EXERCÍCIOS WINDOW FUNCTIONS

Para resolver os exercícios 1 a 4, crie uma View chamada vwProdutos, que contenha o agrupamento das colunas BrandName, ColorName e os totais de quantidade vendida por marca/cor e também o total de receita por marca/cor.

```
CREATE VIEW vwProdutos AS
SELECT
       BrandName AS 'Marca',
       ColorName AS 'Cor',
       COUNT(*) AS 'Quantidade_Vendida',
       ROUND(SUM(SalesAmount), 2) AS 'Receita_Total'
FROM DimProduct
INNER JOIN FactSales
       ON DimProduct.ProductKey = FactSales.ProductKey
GROUP BY BrandName, ColorName
```

1. Utilize a View **vwProdutos** para criar uma coluna extra calculando a quantidade total vendida dos produtos.

2. Crie mais uma coluna na consulta anterior, incluindo o total de produtos vendidos para cada marca.

3. Calcule o % de participação do total de vendas de produtos por marca.

Ex: A marca A. Datum teve uma quantidade total de vendas de 199.041 de um total de 3.406.089 de vendas. Isso significa que a da marca A. Datum é 199.041/3.406.089 = 5,84%.

4. Crie uma consulta à View **vwProdutos**, selecionando as colunas Marca, Cor, Quantidade_Vendida e também criando uma coluna extra de Rank para descobrir a posição de cada Marca/Cor. Você deve obter o resultado abaixo. Obs: Sua consulta deve ser filtrada para que seja mostrada apenas a marca Contoso.

Marca	Cor	Quantidade_Por_Produto	Rank
Contoso	Black	239810	1
Contoso	White	230199	2
Contoso	Silver	149081	3
Contoso	Grey	133244	4
Contoso	Blue	52819	5
Contoso	Red	50168	6
Contoso	Pink	21709	7
Contoso	Brown	21014	8
Contoso	Green	19617	9
Contoso	Orange	10105	10
Contoso	Yellow	9295	11
Contoso	Gold	7502	12
Contoso	Silver Grey	6419	13
Contoso	Purple	1752	14
Contoso	Transparent	1655	15

Exercício Desafio 1.

Para responder os próximos 2 exercícios, você precisará criar uma View auxiliar. Diferente do que foi feito anteriormente, você não terá acesso ao código dessa view antes do gabarito.

A sua view deve se chamar vwHistoricoLojas e deve conter um histórico com a quantidade de lojas abertas a cada Ano/Mês. Os desafios são:

(1) Criar uma coluna de ID para essa View
(2) Relacionar as tabelas DimDate e DimStore

Dicas:

1- A coluna de ID será criada a partir de uma função de janela. Você deverá se atentar a forma como essa coluna deverá ser ordenada, pensando que queremos visualizar uma ordem de Ano/Mês que seja: 2005/january, 2005/February... e não 2005/April, 2005/August...

2- As colunas Ano, Mês e Qtd_Lojas correspondem, respectivamente, às seguintes colunas: CalendarYear e CalendarMonthLabel da tabela DimDate e uma contagem da coluna OpenDate da tabela DimStore.

ID	Ano	Mês	Qtd_Lojas
1	2005	January	0
2	2005	February	0
3	2005	March	0
4	2005	April	0
5	2005	May	2
6	2005	June	4
7	2005	July	1
8	2005	August	0
9	2005	September	0
10	2005	October	0
11	2005	November	1
12	2005	December	0
13	2006	January	2
14	2006	February	3
15	2006	March	0
16	2006	April	1
17	2006	May	0
18	2006	June	0
19	2006	July	1
20	2006	August	1
21	2006	September	0
22	2006	October	0

5. A partir da view criada no exercício anterior, você deverá fazer uma soma móvel considerando sempre o mês atual + 2 meses para trás.

6. Utilize a **vwHistoricoLojas** para calcular o acumulado de lojas abertas a cada ano/mês.

Exercício Desafio 2

Neste desafio, você terá que criar suas próprias tabelas e views para conseguir resolver os exercícios 7 e 8. Os próximos exercícios envolverão análises de novos clientes. Para isso, será necessário criar uma nova tabela e uma nova view.

Abaixo, temos um passo a passo para resolver o problema por partes.

PASSO 1: Crie um novo banco de dados chamado Desafio e selecione esse banco de dados criado.

PASSO 2: Crie uma tabela de datas entre o dia 1 de janeiro do ano com a compra (DateFirstPurchase) mais antiga e o dia 31 de dezembro do ano com a compra mais recente.

Obs1: Chame essa tabela de Calendario.
Obs2: A princípio, essa tabela deve conter apenas 1 coluna, chamada data e do tipo DATE.

PASSO 3: Crie colunas auxiliares na tabela Calendario chamadas: Ano, Mes, Dia, AnoMes e NomeMes. Todas do tipo INT.

PASSO 4: Adicione na tabela os valores de Ano, Mês, Dia, AnoMes e NomeMes (nome do mês em português). Dica: utilize a instrução CASE para verificar o mês e retornar o nome certo.

PASSO 5: Crie a View vwNovosClientes, que deve ter as colunas mostradas abaixo.

ID	Ano	NomeMes	Novos_Clientes
1	2001	Janeiro	0
2	2001	Fevereiro	0
3	2001	Março	0
4	2001	Abril	0
5	2001	Maio	0
6	2001	Junho	0
7	2001	Julho	146
8	2001	Agosto	156
9	2001	Setembro	146
10	2001	Outubro	161
11	2001	Novembro	169
12	2001	Dezembro	235
13	2002	Janeiro	188
14	2002	Fevereiro	171
15	2002	Março	199
16	2002	Abril	207
17	2002	Maio	214
18	2002	Junho	214
19	2002	Julho	253
20	2002	Agosto	281
21	2002	Setembro	198
22	2002	Outubro	229
23	2002	Novembro	193
24	2002	Dezembro	330
25	2003	Janeiro	244
26	2003	Fevereiro	272
27	2003	Março	272
28	2003	Abril	294
29	2003	Maio	335
30	2003	Junho	321
31	2003	Julho	202
32	2003	Agosto	1210
33	2003	Setembro	1112

7.
a) Faça um cálculo de soma móvel de novos clientes nos últimos 2 meses.
b) Faça um cálculo de média móvel de novos clientes nos últimos 2 meses.
c) Faça um cálculo de acumulado dos novos clientes ao longo do tempo.
d) Faça um cálculo de acumulado intra-ano, ou seja, um acumulado que vai de janeiro a dezembro de cada ano, e volta a fazer o cálculo de acumulado no ano seguinte.

8. Faça os cálculos de MoM e YoY para avaliar o percentual de crescimento de novos clientes, entre o mês atual e o mês anterior, e entre um mês atual e o mesmo mês do ano anterior.

Solução

1.

```
SELECT
	*,
	SUM(Quantidade_Por_Produto) OVER() AS 'Qtd Total Produtos'
FROM vwProdutos
ORDER BY Marca
```

2.

```
SELECT
	*,
	SUM(Quantidade_Por_Produto) OVER() AS 'Qtd Total Produtos',
	SUM(Quantidade_Por_Produto) OVER(PARTITION BY Marca) AS 'Qtd Total Por
Marca'
FROM vwProdutos
ORDER BY Marca
```

3.

```
SELECT
	*,
	SUM(Quantidade_Por_Produto) OVER() AS 'Qtd1',
	SUM(Quantidade_Por_Produto) OVER(PARTITION BY Marca) AS 'Qtd2',
	FORMAT(1.0*(SUM(Quantidade_Por_Produto) OVER(PARTITION BY
Marca))/SUM(Quantidade_Por_Produto) OVER(), '0.00%')
FROM vwProdutos
ORDER BY Marca
```

4.

```
SELECT
```

```
	*,
	RANK() OVER(ORDER BY Quantidade_Por_Produto DESC) AS 'Rank'
FROM vwProdutos
WHERE Marca = 'Contoso'


Exercício Desafio 1

CREATE VIEW vwHistoricoLojas AS
SELECT
	ROW_NUMBER() OVER(ORDER BY CalendarMonth) AS 'ID',
	CalendarYear AS 'Ano',
	CalendarMonthLabel AS 'Mês',
	COUNT(DimStore.OpenDate) AS'Qtd_Lojas'
FROM DimDate
LEFT JOIN DimStore
	ON DimDate.Datekey = DimStore.OpenDate
GROUP BY CalendarYear, CalendarMonthLabel, CalendarMonth


5.

SELECT
	*,
	SUM(Qtd_Lojas) OVER(ORDER BY ID ROWS BETWEEN 2 PRECEDING AND CURRENT ROW)
FROM vwHistoricoLojas

6.

SELECT
	*,
	SUM(Qtd_Lojas) OVER(ORDER BY ID ROWS BETWEEN UNBOUNDED PRECEDING AND
CURRENT ROW)
FROM vwHistoricoLojas


Exercício Desafio 2

--1) Passo 1: Criar o Banco de Dados Desafio

CREATE DATABASE Desafio
USE Desafio

--2) Passo 2: Criar uma tabela de datas entre o dia 1 de janeiro do ano com a
primeira compra de cliente mais antiga até o dia 31 de dezembro do ano com a
última compra

CREATE TABLE Calendario (
  data DATE
)


DECLARE @varAnoInicial INT = YEAR((SELECT MIN(DateFirstPurchase) FROM
ContosoRetailDW.dbo.DimCustomer))
DECLARE @varAnoFinal INT = YEAR((SELECT MAX(DateFirstPurchase) FROM
ContosoRetailDW.dbo.DimCustomer))

DECLARE @varDataInicial DATE = DATEFROMPARTS(@varAnoInicial, 1, 1)
DECLARE @varDataFinal DATE = DATEFROMPARTS(@varAnoFinal, 12, 31)


WHILE @varDataInicial <= @varDataFinal
```

```
BEGIN
	INSERT INTO Calendario(data) VALUES(@varDataInicial)
	SET @varDataInicial = DATEADD(DAY, 1, @varDataInicial)
END

SELECT * FROM Calendario

--3) Passo 3: Criar colunas auxiliares na tabela Calendario

ALTER TABLE Calendario
ADD Ano INT,
	Mes INT,
	Dia INT,
	AnoMes INT,
	NomeMes VARCHAR(50)

--4) Passo 4: Adicionar os valores às colunas

UPDATE Calendario SET Ano = YEAR(data)
UPDATE Calendario SET Mes = MONTH(data)
UPDATE Calendario SET Dia = DAY(data)
UPDATE Calendario SET AnoMes = CONCAT(YEAR(data), FORMAT(MONTH(data), '00'))
UPDATE Calendario SET NomeMes =
		CASE
			WHEN MONTH(data) = 1 THEN 'Janeiro'
			WHEN MONTH(data) = 2 THEN 'Fevereiro'
			WHEN MONTH(data) = 3 THEN 'Março'
			WHEN MONTH(data) = 4 THEN 'Abril'
			WHEN MONTH(data) = 5 THEN 'Maio'
			WHEN MONTH(data) = 6 THEN 'Junho'
			WHEN MONTH(data) = 7 THEN 'Julho'
			WHEN MONTH(data) = 8 THEN 'Agosto'
			WHEN MONTH(data) = 9 THEN 'Setembro'
			WHEN MONTH(data) = 10 THEN 'Outubro'
			WHEN MONTH(data) = 11 THEN 'Novembro'
			WHEN MONTH(data) = 12 THEN 'Dezembro'
		END

--5) Passo 5: Crie uma View

CREATE VIEW vwNovosClientes AS
SELECT
	ROW_NUMBER() OVER(ORDER BY AnoMes) AS 'ID',
	Ano,
	NomeMes,
	COUNT(DimCustomer.DateFirstPurchase) AS  'Novos_Clientes'
FROM Calendario
LEFT JOIN ContosoRetailDW.dbo.DimCustomer
	ON Calendario.data = DimCustomer.DateFirstPurchase
GROUP BY Ano, NomeMes, AnoMes

SELECT * FROM vwNovosClientes



7.


SELECT
	*,
	SUM(Novos_Clientes) OVER(ORDER BY ID ROWS BETWEEN 2 PRECEDING AND CURRENT
ROW) AS 'Soma Móvel (2 meses)',
```

```
	AVG(Novos_Clientes) OVER(ORDER BY ID ROWS BETWEEN 2 PRECEDING AND CURRENT
ROW) AS 'Média Móvel (2 meses)',
	SUM(Novos_Clientes) OVER(ORDER BY ID ROWS BETWEEN UNBOUNDED PRECEDING AND
CURRENT ROW) AS 'Acumulado Total',
	SUM(Novos_Clientes) OVER(PARTITION BY Ano ORDER BY ID ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW) AS 'Acumulado (YTD)'
FROM vwNovosClientes
```

8. Vai ter que dividir em duas partes para corrigir o problema do CONVERT

```
SELECT
	*,
	FORMAT(CONVERT(FLOAT, Novos_Clientes)/NULLIF(LAG(Novos_Clientes, 1)
OVER(ORDER BY ID), 0) - 1, '0.00%') AS '% MoM',
	FORMAT(CONVERT(FLOAT, Novos_Clientes)/NULLIF(LAG(Novos_Clientes, 12)
OVER(ORDER BY ID), 0) - 1, '0.00%') AS '% YoY'
FROM vwNovosClientes
```