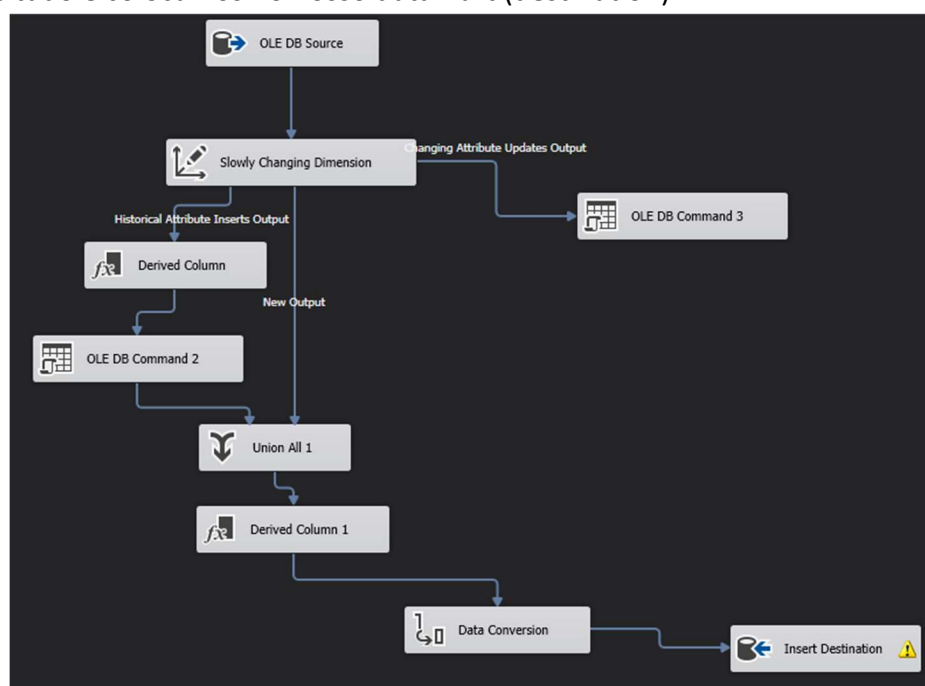
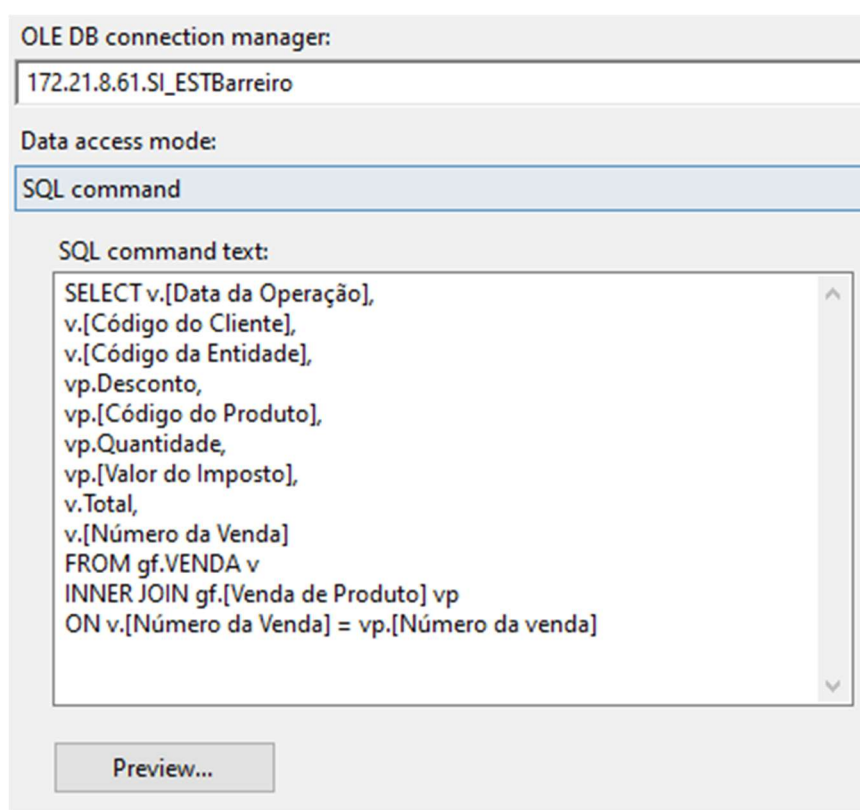


Figura 9

## Preenchimento da tabela de Factos

Após concluir todas as Packages das dimensões, pudemos completar a package da tabela de factos, sendo que esta através de lookups regista apenas os registos que estão válidos, sendo que para os inválidos ou inexistentes é criada através da “técnica dummy”, um tratamento para não se perder informação. Esta técnica agrupa todos os outputs que não deram match em cada atributo e cria uma key específica para, novamente, não perder dados importantes.

Começámos por ir buscar a seguinte Source às tabelas Venda e Venda do Produto:



OLE DB connection manager:

172.21.8.61.SI\_ESTBarreiro

Data access mode:

SQL command

SQL command text:

```
SELECT v.[Data da Operação],  
v.[Código do Cliente],  
v.[Código da Entidade],  
vp.Desconto,  
vp.[Código do Produto],  
vp.Quantidade,  
vp.[Valor do Imposto],  
v.Total,  
v.[Número da Venda]  
FROM gf.VENDA v  
INNER JOIN gf.[Venda de Produto] vp  
ON v.[Número da Venda] = vp.[Número da venda]
```

Preview...

Figura 10

De seguida contruímos o resto do data flow fazendo as seguintes tasks já antes referidas:

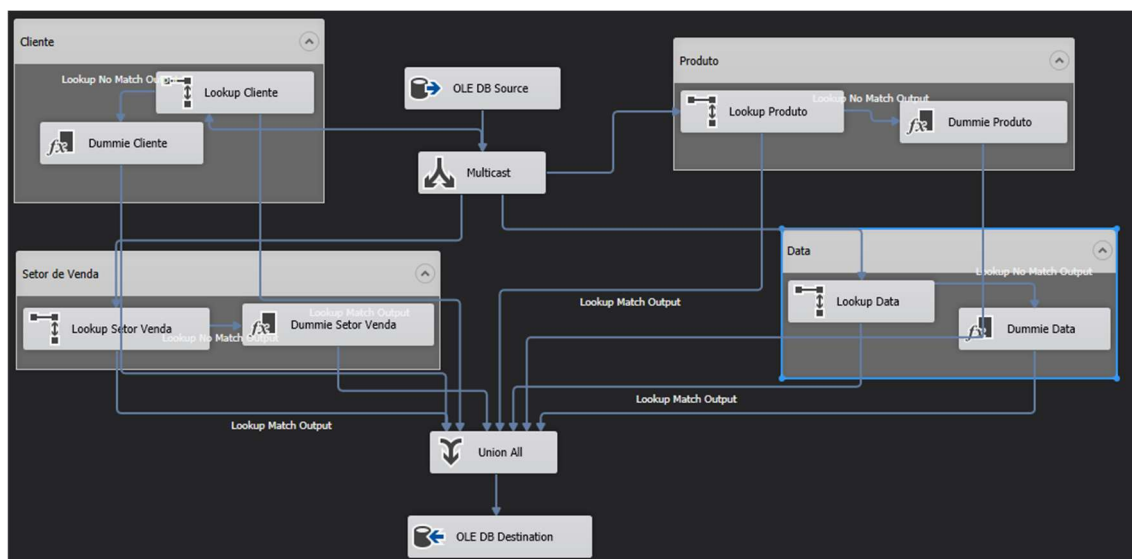


Figura 11

Para cada dimensão o objetivo era colocar as chaves Surrogate na tabela de factos de modo a contruir os registos. Daí usarmos o lookup da seguinte maneira, por exemplo para a dimensão Cliente:

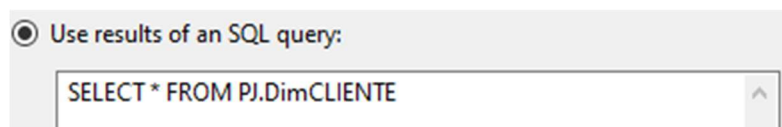


Figura 12

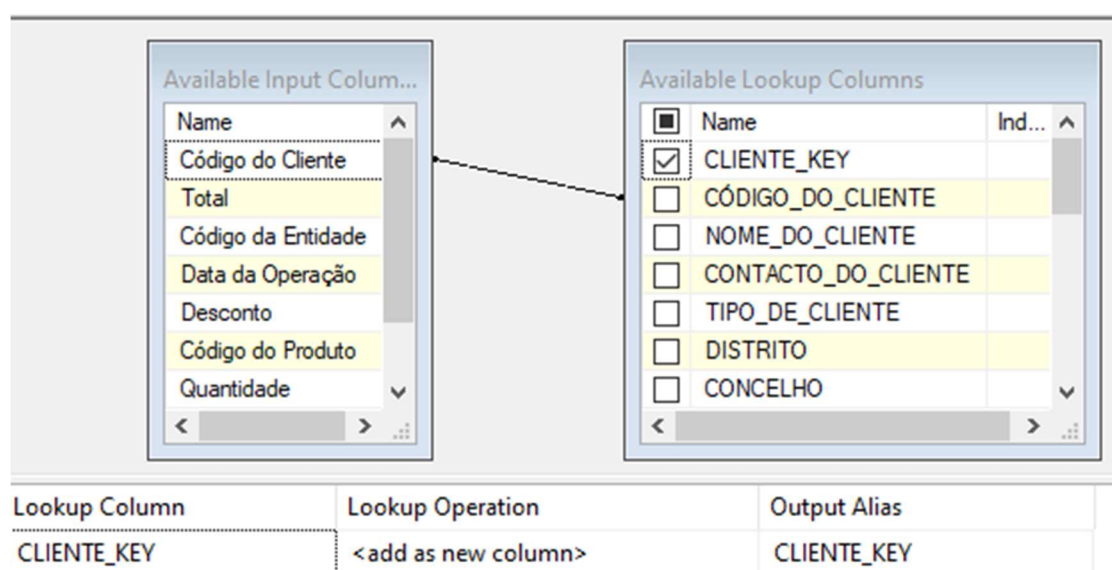


Figura 13



Para os no match outputs, usamos a técnica Dummy, com a task Derived Column, de modo a fazer um controlo sobre os registos que não têm atributos, ou que estes são inválidos, atribuindo neste caso o valor (-1):

Derived Column Name	Derived Column	Expression	Data Type	Le
CLIENTE_KEY	<add as new column>	-1	four-byte signed integ...	

Figura 14

Ainda para facilitar a visualização deste data flow, devido ao elevado número de tasks e ligações feitas, surgiu a necessidade de criar grupos e assim compreender melhor as dimensões trabalhadas neste Package.

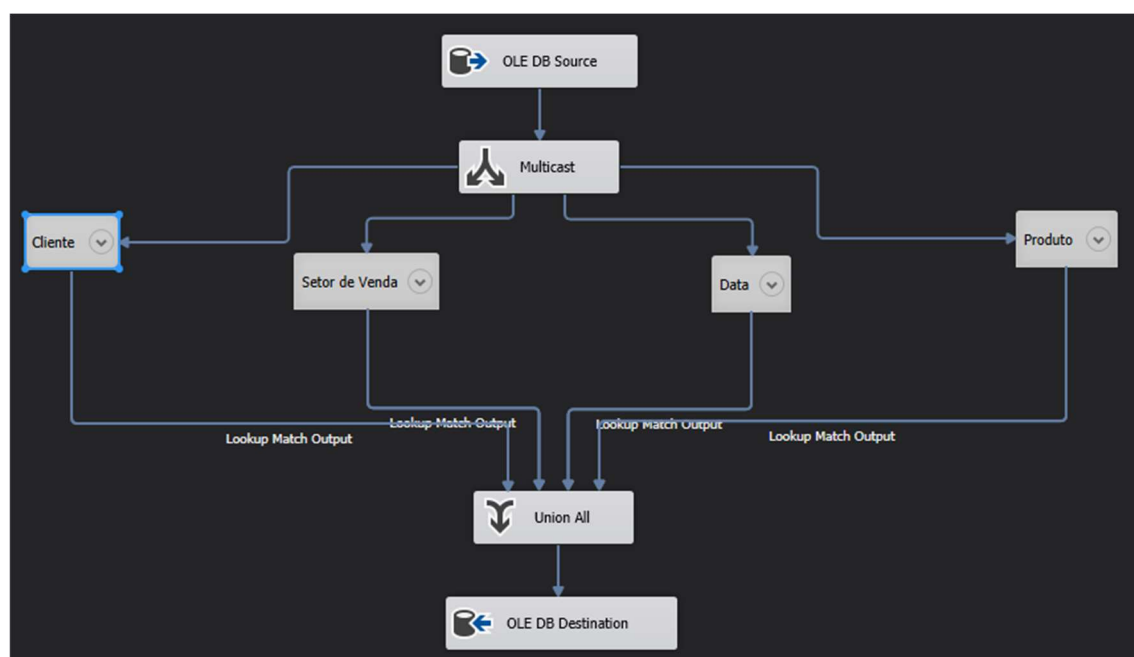


Figura 15

## Registo de Auditoria

Ao longo de todo este processo de ETL, foi-nos pedido para fazer um registo de Auditoria, com o objetivo de saber quantas linhas uma tabela tinha no início e no fim do processo ETL, registar as datas da ocorrência (inicial e final), bem como identificar o nome das Tabelas e dos Packages envolvidos e assim obter a Key para preencher na tabela de Auditoria.

Este processo esteve presente maioritariamente na parte do Control Flow, e para isso foi necessário criar as seguintes variáveis em todas as packages:

Name	Scope	Data type	Value	Expression
AUDITORIA_KEY_PAI	Package2	Int32	0	
DATA_FIM	Package2	DateTime	21/01/2020 12:55	GETDATE()
DATA_INICIO	Package2	DateTime	21/01/2020 12:55	GETDATE()
NOME_PACKAGE	Package2	String	FactTRANSAÇÃO	
NOME_TABELA	Package2	String	Table_FactTRANSAÇÃO	
ROWS_COUNT_FINAL	Package2	Int32	0	
ROWS_COUNT_INICIAL	Package2	Int32	0	

Figura 16

Depois já no Control Flow, que foi igual para todas as tabelas, foi feito da seguinte forma:

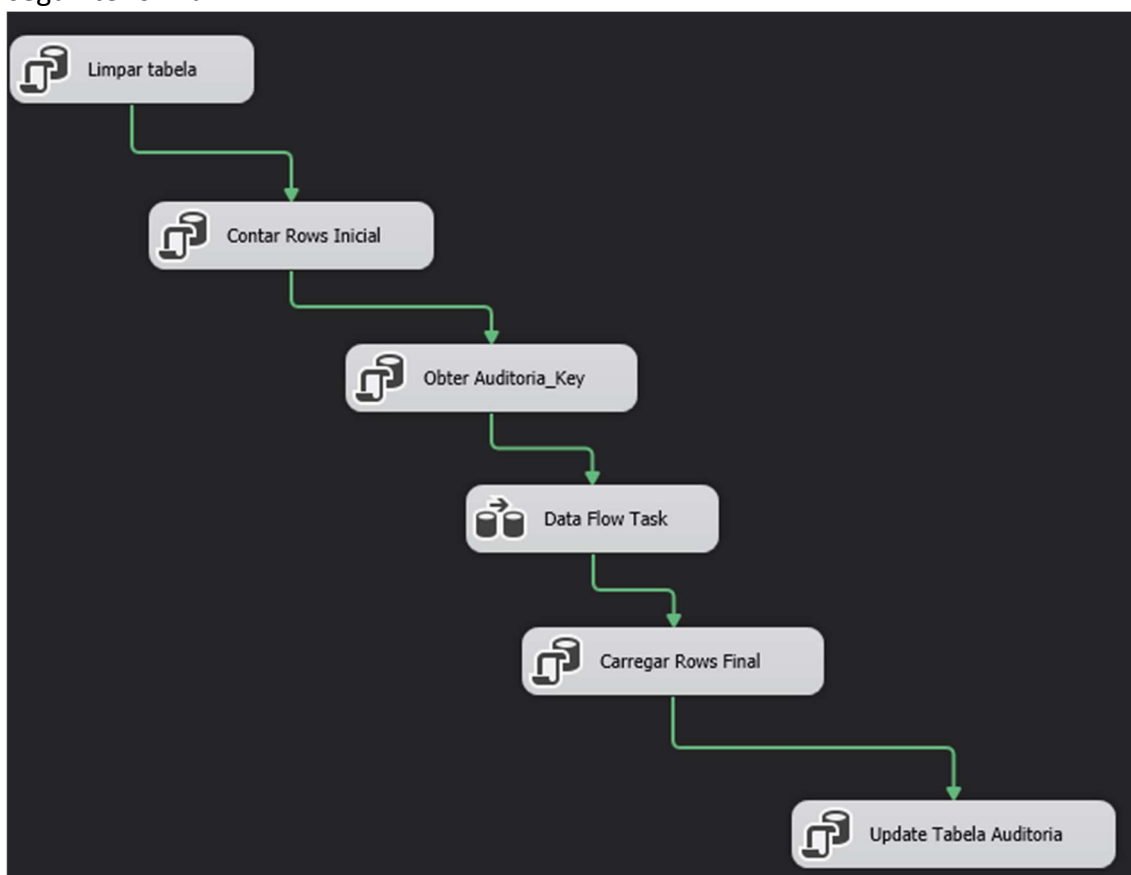


Figura 17

Primeiro limpamos sempre os registos antigos através de uma Query :  
“TRUNCATE TABLE”

Em segundo lugar é onde é feita a contagem de linhas iniciais numa “Execute SQL Task”:

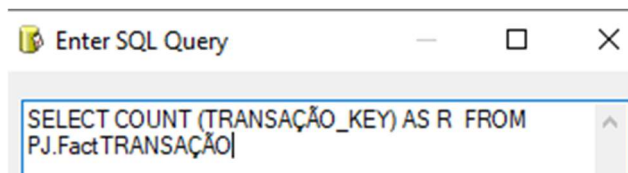


Figura 18

Result Name	Variable Name
R	User::ROWS_COUNT_INICIAL

Figura 19

Depois obtém-se uma chave de auditoria com o seguinte Query:

“INSERT INTO PJ.AUDITORIA

(AUDITORIA\_KEY\_PAI, NOME\_TABELA, NOME\_PACKAGE, ROWS\_COUNT\_INICIAL, DATA\_INICIO);

VALUES (@AUDITORIA\_KEY\_PAI, @NOME\_TABELA, @NOME\_PACKAGE, @ROWS\_COUNT\_INICIAL, @DATA\_INICIO)”

É feito o Data Flow e depois a contagem das linhas finais é feita da mesma maneira que a das iniciais, apenas mudando a devida variável.

Por fim, o update da Tabela com os valores obtidos:

“UPDATE PJ.AUDITORIA SET DATA\_FIM = getdate(),

ROWS\_INSERTED = @ROWS\_COUNT\_FINAL - @ROWS\_COUNT\_INICAL,

ROWS\_EXTRACTED = @ROWS\_EXTRACTED,

ROWS\_UPDATED= @ROWS\_UPDATED,

ROWS\_COUNT\_FINAL = @ROWS\_COUNT\_FINAL WHERE AUDITORIA\_KEY=@AUDITORIA\_KEY”

## Master Package

Temos finalmente tudo pronto para fazer então a Master Package, esta será a única package que irá correr no processo. Decidimos fazer um corrimento paralelo (2 a 2), por questões de performance. Nesta package há também um registo de auditoria e é onde todas as packages previamente criadas irão correr.

É muito importante que liguemos todas as dimensões à package da tabela de Factos:

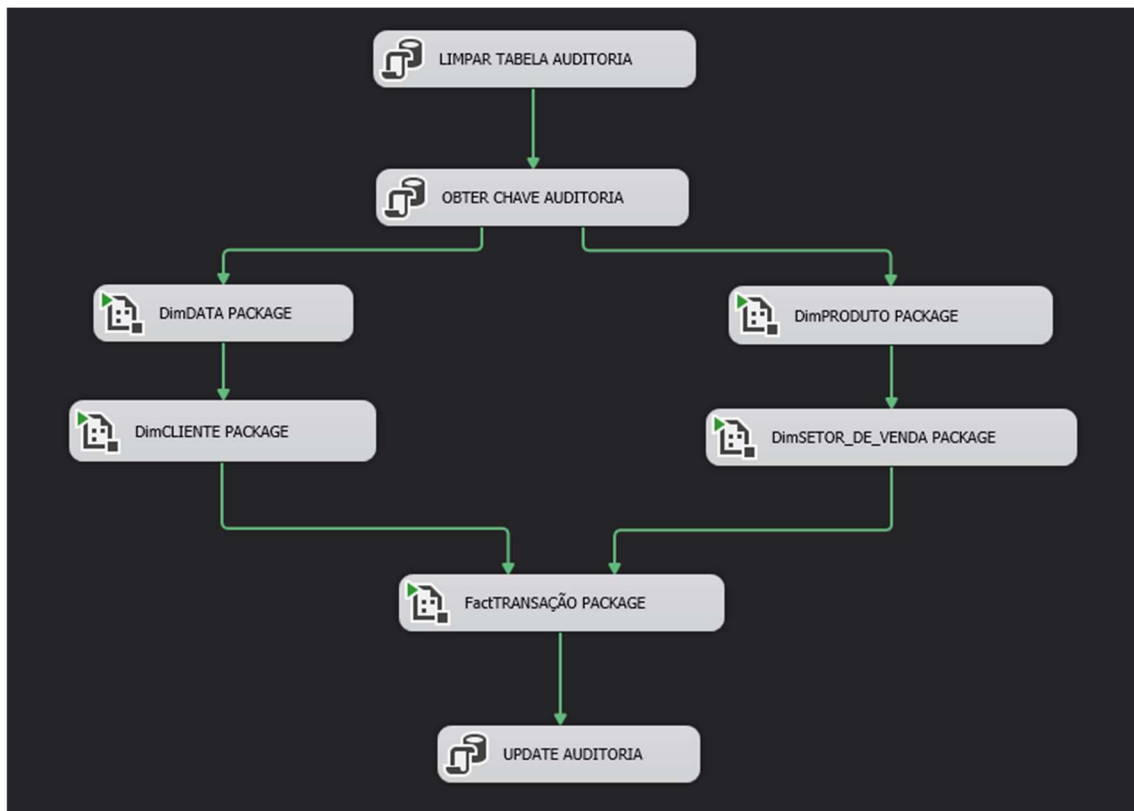


Figura 20

A forma como se obtém a chave para a auditoria é ligeiramente diferente:

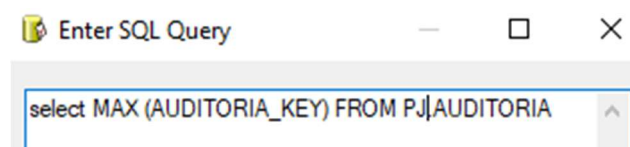


Figura 21

## SSAS

Realizado agora todo o projeto de SSIS foi-nos apresentado outro desafio, um muito simples projeto de SQL Server Analysis Services, SSAS. Este projeto teve como fonte de dados a Microsoft Adventure Works, uma base de dados da Microsoft com a finalidades académicas.

Este tipo de projeto tem como base a possibilidade de nos dar uma maneira de identificar valores valiosos, calcular valores que sejam desejados, encontrar padrões ou tendências e inclusive a construção de modelos para *data mining*. É de salientar que os componentes mais importantes são o Cubo onde armazenamos os dados, as Dimensões onde organizamos os dados, as Hierarquias que podem ou não ser definidas pelo utilizador da ferramenta, os atributos e por fim as medidas calculadas que nos permitem definir uma variável ao qual é atribuída uma fórmula para calcular um valor que seja importante.

## Desenvolvimento do Cubo

Numa terceira parte do trabalho, foi pedido para fazer um projeto de Análise a duas dimensões presentes no DW AdventureWorksDW2017.

Então começámos por criar uma solução no Visual Studio de Analysis Services e dividimos este projeto em fases:

- Primeiro fomos obter uma “Source”, esta provém do Data Warehouse AdventureWorksDW2017.
- De seguida definimos uma “View”, onde fomos obter duas tabelas de Factos (Internet Sales e Reseller Sales) também como as suas dimensões agregadas
- Depois criámos um Cubo

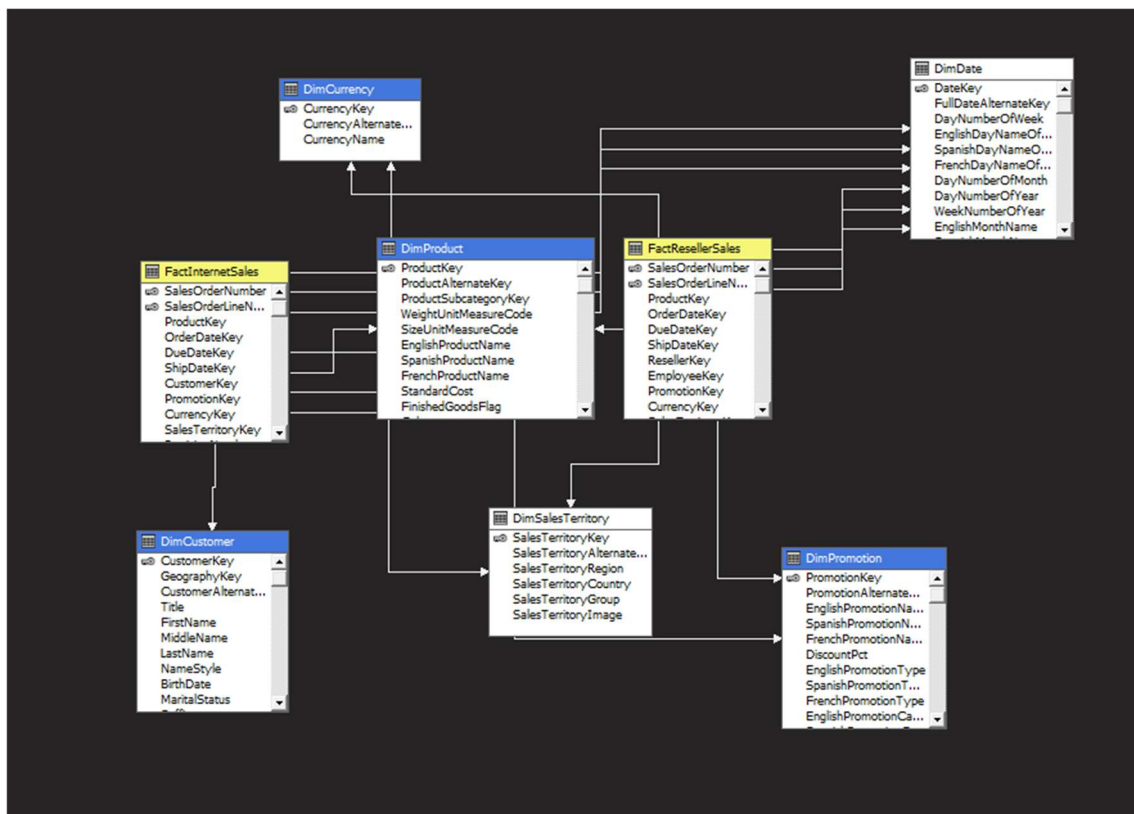


Figura 22

-Foi pedido que fizéssemos duas hierarquias, nós decidimos fazer uma relativamente à data (figura) e outra relativamente ao Território/Região (figura)



Figura 23

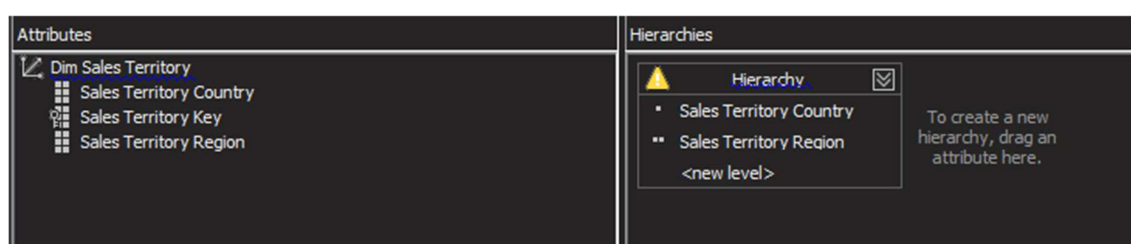


Figura 24

- Outro pedido feito foi para fazermos “Calculations”, então decidimos criar duas, uma para somar os “Total Costs” de ambas as tabelas de Facto (figura) e outro para somar a “Order Quantity” das duas tabelas também(figura). Estas decisões foram feitas com o objetivo de saber o número total de gastos e de encomendas feitas pela empresa.



Figura 25

Name: [Total Order Quantity]

Parent Properties

Parent hierarchy: Measures

Parent member: [Empty]

Change

Expression

[Measures].[Order Quantity - Fact Reseller Sales] + [Measures].[Order Quantity]

No issues found

Ln: 1 Ch: 80 SPC CRLF

Additional Properties

Format string: Standard

Visible: [Checked]

Non-empty behavior: Default

Associated measure group: [Empty]

Display folder: [Empty]

Color Expressions: [Checked]

Font Expressions: [Checked]

Figura 26

Para concluir esta fase do projeto é importante referir que obtivemos uma dimensão Snowflake que foi a (SubCategory) na dimensão Produto presente no cubo.

Depois de tudo feito está pronto para ser feito o “Build”, seguido de um “Deploy” para a nossa base de dados em SSMS e possibilitando assim ser feita a consulta futura num excel ou mesmo em SQL.



## Conclusão

Chegando agora ao fim não encontramos necessidade de concluir mais uma vez tudo que foi sendo referido ao longo do trabalho e explicado, porém, gostaríamos de deixar umas últimas palavras e opiniões.

Durante o projeto surgiu também a necessidade de pensarmos como iríamos resolver os nulls e chegamos à conclusão que a maneira mais fácil seria tratar manualmente com linguagem T-SQL como mostrado na figura XXXX com updates. É de salientar ainda que não foram todos tratados como discutido com o docente.

Por ultimo, gostaríamos de deixar o nosso apreço pelas várias fases do projeto pois embora tenhamos tido alguns problemas foram todos ultrapassados com a ajuda do docente e dedicação ao mesmo. Foi enriquecedor tanto a nível de T-SQL como de aprender a usar SSIS e SSAS embora o tempo de contacto com o último não tivesse sido o tempo que gostaríamos de ter tido.

## Bibliografia

KIMBAL,Ralph ; MARGY,Ross. The Data WareHouse Toolkit: The Definitive Guide to Dimensional Modeling .3 ed. Indianapolis, 2013

Slides DataWareHouse moodle.ips.pt