

Trabalho armazéns de dados

para
Armazéns de dados

Preparado por:
Tiago Nora(1201050)
João Gaspar(1200884)

*Instituto Superior de Engenharia do Porto, Porto
Sistemas de Informação e Conhecimento
Armazéns de dados
Paulo Jorge Machado Oliveira (JPO)*

Porto, 8 de janeiro de 2024

Conteúdo

Lista de Figuras	v
Lista de Tabelas	ix
Listagens	x
1 Introdução	1
1.1 Contextualização	1
1.2 Objetivos	1
1.3 Divisão do documento	2
2 Modelo Relacional	3
2.1 Modelo Relacional	3
2.2 Tabelas	4
3 Estrutura de dados	12
3.1 Staging Area	12
3.2 Operações de transformação	22
3.3 Operações de limpeza	24
4 Modelo Dimensional	25
4.1 Modelo Dimensional	25
4.2 Metodologia de Kimball	27
4.2.1 Área de negócio	27

4.2.2	Nível de granularidade	27
4.2.3	Dimensões identificadas	27
4.2.4	Tabelas de facto identificadas	27
4.3	Descrição das dimensões	28
4.4	Descrição das tabelas de facto	33
4.5	Justificação de cada dimensão	34
4.6	Justificação de cada tabela de factos	35
4.7	Justificação das chaves primárias das tabelas de facto	36
4.8	Mapeamento das colunas	36
5	Arquitetura da solução	37
5.1	Arquitetura da solução	37
6	Implementação da staging area e do datamart	39
6.1	Organização do projeto	39
6.2	Criação da staging area	41
6.2.1	Criação das tabelas	41
6.2.2	Criação dos lookups	41
6.2.3	Criação dos DQPs	42
6.2.4	Criação da base de dados da staging area	42
6.2.5	Criação das tabelas na base de dados	43
6.2.6	Carregamentos dos dados na staging area	44
6.3	Criação do datamart	45
6.3.1	Criação dos scripts	46
6.3.2	Criação da base de dados datamart	48
6.3.3	Criação das tabelas para o datamart	49
6.3.4	DimDate	50
6.3.5	Dimensões	53
6.3.6	DimCustomers	56
6.3.7	DimAdddressess	57

6.3.8	DimSalesTerritories	58
6.3.9	DimCurrency	59
6.3.10	DimProducts	60
6.3.11	DimShipMethods	63
6.3.12	DimSalesPersons	64
6.3.13	FactSales	67
6.3.14	FactCurrencyRates	79
7	Construção do cubo	89
8	Análises	90
8.1	Análise 1	90
8.2	Análise 2	91
8.3	Análise 3	91
8.4	Análise 4	92
8.5	Análise 5	93
8.6	Análise 6	93
8.7	Análise 7	94
8.8	Análise 8	95
8.9	Análise 9	95
8.10	Análise 10	96
8.11	Análise 11	96
8.12	Análise 12	97
8.13	Análise 13	98
8.14	Análise 14	98
8.15	Análise 15	99
8.16	Análise 16	100
8.17	Análise 17	102
8.18	Análise 18	103
8.19	Análise 19	103

8.20 Análise 20	104
9 Conclusões	106
9.1 Trabalho futuro	106
9.2 Conclusões ao nível dos dados	107
9.3 Conclusões ao nível do desenvolvimento	107

Listas de Figuras

2.1	Modelo Relacional	4
3.1	DQP para os “Customers”	13
3.2	DQP para os “Addresses”	14
3.3	DQP para os “SalesTerritories”	15
3.4	DQP para os “Products”	16
3.5	DQP para os “SalesPersons”	18
3.6	DQP para os “ShipMethods”	19
3.7	DQP para os “Currencies”	20
3.8	DQP para os “FactSales”	21
3.9	DQP para os “FactCurrency”	22
4.1	Modelo Dimensional	26
5.1	Arquitetura da Solução	38
6.1	Exemplo de uma tabela criada para a staging area	40
6.2	Caminho uniformizado	40
6.3	Exemplo de uma tabela criada para a staging area	41
6.4	Exemplo de um lookup	41
6.5	Exemplo de um DQP	42
6.6	Criação da base de dados	42
6.7	Pacote para a criação das tabelas	43

6.8	Criação das tabelas	44
6.9	Caminho para a staging area	44
6.10	Carregamento dos dados na staging area	45
6.11	Dataflow para o carregamento dos dados na staging area	45
6.12	Fluxo para o carregamento dos dados	45
6.13	Exemplo do script criado para uma das dimensões	46
6.14	Index para a dimensão “Addresses”	46
6.15	Index para a dimensão “Currency”	47
6.16	Index para a dimensão “Customers”	47
6.17	Index para a dimensão “Date”	47
6.18	Index para a dimensão “Products”	47
6.19	Index para a dimensão “SalesPersons”	48
6.20	Index para a dimensão “SalesTerritories”	48
6.21	Index para a dimensão “ShipMethods”	48
6.22	Criação da base de dados DataMart	49
6.23	Pacote para a criação das tabelas do datamart	49
6.24	Ciclo para a criação das tabelas do datamart	50
6.25	Caminho para o datamart	50
6.26	Pacote para o carregamento da dimensão data	52
6.27	Dataflow para o carregamento da dimensão data	52
6.28	Carregamento da dimensão date	53
6.29	Pacote do carregamento das dimensões de negócio	54
6.30	Dataflows de carregamento para cada dimensão	55
6.31	Dimensão “Customer”	56
6.32	Validação dimensão “Customer”	57
6.33	DQP para a dimensão “Customer”	57
6.34	Dimensão “Addresses”	58
6.35	Dimensão “SalesTerritories”	59
6.36	Dados provenientes da tabela “Currency”	59

6.37 Dimensão “Currency”	60
6.38 Dimensão “Products”	62
6.39 Flag para Dimensão “Products”	62
6.40 Validação para a Dimensão “Products”	62
6.41 DQP para a Dimensão “Products”	63
6.42 Dados provenientes da tabela “ShipMethods”	63
6.43 Dimensão “ShipMethods”	63
6.44 Dimensão “SalesPersons”	64
6.45 Transformação do “Gender”	65
6.46 Transformação do “SalariedFlag”	65
6.47 Seleção da tabela para o mapeamento	65
6.48 Mapeamento do atributo “MaritalStatus”	66
6.49 DQP da Dimensão “SalesPersons”	66
6.50 Validação feita ao atributo “NationalIDNumber”	66
6.51 DQP da Dimensão “SalesPersons”	67
6.52 Fluxo de remoção e criação de chaves estrangeiras em “FactSales”	67
6.53 Fluxo de dados Parte 1	70
6.54 Fluxo de dados Parte 2	71
6.55 OrderDateKey	72
6.56 Query para selecionar as CustomerKeys	73
6.57 Mapeamento do CustomerID que coloca a CustomerKey na tabela de factos	74
6.58 Mapeamento da conversão das colunas uniformizadas e com a granularidade ao nível da linha	76
6.59 Mapeamento da uniformização de colunas já ao nível da linha	78
6.60 Mapeamento final da tabela de factos	79
6.61 Fluxo de dados da FactCurrencyRates	80
6.62 Fluxo de dados da FactCurrencyRates	81
6.63 Mapeamento Inicial FactCurrencyRate	83
6.64 Query para selecionar a key consoante o código da moeda	84

6.65 Mapeamento da CurrencyKey	85
6.66 Mapeamento da data na tabela FactCurrencyRates	86
6.67 Mapeamento final da tabela FactCurrencyRates	87
6.68 Tabelas de facto do rácio das moedas populadas	88
7.1 Implementação do Cubo	89
8.1 Análise 1	90
8.2 Análise 2	91
8.3 Análise 3	92
8.4 Análise 4	92
8.5 Análise 5	93
8.6 Análise 6	94
8.7 Análise 7	94
8.8 Análise 8	95
8.9 Análise 9	96
8.10 Análise 10	96
8.11 Análise 11	97
8.12 Análise 12	97
8.13 Análise 13	98
8.14 Análise 14	99
8.15 Análise 15	100
8.16 Análise 16	101
8.17 Análise 17	102
8.18 Análise 18	103
8.19 Análise 19	104
8.20 Análise 20	105
9.1 Produtos Semelhantes	106
9.2 Sucesso do fluxo dos packages	108

Listas de Tabelas

3.1	Transformações do Marital Status	23
3.2	Transformações do Gender	23
3.3	Transformações do SalariedFlag	23
3.4	Transformações do FinishedGoodsFlag	24

Listagens

6.1	Script para a criação da base de dados Staging Area	43
6.2	Script para a criação da base de dados DataMart	49
6.3	Fórmula para a coluna Year	51
6.4	Fórmula para a coluna Month	51
6.5	Fórmula para a coluna IsLastDayOfTheMonth	51
6.6	Fórmula para a coluna Season	52
6.7	Script de agregação de tabelas	56
6.8	Script de agregação de tabelas	57
6.9	Script de agregação de tabelas	58
6.10	Script de agregação de tabelas	61
6.11	Script de agregação de tabelas	64
6.12	Eliminação das restrições de chaves	68
6.13	Criação das restrições de chaves	69
6.14	Representação monetária ao nível da linha	75
6.15	Representação com uniformização	77
6.16	Eliminação das restrições de chaves	80
6.17	Criação das restrições de chaves	81
6.18	Query para obter as conversões de cada moeda para EUR	82

Capítulo 1

Introdução

Neste capítulo é feita uma contextualização do problema que surgiu numa empresa fictícia, os objetivos identificados e como foi feita a divisão do documento.

1.1 Contextualização

A empresa Bikes & Bikes, especializada na produção e venda de bicicletas e acessórios, enfrenta limitações nas suas capacidades analíticas devido à natureza restrita do seu sistema operativo. A necessidade de análises detalhadas e históricas das vendas realizadas aos clientes motivou a proposta de desenvolvimento de um armazém de dados. Este armazém de dados visa superar as limitações existentes, proporcionando uma base sólida para análises abrangentes e flexíveis.

1.2 Objetivos

No que toca a objetivos desta parte do trabalho foram identificados os seguintes:

- Explorar a base de dados fornecida
- Desenvolver o modelo dimensional tendo em conta o modelo relacional
- Mapeamento dos dados entre sistemas
- Identificar operações de transformação e limpeza

1.3 Divisão do documento

O presente documento está divido nos seguintes capítulos:

- Capítulo 1: Introdução
- Capítulo 2: Modelo Relacional
- Capítulo 3: Modelo Dimensional
- Capítulo 4: Arquitetura da solução
- Capítulo 5: Operações de transformação e limpeza
- Capítulo 6: Implementação da staging area e datamart
- Capítulo 7: Construção do cubo
- Capítulo 8: Análises
- Capítulo 9: Conclusão

Capítulo 2

Modelo Relacional

Neste capítulo é descrito o modelo relacional apresentado, as suas tabelas e as suas colunas, juntamente dos seus tipos de dados.

2.1 Modelo Relacional

O modelo relacional foi construído recorrendo a ferramentas como, por exemplo:

- SQL Server Management Studio 19
- Visual Paradigm 17.0

Inicialmente foi gerado um script da base de dados. Nesta etapa o script foi gerado com recurso ao SQL Server Management Studio 19. Posteriormente, com o script criado foi utilizada a ferramenta 'Reverse DDL' do Visual Paradigm para construir-se automaticamente o modelo com as colunas, relacionamentos entre tabelas e tipos de dados. Na Figura está representado o modelo construído, neste está incluído todas as tabelas, colunas e os respetivos tipos de dados.

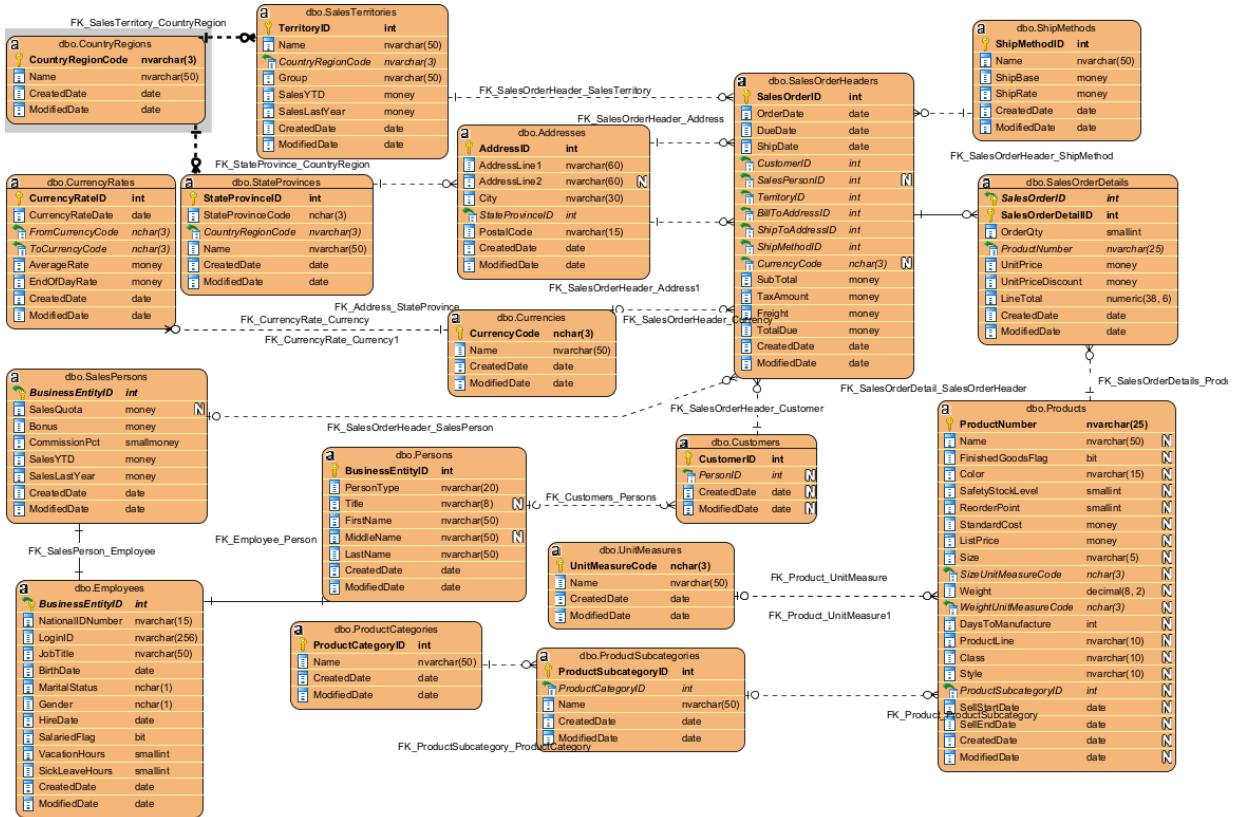


Figura 2.1: Modelo Relacional

2.2 Tabelas

No modelo relacional apresentado em cima foram identificadas as seguintes tabelas:

- Addresses
- CountryRegions
- Currencies
- CurrencyRates
- Customers
- Employees
- Persons
- ProductCategories

- Products
- ProductSubcategories
- SalesOrderDetails
- SalesOrderHeaders
- SalesPersons
- SalesTerritories
- ShipMethods
- StateProvinces
- UnitMeasures

Na tabela 'Addresses' foram encontradas as seguintes colunas:

- Chave Primária: AddressID (int)
- AddressLine1 (nvarchar(60))
- AddressLine2 (nvarchar(60))
- City (nvarchar(30))
- StateProvinceID (int, Chave Estrangeira: StateProvinces.StateProvinceID)
- PostalCode (nvarchar(15))
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'CountryRegions' foram encontradas as seguintes colunas:

- Chave Primária: CountryRegionCode (nvarchar(3))
- Name (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'Currencies' foram encontradas as seguintes colunas:

- Chave Primária: CurrencyCode (nchar(3))
- Name (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'CurrencyRates' foram encontradas as seguintes colunas:

- Chave Primária: CurrencyRateID (int)
- CurrencyRateDate (date)
- FromCurrencyCode (nchar(3), Chave Estrangeira: Currencies.CurrencyCode)
- ToCurrencyCode (nchar(3), Chave Estrangeira: Currencies.CurrencyCode)
- AverageRate (money)
- EndOfDayRate (money)
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'Customers' foram encontradas as seguintes colunas:

- Chave Primária: CustomerID (int)
- PersonID (int, Chave Estrangeira: Persons.BusinessEntityID)
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'Employees' foram encontradas as seguintes colunas:

- Chave Primária: BusinessEntityID (int)
- NationalIDNumber (nvarchar(15))
- LoginID (nvarchar(256))
- JobTitle (nvarchar(50))
- BirthDate (date)

- MaritalStatus (nchar(1))
- Gender (nchar(1))
- HireDate (date)
- SalariedFlag (bit)
- VacationHours (smallint)
- SickLeaveHours (smallint)
- CreatedDate (date)
- ModifiedDate (date)
- BusinessEntityID (int, Chave Estrangeira: Persons.BusinessEntityID)

Na tabela 'Persons' foram encontradas as seguintes colunas:

- Chave Primária: BusinessEntityID (int)
- PersonType (nvarchar(20))
- Title (nvarchar(8))
- FirstName (nvarchar(50))
- MiddleName (nvarchar(50))
- LastName (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'ProductCategories' foram encontradas as seguintes colunas:

- Chave Primária: ProductCategoryID (int)
- Name (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'Products' foram encontradas as seguintes colunas:

- Chave Primária: ProductNumber (nvarchar(25))
- ProductSubcategoryID (int, Chave Estrangeira: ProductSubcategories.ProductSubcategoryID)
- SizeUnitMeasureCode (nchar(3), Chave Estrangeira: UnitMeasures.UnitMeasureCode)
- WeightUnitMeasureCode (nchar(3), Chave Estrangeira: UnitMeasures.UnitMeasureCode)
- Name (nvarchar(50))
- FinishedGoodsFlag (bit)
- Color (nvarchar(15))
- SafetyStockLevel (smallint)
- ReorderPoint (smallint)
- StandardCost (money)
- ListPrice (money)
- Size (nvarchar(5))
- Weight (decimal(8, 2))
- DaysToManufacture (int)
- ProductLine (nvarchar(10))
- Class (nvarchar(10))
- Style (nvarchar(10))
- SellStartDate (date)
- SellEndDate (date)
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'ProductSubcategories' foram encontradas as seguintes colunas:

- Chave Primária: ProductSubcategoryID (int)
- ProductCategoryID (int, Chave Estrangeira: ProductCategories.ProductCategoryID)
- Name (nvarchar(50))
- CreatedDate (date)

- ModifiedDate (date)

Na tabela 'SalesOrderDetails' foram encontradas as seguintes colunas:

- Chave Primária: SalesOrderID (int)
- SalesOrderDetailID (int)
- OrderQty (smallint)
- ProductNumber (nvarchar(25), Chave Estrangeira: Products.ProductNumber)
- UnitPrice (money)
- UnitPriceDiscount (money)
- LineTotal (numeric(38, 6))
- CreatedDate (date)
- ModifiedDate (date)
- SalesOrderID (int, Chave Estrangeira: SalesOrderHeaders.SalesOrderID)

Na tabela 'SalesOrderHeaders' foram encontradas as seguintes colunas:

- Chave Primária: SalesOrderID (int)
- OrderDate (date)
- DueDate (date)
- ShipDate (date)
- CustomerID (int, Chave Estrangeira: Customers.CustomerID)
- SalesPersonID (int, Chave Estrangeira: SalesPersons.BusinessEntityID)
- TerritoryID (int, Chave Estrangeira: SalesTerritories.TerritoryID)
- BillToAddressID (int, Chave Estrangeira: Addresses.AddressID)
- ShipToAddressID (int, Chave Estrangeira: Addresses.AddressID)
- ShipMethodID (int, Chave Estrangeira: ShipMethods.ShipMethodID)
- CurrencyCode (nchar(3), Chave Estrangeira: Currencies.CurrencyCode)
- SubTotal (money)

- TaxAmount (money)
- Freight (money)
- TotalDue (money)
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'SalesPersons' foram encontradas as seguintes colunas:

- Chave Primária: BusinessEntityID (int)
- SalesQuota (money)
- Bonus (money)
- CommissionPct (smallmoney)
- SalesYTD (money)
- SalesLastYear (money)
- CreatedDate (date)
- ModifiedDate (date)
- BusinessEntityID (int, Chave Estrangeira: Employees.BusinessEntityID)

Na tabela 'SalesTerritories' foram encontradas as seguintes colunas:

- Chave Primária: TerritoryID (int)
- Name (nvarchar(50))
- CountryRegionCode (nvarchar(3), Chave Estrangeira: CountryRegions.CountryRegionCode)
- Group (nvarchar(50))
- SalesYTD (money)
- SalesLastYear (money)
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'ShipMethods' foram encontradas as seguintes colunas:

- Chave Primária: ShipMethodID (int)
- Name (nvarchar(50))
- ShipBase (money)
- ShipRate (money)
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'StateProvinces' foram encontradas as seguintes colunas:

- Chave Primária: StateProvinceID (int)
- StateProvinceCode (nchar(3))
- CountryRegionCode (nvarchar(3), Chave Estrangeira: CountryRegions.CountryRegionCode)
- Name (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Na tabela 'UnitMeasures' foram encontradas as seguintes colunas:

- Chave Primária: UnitMeasureCode (nchar(3))
- Name (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Capítulo 3

Estrutura de dados

No presente capítulo é descrito a estrutura dos dados na Staging Area.

3.1 Staging Area

A Staging Area é uma área intermédia onde os dados são preparados, transformados e organizados. Nesta fase devem existir as tabelas que existem na base de dados fornecidos com a adição dos lookups, tabelas DQP (Data Quality Processes) que armazenam os dados que apresentam problemas de qualidades e tabelas de duplicados que armazenam os dados identificados como registos duplicados ou idênticos.

As tabelas DQP a serem criadas devem conter cada uma delas a coluna DQP (nvarchar(100)), esta coluna serve para registar a razão pela qual aquele registo não foi carregado na respetiva dimensão, para além dos outros atributos da tabela que originou o DQP. Para este efeito foram criadas as seguintes tabelas:

- CustomersDQP
- AddressesDQP
- SalesTerritoriesDQP
- ProductsDQP
- SalesPersonsDQP
- ShipMethodDQP
- CurrencyDQP
- FactSalesDQP

- FactCurrencyDQP

Na Figura 3.1 é apresentado a tabela DQP para os “Customers”.

dbo.CustomersDQP			
CustomerID	int		N
PersonType	nvarchar(20)		N
Title	nvarchar(8)		N
FirstName	nvarchar(50)		N
MiddleName	nvarchar(50)		N
LastName	nvarchar(50)		N
CreatedDate	date		N
ModifiedDate	date		N
DQP	nvarchar(100)		N

Figura 3.1: DQP para os “Customers”

Na Figura 3.2 é apresentado a tabela DQP para os “Addresses”.

dbo.AddressesDQP		
AddressID	int	N
AddressLine	nvarchar(60)	N
AddressLine2	nvarchar(60)	N
City	nvarchar(30)	N
PostalCode	nvarchar(15)	N
CountryRegionName	nvarchar(50)	N
StateProvinceCode	nchar(3)	N
StateProvinceName	nvarchar(50)	N
CreatedDate	date	N
Modified Date	date	N
DQP	nvarchar(100)	N

Figura 3.2: DQP para os “Addresses”

Na Figura 3.3 é apresentado a tabela DQP para os “SalesTerritories”.

dbo.SalesTerritoriesDQP		
	TerritoryID	int
	TerritoryName	nvarchar(50)
	Group	nvarchar(50)
	SalesYTD	money
	SalesLastYear	money
	CountryRegionName	nvarchar(50)
	CreatedDate	date
	ModifiedDate	date
	DQP	nvarchar(100)

Figura 3.3: DQP para os “SalesTerritories”

Na Figura 3.4 é apresentado a tabela DQP para os “Products”.

dbo.ProductsDQP		
ProductID	nvarchar(25)	N
Name	nvarchar(50)	N
Finished Goods Flag	nvarchar(3)	N
Color	nvarchar(15)	N
SafetyStockLevel	smallint	N
ReorderPoint	smallint	N
StandardCost	money	N
ListPrice	money	N
Size	nvarchar(5)	N
Weight	decimal(8, 2)	N
DaysToManufacture	int	N
ProductLine	nvarchar(10)	N
Class	nvarchar(10)	N
Style	nvarchar(10)	N
WeightUnitMeasureName	nvarchar(50)	N
SizeUnitMeasureName	nvarchar(50)	N
SubCategoryName	nvarchar(50)	N
CategoryName	nvarchar(50)	N
SellStartDate	date	N
SellEndDate	date	N
CreatedDate	date	N
ModifiedDate	date	N
DQP	nvarchar(100)	N

Figura 3.4: DQP para os “Products”

Na Figura 3.5 é apresentado a tabela DQP para os “SalesPersons”.

dbo.SalesPersonsDQP		
BusinessEntityID	int	N
SalesQuota	money	N
Bonus	money	N
ComissionPct	smallmoney	N
SalesYTD	money	N
SalesLastYear	money	N
NationalIDNumber	nvarchar(15)	N
LoginID	nvarchar(256)	N
JobTitle	nvarchar(50)	N
BirthDate	date	N
MaritalStatus	nchar(10)	N
Gender	nchar(6)	N
HireDate	date	N
SalariedFlag	nvarchar(3)	N
VacationHours	smallint	N
SickLeaveHours	smallint	N
PersonType	nvarchar(20)	N
Title	nvarchar(8)	N
FirstName	nvarchar(50)	N
MiddleName	nvarchar(50)	N
LastName	nvarchar(50)	N
CreatedDate	date	N
ModifiedDate	date	N
DQP	nvarchar(100)	N

Figura 3.5: DQP para os “SalesPersons”

Na Figura 3.6 é apresentado a tabela DQP para os “ShipMethods”.

dbo.ShipMethods DQP		
ShipMethodID	int	N
Name	nvarchar(50)	N
ShipBase	money	N
ShipRate	money	N
CreatedDate	date	N
ModifiedDate	date	N
DQP	nvarchar(100)	N

Figura 3.6: DQP para os “ShipMethods”

Na Figura 3.7 é apresentado a tabela DQP para os “Currencies”.

dbo.Currencies DQP			
	CurrencyCode	nchar(3)	N
	Name	nvarchar(50)	N
	CreatedDate	date	N
	ModifiedDate	date	N
	DQP	nvarchar(100)	N

Figura 3.7: DQP para os “Currencies”

Na Figura 3.8 é apresentado a tabela DQP para os “FactSales”.

dbo.FactSalesDQP		
ProductNumber	nvarchar(25)	N
OrderDate	date	N
CustomerID	int	N
DueDate	date	N
BusinessEntityID	int	N
ShipMethodID	int	N
TerritoryID	int	N
CurrencyCode	nchar(3)	N
ShipDate	date	N
BillToAddress	int	N
ShipToAddress	int	N
SalesOrderDetailID	int	N
SalesOrderID	int	N
OrderQty	smallint	N
UnitPriceLocal	money	N
UnitPriceStandard	money	N
UnitPriceDiscountLocal	money	N
UnitPriceDiscountStandard	money	N
LineTotalLocal	numeric(38, 6)	N
LineTotalStandard	numeric(38, 6)	N
FreightLocal	money	N
FreightStandard	money	N
TotalDueLocal	money	N
TotalDueStandard	money	N
TaxAmountLocal	money	N
TaxAmountStandard	money	N
SubTotalLocal	money	N
SubTotalStandard	money	N
DQP	nvarchar(100)	N

Figura 3.8: DQP para os “FactSales”

Na Figura 3.9 é apresentado a tabela DQP para os “FactCurrency”.

dbo.FactCurrencyDQP		
	CurrencyRateDate	date
	FromCurrencyKey	int
	AverageRate	money
	EndOfDayRate	money
	DQP	nvarchar(100)

Figura 3.9: DQP para os “FactCurrency”

Por outro lado, a tabela que vai registar os registo duplicados ou idênticos, deve constar os registo identificados como duplicados ou idênticos e os seus fatores de similaridades. Para este efeito foi criada a seguinte tabela:

- ProductsDuplicated

3.2 Operações de transformação

No que diz respeito a operações de transformação devem ser realizadas as seguintes transformações no carregamento das dimensões:

- Transformação do ‘Marital Status’ da tabela Customer e Employees
- Transformação do ‘Gender’ da tabela Customer e Employees
- Transformação do ‘SalariedFlag’ da tabela Employees
- Transformação do ‘FinishedGoodsFlag’ da tabela Products

Na Tabela 3.1 são apresentadas as transformações que devem ser feitas na coluna ‘Marital Status’.

Tabela 3.1: Transformações do Marital Status

Transformação	
Atributo	Atributo
NULL	Unknown
S	Single
M	Married
D	Divorced
W	Widowed

Na Tabela 3.2 são apresentadas as transformações que devem ser feitas na coluna 'Gender'.

Tabela 3.2: Transformações do Gender

Transformação	
Atributo	Atributo
F	Female
M	Male

Na Tabela 3.3 são apresentadas as transformações que devem ser feitas na coluna 'Salaried Flag'.

Tabela 3.3: Transformações do SalariedFlag

Transformação	
Atributo	Atributo
1	Yes
0	No

Na Tabela 3.4 são apresentadas as transformações que devem ser feitas na coluna 'Finished Goods Flag'.

Tabela 3.4: Transformações do FinishedGoodsFlag

Transformação	
Atributo	Atributo
1	Yes
0	No

3.3 Operações de limpeza

Em relação, a operações de limpeza, devem ser filtrados todos os registo que apresentem problemas de qualidade, infrinjam as regras de negócio e aqueles que forem classificados como registo duplicados ou aproximadamente iguais. Os registo não devem ser carregados nas dimensões e por sua vez devem ser encaminhados para as tabelas que guardam os dados rejeitados para posteriormente serem corrigidos. Foram identificados alguns problemas de qualidade de dados como, por exemplo:

- Na tabela products, os produtos com os identificadores 'FR-M21B-40' e 'FR-M21B-52', entre outros, apresentam os restantes dos seus atributos com valor NULL
- Na tabela employee, os empregados com os identificadores 14 e 15, entre outros, apresentam o valor NationalIDNumber incorreto devido a apresentarem só 8 dígitos

Capítulo 4

Modelo Dimensional

Neste capítulo é descrito as dimensões e tabela de factos pensadas para o problema descrito no início do documento.

4.1 Modelo Dimensional

Na Figura 4.1 está representado o modelo dimensional proposto com as dimensões e respetiva tabela de factos.

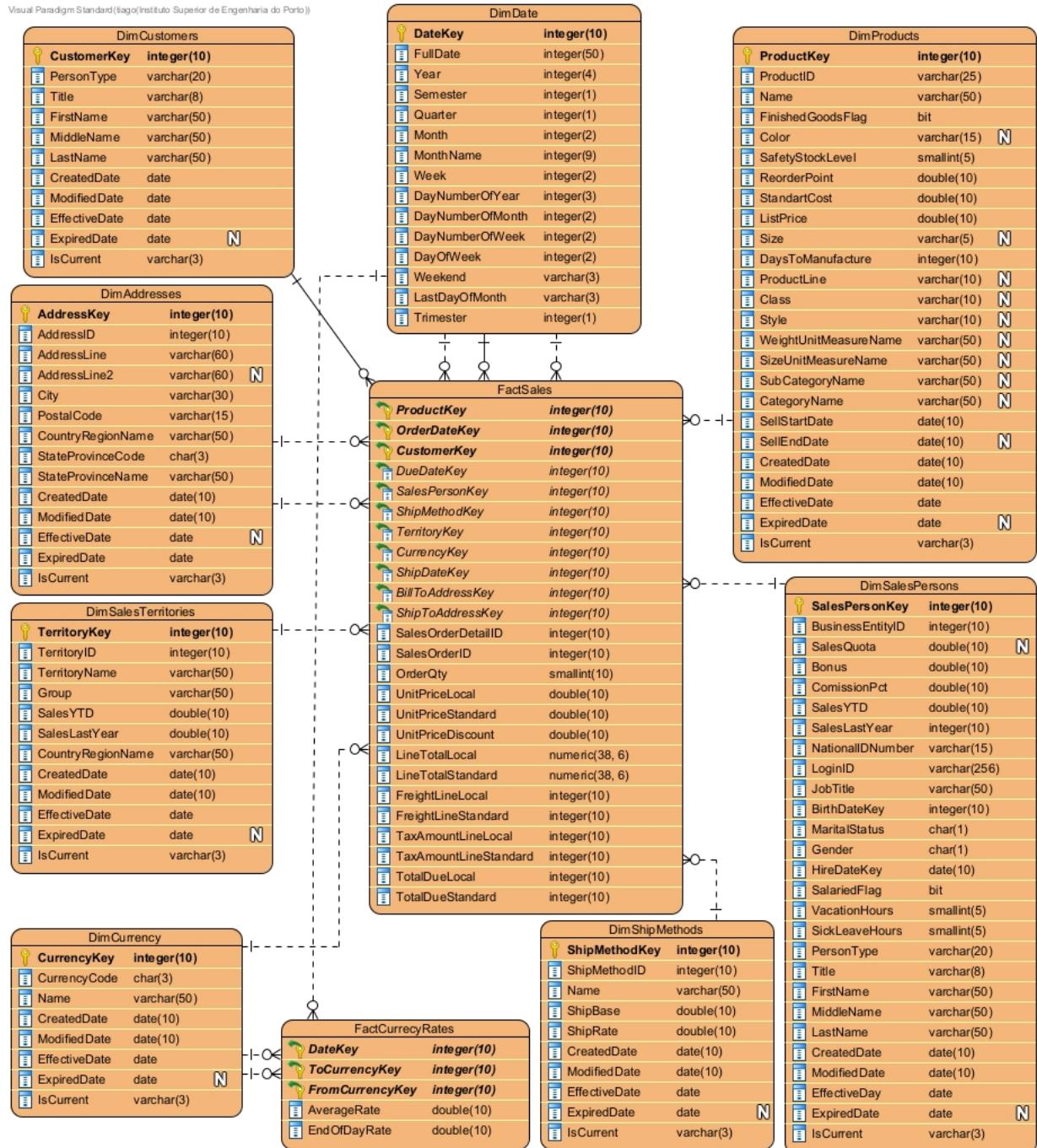


Figura 4.1: Modelo Dimensional

4.2 Metodologia de Kimball

4.2.1 Área de negócio

Este projeto está inserido na área de negócio de venda, neste caso venda de bicicletas e acessórios.

4.2.2 Nível de granularidade

O armazém de dados a desenvolver deverá ser concebido de modo a permitir a realização de consultas/análises aos dados das vendas no nível de granularidade mais elementar, ou seja, as vendas da empresa por dia, por produto, por cliente, por moeda, por empregado, por método de envio, por endereço, por territórios e por detalhe da compra.

4.2.3 Dimensões identificadas

As dimensões identificadas foram as seguintes:

- DimDate
- DimCurrency
- DimSalesPerson
- DimCustomer
- DimProduct
- DimAddress
- DimShipMethod
- DimTerritories

4.2.4 Tabelas de facto identificadas

As tabelas de facto identificadas foram as seguintes:

- FactSales
- FactCurrencyRate

4.3 Descrição das dimensões

Na dimensão 'DimDate' foram identificadas as seguintes colunas:

- Chave Primaria: DateKey (int)
- FullDate (datetime)
- Year (int)
- Semester (tinyint)
- Quarter (tinyint)
- Month (tinyint)
- MonthName (nvarchar(10))
- Week (tinyint)
- DayNumberOfYear (int)
- DayNumberOfMonth (tinyint)
- DayNumberOfWeek (tinyint)
- DayOfWeek (nvarchar(10))
- Weekend (nvarchar(3))
- LastDayOfTheMonth (nvarchar(3))
- Trimester (tinyint)

Na dimensão 'DimCurrency' foram identificadas as seguintes colunas:

- Chave Primaria: CurrecyKey (int)
- CurrencyCode (nchar(3))
- Name (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)

Na dimensão 'DimSalesPerson' foram identificadas as seguintes colunas:

- Chave Primaria: SalesPersonKey (inr)
- BusinessEntityID (int)
- SalesQuota (money)
- Bonus (money)
- ComissionPct (money)
- SalesYTD (money)
- SalesLastYear (money)
- NationalIDNumber (nvarchar(15))
- LoginID (nvarchar(256))
- JobTitle (nvarchar(50))
- BirthDate (date)
- MaritalStatus (nchar(10))
- Gender (nchar(6))
- HireDate (date)
- SalariedFlag (nvarchar(3))
- VacationHours (smallint)
- SickLeaveHours (smallint)
- PersonType (nvarchar(20))
- Title (nvarchar(8))
- FirstName (nvarchar(50))
- MiddleName (nvarchar(50))
- LastName (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)
- EffectiveDate (date)
- ExpireDate (date)
- IsCurrent (nvarchar(3))

Na dimensão 'DimCustomer' foram identificadas as seguintes colunas:

- Chave Primaria: CustomerKey (int)
- CustomerID (int)
- CreatedDate (date)
- ModifiedDate (date)
- PersonType (nvarchar(20))
- Title (nvarchar(8))
- FirstName (nvarchar(50))
- MiddleName (nvarchar(50))
- LastName (nvarchar(50))
- CreatedDate (date)
- ModifiedDate (date)
- EffectiveDate (date)
- ExpireDate (date)
- IsCurrent (nvarchar(3))

Na dimensão 'DimProduct' foram identificadas as seguintes colunas:

- Chave Primaria: ProductKey (int)
- ProductID (nvarchar(25))
- Name (nvarchar(50))
- FinishedGoodsFlag (nvarchar(3))
- Color (nvarchar(15))
- SafetyStockLevel (smallint)
- ReorderPoint (smallint)
- StandardCost (money)
- ListPrice (money)

- Size (nvarchar(5))
- DaysToManufacture (int)
- ProductLine (nvarchar(10))
- Class (nvarchar(10))
- Style (nvarchar(10))
- WeightUnitMeasureName (nvarchar(50))
- SizeUnitMeasureName (nvarchar(50))
- SubCategoryName (nvarchar(50))
- CategoryName (nvarchar(50))
- SellStartDate (date)
- SellEndDate (date)
- CreatedDate (date)
- ModifiedDate (date)
- EffectiveDate (date)
- ExpireDate (date)
- IsCurrent (nvarchar(3))

Na dimensão 'DimAddress' foram identificadas as seguintes colunas:

- Chave Primaria: AddressKey (int)
- AddressID (int)
- AddressLine (nvarchar(60))
- AddressLine2 (nvarchar(60))
- City (nvarchar(30))
- PostalCode (nvarchar(15))
- CountryRegionName (nvarchar(50))
- StateProvinceCode (nchar(3))
- StateProvinceName (nvarchar(60))

- CreatedDate (date)
- ModifiedDate (date)
- EffectiveDate (date)
- ExpireDate (date)
- IsCurrent (nvarchar(3))

Na dimensão 'DimShipMethod' foram identificadas as seguintes colunas:

- Chave Primaria: ShipMethodKey (int)
- ShipMethodID (int)
- Name (nvarchar(50))
- ShipBase (money)
- ShipRate (money)
- ModifiedDate (date)
- CreatedDate (date)
- EffectiveDate (date)
- ExpireDate (date)
- IsCurrent (nvarchar(3))

Na dimensão 'DimTerritories' foram identificadas as seguintes colunas:

- Chave Primaria: TerritoryKey (int)
- TerritoryID (int)
- TerritoryName (nvarchar(50))
- Group (nvarchar(50))
- SalesYTD (money)
- SalesLastYear (money)
- CountryRegionName (nvarchar(50))
- CreatedDate (date)

- ModifiedDate (date)
- EffectiveDate (date)
- ExpireDate (date)
- IsCurrent (nvarchar(3))

4.4 Descrição das tabelas de facto

Na tabela de factos 'FactSales' foram identificadas as seguintes colunas:

- Chave Primaria: ProductKey (int, Chave Estrangeira: DimProducts_ProductKey)
- SalesPersonKey (int, Chave Estrangeira: DimSalesPerson_SalesPersonKey)
- Chave Primaria: CustomerKey (int, Chave Estrangeira: DimCustomers_CustomerKey)
- ShipMethodKey (int, Chave Estrangeira: DimShipMethod_ShipMethodKey)
- BillToAddress (int, Chave Estrangeira: DimAddress_AddressKey)
- ShipToAddress (int, Chave Estrangeira: DimAddress_AddressKey)
- TerritoryKey (int, Chave Estrangeira: DimTerritories_TerritoryKey)
- CurrencyKey (int, Chave Estrangeira: DimCurrency_CurrencyKey)
- Chave Primaria: OrderDateKey (int, Chave Estrangeira: DimDate_DateKey)
- DueDateKey (int, Chave Estrangeira: DimDate_DateKey)
- ShipDateKey (int, Chave Estrangeira: DimDate_DateKey)
- SalesOrderID (int)
- SalesOrderDetailID (int)
- OrderQty (smallint)
- UnitPriceLocal (money)
- UnitPriceStandard (money)
- UnitPriceDiscountLocal (money)
- UnitPriceDiscountStandard (money)
- FreightLineLocal (money)

- FreightLineStandard (money)
- TaxAmountLineLocal (money)
- TaxAmountLineStandard (money)
- LineTotalLocal (numeric(38,6))
- LineTotalStandard (numeric(38,6))

Na tabela de factos 'FactCurrencyRate' foram identificadas as seguintes colunas:

- Chave Primaria: DateKey (int, Chave Estrangeira: DimDate_DateKey)
- Chave Primaria: ToCurrencyKey (int, Chave Estrangeira: DimCurrency_CurrencyKey)
- Chave Primaria: FromCurrencyKey (int, Chave Estrangeira: DimCurrency_CurrencyKey)
- AverageRate (money)
- EndOfDayRate (money)

4.5 Justificação de cada dimensão

Para melhor se compreender as decisões da implementação das dimensões esta secção foi criada, para haver uma breve justificação da transformação do modelo relacional no dimensional.

Primeiramente pode começar-se pelo desdobramento das 3 tabelas 'Persons', 'Employee' e 'Customers' que foram transformadas em DimSalesPerson e DimCustomers, tendo a tabela 'Persons' sido embebida nas colunas de cada uma das tabelas 'DimSalesPerson' e 'Customers', gerando, assim, estas dimensões. Deste modo, já que a tabela 'Persons' foi introduzida nas novas dimensões, a chave primária da tabela 'Persons' ('BusinessEntityID') pode ser descartada visto que já não tem utilidade.

Além da tabela 'Persons' foi também descartada a tabela 'CountryRegions', onde se aproveitou apenas o nome do país, incorporando-o na dimensão 'DimSalesTerritories' e na dimensão 'DimAddress', provenientes das tabelas 'Address' e 'SalesTerritories', onde os restantes atributos destas se mantiveram na transformação para as dimensões. Além disso, a 'DimAddress' também integra os nomes da tabela 'StateProvince', sendo o restante conteúdo desta descartado.

Seguidamente, tem-se o desdobramento das tabelas Currency e CurrencyRate na dimensão Currency e na tabela de factos FactCurrencyRate. Esta tabela de factos é criada para poder haver paralelamente à tabela de factos, algo que consiga converter a moeda em que foi paga a venda noutra, para ser mais explícito o valor pago.

Outra alteração relevante, tendo em conta o modelo original, é a agregação das tabelas 'ProductSubCategories', 'ProductCategories' e 'Products', numa única dimensão, 'DimProducts'. Desta forma, aproveitam-se as colunas 'Name' das duas primeiras tabelas, integrando-as na 'DimProducts', onde estarão as colunas originais da tabela 'Products'.

Tem-se também as dimensões 'DimDate' e 'DimShipMethods', sendo a primeira uma dimensão que não é proveniente de nenhuma tabela em concreto, sendo criada de raiz. Por outro lado, a 'DimShipMethods' é diretamente proveniente da tabela 'ShipMethods' não sofrendo alterações.

Finalmente, tem-se a tabela de factos principal, afeta ao negócio. Neste caso o negócio prende-se com as vendas, pelo que as informações relativas às vendas serão as que devem estar na tabela de factos, neste caso 'FactSales', que será composta pelas 'Keys' todas das dimensões criadas anteriormente.

Note-se que ao longo da transformação do modelo relacional no modelo dimensional foram descartadas as colunas 'Creation Date' e 'Modified Date' das tabelas que foram eliminadas, como 'CountryRegions' ou 'ProductCategories', já que as dimensões nas quais foram incorporadas já possuem esses dois atributos. Assim, se na dimensão 'DimProduct' se pretender atualizar a categoria do produto, pode usar-se a 'key' da dimensão e atualizar a 'Modified Date' dessa dimensão com a categoria, não tendo cabimento a incorporação de dois registo de 'Creation Date' e 'Modified Date', os provenientes das tabelas incorporadas em dimensões.

4.6 Justificação de cada tabela de factos

Para melhor se compreender as decisões da implementação das tabelas de factos esta secção foi criada, para haver uma breve justificação da transformação do modelo relacional no dimensional.

No que diz respeito a tabela de factos "FactSales", esta apresenta as chaves primárias de todas as dimensões criadas com a adição das métricas encontradas na base de dados fonte relacionadas com as vendas ao nível de cada linha de uma encomenda. Algumas das métricas encontradas encontravam ao nível da encomenda, por isso foi preciso dividir os valores pelas várias linhas da mesma encomenda para manter-se a mesma granularidade entre os vários elementos da tabela de factos. Além disso, as métricas monetárias são apresentadas na sua moeda local tal como na moeda standard, neste caso, o euro.

E finalmente, a tabela de factos "FactCurrencyRate", esta apresenta uma dupla ligação à dimensão "DimCurrency" que vai representar as moedas na qual se está a fazer o cambio e as respetivas métricas que neste caso são os valores de conversão.

4.7 Justificação das chaves primárias das tabelas de facto

Para que se possa identificar unicamente os registos na tabela de factos, tem de se identificar quais são as chaves primárias de cada tabela de factos (tendo em conta que nas dimensões a chave primária será sempre a Key de cada uma).

Deste modo, existem 3 abordagens possíveis, dependendo de certas restrições na simbiose entre as vendas e o seu armazém de dados.

Primeiramente, assumindo que o cliente só pode fazer uma compra por dia, e que só pode comprar um produto por compra, a chave primária composta será constituída pelo 'CustomerKey' e pelo 'DateKey', já que com a data e com o identificador do cliente, consegue obter-se o resto dos registos da venda.

Já numa perspetiva onde cada venda por dia pode conter mais do que um produto, ao nível da linha, será necessário juntar a 'ProductKey' à chave primária composta para identificar qual o produto comprado.

Por fim, no caso de não haver restrições sobre se se pode fazer mais de uma compra por dia e com quantos produtos, a chave primária deverá ser composta pela 'TransactionKey' e pela 'ProductKey'. Isto deve-se ao facto de a 'DimTransaction' ser possível identificar o número da fatura, fatura essa que é de um cliente numa data, pelo que se pode trocar essas duas chaves pela 'TransactionKey', adicionando-se a 'ProductKey' para diferenciar os produtos na mesma fatura. Uma solução que se poderia implementar neste caso, seria adicionar uma 'DimTime', para que se possa registar as compras feitas no mesmo dia, mas tendo em conta que não existe no modelo relacional essa indicação, não foi incorporado no novo modelo relacional.

Quanto à FactCurrencyRate, a chave primária é composta pela 'DateKey', 'ToCurrencyKey' e 'FromCurrencyKey', já que com uma data e uma transformação de moeda é possível obter qual foi o rácio para esse dia.

4.8 Mapeamento das colunas

Para demonstrar o mapeamento entre sistemas fontes, foi utilizado um Excel, cujo conteúdo está presente no ficheiro anexo.

Capítulo 5

Arquitetura da solução

No presente capítulo é descrito a arquitetura da solução pensada.

5.1 Arquitetura da solução

Na Figura 5.1 é apresentada a arquitetura da solução pensada. A arquitetura é dividida em quatro partes, nomeadamente:

- Fonte
- Staging Area
- Data Mart
- Análise

No caso, da fonte, este é a primeira etapa da arquitetura. A fonte dos dados pode ser bases de dados, sistemas externos ou ficheiros de dados, neste caso, a fonte de dados é uma base de dados (Bikes & Bikes), uma base de dados relacional cujas tabelas e registos já estão povoados, passíveis de serem recolhidos para a Staging Area. Este é o local onde os dados são armazenados temporariamente, os dados nesta fase são transformados, limpos e estruturados. Esta etapa permite garantir a qualidade dos dados através da criação de Lookups e DQP que possam assegurar que os dados a serem utilizados serão únicos e coerentes. Os Lookups que estarão contidos na Staging Area terão o propósito de transformar registos de uma letra numa palavra como, por exemplo, o 'Marital Status' (de 'S' para 'Single'), de modo a haver uma explicitação maior do valor do registo. Por outro lado, haverá um DQP para as seguintes tabelas: 'Customer', 'Address', 'SalesPerson', 'SalesTerritórios', 'Address' de modo a limpar os dados. Finalmente, na StagingArea, serão criadas tabelas para duplicados sobre as

mesmas tabelas. A seguir, os dados são organizados e armazenados de uma maneira mais estruturada no Data Mart, desta maneira os dados são acedidos mais rapidamente e com uma maior facilidade, podendo desta forma, facilitar as consultas e a elaboração de relatórios, com o modelo dimensional desenvolvido, apropriado para otimizar as consultas de dados em armazéns com uma dimensão elevada de registo. Por último, por meio de ferramentas diversas de análise de dados é possível extrair-se informações valiosas a partir dos dados armazenados no Data Mart.

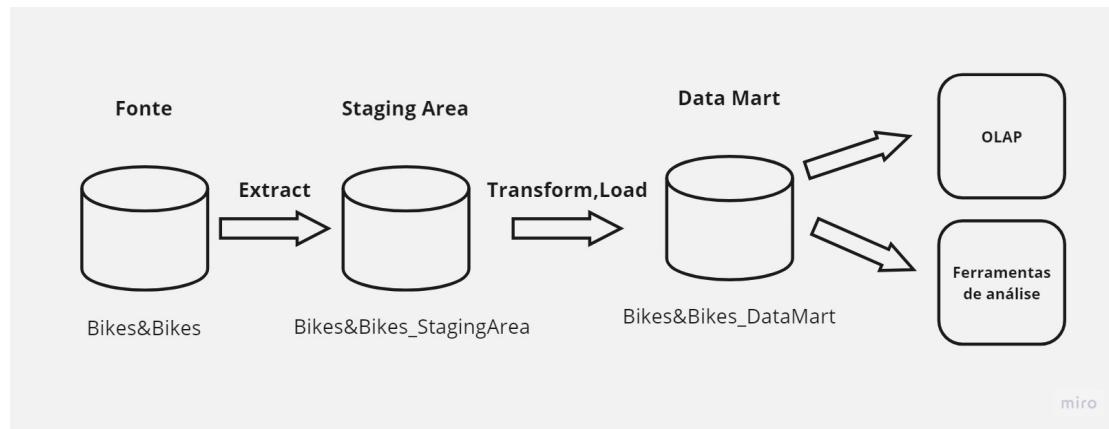


Figura 5.1: Arquitetura da Solução

Capítulo 6

Implementação da staging area e do datamart

Neste capítulo são descritos os passos realizados na criação da staging area tal como do datamart.

6.1 Organização do projeto

Na Figura 6.1 está representada de como o projeto foi organização. Para facilitar e evitar erros, tal como, conflitos no projeto correu-se a pacotes. Cada pacote contem as várias operações realizadas para se chegar ao objetivo final.

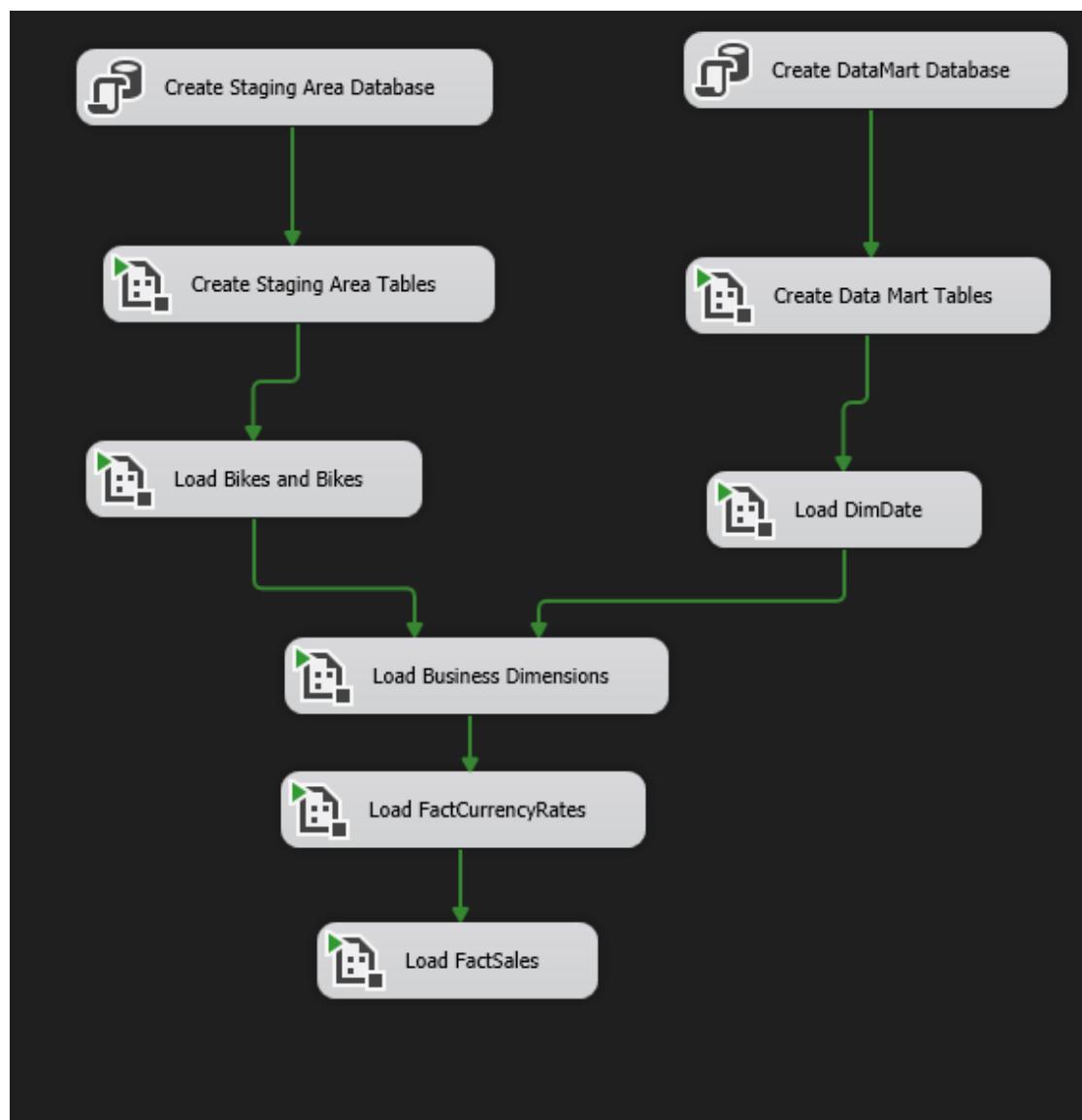


Figura 6.1: Exemplo de uma tabela criada para a staging area

Além disso, foi contemplado que o projeto pode ser executado em ambientes diferentes daqueles que são as máquinas de cada estudante, e por isso, houve um cuidado para que no “Project params” isso fosse contemplado, por meio de um caminho base, que pode ser visto na Figura 6.2.

Name	Data type	Value	Sensitive	Required
PackagePath	String	C:\VARMDD	False	True

Figura 6.2: Caminho uniformizado

6.2 Criação da staging area

Para a criação da staging area, correu-se ao próprio SQL server para este gerar um script com o schema de todas as tabelas da base de dados fonte. Com este script fez a divisão das tabelas em diferentes ficheiros.

6.2.1 Criação das tabelas

Como mencionado anteriormente, realizou-se uma divisão do ficheiro em vários ficheiros, sendo que, cada um contém o script de criação de uma só tabela. Na Figura 6.3 é possível observar um exemplo de um dos scripts criados para o efeito.

```
IF NOT EXISTS (SELECT name from sys.tables WHERE name = 'Addresses')
    CREATE TABLE [dbo].[Addresses](
        [AddressID] [int] NOT NULL,
        [AddressLine1] [nvarchar](60) NOT NULL,
        [AddressLine2] [nvarchar](60) NULL,
        [City] [nvarchar](30) NOT NULL,
        [StateProvinceID] [int] NOT NULL,
        [PostalCode] [nvarchar](15) NOT NULL,
        [CreatedDate] [date] NOT NULL,
        [ModifiedDate] [date] NOT NULL
    )
ELSE
    TRUNCATE TABLE Addresses
```

Figura 6.3: Exemplo de uma tabela criada para a staging area

6.2.2 Criação dos lookups

Na Figura 6.4 é possível observar-se um dos lookups criados. Neste é possível averiguar as transformações que a coluna MaritalStatus sofrerá posteriormente.

```
IF NOT EXISTS(SELECT name FROM sys.tables WHERE name = 'MaritalStatusDescription_Lookup')
BEGIN
    CREATE TABLE [dbo].[MaritalStatusDescription_Lookup](
        [MaritalStatus] [char](1) NULL,
        [MaritalStatusDescription] [nvarchar](10) NOT NULL
    ) ON [PRIMARY]
    INSERT [dbo].[MaritalStatusDescription_Lookup] ([MaritalStatus], [MaritalStatusDescription]) VALUES (NULL, N'Unknown')
    INSERT [dbo].[MaritalStatusDescription_Lookup] ([MaritalStatus], [MaritalStatusDescription]) VALUES (N'S', N'Single')
    INSERT [dbo].[MaritalStatusDescription_Lookup] ([MaritalStatus], [MaritalStatusDescription]) VALUES (N'M', N'Married')
    INSERT [dbo].[MaritalStatusDescription_Lookup] ([MaritalStatus], [MaritalStatusDescription]) VALUES (N'D', N'Divorced')
    INSERT [dbo].[MaritalStatusDescription_Lookup] ([MaritalStatus], [MaritalStatusDescription]) VALUES (N'W', N'Widowed')
END
```

Figura 6.4: Exemplo de um lookup

6.2.3 Criação dos DQPs

Na Figura 6.5 é possível observar-se um dos DQP criados neste caso para a tabela “Address”. Nesta tabela, em comparação à original, foi adicionada a coluna “DQP” do tipo “nvarchar(100)”.

```
IF NOT EXISTS (SELECT name from sys.tables WHERE name = 'AddressesDQP')
    CREATE TABLE [dbo].[AddressesDQP](
        [AddressID] [int] NOT NULL,
        [AddressLine1] [nvarchar](60) NOT NULL,
        [AddressLine2] [nvarchar](60) NULL,
        [City] [nvarchar](30) NOT NULL,
        [StateProvinceID] [int] NOT NULL,
        [PostalCode] [nvarchar](15) NOT NULL,
        [CreatedDate] [date] NOT NULL,
        [ModifiedDate] [date] NOT NULL,
        DQP nvarchar(100)
    )
ELSE
    TRUNCATE TABLE AddressesDQP
```

Figura 6.5: Exemplo de um DQP

6.2.4 Criação da base de dados da staging area

A criação da staging area foi iniciada pela criação da base de dados desta e por isso recorreu-se a um “Execute SQL Task” para ser executado o script de criação desta. Na Figura 6.7 está representado o componente usado para realizar-se essa tarefa. Neste componente foi introduzido o script disponível na Listagem 6.1.



Figura 6.6: Criação da base de dados

```
1 IF NOT EXISTS(SELECT name FROM sys.databases WHERE name = "BikesAndBikes_StagingArea") CREATE DATABASE BikesAndBikes_StagingArea
```

Listagem 6.1: Script para a criação da base de dados Staging Area

6.2.5 Criação das tabelas na base de dados

Após criada a base de dados procedeu-se a criação das tabelas na base de dados criada. Para isto foi criado um pacote, Figura 6.7 e dentro deste foi criado um ciclo no qual este percorrerá todos os ficheiros .sql que se encontrarem na pasta selecionada, Figura 6.8.



Figura 6.7: Pacote para a criação das tabelas

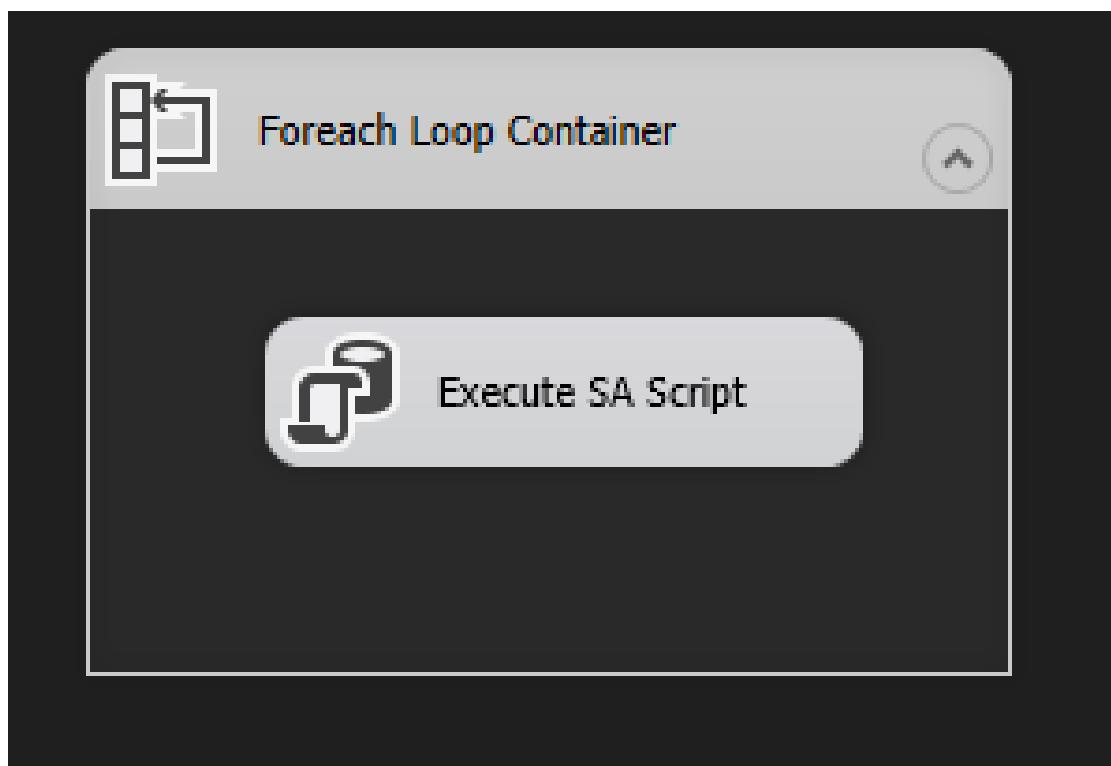


Figura 6.8: Criação das tabelas

Na Figura 6.9 é demonstrado o uso do parâmetro do projeto para o carregamento dos scripts para a staging area.

Foreach Loop Editor	
Enumerator	Foreach File Enumerator
Expressions	<code>@[\${Project::PackagePath} + "\\Folder\\StagingArea"]</code>

Figura 6.9: Caminho para a staging area

6.2.6 Carregamentos dos dados na staging area

Já com as tabelas criadas na base de dados da staging area procedeu-se à criação de um novo pacote, Figura 6.10, onde por sua vez criou-se um dataflow, Figura 6.11 e finalmente dentro deste foram feitas as ligações entre a base de dados fonte e a base de dados destino. Cada ligação corresponde à transferência dos dados de uma só tabela da tabela fonte para a tabela destino, Figura 6.12.



Figura 6.10: Carregamento dos dados na staging area

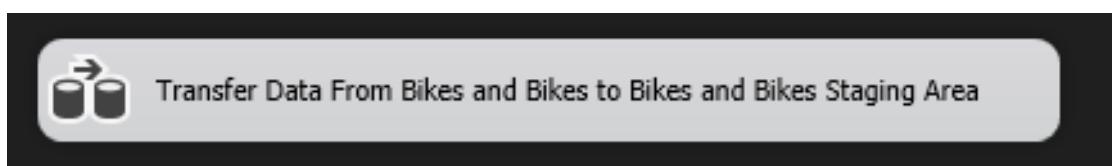


Figura 6.11: Dataflow para o carregamento dos dados na staging area

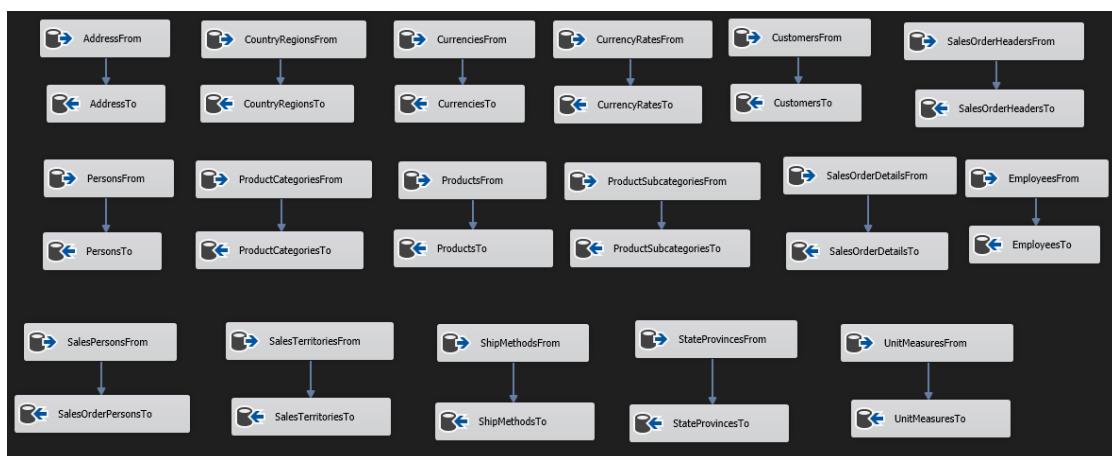


Figura 6.12: Fluxo para o carregamento dos dados

6.3 Criação do datamart

Para a criação do datamart começou-se pela criação dos scripts, a seguir fez o carregamento dos dados nas respetivas dimensões e por último desenvolveu-se as tabelas de facto.

6.3.1 Criação dos scripts

No que toca aos scripts desenvolvidos, na Figura 6.13 está apresentado um exemplo de um script desenvolvido, neste caso, para a dimensão dos “Addresses”.

```
IF NOT EXISTS (SELECT name from sys.tables WHERE name = 'DimAddresses')
BEGIN
    CREATE TABLE [dbo].[dimAddresses](
        [AddressKey] [int] IDENTITY(1,1) NOT NULL,
        [AddressID] [int] NOT NULL,
        [AddressLine] [nvarchar](60) NOT NULL,
        [AddressLine2] [nvarchar](60) NULL,
        [City] [nvarchar](30) NOT NULL,
        [PostalCode] [nvarchar](15) NOT NULL,
        [CountryRegionName] [nvarchar](50) NOT NULL,
        [StateProvinceCode] [nchar](3) NOT NULL,
        [StateProvinceName] [nvarchar](50) NOT NULL,
        [CreatedDate] [date] NOT NULL,
        [ModifiedDate] [date] NOT NULL,
        [EffectiveDate] [datetime] NOT NULL,
        [ExpiredDate] [datetime] NULL,
        [Is_Current] [nchar] (3) NOT NULL,
    )
    CONSTRAINT [PK_DimAddress] PRIMARY KEY CLUSTERED
    (
        [AddressKey] ASC
    )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
)
CREATE NONCLUSTERED INDEX [NonClusteredIndex-AddressKey] ON [dbo].[DimAddresses]
(
    [AddressKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
END
```

Figura 6.13: Exemplo do script criado para uma das dimensões

Para fins de otimização foram utilizados indexes, este permitem uma localização mais rápida de um certo registo quando é efetuada uma consulta. Na Figura 6.14 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “AddressKey” na dimensão “Address”.

```
CREATE NONCLUSTERED INDEX [NonClusteredIndex-AddressKey] ON [dbo].[DimAddresses]
(
    [AddressKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING = OFF, ONLINE = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
END
```

Figura 6.14: Index para a dimensão “Addresses”

Na Figura 6.15 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “CurrencyKey” na dimensão “Currency”.

```

CREATE NONCLUSTERED INDEX [NonClusteredIndex-CurrencyKey] ON [dbo].[DimCurrency]
(
    [CurrencyKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
END

```

Figura 6.15: Index para a dimensão “Currency”

Na Figura 6.16 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “CustomerKey” na dimensão “Customers”.

```

CREATE NONCLUSTERED INDEX [NonClusteredIndex-CustomerKey] ON [dbo].[DimCustomers]
(
    [CustomerKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
END

```

Figura 6.16: Index para a dimensão “Customers”

Na Figura 6.17 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “DateKey” na dimensão “Date”.

```

CREATE NONCLUSTERED INDEX [NonClusteredIndex-DateKey] ON [dbo].[DimDate]
(
    [DateKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
END

```

Figura 6.17: Index para a dimensão “Date”

Na Figura 6.18 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “ProductKey” na dimensão “Products”.

```

CREATE NONCLUSTERED INDEX [NonClusteredIndex-ProductKey] ON [dbo].[DimProducts]
(
    [ProductKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON)
END

```

Figura 6.18: Index para a dimensão “Products”

Na Figura 6.19 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “SalesPersonKey” na dimensão “SalesPersons”.

```
CREATE NONCLUSTERED INDEX [NonClusteredIndex-SalesPersonKey] ON [dbo].[DimSalesPersons]
(
    [SalesPersonKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING =
END
```

Figura 6.19: Index para a dimensão “SalesPersons”

Na Figura 6.20 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “SalesTerritoryKey” na dimensão “SalesTerritories”.

```
CREATE NONCLUSTERED INDEX [NonClusteredIndex-TerritoryKey] ON [dbo].[DimSalesTerritories]
(
    [TerritoryKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING =
END
```

Figura 6.20: Index para a dimensão “SalesTerritories”

Na Figura 6.21 é demonstrado a adição feita ao script de criação para realizar-se o registo do index para a chave “ShipMethodKey” na dimensão “ShipMethods”.

```
CREATE NONCLUSTERED INDEX [NonClusteredIndex-ShipMethodKey] ON [dbo].[DimShipMethods]
(
    [ShipMethodKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, SORT_IN_TEMPDB = OFF, DROP_EXISTING =
END
```

Figura 6.21: Index para a dimensão “ShipMethods”

6.3.2 Criação da base de dados datamart

A criação da staging area foi iniciada pela criação da base de dados desta e por isso recorreu-se a um “Execute SQL Task” para ser executado o script de criação desta. Na Figura 6.22 está representado o componente usado para realizar-se essa tarefa. Neste componente foi introduzido o script disponível na Listagem 6.2.



Figura 6.22: Criação da base de dados DataMart

```
1 IF NOT EXISTS(SELECT name FROM sys.databases WHERE name = "BikesAndBikes_DataMart") CREATE DATABASE BikesAndBikes_DataMart
```

Listagem 6.2: Script para a criação da base de dados DataMart

6.3.3 Criação das tabelas para o datamart

Após criada a base de dados procedeu-se a criação das tabelas na base de dados criada. Para isto foi criado um pacote, Figura 6.23 e dentro deste foi criado um ciclo no qual este percorrerá todos os ficheiros .sql que se encontrarem na pasta selecionada, Figura 6.24.



Figura 6.23: Pacote para a criação das tabelas do datamart

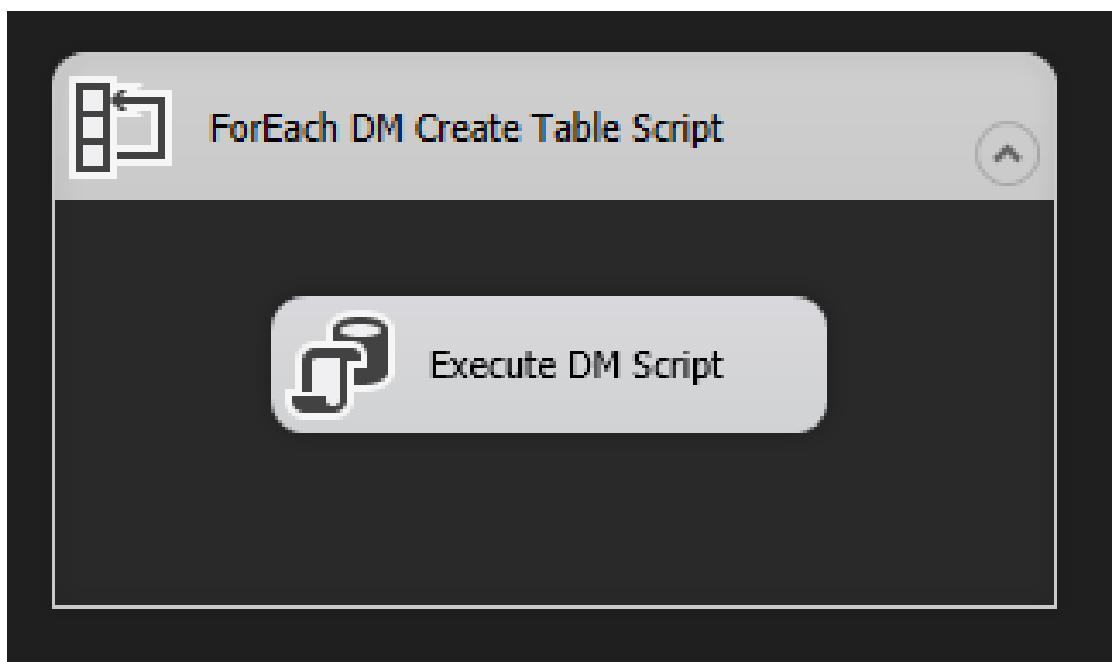


Figura 6.24: Ciclo para a criação das tabelas do datamart

Na Figura 6.25 é demonstrado o uso do parâmetro do projeto para o carregamento dos scripts para o datamart.

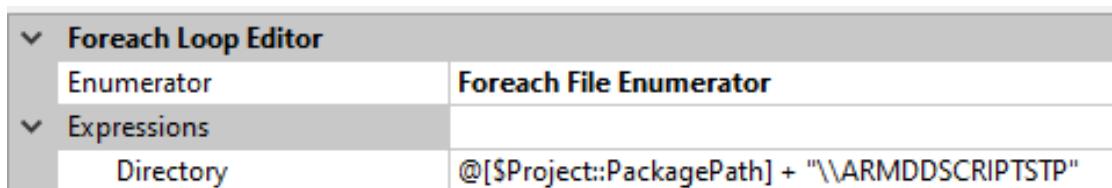


Figura 6.25: Caminho para o datamart

6.3.4 DimDate

Para se realizar a criação da dimensão “Date” foi preciso recorrer a uma “Analysis Services Multidimensional Project”, neste projeto é gerado uma tabela em que cada uma linha representa uma data. Neste caso foi escolhido como a data inicial 1 de janeiro de 2001 e data fim 31 de dezembro de 2040, dado o intervalo de valores apresentados nas informações fornecidas e foi também selecionado todos os períodos para criar uma dimensão “Date” mais complexa. Estes dados foram criados numa tabela situada na staging area.

Após criados os dados foi necessário criar um “Integration Services Project” para se

guardar os dados num ficheiro csv, para assim serem adicionados novos atributos para satisfazer assim as necessidades do projeto. Foram adicionados os seguintes parâmetros:

- Year
- MonthName
- DayOfTheWeek
- IsWeekend
- IsLastDayOfTheMonth

No caso da coluna “Year”, este foi preenchida com a fórmula na Listagem 6.3.

`1 =YEAR(C2)`

Listagem 6.3: Fórmula para a coluna Year

Por outro lado, a coluna “MonthName” foi preenchida com a fórmula na Listagem 6.4.

`1 =LEFT(L2;SEARCH“(;L2)-1)`

Listagem 6.4: Fórmula para a coluna Month

A coluna “DayOfTheWeek” foi preenchida através do preenchimento manual dos dias da semana para a primeira semana do ano e duplicação dos sete dias da semana para as restantes semanas.

A seguir, a coluna “IsWeekend” foi preenchida de uma forma semelhante à coluna anterior, onde a primeira semana foi preenchida com as palavras “Yes” (dia da semana em que é fim de semana) ou “No” (não é dia de fim de semana) e depois duplicado para as restantes semanas.

A coluna “IsLastDayOfTheMonth” foi preenchida através da fórmula na Listagem 6.5.

`1 =IF(BK2=EOMONTH(BK2; 0);”Yes”;”No”)`

Listagem 6.5: Fórmula para a coluna IsLastDayOfTheMonth

E por fim, a coluna “Season” foi preenchida através da fórmula na Listagem 6.6. Para a construção desta coluna foi assumido que o hemisfério norte e as estações meteorológicas.

```
1 =CHOOSE(MONTH(BK2); "Winter"; "Winter"; "Spring"; "Spring"; "Spring"; "Summer"; "Summer"; "Summer"; "Autumn"; "Autumn"; "Autumn"; "Winter")
```

Listagem 6.6: Fórmula para a coluna Season

Para o carregamento da dimensão “Date”, deu-se primeiro à criação de um package 6.26, de seguida foi criado o dataflow, Figura 6.27 onde a dimensão “Date” foi carregada a partir de um ficheiro .csv, Figura 6.28.

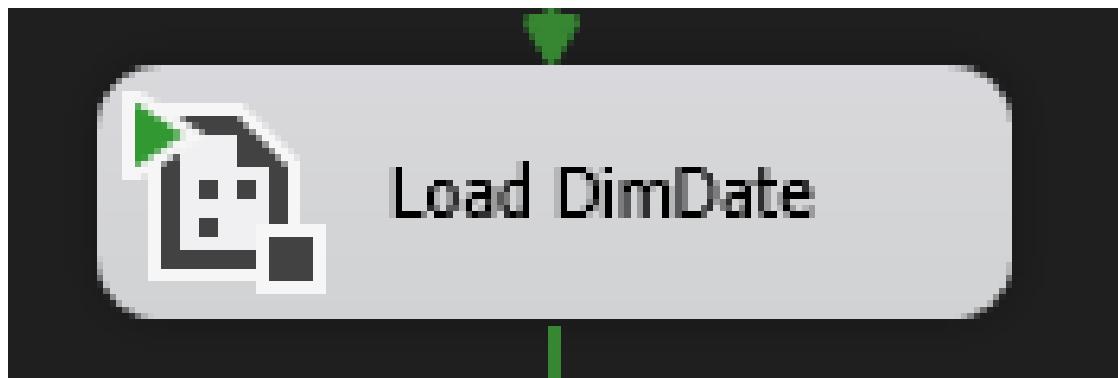


Figura 6.26: Pacote para o carregamento da dimensão data

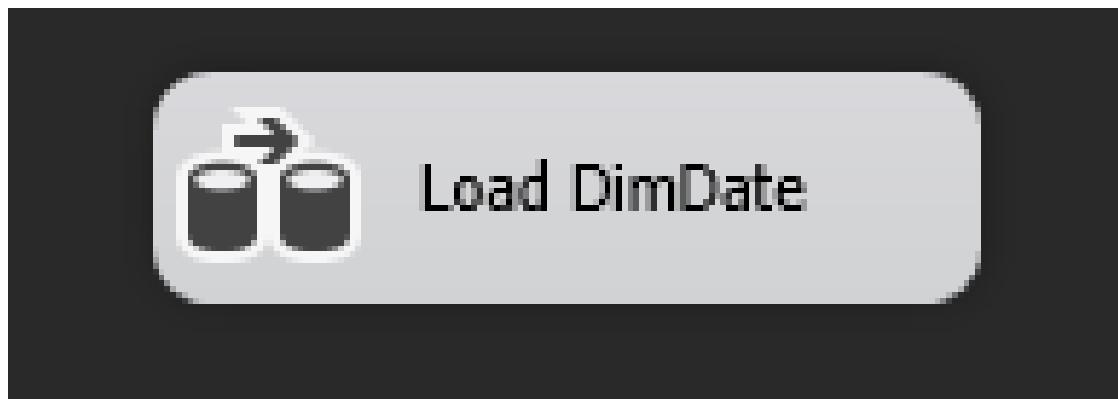


Figura 6.27: Dataflow para o carregamento da dimensão data

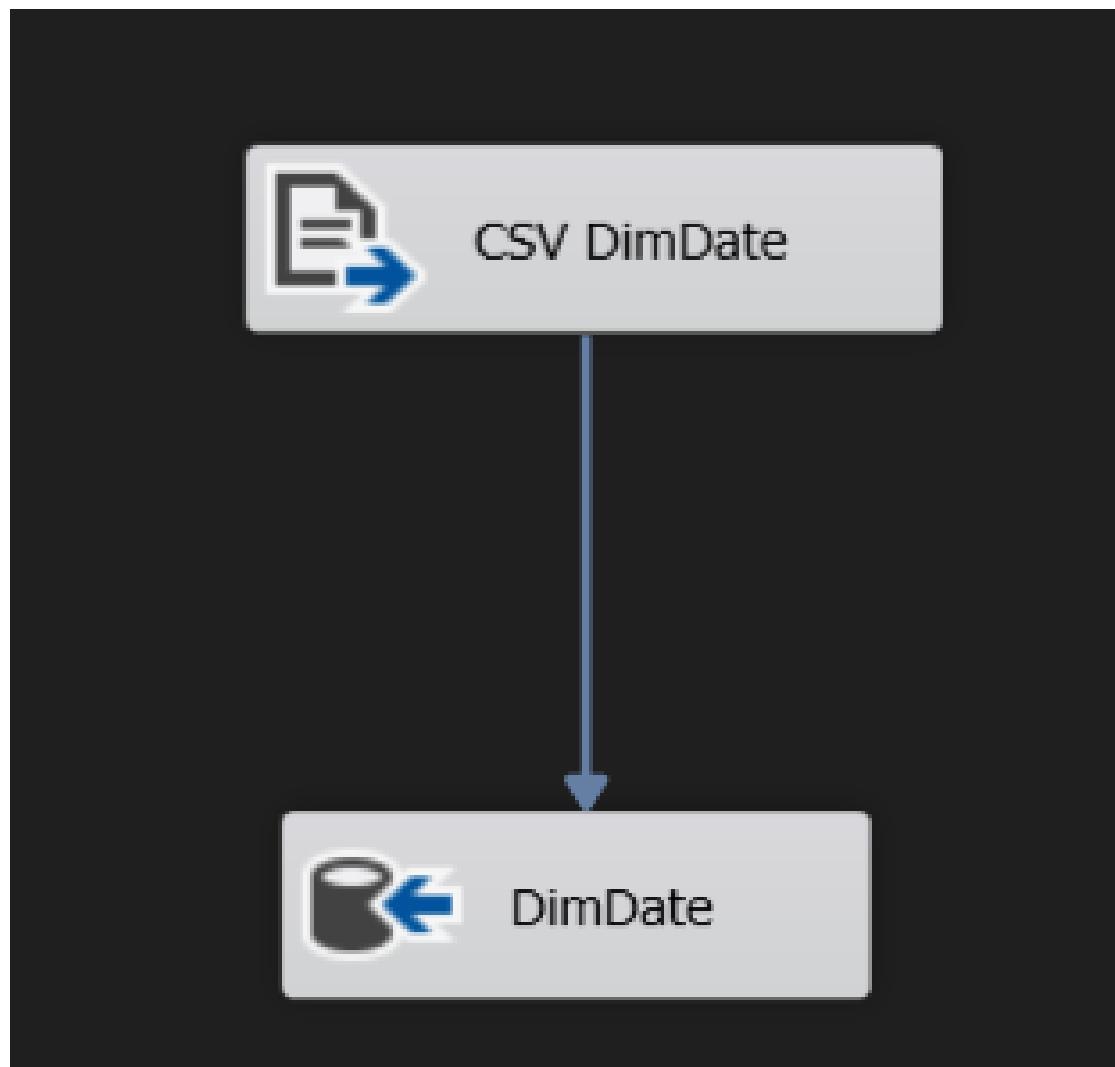


Figura 6.28: Carregamento da dimensão date

6.3.5 Dimensões

Já na parte do carregamento das dimensões de negócio começou-se pela criação do pacote, Figura 6.29, que por sua vez, contem os vários dataflows para as várias dimensões, Figura 6.30.



Figura 6.29: Pacote do carregamento das dimensões de negócio



Figura 6.30: Dataflows de carregamento para cada dimensão

6.3.6 DimCustomers

Na Listagem 6.7 é apresentado o script usado para realizar-se a agregação das tabelas.

```

1  SELECT      Customers.CustomerID, Persons.PersonType, Persons.Title, Persons.
   FirstName, Persons.MiddleName, Persons.LastName, Customers.CreatedDate,
   Customers.ModifiedDate
2  FROM        Customers left JOIN
3                  Persons ON Customers.PersonID = Persons.BusinessEntityID

```

Listagem 6.7: Script de agregação de tabelas

Na Figura 6.31 é apresentada a estrutura de carregamento dos dados para a dimensão “Customer”.

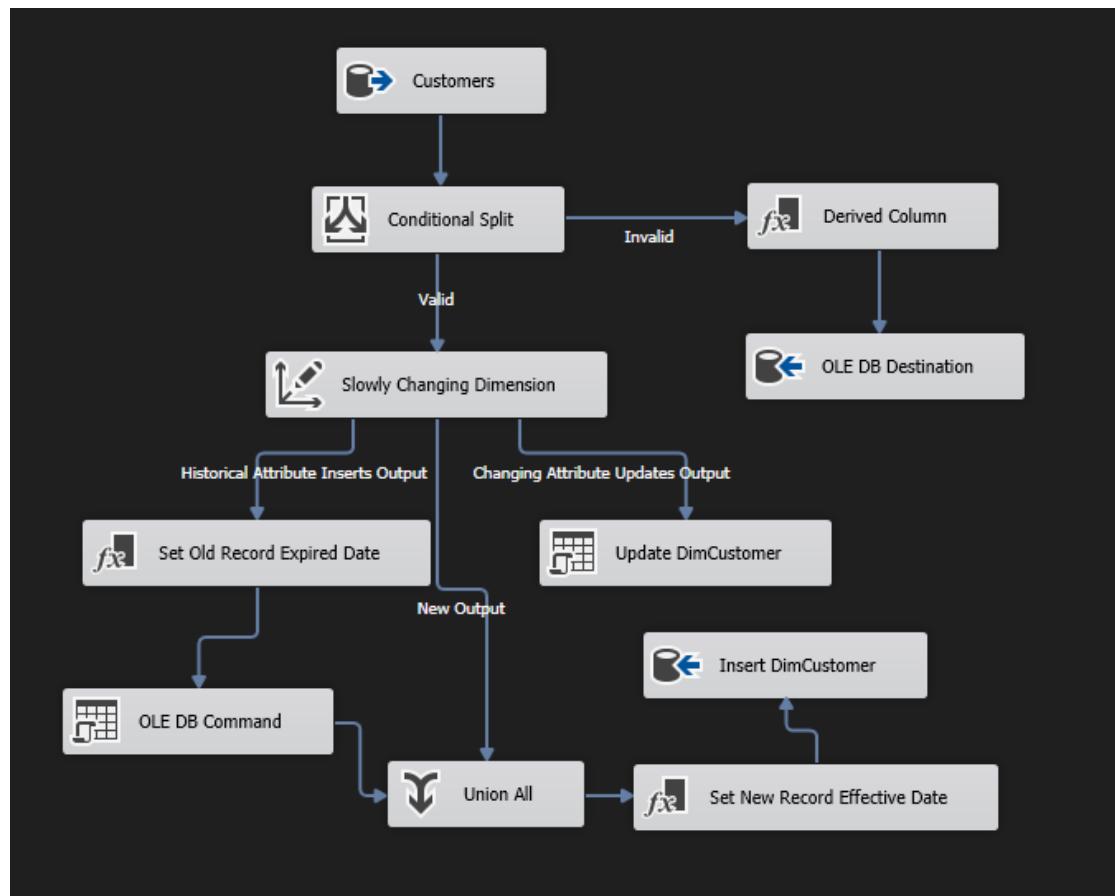


Figura 6.31: Dimensão “Customer”

Na Figura 6.32 é apresentada a validação realizada as linhas provenientes da query

de agregação das tabelas. Uma linha é considerada válida quando o parâmetro “PersonType” não é null.

Order	Output Name	Condition
1	Valid	!(ISNULL(PersonType))

Figura 6.32: Validação dimensão “Customer”

Na Figura 6.33 é apresentada a mensagem de erro introduzida com a linha classificada com inválida, originada do componente anterior.

Derived Column Name	Derived Column	Expression	Data Type	Length
DQP	<add as new column>	"Invalid customer"	Unicode string [DT_WS...]	1

Figura 6.33: DQP para a dimensão “Customer”

6.3.7 DimAddddresses

Na Listagem 6.8 é apresentado o script usado para realizar-se a agregação das tabelas.

```

1  SELECT      Addresses.AddressID, Addresses.AddressLine1, Addresses.AddressLine2,
2          Addresses.City, Addresses.PostalCode, StateProvinces.StateProvinceCode,
3          StateProvinces.Name AS StateProvinceName,
4          CountryRegions.Name AS CountryRegionName, Addresses.
5          CreatedDate, Addresses.ModifiedDate
6
7  FROM        Addresses INNER JOIN
8          StateProvinces ON Addresses.StateProvinceID =
9          StateProvinces.StateProvinceID INNER JOIN
10         CountryRegions ON StateProvinces.CountryRegionCode =
11         CountryRegions.CountryRegionCode

```

Listagem 6.8: Script de agregação de tabelas

Na Figura 6.35 é apresentada a estrutura de carregamento dos dados para a dimensão “Addresses”.

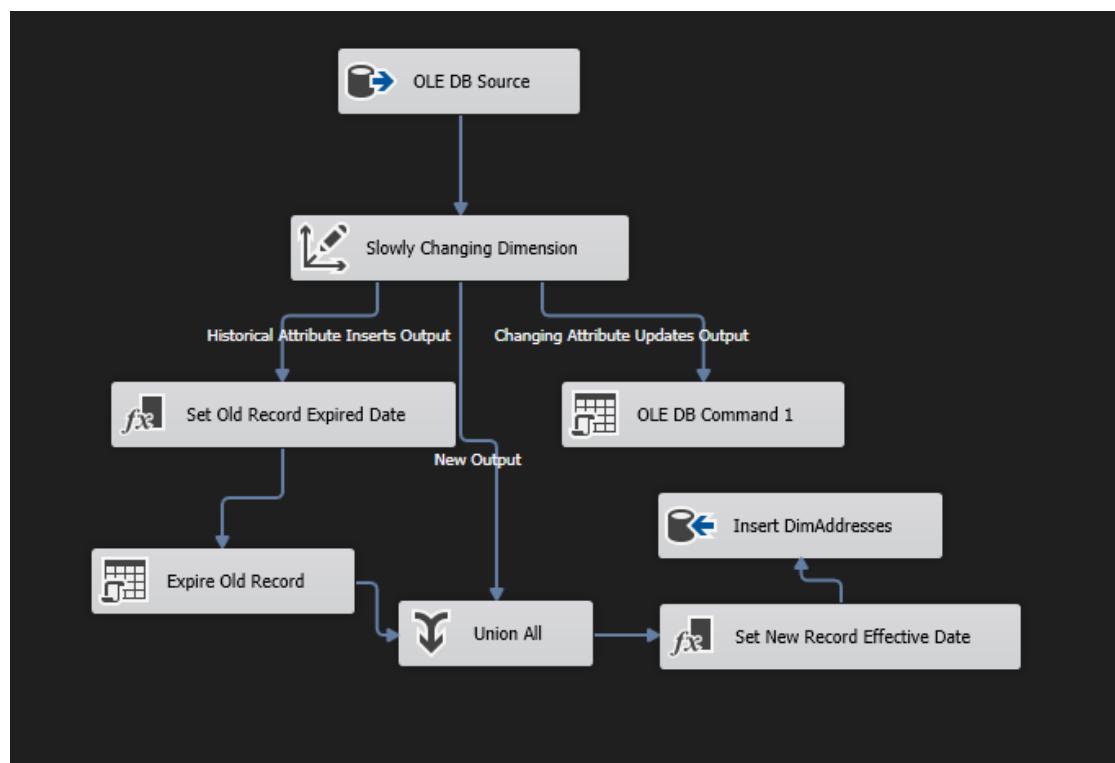


Figura 6.34: Dimensão “Addresses”

6.3.8 DimSalesTerritories

Na Listagem 6.9 é apresentado o script usado para realizar-se a agregação das tabelas.

```

1  SELECT      SalesTerritories.TerritoryID, SalesTerritories.Name, SalesTerritories.[
   Group], SalesTerritories.SalesYTD, SalesTerritories.SalesLastYear,
   CountryRegions.Name AS CountryRegionName, SalesTerritories.CreatedDate,
   SalesTerritories.ModifiedDate
2  FROM        CountryRegions INNER JOIN
3   SalesTerritories ON CountryRegions.CountryRegionCode =
   SalesTerritories.CountryRegionCode

```

Listagem 6.9: Script de agregação de tabelas

Na Figura 6.35 é apresentada a estrutura de carregamento dos dados para a dimensão “SalesTerritories”.

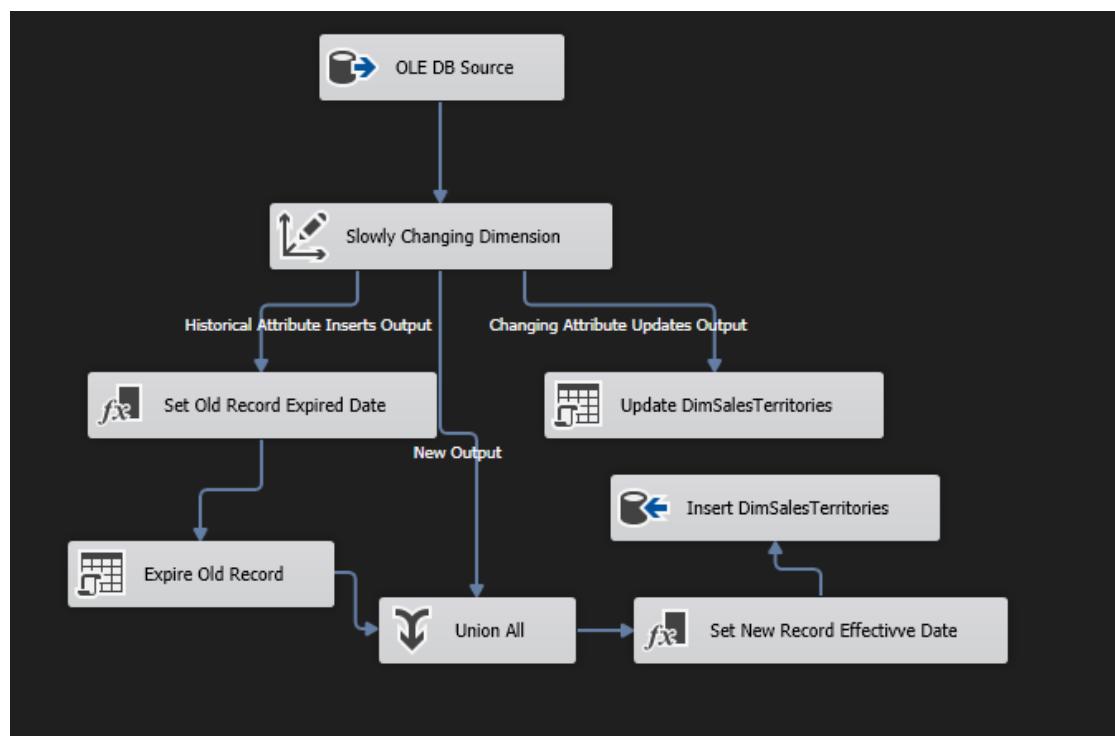


Figura 6.35: Dimensão “SalesTerritories”

6.3.9 DimCurrency

Para o carregamento dos dados para a dimensão “Currency” não foi necessário fazer a agregação com outras tabelas, por isso só foi necessário selecionar a tabela “Currency” localizada na “StagingArea”. Na Figura 6.36 é possível observar a seleção da mesma.

Name of the table or the view:
[dbo].[Currencies]

Figura 6.36: Dados provenientes da tabela “Currency”

Na Figura 6.37 está representada a forma de como a dimensão “Currency” foi desenvolvida. Nesta foi utilizada uma “Slowly Changing Dimension”.

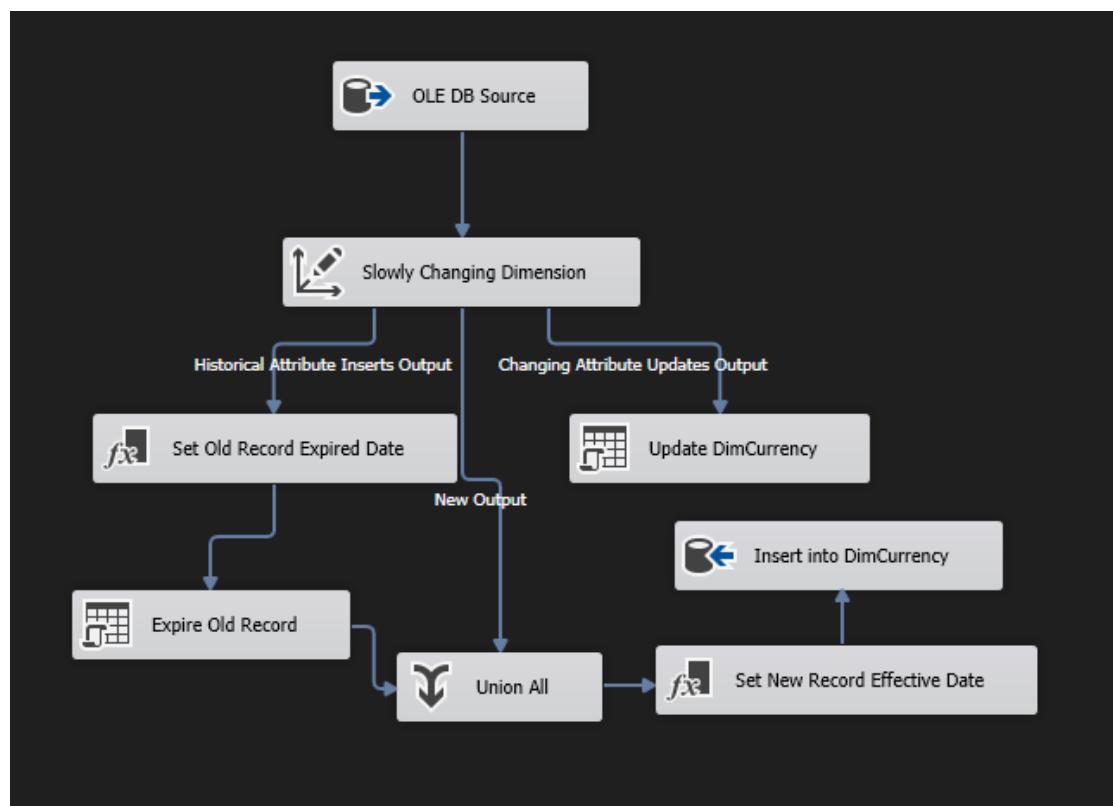


Figura 6.37: Dimensão “Currency”

6.3.10 DimProducts

Na Listagem 6.10 é apresentado o script usado para realizar-se a agregação das tabelas.

```
1 SELECT      Products.ProductNumber, Products.Name, Products.FinishedGoodsFlag,
2          Products.Color, Products.SafetyStockLevel, Products.ReorderPoint, Products.
3          StandardCost, Products.ListPrice, Products.[Size], Products.Weight,
4          Products.DaysToManufacture, Products.ProductLine,
5          Products.Class, Products.Style, Products.SellStartDate,
6          Products.SellEndDate, ProductSubcategories.Name AS
7          ProductSubCategoryName,
8          ProductCategories.Name AS ProductCategoryName,
          UnitMeasures.Name AS SizeUnitMeasureName,
          UnitMeasures_1.Name AS WeightMeasureName,
          Products.CreatedDate, Products.ModifiedDate
4 FROM        ProductSubcategories FULL JOIN
5          Products ON ProductSubcategories.ProductSubcategoryID =
6          Products.ProductSubcategoryID FULL JOIN
7          ProductCategories ON ProductSubcategories.
          ProductCategoryID = ProductCategories.
          ProductCategoryID LEFT JOIN
8          UnitMeasures ON Products.SizeUnitMeasureCode =
          UnitMeasures.UnitMeasureCode LEFT JOIN
          UnitMeasures AS UnitMeasures_1 ON Products.
          WeightUnitMeasureCode = UnitMeasures_1.
          UnitMeasureCode
```

Listagem 6.10: Script de agregação de tabelas

Na Figura 6.38 é apresentada a estrutura de carregamento dos dados para a dimensão “Products”.

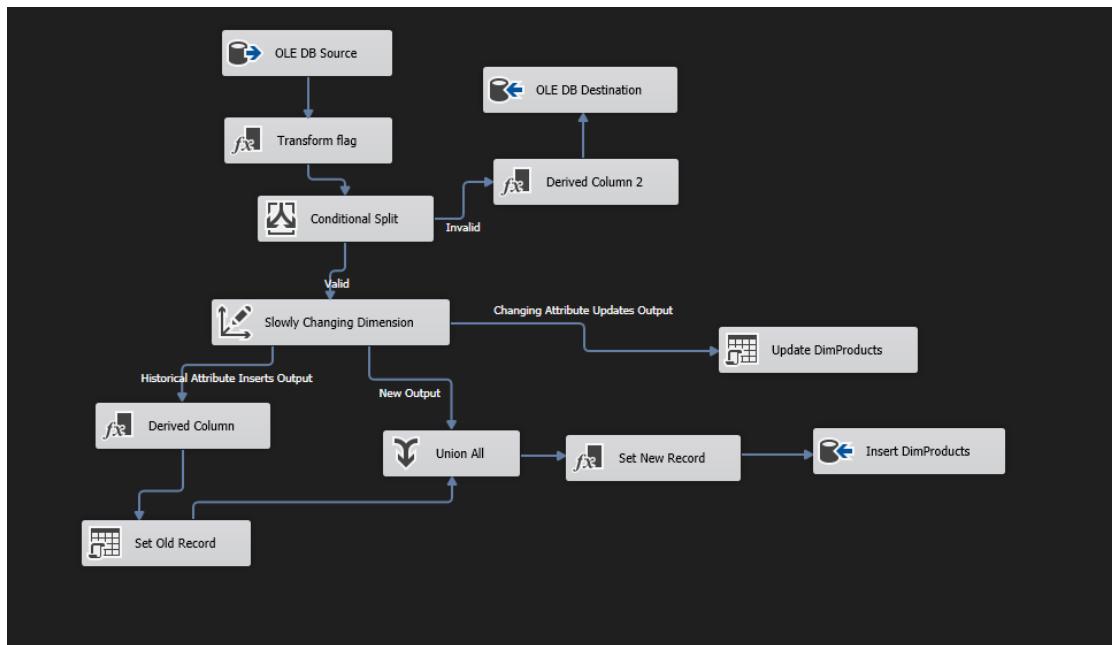


Figura 6.38: Dimensão “Products”

Na Figura 6.39 é apresentada a transformação da flag “FinishedGoodsFlag”. Neste passo foi realizado o mapeamento de 1 para “Yes” e 0 para “No”.

Derived Column Name	Derived Column	Expression	Data Type	Length
FinishedGoodsFlag_Tra...	<add as new column>	FinishedGoodsFlag == TRUE ? "Yes" : "No"	Unicode string [DT_WS...]	3

Figura 6.39: Flag para Dimensão “Products”

Na Figura 6.40 é apresentada a validação realizada para o parâmetro “Name”. Para cada linha é procede-se a esta validação. As linhas que apresentem o parâmetro “Name” a “null” são consideradas como invalidas, por isso, para serem consideradas válido devem apresentar o seu parâmetro “Name” preenchido por um valor diferente de “null”.

Order	Output Name	Condition
1	Valid	!(ISNULL(Name))

Figura 6.40: Validação para a Dimensão “Products”

Na Figura 6.41 é apresentado o processo seguinte realizado caso uma linha seja considerada como invalida. A esta mesma linha é adicionado o parâmetro “DQP” e a seguir,

esta será registada na tabela criada para o efeito.

Derived Column Name	Derived Column	Expression	Data Type	Length
DQP	<add as new column>	"Invalid entry"	Unicode string [DT_WS...]	1

Figura 6.41: DQP para a Dimensão “Products”

6.3.11 DimShipMethods

Para o carregamento dos dados para a dimensão “ShipMethods” não foi necessário fazer a agregação com outras tabelas, por isso só foi necessário selecionar a tabela “ShipMethods” localizada na “StagingArea”. Na Figura 6.42 é possível observar a seleção da mesma.



Figura 6.42: Dados provenientes da tabela “ShipMethods”

Na Figura 6.43 está representada a forma de como a dimensão “ShipMethod” foi desenvolvida. Nesta foi utilizada uma “Slowly Changing Dimension”.

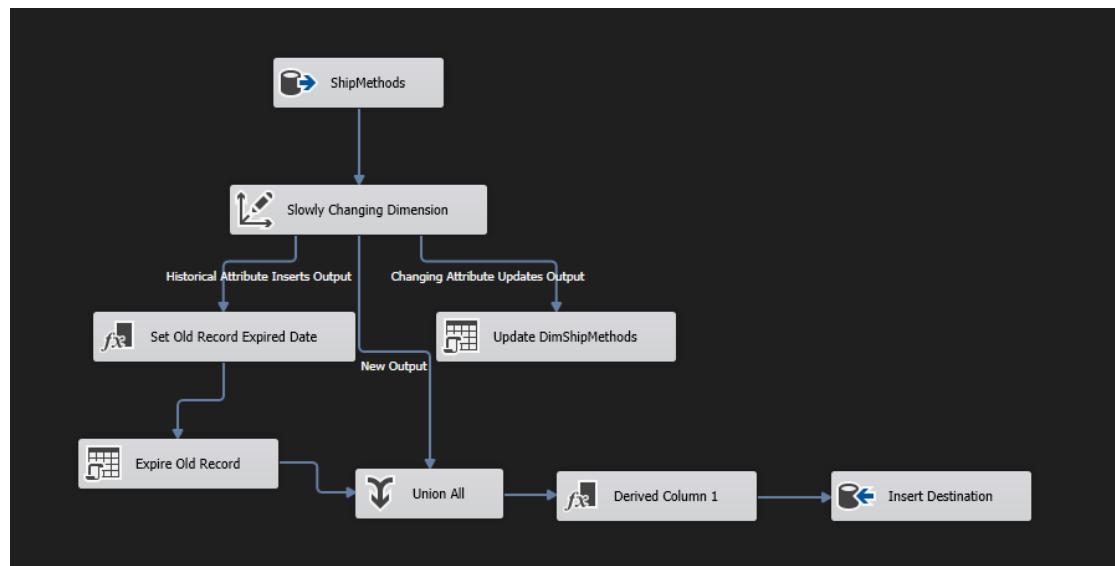


Figura 6.43: Dimensão “ShipMethods”

6.3.12 DimSalesPersons

Na Listagem 6.11 é apresentado o script usado para realizar-se a agregação das tabelas.

```

1  SELECT      SalesPersons.BusinessEntityID, Employees.NationalIDNumber, Employees
   .LoginID, Employees.JobTitle, Employees.BirthDate, Employees.MaritalStatus,
   Employees.Gender, Employees.HireDate, Employees.SalariedFlag,
   Employees.VacationHours, Employees.SickLeaveHours,
   SalesPersons.SalesQuota, SalesPersons.Bonus,
   SalesPersons.CommissionPct, SalesPersons.SalesYTD,
   SalesPersons.SalesLastYear, Persons.PersonType,
   Persons.Title, Persons.FirstName, Persons.MiddleName,
   Persons.LastName, SalesPersons.CreatedDate,
   SalesPersons.ModifiedDate
2
3
4  FROM        Employees INNER JOIN
5          SalesPersons ON Employees.BusinessEntityID = SalesPersons.
   BusinessEntityID INNER JOIN
6          Persons ON Employees.BusinessEntityID = Persons.
   BusinessEntityID

```

Listagem 6.11: Script de agregação de tabelas

Na Figura 6.44 é apresentada a estrutura de carregamento dos dados para a dimensão “SalesPersons”.

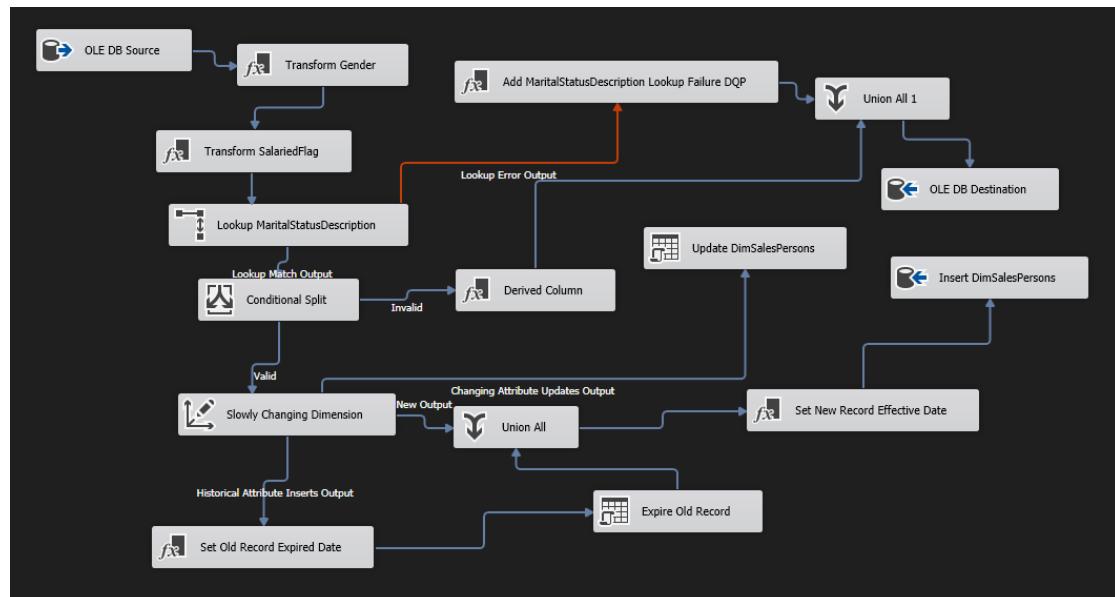


Figura 6.44: Dimensão “SalesPersons”

Na Figura 6.45 é apresentada a transformação realizada ao parâmetro “Gender”. Neste

passo foi realizado o mapeamento de “F” para “Female” e “M” para “Male”.

Derived Column Name	Derived Column	Expression	Data Type	Length
Gender_Transformed	<add as new column>	Gender == "F" ? "Female" : "Male"	Unicode string [DT_WS...]	6

Figura 6.45: Transformação do “Gender”

Na Figura 6.46 é apresentada a transformação realizada ao parâmetro “SalariedFlag”. Neste passo foi realizado o mapeamento de “1” para “Yes” e “0” para “No”.

Derived Column Name	Derived Column	Expression	Data Type	Length
SalariedFlag_Transform...	<add as new column>	SalariedFlag == TRUE ? "Yes" : "No"	Unicode string [DT_WS...]	3

Figura 6.46: Transformação do “SalariedFlag”

Na Figura 6.47 é apresentada a seleção da tabela criada para efeito, onde é realizado o mapeamento das várias alternativas.

(C) Use a table or a view:

[dbo].[MaritalStatusDescription_Lookup]

Figura 6.47: Seleção da tabela para o mapeamento

Na Figura 6.48 é apresentada a ligação entre os “MaritalStatus” e a posterior seleção do “MaritalStatusDescription”.

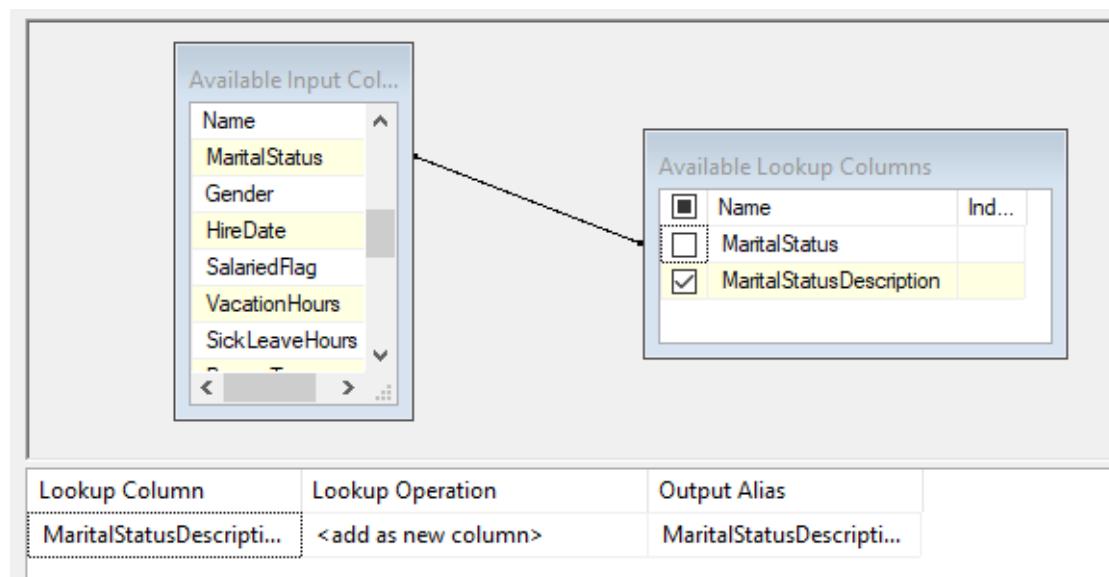


Figura 6.48: Mapeamento do atributo “MaritalStatus”

Na Figura 6.49 é apresentado o processo seguinte realizado caso em alguma linha o processo de mapeamento dê erro. A esta mesma linha é adicionado o parâmetro “DQP” e a seguir, esta será registada na tabela criada para o efeito.

Derived Column Name	Derived Column	Expression	Data Type	Length
DQP	<add as new column>	"Marital status description lookup failed on Marital..."	Unicode string [DT_WS...]	8

Figura 6.49: DQP da Dimensão “SalesPersons”

Na Figura 6.50 é apresentada a validação realizada para o parâmetro “NationalIDNumber”, onde as linhas que têm o parâmetro mencionado com 9 dígitos são considerados um registo válido, por sua vez aqueles que tem menos que 9 dígitos são considerados registo inválidos.

Order	Output Name	Condition
1	Valid	LEN(NationalIDNumber) == 9

Figura 6.50: Validação feita ao atributo “NationalIDNumber”

Na Figura 6.51 é apresentado o processo seguinte realizado caso alguma linha tenha o parâmetro “NationalIDNumber” inválido. A esta mesma linha é adicionado o parâmetro

“DQP” e a seguir, esta será registada na tabela criada para o efeito.

Derived Column Name	Derived Column	Expression	Data Type	Length
DQP	<add as new column>	"Invalid NationalIDNumber"	Unicode string [DT_WS...]	255

Figura 6.51: DQP da Dimensão “SalesPersons”

6.3.13 FactSales

Na Figura 6.52 está representada a sequência da remoção de chaves estrangeiras existentes, carregamento das tabelas de facto e criação de chaves estrangeiras.

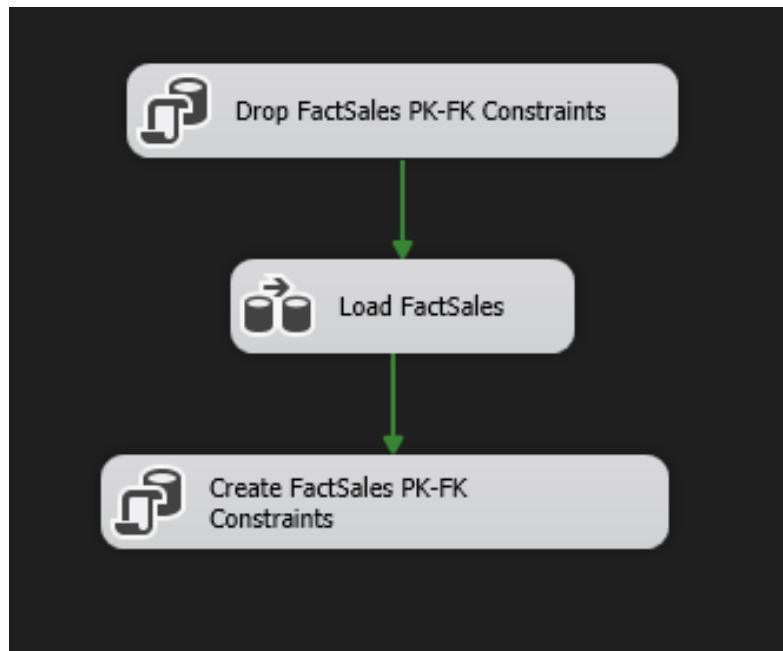


Figura 6.52: Fluxo de remoção e criação de chaves estrangeiras em “FactSales”

Por sua vez, nas Listagens 6.12 e 6.13 são mencionados os scripts usados para eliminação e criação das restrições de chave, respetivamente.

```
1 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimCustomers')
2 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [
    FK_FactSales_DimCustomers]
3 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_OrderDate')
4 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [FK_FactSales_OrderDate]
5 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DueDate')
6 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [FK_FactSales_DueDate]
7 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_ShipDate')
8 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [FK_FactSales_ShipDate]
9 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimProducts')
10 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [FK_FactSales_DimProducts]
    ]
11 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_BillToAddressKey')
12 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [
    FK_FactSales_BillToAddressKey]
13 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_ShipToAddressKey')
14 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [
    FK_FactSales_ShipToAddressKey]
15 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimCurrency')
16 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [FK_FactSales_DimCurrency]
    ]
17 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimProducts')
18 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [FK_FactSales_DimProducts]
    ]
19 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimSalesPersons')
20 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [
    FK_FactSales_DimSalesPersons]
21 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimSalesTerritories')
22 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [
    FK_FactSales_DimSalesTerritories]
23 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactSales_DimShipMethods')
24 ALTER TABLE [dbo].[FactSales] DROP CONSTRAINT [
    FK_FactSales_DimShipMethods]
```

Listagem 6.12: Eliminação das restrições de chaves

```
1 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DimCustomers] FOREIGN KEY([CustomerKey])
2 REFERENCES [dbo].[DimCustomers] ([CustomerKey])
3 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_DimCustomers]
4 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_OrderDate] FOREIGN KEY([OrderDateKey])
5 REFERENCES [dbo].[DimDate] ([DateKey])
6 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [FK_FactSales_OrderDate]
7 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DimProducts] FOREIGN KEY([ProductKey])
8 REFERENCES [dbo].[DimProducts] ([ProductKey])
9 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_DimProducts]
10 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DimShipMethods] FOREIGN KEY(ShipMethodKey)
11 REFERENCES [dbo].[DimShipMethods] ([ShipMethodKey])
12 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_DimShipMethods]
13 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DimSalesPersons] FOREIGN KEY(SalesPersonKey)
14 REFERENCES [dbo].[DimSalesPersons] ([SalesPersonKey])
15 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_DimSalesPersons]
16 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DimSalesTerritories] FOREIGN KEY(TerritoryKey)
17 REFERENCES [dbo].[DimSalesTerritories] ([TerritoryKey])
18 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_DimSalesTerritories]
19 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DimCurrency] FOREIGN KEY(CurrencyKey)
20 REFERENCES [dbo].[DimCurrency] ([CurrencyKey])
21 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_DimCurrency]
22 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_ShipDate] FOREIGN KEY([ShipDateKey])
23 REFERENCES [dbo].[DimDate] ([DateKey])
24 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [FK_FactSales_ShipDate]
25 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_DueDate] FOREIGN KEY([DueDateKey])
26 REFERENCES [dbo].[DimDate] ([DateKey])
27 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [FK_FactSales_DueDate]
28 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_BillToAddressKey] FOREIGN KEY([BillToAddressKey])
29 REFERENCES [dbo].[DimAddresses] ([AddressKey])
30 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_BillToAddressKey]
31 ALTER TABLE [dbo].[FactSales] WITH CHECK ADD CONSTRAINT [
    FK_FactSales_ShipToAddressKey] FOREIGN KEY([ShipToAddressKey])
32 REFERENCES [dbo].[DimAddresses] ([AddressKey])
33 ALTER TABLE [dbo].[FactSales] CHECK CONSTRAINT [
    FK_FactSales_ShipToAddressKey]
```

Listagem 6.13: Criação das restrições de chaves

Com o intuito de criar os dados da tabela de factos, é necessário proceder à seleção das keys das várias dimensões que irão integrar a tabela de factos. A estrutura do fluxo de dados completa para alcançar este objetivo está representada nas Figuras 6.53 e 6.54.

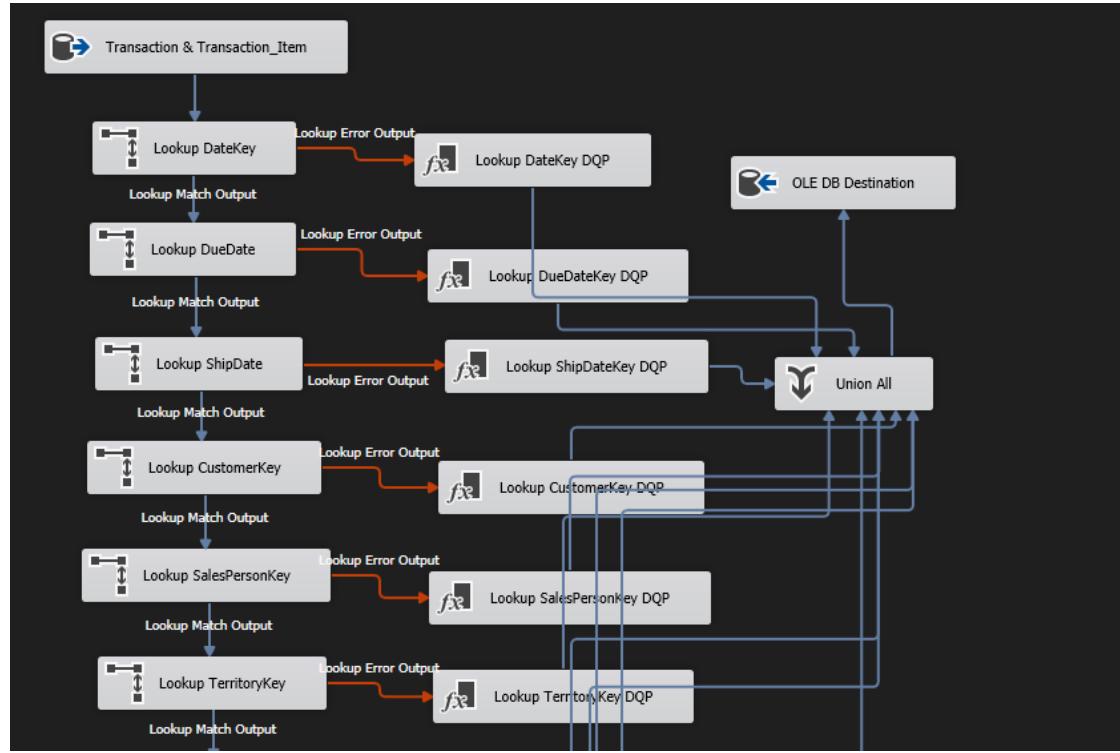


Figura 6.53: Fluxo de dados Parte 1

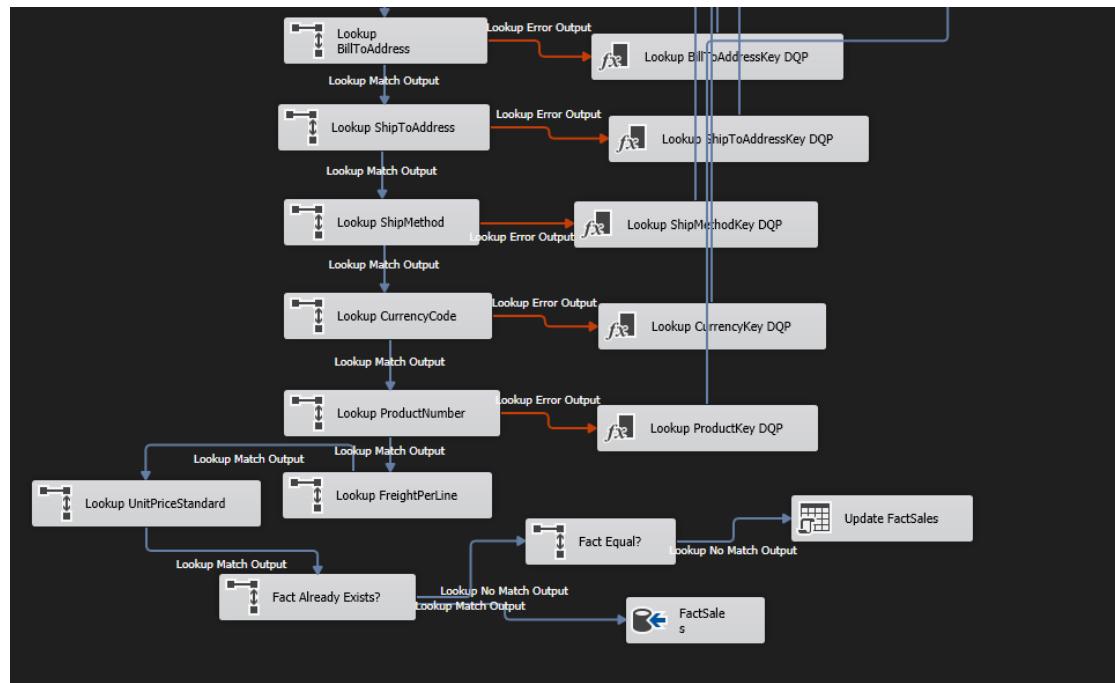


Figura 6.54: Fluxo de dados Parte 2

Primeiramente é necessário recolher as chaves da data para associar à “OrderDate”, “ShipDate” e “DueDate” recorrendo à DimDate. Segue-se uma imagem ilustrativa, Figura 6.55 do “OrderDate” como exemplo que foi seguido para as outras duas colunas nos primeiros 3 lookups.

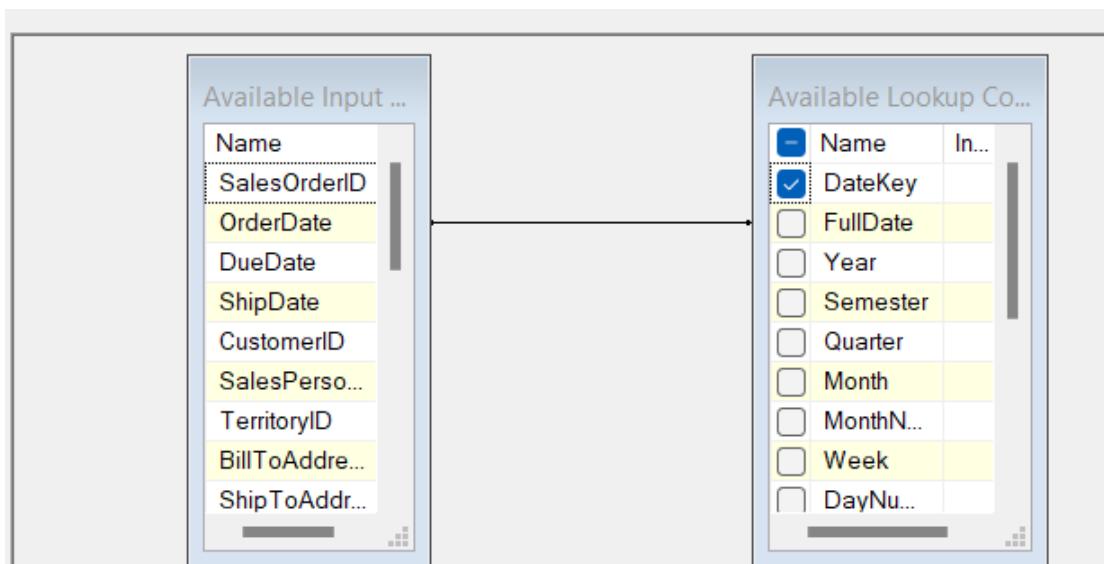


Figura 6.55: OrderDateKey

Seguidamente tem-se o CustomerKey, que se pode colocar na tabela de factos, utilizando o ID da dimensão DimCustomer, pelo que o lookup está presente nas Figuras 6.56 e 6.57.

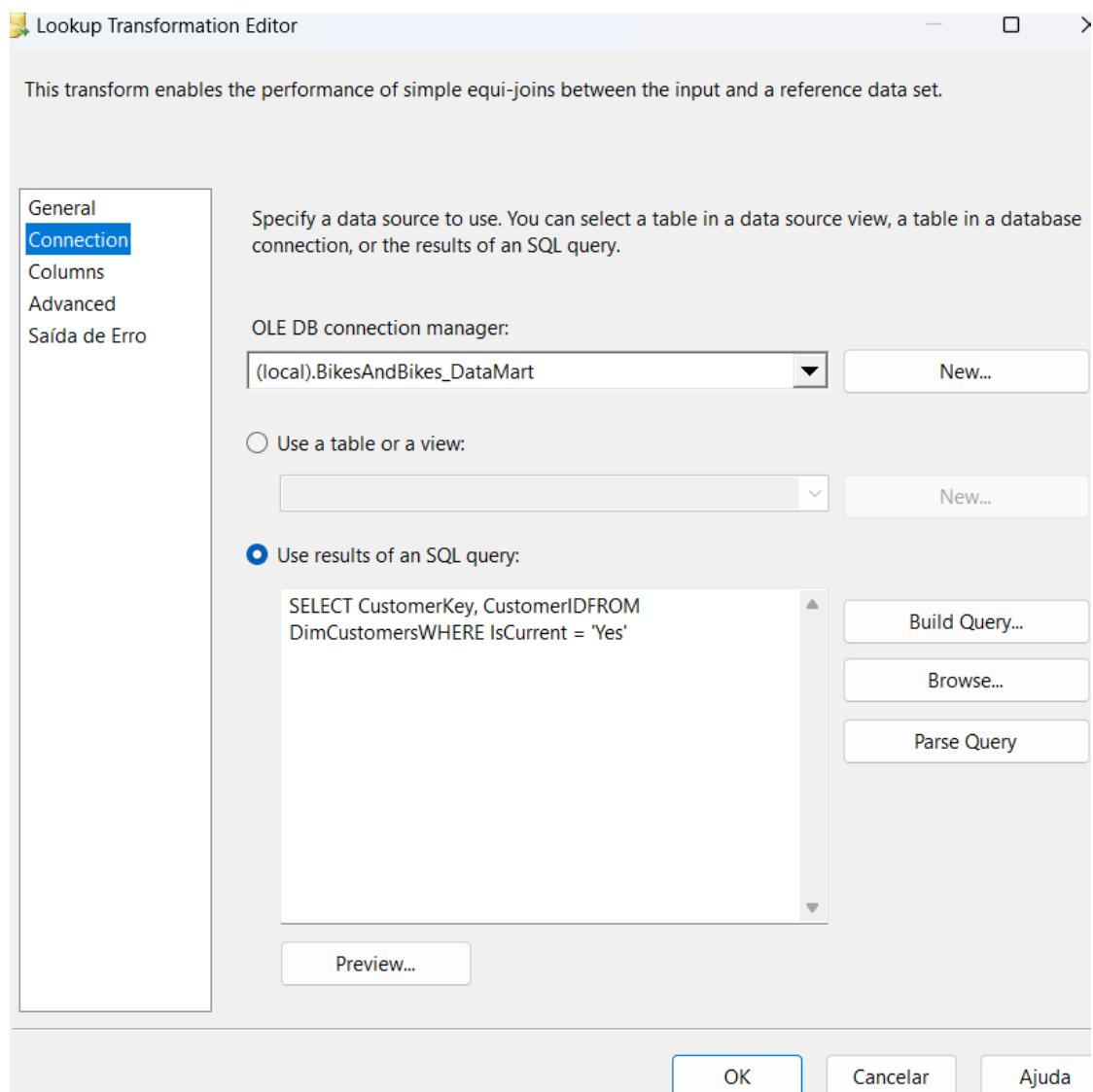


Figura 6.56: Query para selecionar as CustomerKeys

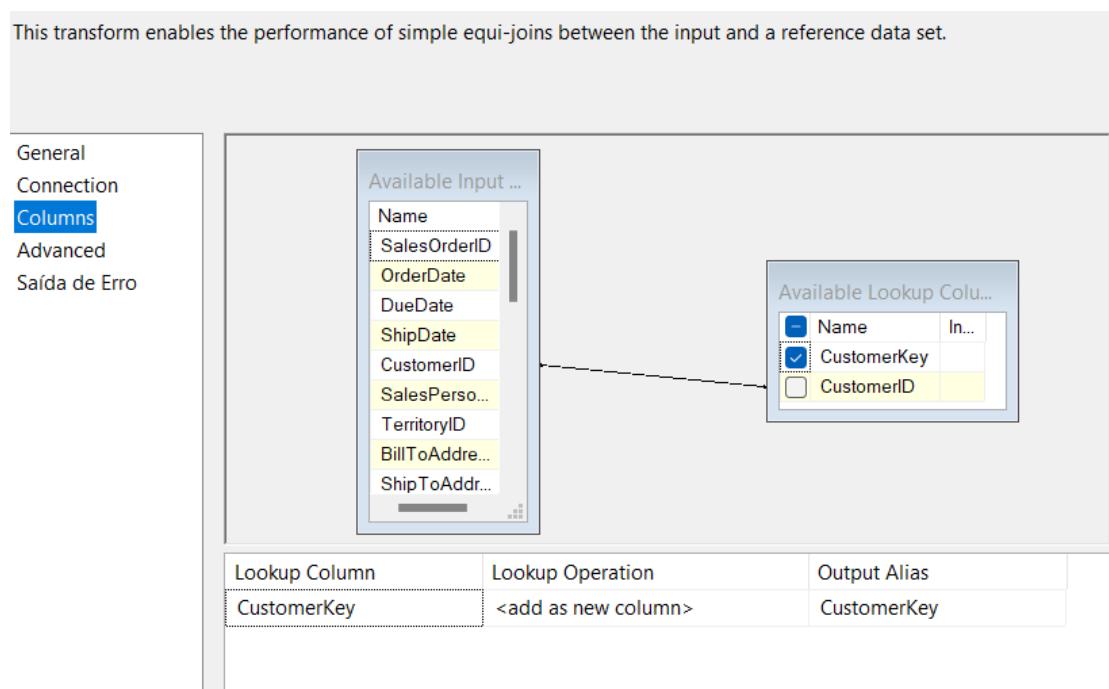


Figura 6.57: Mapeamento do CustomerID que coloca a CustomerKey na tabela de factos

Este modelo foi seguido para as restantes dimensões, como está detalhado na primeira imagem, com o pequeno detalhe que nos endereços de envio e de faturação terá de ser feito o mapeamento duas vezes como foi feito na data, para associar a “AddressKey” a estas duas colunas.

Por fim, para se realizar o populamento da tabela de factos com as unidades corretas, é necessário ter em atenção dois aspectos fulcrais: a conversão dos dados fonte de tabelas em moeda local para moeda uniformizada (neste caso esta moeda será o euro) e a divisão de colunas como o valor de taxa para estar ao nível da linha. Isto quer dizer que se houver 3 registos da mesma compra e o valor total da taxa, por exemplo, ser 300 euros, cada registo dessa compra deverá ter um valor de taxa de 100 euros (300 euros totais / 3 registos). No contexto do problema, este caso estende-se às colunas de “Freight” e de “TotalDue”. Nestas 3 colunas, estas duas operações foram feitas em simultâneo no lookup que possui a query apresenta na Listagem 6.14 que gera os valores desejados.

```
1  SELECT
2    soh.SalesOrderID,
3    sod.SalesOrderDetailID,
4    sod.OrderQty,
5    sod.ProductNumber,
6    soh.OrderDate,
7    soh.Freight / COUNT(*) OVER (PARTITION BY sod.SalesOrderID) AS
8      FreightLineLocal,
9    soh.TaxAmount / COUNT(*) OVER (PARTITION BY sod.SalesOrderID) AS
10      TaxAmountLineLocal,
11    soh.TotalDue / COUNT(*) OVER (PARTITION BY sod.SalesOrderID) AS
12      TotalDueLineLocal,
13    soh.Freight / COUNT(*) OVER (PARTITION BY sod.SalesOrderID) * c.
14      AverageRate AS FreightLineStandard,
15    soh.TaxAmount / COUNT(*) OVER (PARTITION BY sod.SalesOrderID) * c.
16      AverageRate AS TaxAmountLineStandard,
17    soh.TotalDue / COUNT(*) OVER (PARTITION BY sod.SalesOrderID) * c.
18      AverageRate AS TotalDueLineStandard
19  FROM
20    SalesOrderHeaders soh
21  JOIN
22    SalesOrderDetails sod
23  ON
24    soh.SalesOrderID = sod.SalesOrderID
25  JOIN
26    CurrencyRates c
27  ON
28    soh.CurrencyCode = c.FromCurrencyCode
29  AND soh.OrderDate = c.CurrencyRateDate
```

Listagem 6.14: Representação monetária ao nível da linha

O mapeamento resultante foi o apresentado na Figura 6.58.

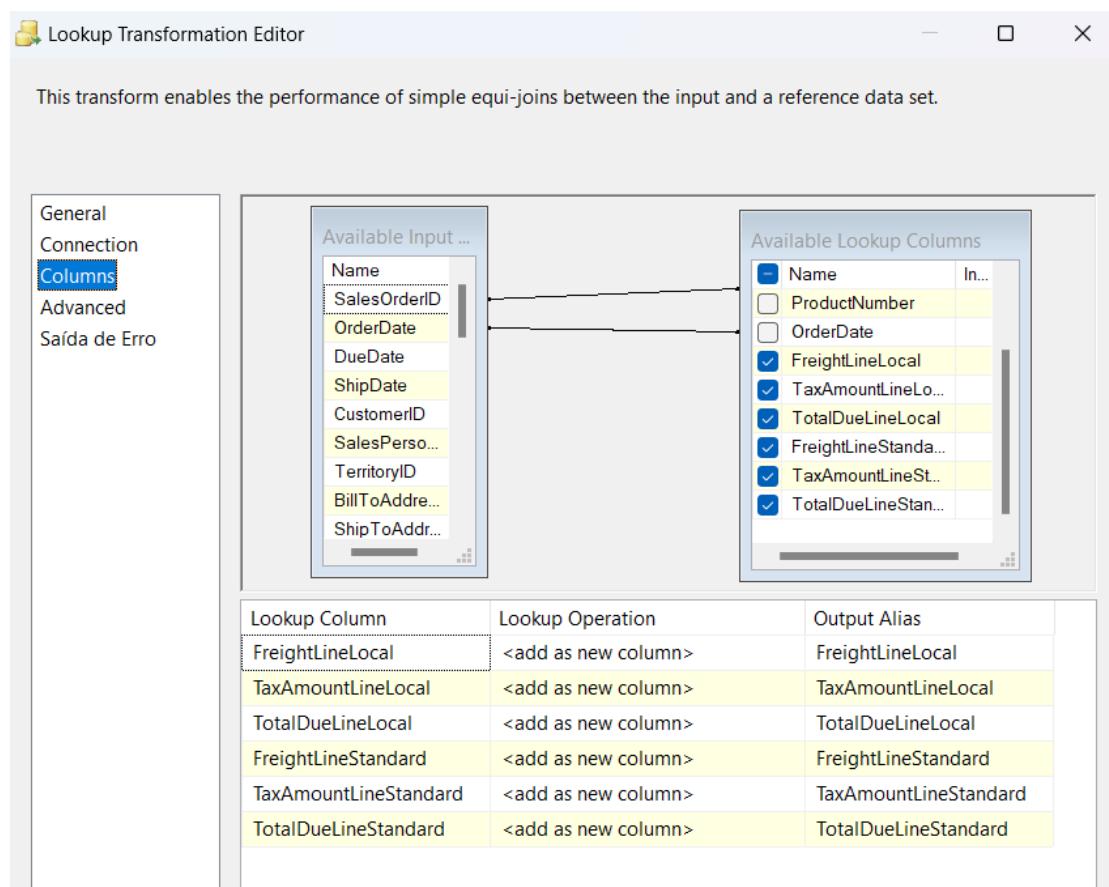


Figura 6.58: Mapeamento da conversão das colunas uniformizadas e com a granularidade ao nível da linha

Deste modo, já tendo a uniformização da moeda e a granularidade ao nível da linha nestas 3 colunas, o lookup final será com os restantes valores monetários que já se encontram ao nível da linha, “UnitPrice”, “UnitPriceDiscount”, “LineTotal”, “SubTotal”. Esta conversão foi feita como no lookup anterior, com a query apresentada na Listagem ??.

```
1  SELECT
2    f.OrderDate,
3    f.CurrencyCode,
4    f.SalesOrderID,
5    sod.UnitPrice * c.AverageRate AS UnitPriceStandard,
6    sod.UnitPrice * sod.UnitPriceDiscount *c.AverageRate As
7      UnitPriceDiscountStandard,
8    sod.LineTotal *c.AverageRate As LineTotalStandard,
9    f.SubTotal *c.AverageRate AS SubTotalStandard,
10   sod.UnitPrice * sod.UnitPriceDiscount As UnitPriceDiscountLocal
11  FROM
12    SalesOrderHeaders f
13  JOIN
14    CurrencyRates c
15  ON
16    f.CurrencyCode = c.FromCurrencyCode
17    AND f.OrderDate = c.CurrencyRateDate
18  JOIN
19    SalesOrderDetails sod
20  ON
21    f.SalesOrderID = sod.SalesOrderID
```

Listagem 6.15: Representação com uniformização

Note-se que existe também uma conversão dos valores de desconto outrora indicados em percentagem, nos quais foi feita a conta total para determinar o preço a pagar, olhando à percentagem do desconto que foi usado. A conversão do desconto para moeda uniformizada foi também executada, usando a mesma conta, mas com o rácio de troca da moeda. O mapeamento realizado é apresentado na Figura 6.59.

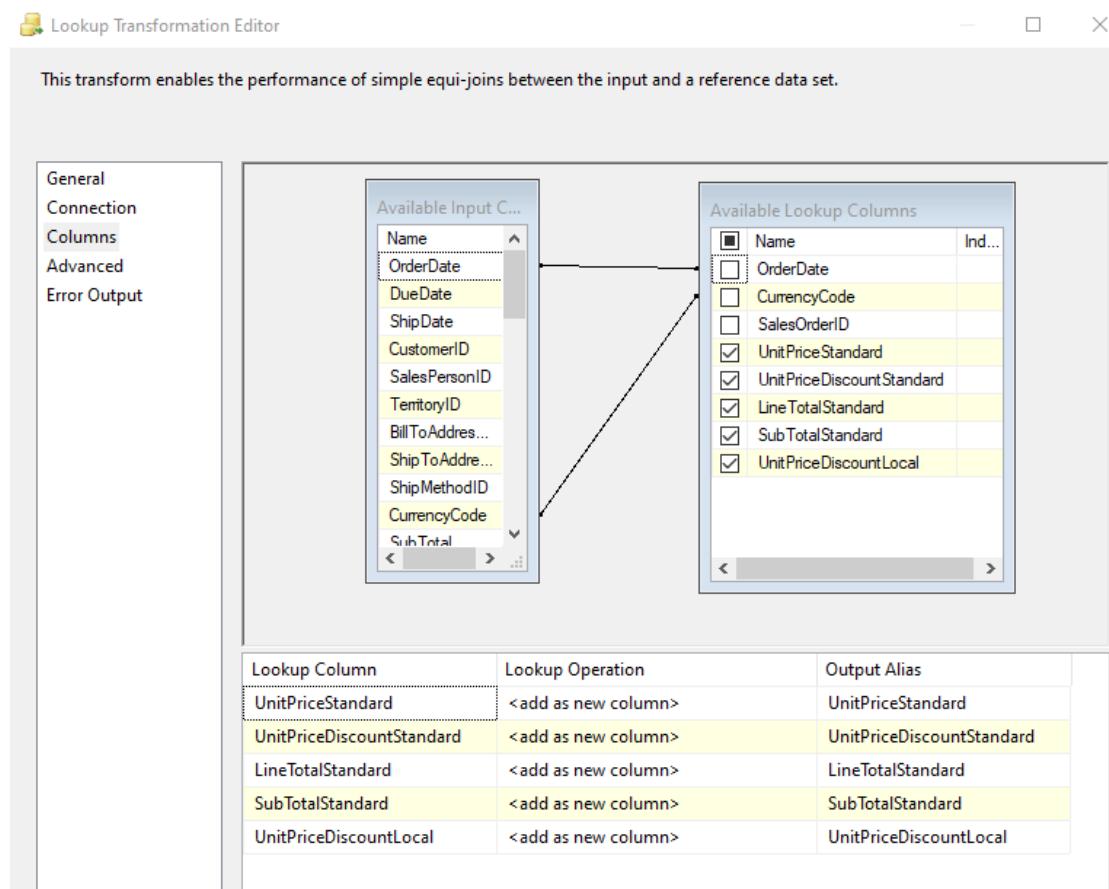


Figura 6.59: Mapeamento da uniformização de colunas já ao nível da linha

Após as várias transformações é apresentado o mapeamento final da tabela de factos que pode ser visualizado na Figura 6.60.

Figura 6.60: Mapeamento final da tabela de factos

6.3.14 FactCurrencyRates

Na tabela de factos de conversão da moeda, foi seguida a mesma sequência de remoção de chaves estrangeiras existentes, carregamento desta tabela de factos e criação destas chaves estrangeiras de novo. Assim, o fluxo da execução da tarefa de carregamento foi a apresentada na Figura 6.61.

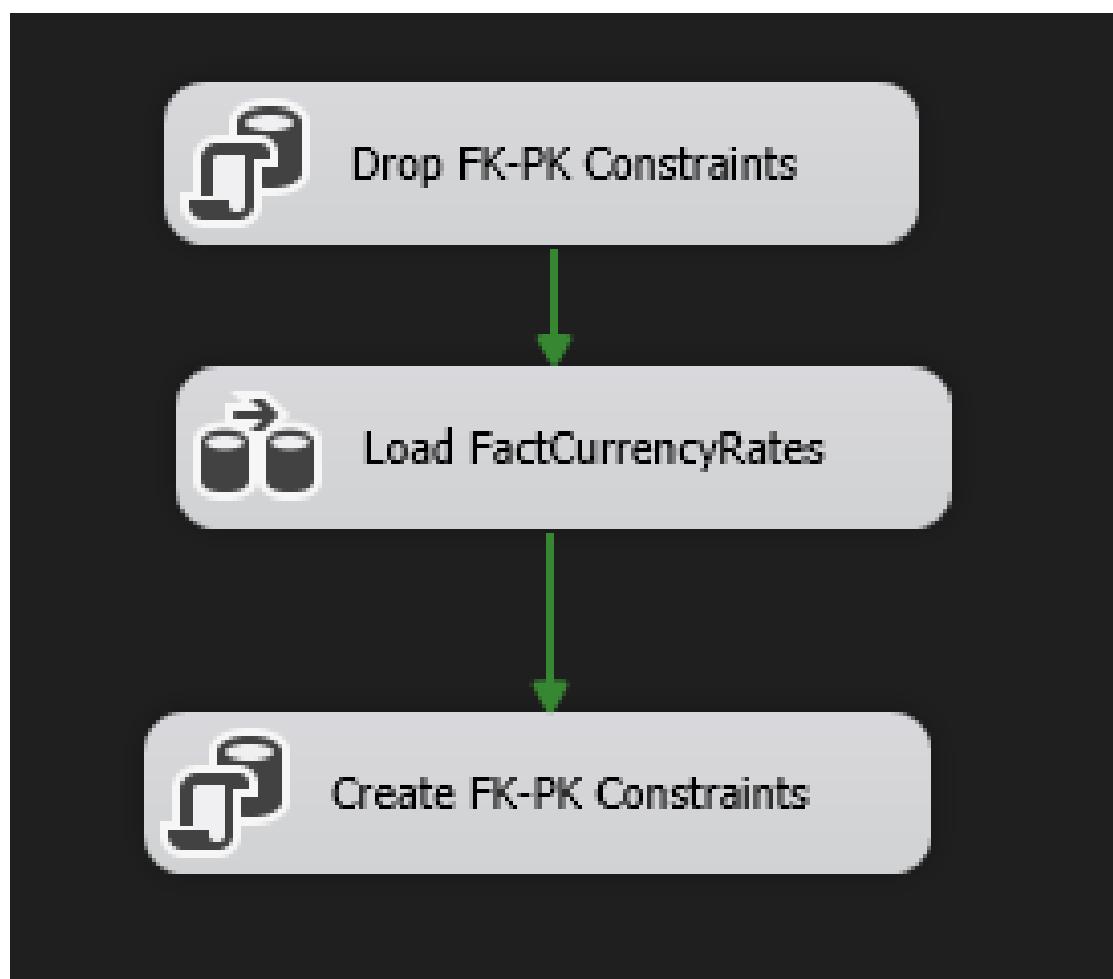


Figura 6.61: Fluxo de dados da FactCurrencyRates

Os scripts utilizados para este fim foram os apresentados nas Listagens 6.16 e 6.17.

```
1 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactCurrencyRates_DimDate')
2 ALTER TABLE [dbo].[FactCurrencyRates] DROP CONSTRAINT [
    FK_FactCurrencyRates_DimDate]
3 IF EXISTS (SELECT name FROM sys.foreign_keys WHERE name = 'FK_FactCurrencyRates_DimCurrency')
4 ALTER TABLE [dbo].[FactCurrencyRates] DROP CONSTRAINT [
    FK_FactCurrencyRates_DimCurrency]
```

Listagem 6.16: Eliminação das restrições de chaves

```

1 ALTER TABLE [dbo].[FactCurrencyRates] WITH CHECK ADD CONSTRAINT [
2   FK_FactCurrencyRates_DimDate] FOREIGN KEY([DateKey])
3 REFERENCES [dbo].[DimDate] ([DateKey])
4 ALTER TABLE [dbo].[FactCurrencyRates] CHECK CONSTRAINT [
5   FK_FactCurrencyRates_DimDate]
6 ALTER TABLE [dbo].[FactCurrencyRates] WITH CHECK ADD CONSTRAINT [
7   FK_FactCurrencyRates_DimCurrency] FOREIGN KEY([FromCurrencyKey])
8 REFERENCES [dbo].[DimCurrency] ([CurrencyKey])
9 ALTER TABLE [dbo].[FactCurrencyRates] CHECK CONSTRAINT [
10  FK_FactCurrencyRates_DimCurrency]

```

Listagem 6.17: Criação das restrições de chaves

Quanto à tabela de factos da conversão de moeda, o fluxo está representado na Figura 6.62.

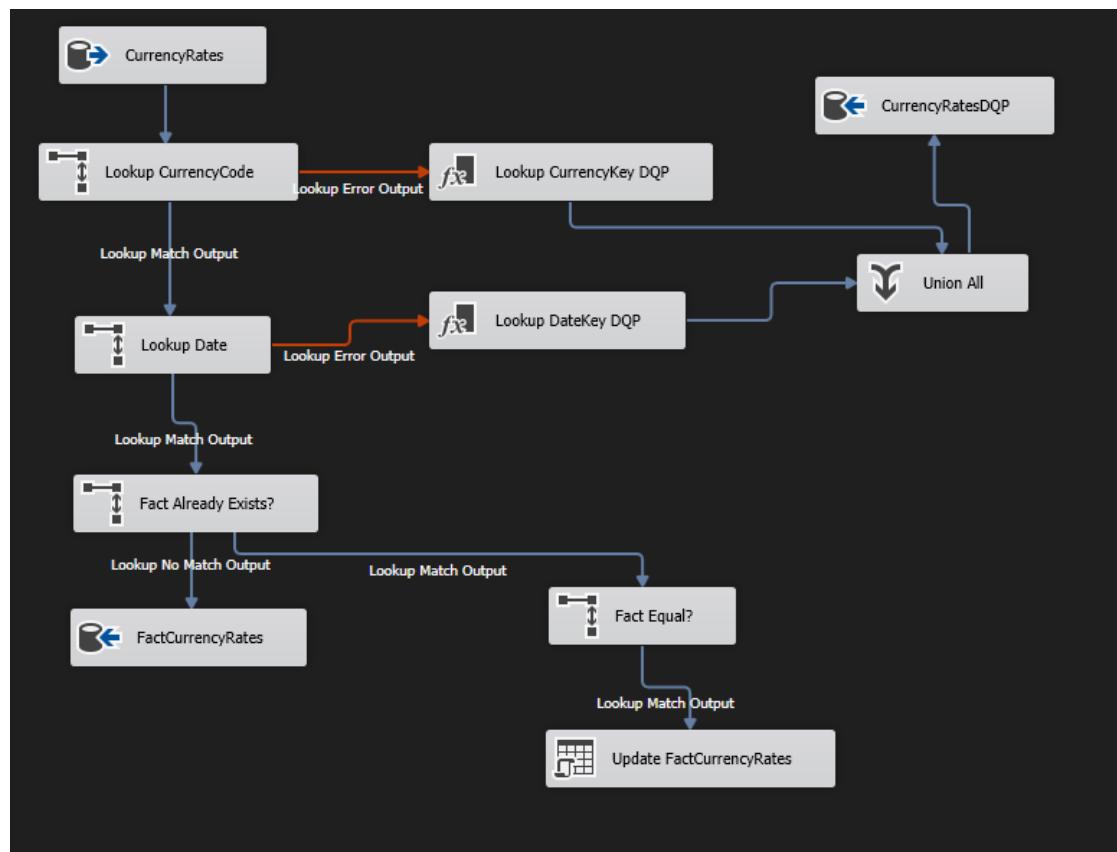


Figura 6.62: Fluxo de dados da FactCurrencyRates

Para obter os dados de cada coluna, pode utilizar-se a query apresentada na Listagem 6.18.

```
1 SELECT
2     CurrencyRates.CurrencyRateID,
3     CurrencyRates.CurrencyRateDate,
4     CurrencyRates.AverageRate,
5     CurrencyRates.EndOfDayRate,
6     FromCurrency.Name AS FromCurrencyName,
7     ToCurrency.Name AS ToCurrencyName
8 FROM
9     CurrencyRates
10 LEFT JOIN Currencies AS FromCurrency ON FromCurrency.CurrencyCode =
11     CurrencyRates.FromCurrencyCode
11 LEFT JOIN Currencies AS ToCurrency ON ToCurrency.CurrencyCode =
12     CurrencyRates.ToCurrencyCode;
```

Listagem 6.18: Query para obter as conversões de cada moeda para EUR

E o mapeamento realizado é apresentado na Figura 6.63.

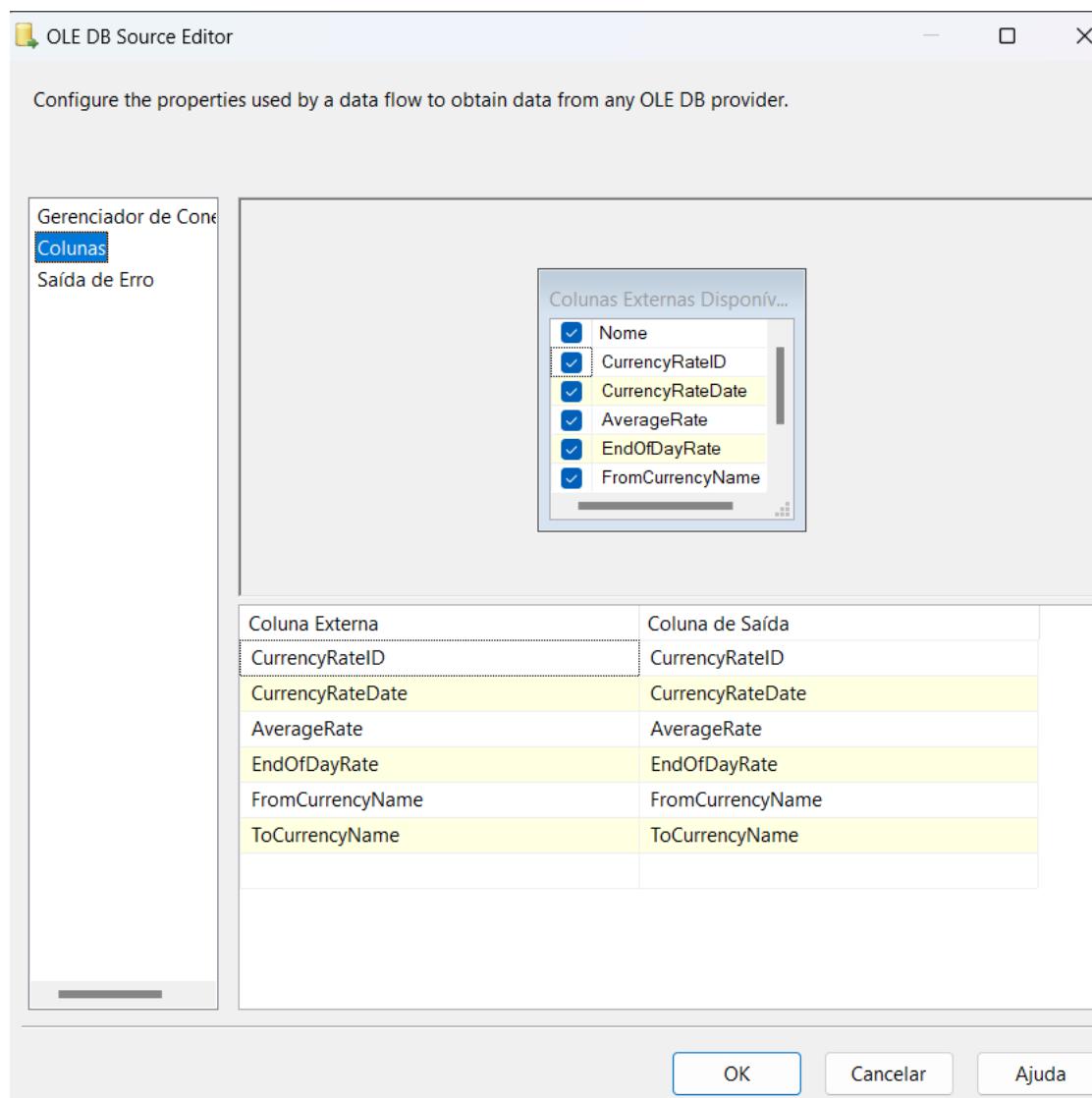


Figura 6.63: Mapeamento Inicial FactCurrencyRate

Como é possível verificar na primeira imagem, existem dois lookups, que servem para atribuir as chaves da data e da moeda. O primeiro lookup, da moeda, pode ser feito da seguinte forma, como no exemplo do Customer no capítulo anterior e apresentado nas Figuras 6.64 e 6.65.

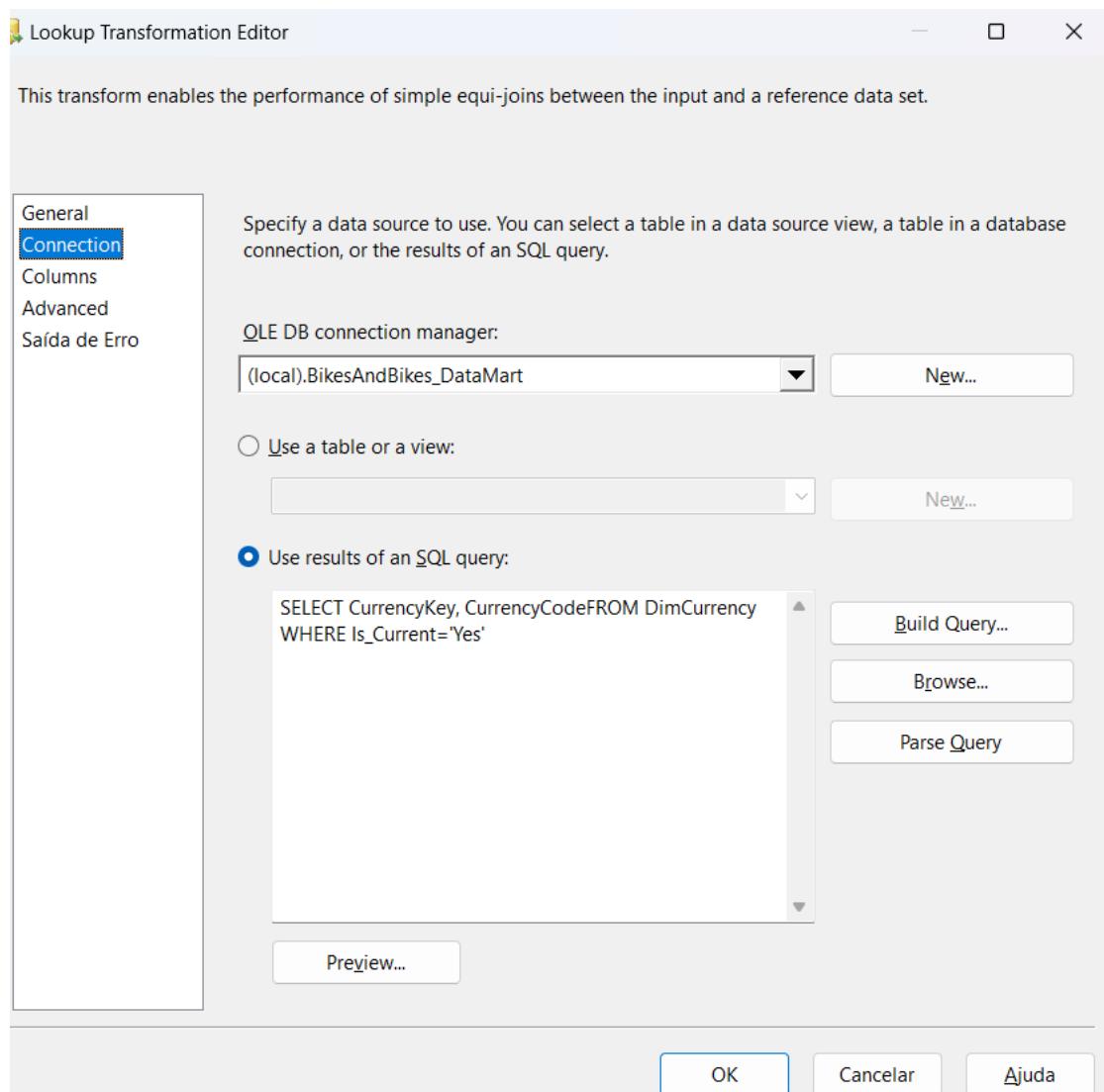


Figura 6.64: Query para selecionar a key consoante o código da moeda

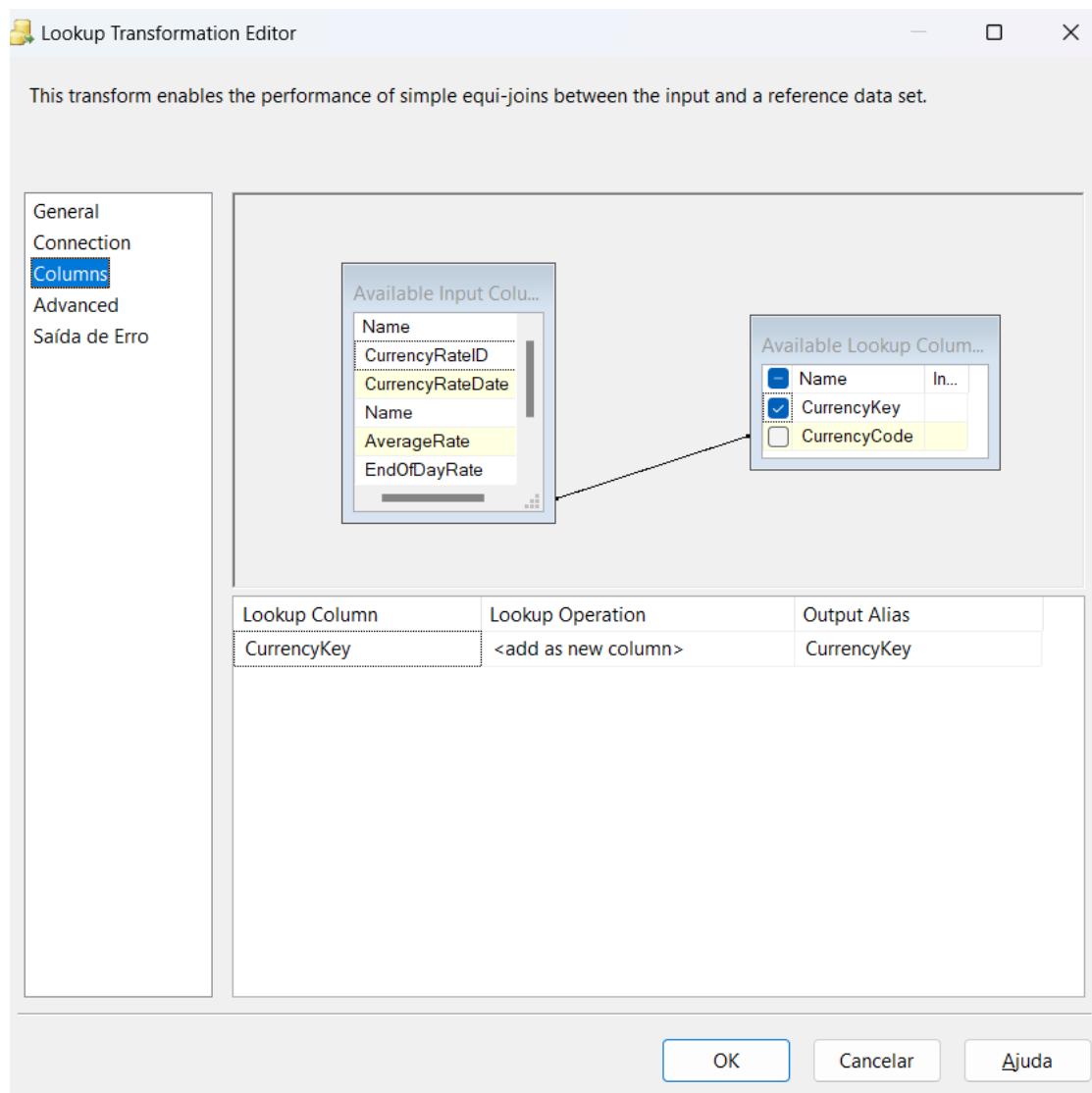


Figura 6.65: Mapeamento da CurrencyKey

Já a seleção da DateKey foi feito como o exemplo da data no capítulo anterior, como é possível verificar na Figura 6.66.

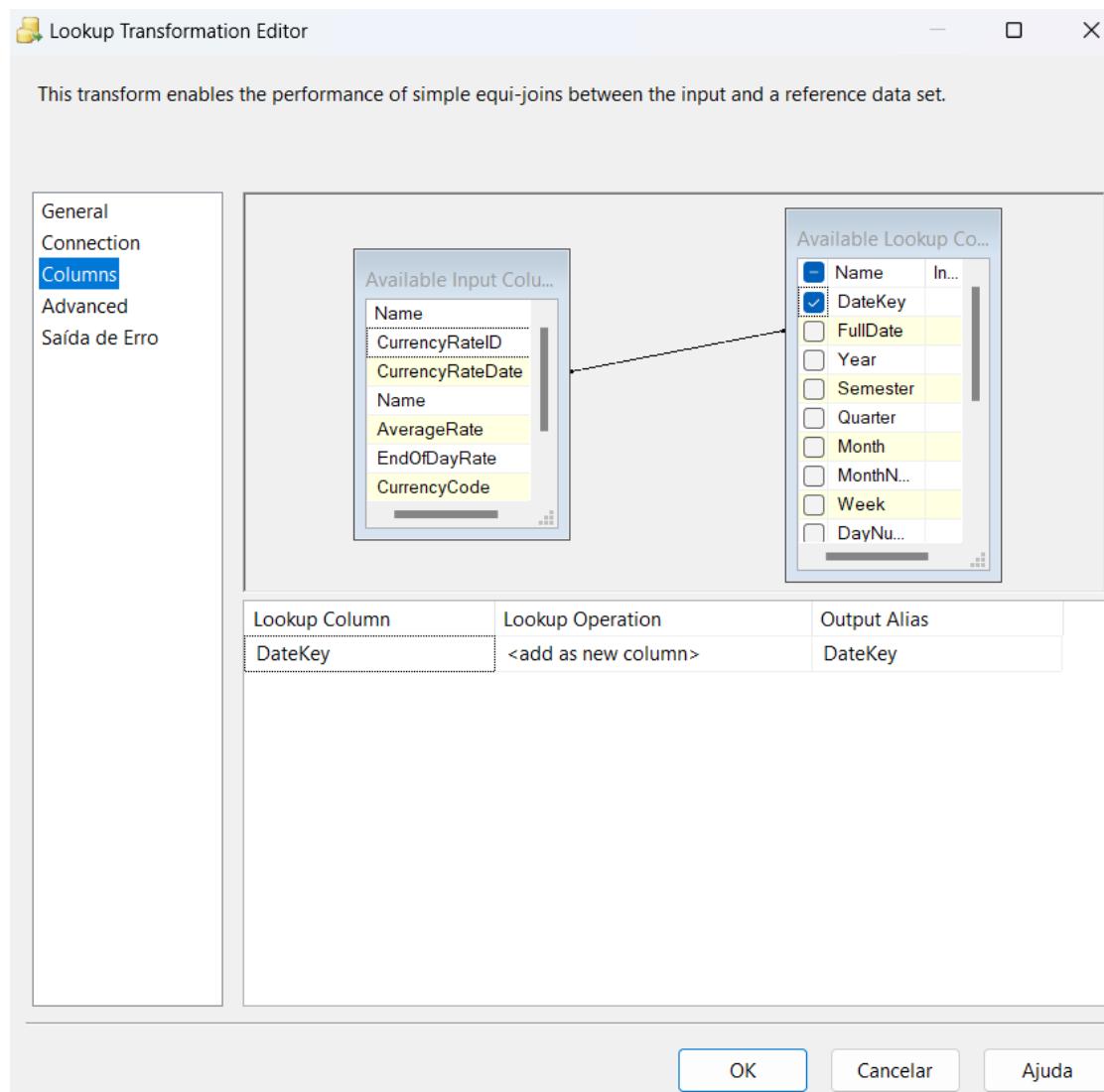


Figura 6.66: Mapeamento da data na tabela FactCurrencyRates

Desta forma, o mapeamento final é dado pela Figura 6.67.

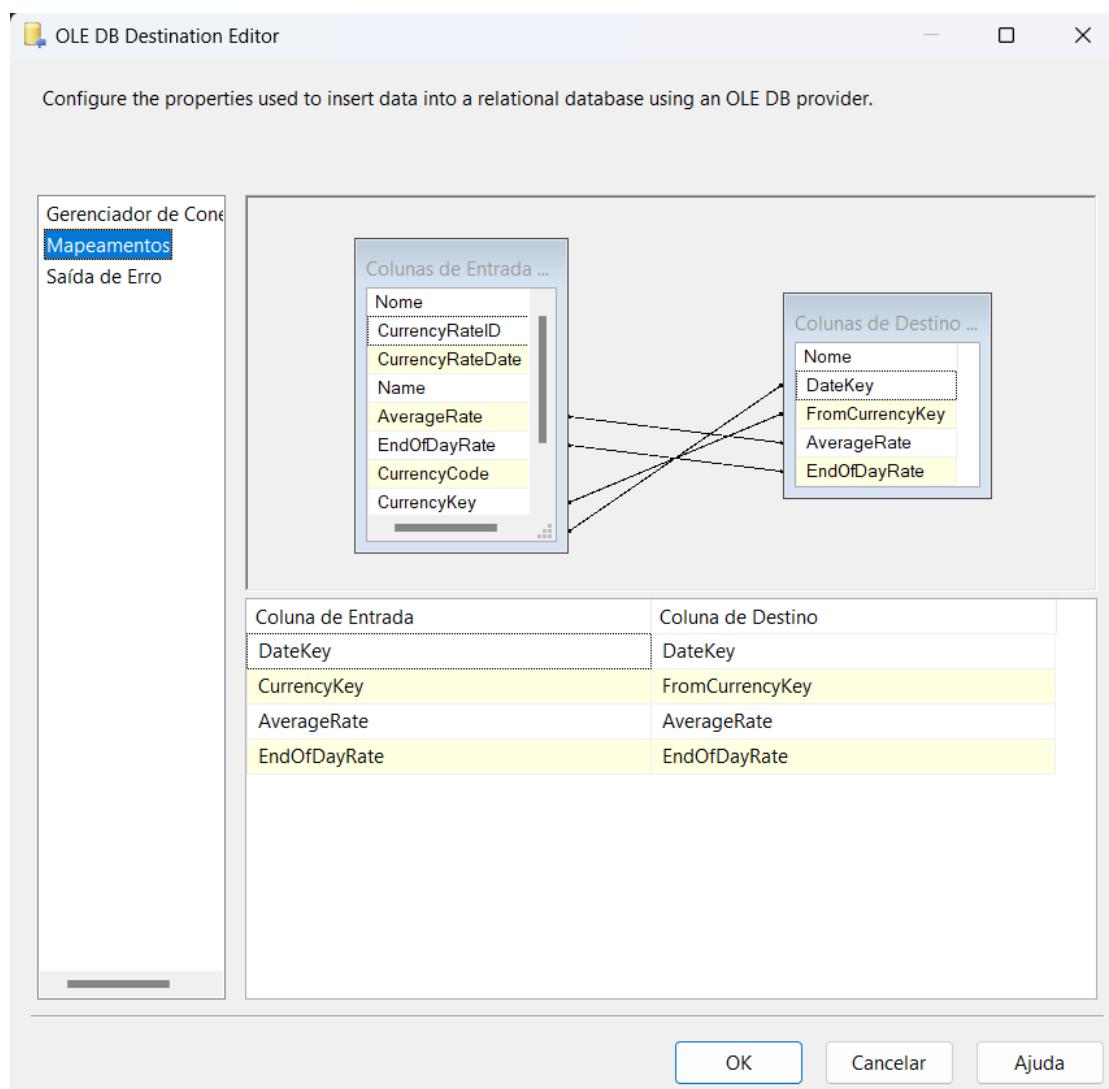
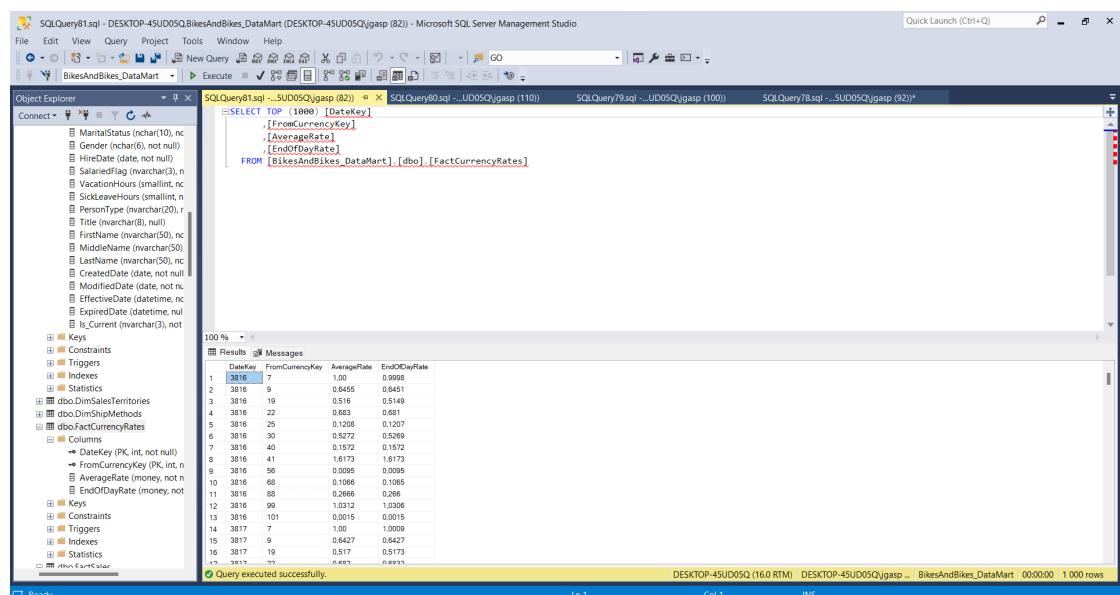


Figura 6.67: Mapeamento final da tabela FactCurrencyRates

Seguindo este processo, a tabela de factos da conversão da moeda pode ser populada executando esse fluxo de dados. Como é possível observar, esta tabela da base de dados está agora populada. Note-se que se removeu a moeda de destino por ser sempre o euro e, pela redundância associada a essa conversão, foi decidido remover essa coluna. Desta forma, a tabela de factos está visível na Figura 6.68.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, there is a tree view of database objects for the 'BikesAndBikes_DataMart' database, including tables like 'DimSalesTerritories', 'DimShipmentMethods', and 'FactCurrencyRates'. The 'FactCurrencyRates' table is selected.

In the main results pane, a T-SQL query is displayed:

```
SELECT TOP (1000) [DateKey]
      ,[FromCurrencyKey]
      ,[AverageRate]
      ,[EndOfDayRate]
   FROM [BikesAndBikes_DataMart].[dbo].[FactCurrencyRates]
```

The results grid shows 10 rows of data:

DateKey	FromCurrencyKey	AverageRate	EndOfDayRate
3816	7	1.00	0.9441
3816	9	0.9455	0.9441
3816	19	0.516	0.5149
3816	22	0.683	0.6831
3816	25	0.1208	0.1207
3816	30	0.5272	0.5269
3816	40	0.5152	0.5152
3816	41	1.6178	1.6173
3816	56	0.0095	0.0095
3816	68	0.1066	0.1065
3816	88	0.2666	0.266
3816	99	1.0312	1.0306
3816	101	0.0015	0.0015
3817	7	1.0009	1.0009
3817	9	0.6427	0.6427
3817	19	0.517	0.5173
3817	59	0.655	0.655

At the bottom of the results pane, it says "Query executed successfully." and "1 000 rows".

Figura 6.68: Tabelas de facto do rácio das moedas populadas

Capítulo 7

Construção do cubo

Para a construção do cubo foram usadas as dimensões todas assim como as duas tabelas de facto, onde foram selecionados todos os atributos de cada tabela, para não haver restrições nas análises posteriores que se pretende fazer. Assim a criação do cubo foi implementada conforme apresentada na Figura 7.1.

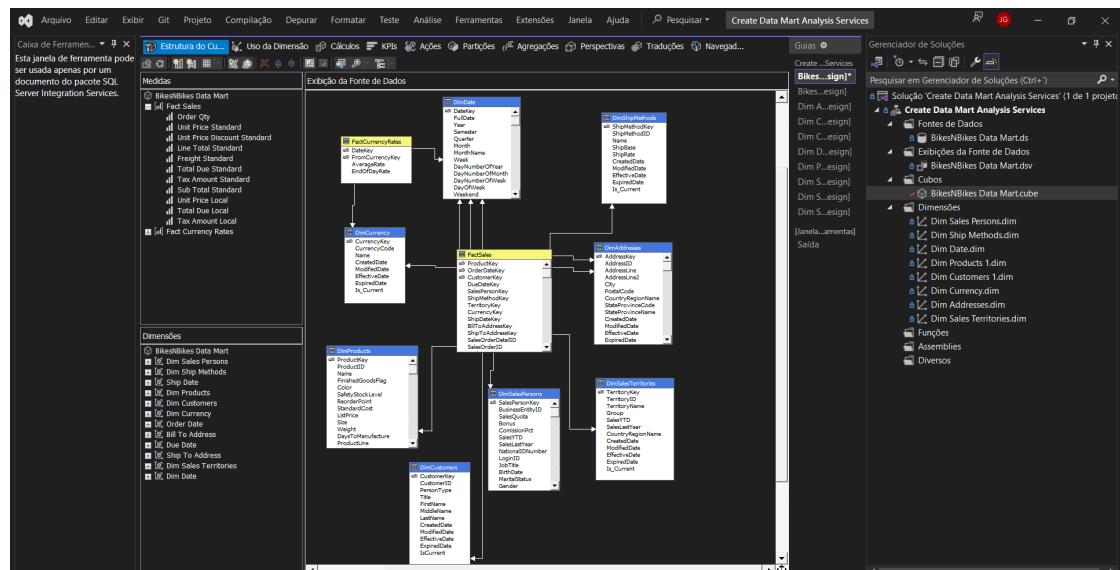


Figura 7.1: Implementação do Cubo

Capítulo 8

Análises

Nestes capítulos são descritas as análises realizadas e apresentados os valores resultantes dessas análises.

8.1 Análise 1

Na análise 1 são apresentados os valores totais (incluindo frete e imposto) em dólares australianos (AUD) referentes às vendas efetuadas nesta moeda, no primeiro semestre de 2012, detalhados por vendedor e por categoria do produto. Na Figura 8.1 é apresentada a seguinte análise.

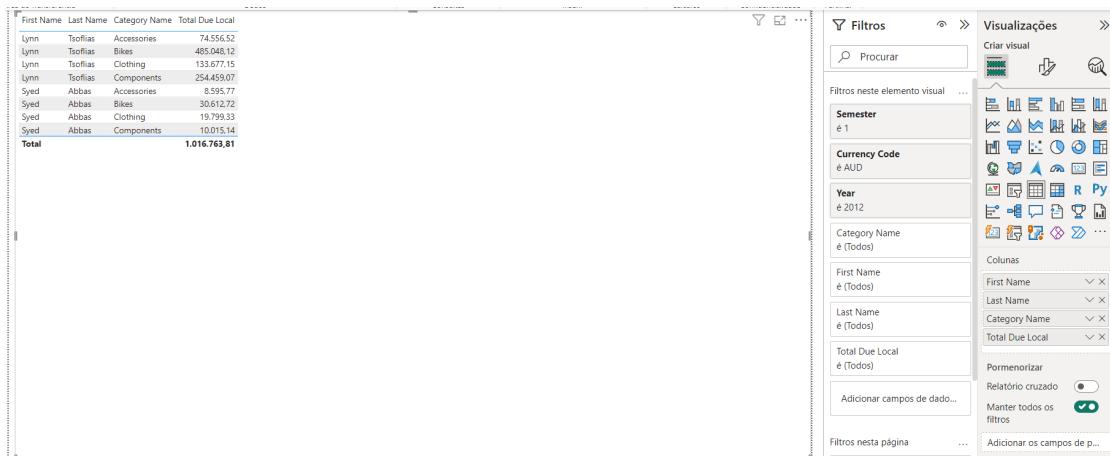


Figura 8.1: Análise 1

8.2 Análise 2

Na análise 2 são apresentados os valores totais dos fretes suportados no transporte dos produtos vendidos durante o mês de dezembro de 2012, detalhados por método de envio e por subcategoria de produto. Na Figura 8.2 é apresentada a seguinte análise.

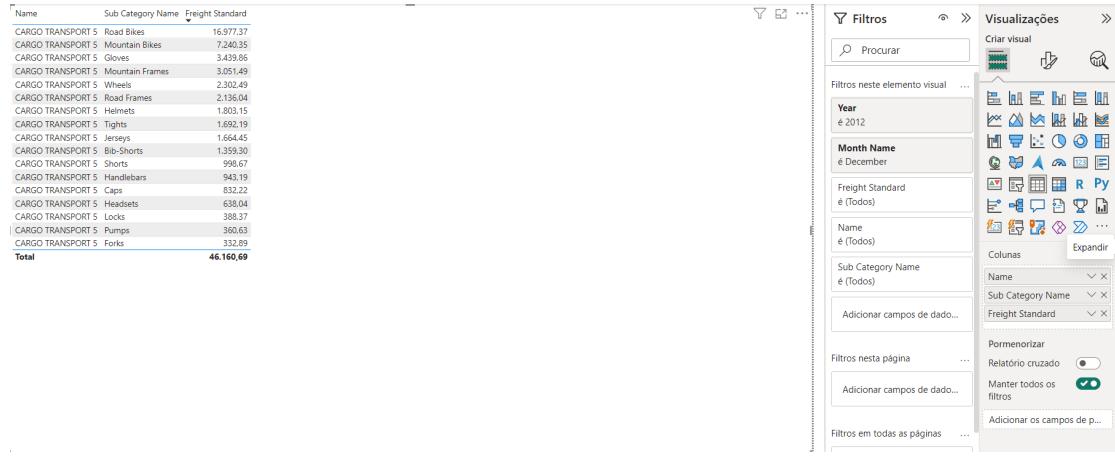


Figura 8.2: Análise 2

8.3 Análise 3

Na análise 3 são apresentados os valores totais dos impostos referentes às vendas realizadas durante o ano de 2014, com possibilidade de análise detalhada (i.e., drill down) ao nível do semestre, trimestre e mês, detalhados por tipo de cliente (atributo PersonType). Na Figura 8.3 é apresentada a seguinte análise.

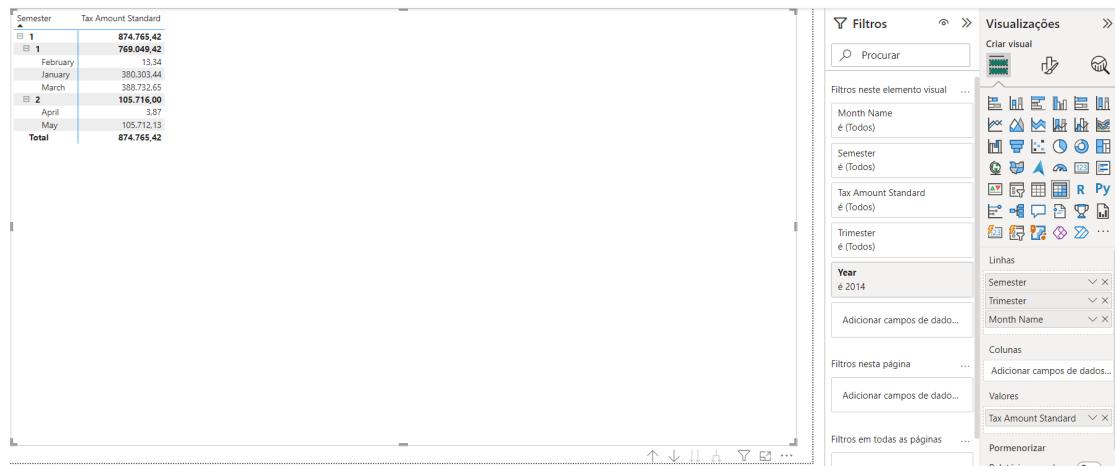


Figura 8.3: Análise 3

8.4 Análise 4

Na análise 4 são apresentados os valores totais (incluindo frete e impostos) e respectivas quantidades das vendas efetuadas a clientes, por cada mês do ano de 2013, com possibilidade de análise agregada (i.e., roll up) ao nível do trimestre, semestre ou ano. Na Figura 8.4 é apresentada a seguinte análise.

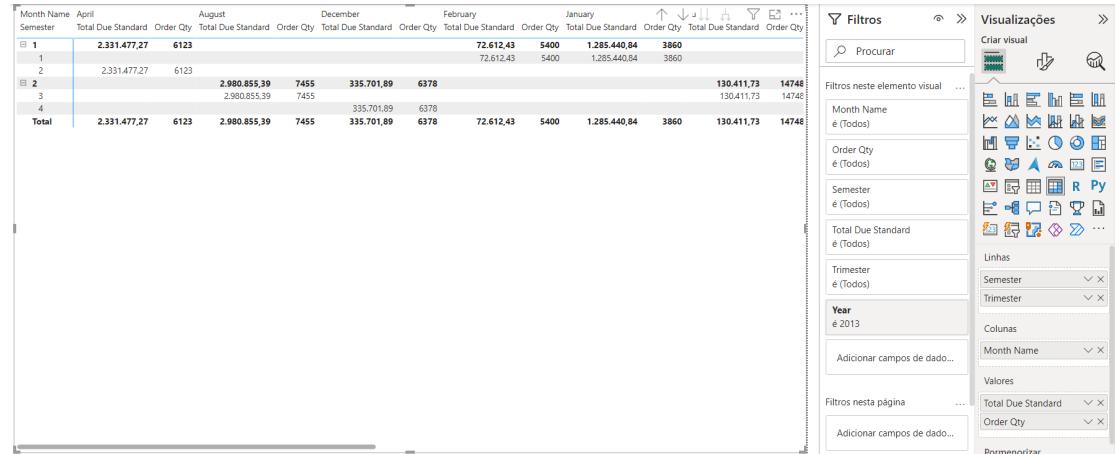


Figura 8.4: Análise 4

8.5 Análise 5

Na análise 5 são apresentados os valores totais das vendas a clientes (sem incluir frete e imposto) efetuadas no primeiro trimestre de 2013, detalhados por vendedor e pela categoria do produto, com possibilidade de análise detalhada (i.e., drill down) ao nível da subcategoria e do produto. Na Figura 8.5 é apresentada a seguinte análise.

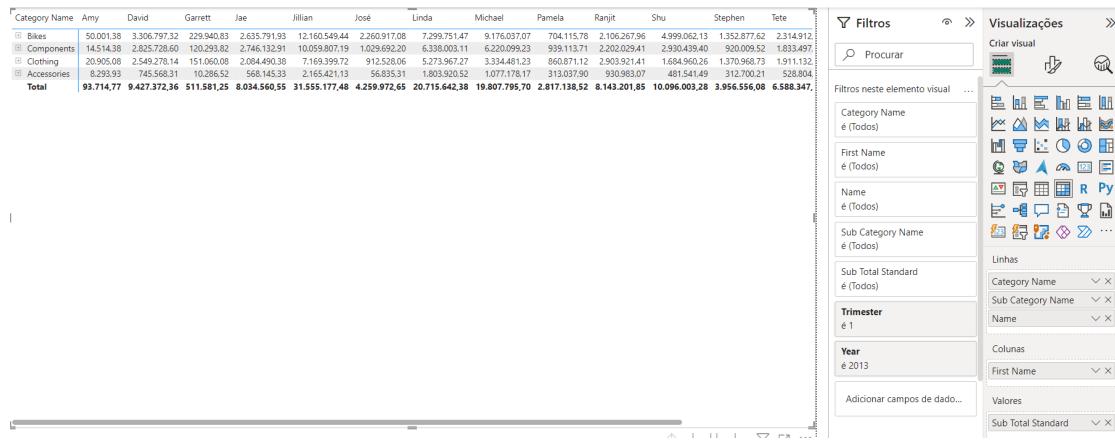


Figura 8.5: Análise 5

8.6 Análise 6

Na análise 6 são apresentados os valores totais dos impostos em libras (GBP) referentes às vendas realizadas durante a primavera e verão de 2013 efetuadas nesta moeda, detalhados pela subcategoria de produto, com possibilidade de análise agregada (i.e., roll up) ao nível da categoria. Na Figura 8.6 é apresentada a seguinte análise.

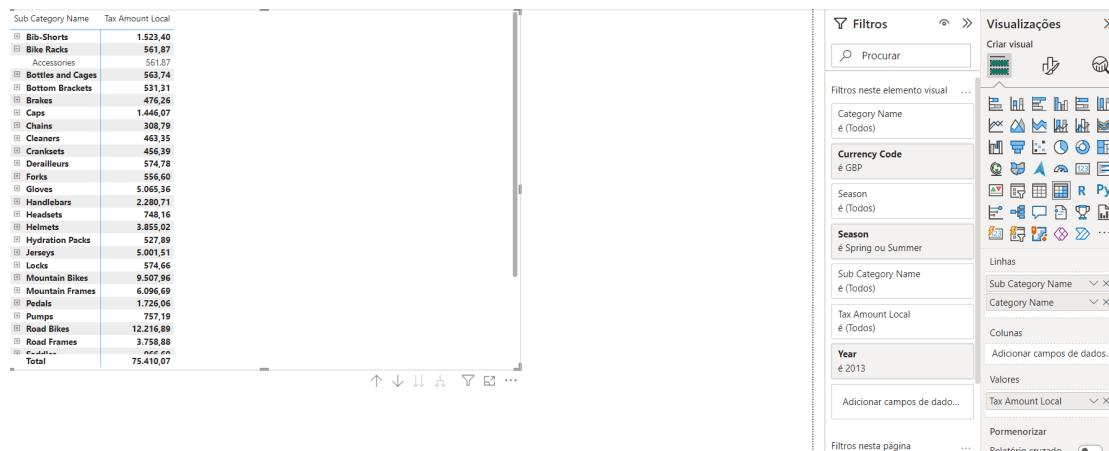


Figura 8.6: Análise 6

8.7 Análise 7

Na análise 7 são apresentados os valores totais referentes aos descontos praticados sobre o preço unitário de venda durante o ano de 2013, com possibilidade de análise detalhada (i.e., drill down) ao nível do semestre, trimestre e mês, detalhados pelos territórios de venda, com possibilidade de análise agregada (i.e., roll up) ao nível do país/região. Na Figura 8.7 é apresentada a seguinte análise.

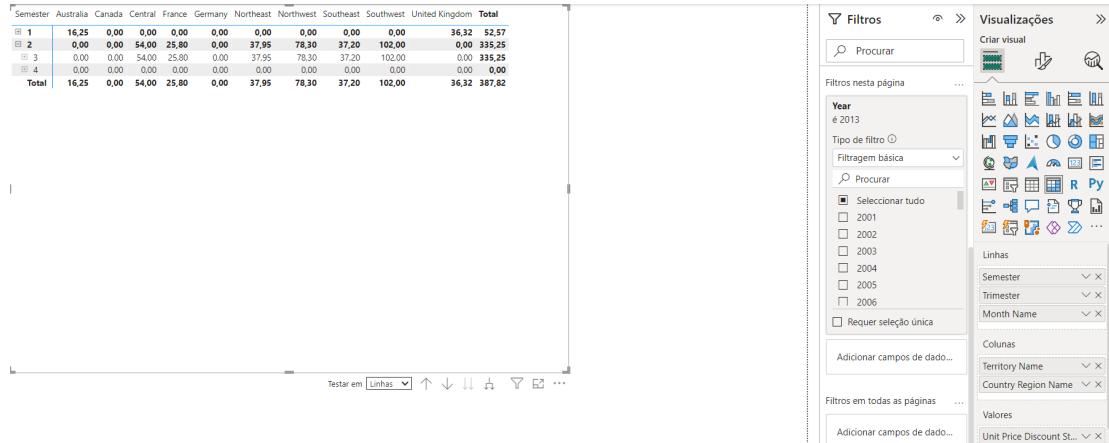


Figura 8.7: Análise 7

8.8 Análise 8

Na análise 8 são apresentados as quantidades vendidas a clientes por vendedor e por cidade da morada de expedição, com possibilidade de análise agregada (i.e., roll up) ao nível do estado/província, para as vendas expedidas no último dia de cada mês do ano de 2013. Na Figura 8.8 é apresentada a seguinte análise.

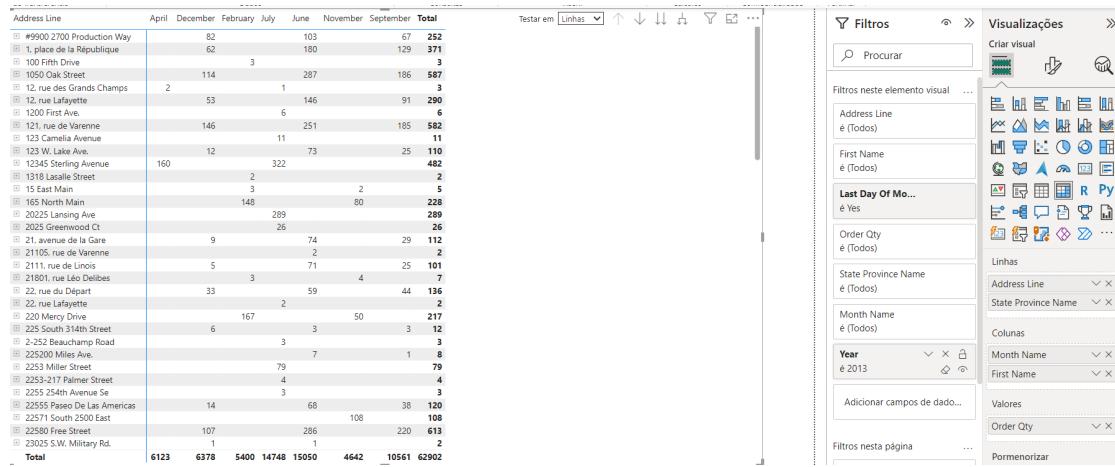


Figura 8.8: Análise 8

8.9 Análise 9

Na análise 9 são apresentados os valores totais (incluindo fretes e impostos) das vendas a clientes por cidade da morada de faturação e por categoria de produto, com possibilidade de análise detalhada (i.e., drill down) ao nível da subcategoria de produto, para o ano de 2013. Na Figura 8.9 é apresentada a seguinte análise.

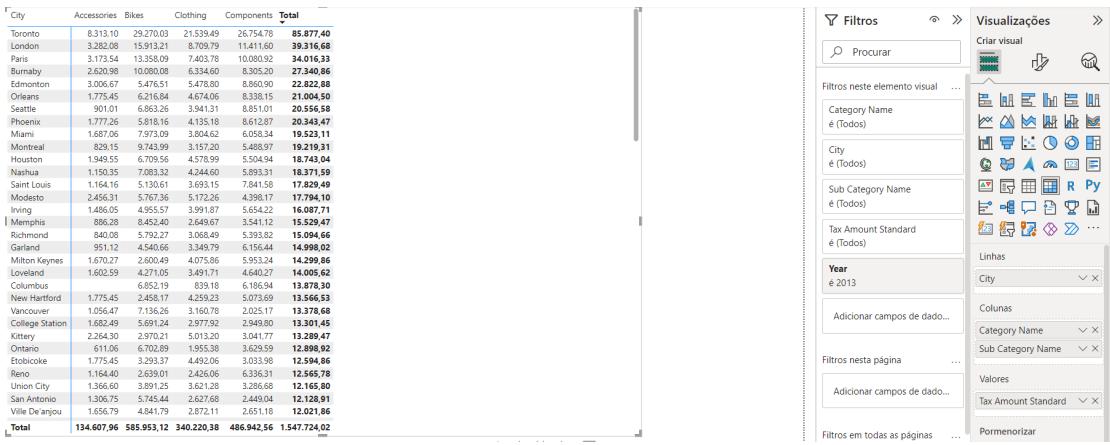


Figura 8.9: Análise 9

8.10 Análise 10

Na análise 10 são apresentados os valores totais das vendas (sem incluir frete e imposto) por moeda e por território, com possibilidade de análise agregada (i.e., roll up) ao nível do país/região do cliente, no terceiro quadrimestre de 2013. Na Figura 8.10 é apresentada a seguinte análise.

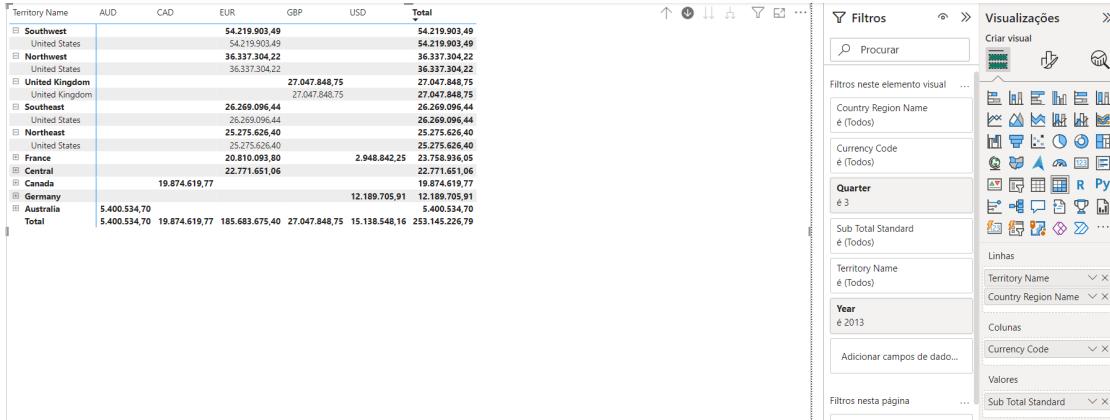


Figura 8.10: Análise 10

8.11 Análise 11

Na análise 11 são apresentados os valores referentes à quantidade de produtos vendidos pelo nome de cada produto. Na Figura 8.11 é apresentada a seguinte análise.

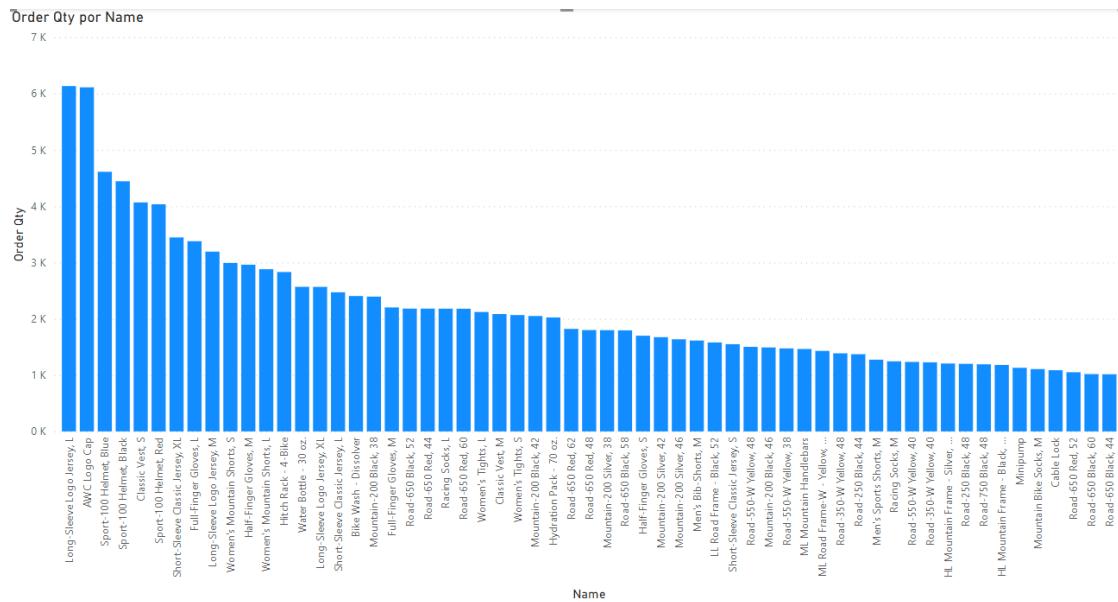
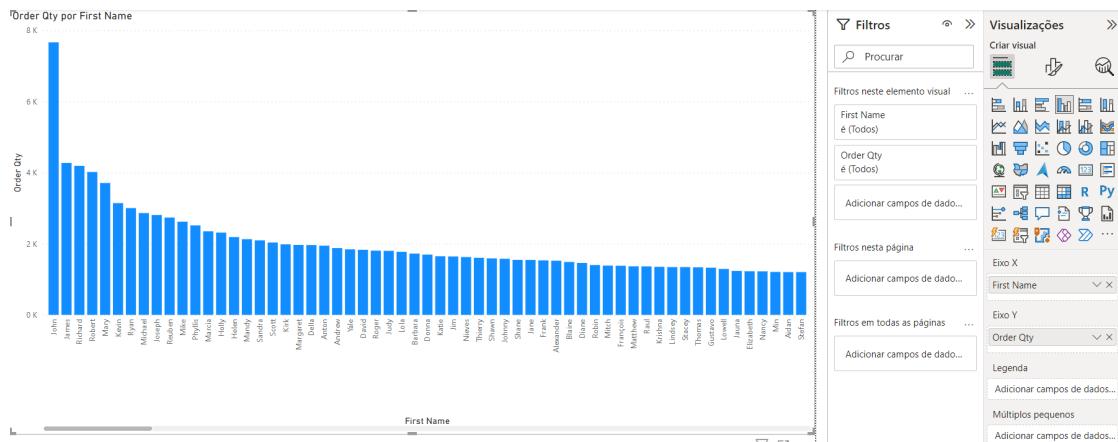


Figura 8.11: Análise 11

8.12 Análise 12

Na análise 12 são apresentados os valores da quantidade de produtos vendidos por cada primeiro nome de cada cliente. Na Figura 8.12 é apresentada a seguinte análise.



8.13 Análise 13

Na análise 13 são apresentados o valor subtotal de uma compra (sem frete e taxas) pelo nome de cada produto, isto é, quanto dinheiro é que a empresa ganhou com as vendas totais desse produto. Na Figura 8.13 é apresentada a seguinte análise.

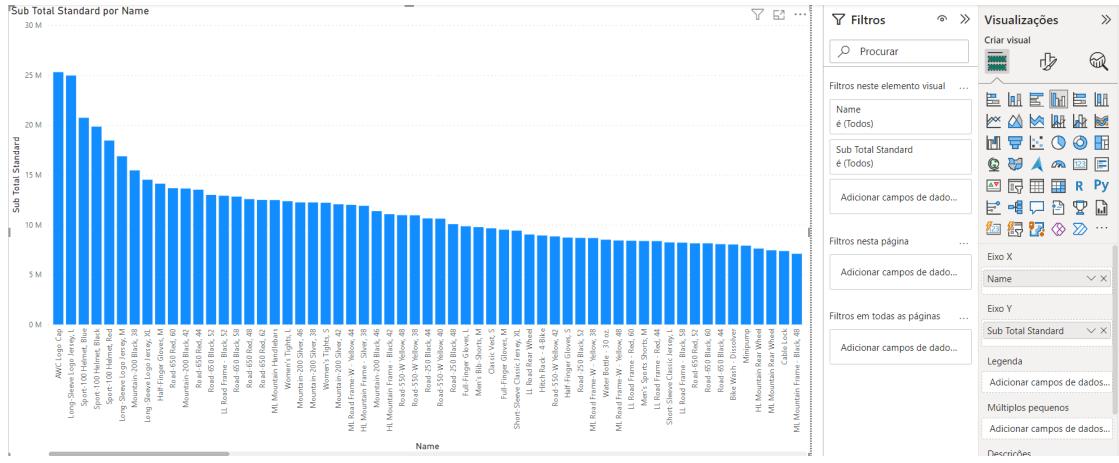


Figura 8.13: Análise 13

8.14 Análise 14

Na análise 14 são apresentadas as quantidades de produtos comprados por cliente. Na Figura 8.14 é apresentada a seguinte análise.

First Name	Order Qty
[+ A.	5
[+ Aaron	128
[+ Abe	15
[+ Abigail	680
[+ Abraham	28
[+ Adam	309
[+ Adrian	5
[+ Aidan	1203
[+ Ajay	5
[+ Alan	27
[+ Aldeen	1
[+ Alejandro	7
[+ Alexander	1523
[+ Alice	123
[+ Alvaro	127
[+ Alvin	1
[+ Amy	756
[+ Andrea	4
[+ Andrew	1878
[+ Andy	47
[+ Anita	239
[+ Ann	26
[+ Anna	817
[+ Anthony	258
[+ Anton	1946
[+ Arlene	1
[+ Barbara	1724
[+ Baris	1104
[+ Barry	105
[+ Bart	258
[+ Ben	38
[+ Benjamin	150
[+ Bernard	701
Total	214278

Figura 8.14: Análise 14

8.15 Análise 15

Na análise 15 são apresentados os valores da faturação de cada ano (excluindo frete e taxas), com possibilidade de drill down na granularidade de semestre, trimestre e mês. Na Figura 8.15 é apresentada a seguinte análise.

Year	Total Due Standard
2011	3.717.573,42
2012	18.235.771,10
2013	19.535.823,42
2014	10.587.156,93
Total	52.076.324,87

Figura 8.15: Análise 15

8.16 Análise 16

Na análise 16 são apresentados os valores da faturação de cada ano por produto com possibilidade de drill down do nome do cliente. Na Figura 8.16 é apresentada a seguinte análise.

Name	Total Due Standard
AWC Logo Cap	1.080.369,81
Aaron	4.924,21
Adam	3.944,45
Aidan	7.617,56
Alexander	6.351,87
Amy	5.106,12
Andrew	1.057,98
Ann	230,63
Anna	2.933,38
Anton	6.157,33
Barbara	6.769,98
Baris	2.226,13
Benjamin	1.082,24
Bernard	230,63
Beth	4.929,97
Bev	4.929,97
Blaine	11.854,75
Brannon	5.697,42
Brenda	5.969,20
Bruno	161,24
Bryan	849,63
Carla	1.919,15
Carolee	3.467,39
Caroline	8.039,51
Vicknair	8.039,51
Catherine	2.933,38
Cecelia	2.862,21
Cecil	5.697,42
Charles	2.021,63
Cheryl	4.800,03
Chris	2.933,38
Christie	4.800,03
Christopher	8.039,51
Total	52.076.324,87

Figura 8.16: Análise 16

8.17 Análise 17

Na análise 17 são apresentados os valores totais das taxas para cada endereço de expedição. Na Figura 8.17 é apresentada a seguinte análise.

Address Line	Tax Amount Standard
#9900 2700 Production Way	20.616,19
1 Corporate Center Drive	1.848,49
1, place de la République	6.523,84
100 Fifth Drive	3.790,95
1050 Oak Street	24.664,56
12, rue des Grands Champs	816,15
12, rue Lafayette	8.846,37
1200 First Ave.	718,34
121, rue de Varenne	8.733,32
123 Camelia Avenue	1.765,68
123 Union Square South	2.095,47
123 W. Lake Ave.	3.185,31
12345 Sterling Avenue	34.961,21
1318 Lasalle Street	7,80
15 East Main	2.288,74
165 North Main	12.912,79
2000 300th Street	7.187,76
20225 Lansing Ave	9.697,73
2025 Greenwood Ct	1.454,92
20500 S.W. 2512th Ave	117,03
21, avenue de la Gare	2.399,10
21105, rue de Varenne	168,62
2111, rue de Linois	4.303,77
21801, rue Léo Delibes	1.011,94
22, rue du Départ	5.983,59
22, rue Lafayette	251,47
220 Mercy Drive	20.253,69
225 South 314th Street	900,50
2251 Elliot Avenue	10.073,20
2-252 Beauchamp Road	212,69
225200 Miles Ave.	9.164,30
2253 Miller Street	1.061,23
2253-217 Palmer Street	757,11
Total	4.446.423,58

Figura 8.17: Análise 17

8.18 Análise 18

Na análise 18 são apresentados os valores totais das quantidades vendidas tendo em conta a cor dos produtos. Na Figura 8.18 é apresentada a seguinte análise.

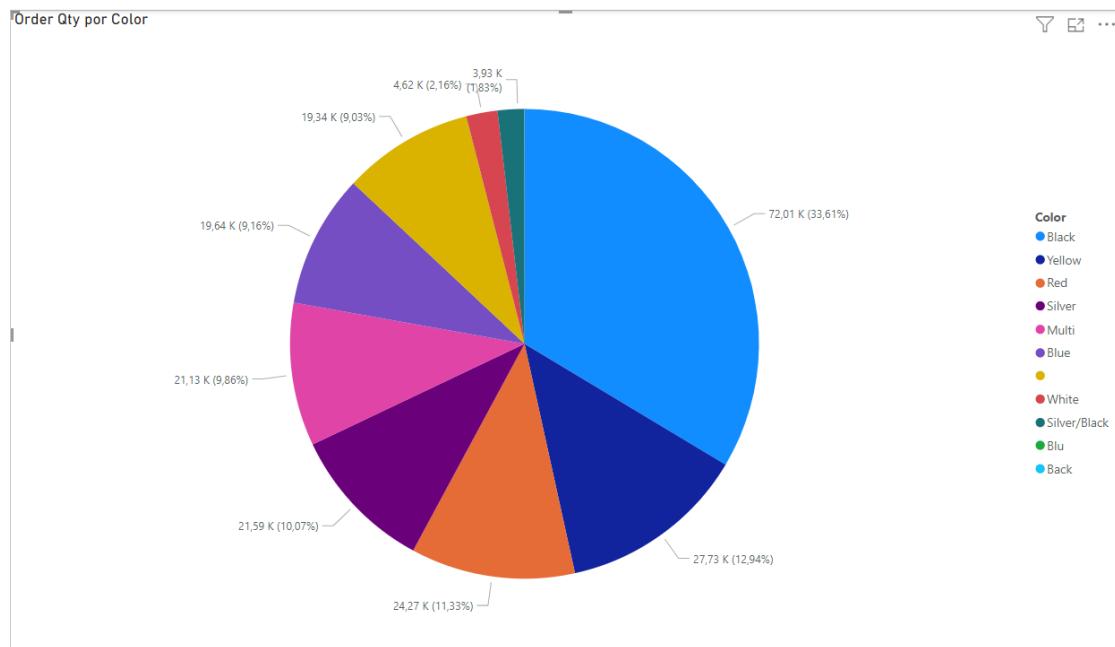


Figura 8.18: Análise 18

8.19 Análise 19

Na análise 19 são apresentados os valores monetários vendidos de cada unidade tendo em conta o género da pessoa vendedora. Na Figura 8.19 é apresentada a seguinte análise.



Figura 8.19: Análise 19

8.20 Análise 20

Na análise 20 é apresentada a comparação de preço unitário de cada produto, tendo em conta os dias que foram necessários para produzir esse produto. Na Figura 8.20 é apresentada a seguinte análise.



Figura 8.20: Análise 20

Capítulo 9

Conclusões

9.1 Trabalho futuro

No futuro seria interessante desenvolver o data mart para conseguir-se realizar outro tipo de análises e desenvolver um serviço para obtenção dos câmbios diários automaticamente. Por outro lado, a implementação de deteção de produtos semelhantes pode ser alastrada às outras dimensões já que podem ter sido introduzidos registos com identificadores diferentes, no entanto, com valores iguais nas dimensões restantes. Essa implementação pode seguir os moldes daquilo que foi o exemplo para o produto, presente na Figura 9.1.

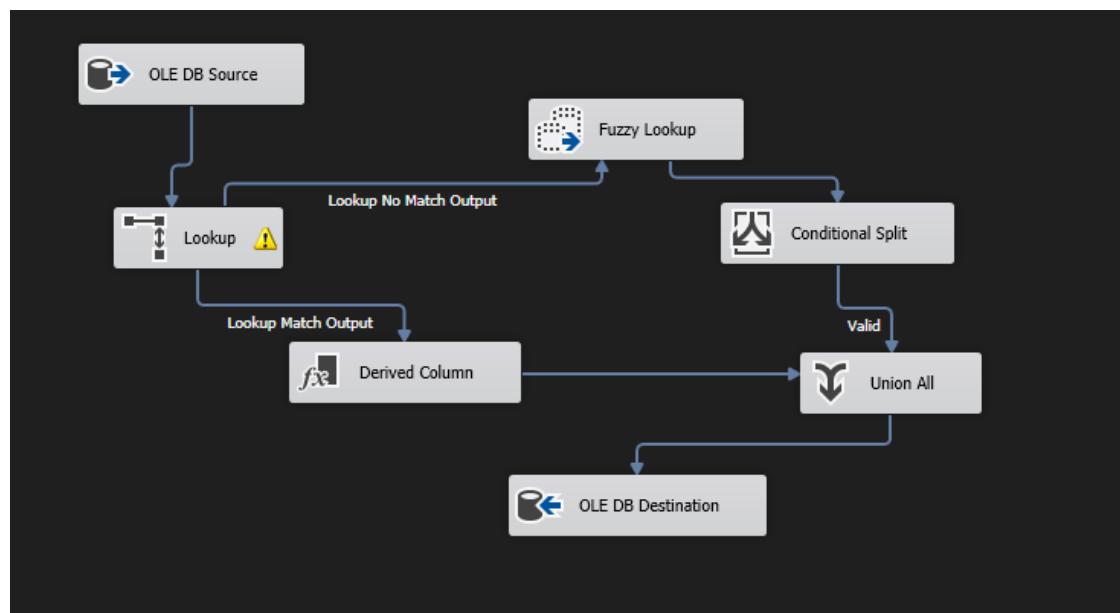


Figura 9.1: Produtos Semelhantes

O threshold utilizado foi 90 por cento, visto que há registos para o mesmo produto que tem parâmetros como o tamanho que diferem e por isso trata-se do mesmo produto, ou seja, deve dar-se uma folga maior para abranger estes casos.

9.2 Conclusões ao nível dos dados

A análise de dados realizada no desenvolvimento do projeto revela que em termos gerais, a faturação da empresa foi exponencialmente maior em 2012 e 2013, o que pode ser explicado pelo facto de em 2011 e 2014 haver dados apenas para parte do ano. Por outro lado, o produto mais vendido em termos de quantidade foi a “Long Sleeve Logo Jersey” seguida de perto pela “AWC Logo Cap”, com aproximadamente 6000 unidades vendidas no total. Fechando o pódio tem-se o “Sport 1000 Helmet, Blue” com 4000 unidades vendidas, o que demonstra haver um fosso considerável entre os dois primeiros produtos e o terceiro espelhando as preferências dos consumidores. No entanto, estes têm uma preferência notória pelos produtos que possuem a cor preta, já que em 10 cores possíveis, os produtos vendidos em cor preta representam um terço dos restantes produtos nas cores todas. Por fim, é também notório que a empresa pode apostar em diferentes métodos de envio, já que em 6 registados, existe uma predominância visível do “Cargo Transport 5”.

9.3 Conclusões ao nível do desenvolvimento

Neste trabalho foi possível verificar o desempenho da ferramenta PowerBI na análise de dados, nomeadamente na facilidade com que é possível fazer “rollup” e “drill down” assim como selecionar os mesmos valores de algumas dimensões dentro da mesma página. Além disso, a divisão do trabalho em packages diferentes, permitiu que os dois membros do grupo conciliassem o trabalho evitando conflitos ao nível de controlo de versões no Visual Studio. No parecer do grupo o projeto foi enriquecedor para abordar os tópicos lecionados ao longo das aulas, nomeadamente aqueles que são menos comuns, como a uniformização da moeda, conversão dos atributos ao nível da linha e utilização de duas tabelas de facto. O resultado final deste projeto, que engloba todos os passos detalhados até aqui, está presente na Figura 9.2.

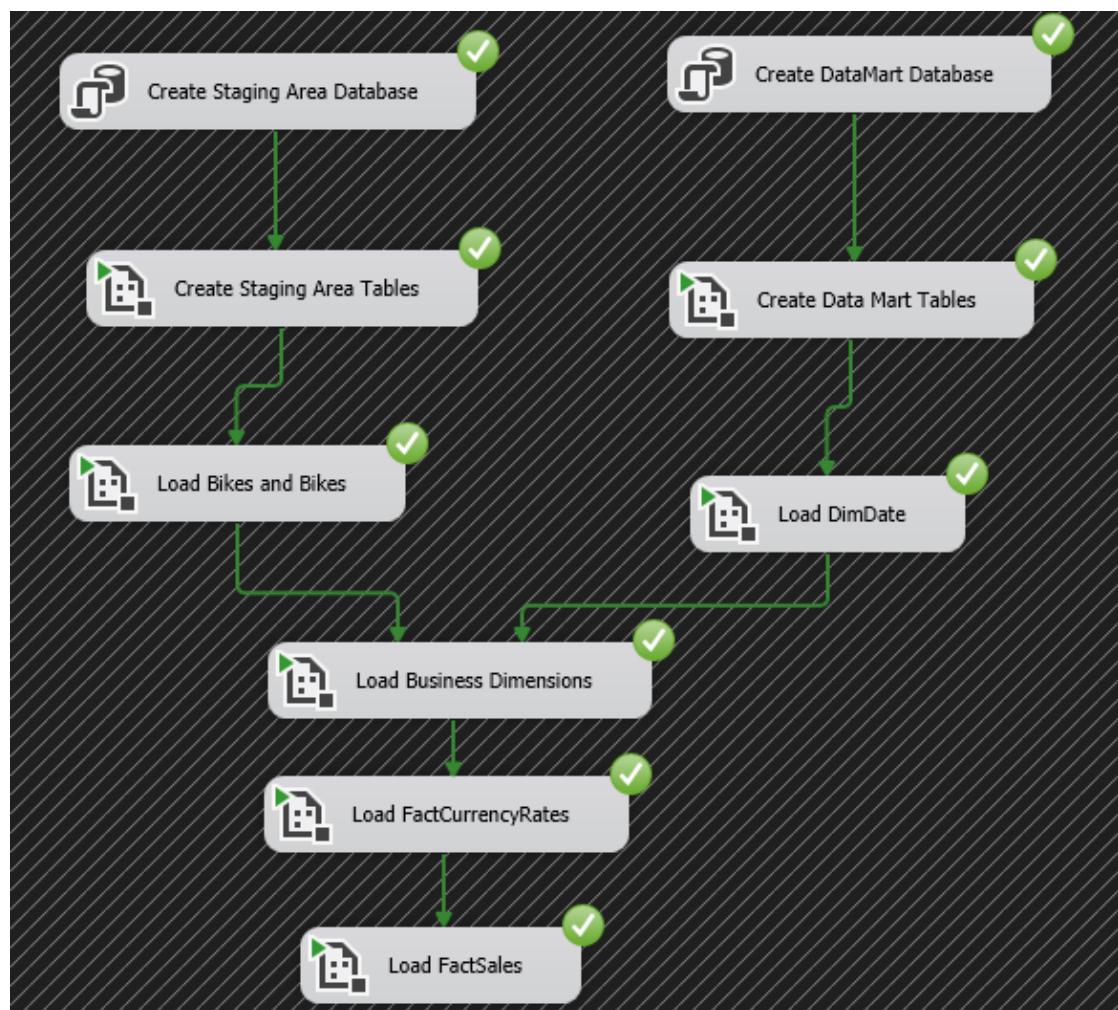


Figura 9.2: Sucesso do fluxo dos packages