-- Inserir comandos Delete e update com top

CREATE DATABASE TESTE

Go

USE TESTE

Go

CREATE TABLE Cliente

(

Cod\_Cli int,

Nome\_Cli char(30),

Rua\_Cli char(20),

Fone\_Cli char(10),

Email\_Cli char(20),

Sal\_Cli decimal(10,2),

Data\_Cli datetime

)

Go

INSERT Cliente VALUES( 1,'Ana de Souza','Rua A','2188-2587','Ana@Hotmail.com',1500.25,'05/15/01' )

INSERT Cliente VALUES( 2,'Mariana de Souza','Rua B','6524-2587','Mariana@Hotmail.com',2800.15,'08/06/02' )

INSERT Cliente VALUES( 3,'Fabiana de Souza','Rua C','4578-5689','Fabiana@Hotmail.com',287.56,'06/24/02' )

INSERT Cliente VALUES( 4,'Rosana de Souza','Rua D','3212-3212','Rosana@Hotmail.com',5589.58,'06/27/02' )

INSERT Cliente VALUES( 5,'Adriana de Souza','Rua E','5263-7551','Adriana@Hotmail.com',6785.59,'06/22/01' )

Go

-- 1. Exibindo todas as colunas.

SELECT \* FROM Cliente

-- 2. Retornando algumas Colunas.

SELECT

Nome\_Cli,

Fone\_Cli,

Email\_Cli

FROM

Cliente

-- 3. Retornando algumas colunas Com apelidos. MAIS RECOMENDADO

SELECT

Nome\_Cli AS Nome, --nao altera nome da coluna apenas muda o apelido

Fone\_Cli AS Telefone,

Email\_Cli AS Email

FROM

Cliente

-- 4. Retornando algumas colunas Com apelidos

SELECT

Nome\_Cli [Nome do Cliente], --para escrever nome com espaços, usar chaves.

Fone\_Cli Telefone,

Email\_Cli Email

FROM

Cliente

-- 5. Retornando algumas colunas Com apelidos.

SELECT

Nome = Nome\_Cli,

Fone = fone\_Cli,

Email = Email\_Cli

FROM

Cliente

-- 6. Retornando algumas colunas Com apelidos contendo espaços (NAO RECOMENDADO)

SELECT

Nome\_Cli AS [ Nome do Cliente ],

Fone\_Cli AS [ Telefone do Cliente ],

Email\_Cli AS [ Email do Cliente ]

FROM

Cliente

SELECT

Nome\_Cli [ Nome do Cliente ],

Fone\_Cli [ Telefone do Cliente ],

Email\_Cli [ Email do Cliente ]

FROM

Cliente

SELECT

[ Nome do Cliente ] = Nome\_Cli,

[ Fone do Cliente ] = fone\_Cli,

[ Email do Cliente ] = Email\_Cli

FROM

Cliente

-- 7. Criando uma Coluna Virtual com o comando Select.

SELECT

'CLIENTE' Tipo, --TEXTO PARA SEPARAÇÃO DE FUNCIONARIOS E CLIENTES ("Tipo" apenas apelida a Coluna antes de "Nome \_Cli")

Nome\_Cli

From Cliente

UNION ALL --UNIAO DOS NUMA SO COLUNA (APENAS UNION NAO APARECE DADOS DUPLICADOS E PODE DAR LENTIDAO, com ALL faz o mesmo mas duplica os dados)

SELECT

'FUNCIONARIO',--TEXTO PARA SEPARAÇÃO DE FUNCIONARIOS E CLIENTES

Nome\_func

From Funcionario

SELECT

Nome\_Cli AS [Nome do Cliente],

Fone\_Cli AS [Telefone do Cliente],

Email\_Cli AS [Email do Cliente],

'Pessoa Fisica' AS [Tipo de Cliente]

FROM

Cliente

SELECT

Nome\_Cli [ Nome do Cliente ],

Fone\_Cli [ Telefone do Cliente ],

Email\_Cli [ Email do Cliente ],

'Pessoa Fisica' [ Tipo de Cliente ]

FROM

Cliente

SELECT

[Nome do Cliente] = Nome\_Cli,

[Fone do Cliente] = fone\_Cli,

[Email do Cliente] = Email\_Cli,

[Tipo de Cliente] = 'Pessoa Fisica'

FROM

Cliente

-- 8. Exibindo um Cálculo com o comando select

SELECT

Nome\_Cli AS [ Nome do Cliente ],

Fone\_Cli AS [ Telefone do Cliente ],

Email\_Cli AS [ Email do Cliente ],

'Pessoa Fisica' AS [ Tipo de Cliente ],

Sal\_Cli AS [ Salário do Cliente ],

Sal\_Cli \* 1.1 AS [ Salário com 10% de Aumento ]

FROM

Cliente

/\*\*\*\*\* DISTINCT \*\*\*\*\*\*\*/

--9. Inserindo mais um Registro na tabela em que existe algumas informações do registro 5

INSERT Cliente VALUES ( 6,'Adriana de Souza','Rua F','3125-4152','Drica@Hotmail.com',2500.00,'06/22/01' )

-- Primeiro Resultado

SELECT Nome\_Cli, Data\_Cli FROM CLIENTE

-- Segundo Resultado

SELECT DISTINCT Nome\_Cli, Data\_Cli FROM CLIENTE --Verifica se tem duplicidade em todas as colunas e aparece apenas uma vez

/\*\*\*\*\*\*\*UPDATE\*\*\*\*\*\*\*\*\*/

INSERT Cliente VALUES( 7,'Cristina de Souza','Rua F','2541-1542','Cristina@Hotmail.com',1500.00,'03/03/03' )

INSERT Cliente VALUES( 8,'Emiliana de Souza','Rua F','9187-6589','Emiliana@Hotmail.com',1500.00,'03/03/03' )

--10. Alterando Dados de apenas uma coluna da tabela Cliente

UPDATE Cliente

SET Sal\_Cli = Sal\_Cli \* 1.1

-- Visualizar Dados após a atualizar Informações do Salário de Cliente

SELECT \* FROM Cliente

---11. Alterando Dados de duas colunas da tabela Cliente

UPDATE Cliente

SET Sal\_Cli = Sal\_Cli \* 1.1,

Data\_Cli = Data\_Cli + 1 --SOMENTE O NUMERO ADICIONA DIAS

WHERE Cod\_Cli = 1-- PARA COLOCAR FILTRO

-- Visualizar Dados após a atualizar Informações do Salário de Cliente

SELECT \* FROM Cliente

--12. DELETANDO DADOS DA TABELA

DELETE FROM Cliente

TRUNCATE TABLE Cliente-- IGUAL UM SHIFT+DEL EXCLUI MAIS RAPIDO OS DADOS DA TABELA, (USAR SOMENTE SE FOR DELETAR TUDO)

--Utilizando a Clasula Where

CREATE TABLE Produto

(

Cod\_Prod int,

Nome\_Prod char(50),

Tipo\_Prod char(20),

Qtd\_Prod int,

Val\_Prod decimal(10,2)

)

INSERT Produto VALUES(1,'Geladeira','Cozinha',500,1500.00)

INSERT Produto VALUES(2,'Armário','Cozinha',700,300.00)

INSERT Produto VALUES(3,'Sofá','Sala',1200,00.00)

INSERT Produto VALUES(4,'Cama','Quarto',300,500.00)

INSERT Produto VALUES(5,'Fogão','Cozinha',400,700.00)

INSERT Produto VALUES(6,'Guarda-Roupa','Quarto',500,100.00)

INSERT Produto VALUES(7,'Poltrona','Sala',600,250.00)

INSERT Produto VALUES(8,'Estante','Sala',800,400.00)

INSERT Produto VALUES(9,'Penteadeira','Sala',200,300.00)

INSERT Produto VALUES(10,'Cadeira','Cozinha',100,150.00)

-- 13. O Comando para realizar a alteração de Preços nos Produtos do tipo Sala é:

select \* from produto

UPDATE Produto

SET Val\_Prod = Val\_Prod \* 1.1

WHERE Tipo\_Prod = 'Sala'

-- 14. Exibindo somente apenas os Produtos do tipo Sala

SELECT \* FROM Produto

WHERE Tipo\_Prod = 'Sala'

/\*15. Aumentar a Quantida de em Estoque de Todos os produtos cuja a quantidade em estoque

seja igual ou inferior a 300 unidades em mais 100 Unidades.\*/

UPDATE Produto

SET Qtd\_Prod = Qtd\_Prod + 100

WHERE Qtd\_Prod <= 300 --Filtro (pode ser usado regras)

Select \* from Produto

WHERE Qtd\_Prod <= 300

--16. Clasula Where Com operadores AND, OR E IN

UPDATE Produto

SET Val\_Prod = Val\_Prod \* 1.1

WHERE Tipo\_Prod = 'Sala'

AND Val\_Prod < 100.00

UPDATE Produto

SET Val\_Prod = Val\_Prod \* 1.1

WHERE Tipo\_Prod = 'Sala'

AND Val\_Prod < 100.00

SELECT \* FROM Produto

WHERE Tipo\_Prod = 'Sala' OR Tipo\_Prod = 'Quarto' --filtra para aparecer os tipos Um ou Outro, existe tambem o AND.

SELECT \* FROM Produto

WHERE Tipo\_Prod IN ('Sala','Quarto') --igual o OR, mas digita menos

SELECT \* FROM Produto

WHERE Tipo\_Prod NOT IN ('Sala','Quarto') --exibe os dados que NAO SEJAM os filtrados

SELECT \* FROM Produto

WHERE Tipo\_Prod <> 'Sala' AND Tipo\_Prod <> 'Quarto'

-- 17. A Clásula WHERE com o operador Between

SELECT \* FROM Produto

WHERE QTD\_Prod BETWEEN 200 AND 400 --mostra intervalo de valores.

SELECT \* FROM Produto

WHERE Qtd\_Prod >= 200 AND Qtd\_Prod <= 400 --contrario do BETWEEN

SELECT \* FROM Produto

WHERE QTD\_Prod NOT BETWEEN 200 AND 400

SELECT \* FROM Produto

WHERE Qtd\_Prod < 200 OR Qtd\_Prod > 400

--A Clasula WHERE com o operador LIKE

--18. Retornando dados em que o nome do produto comece com a letra P

SELECT \* FROM Produto

WHERE Nome\_Prod LIKE 'P%'--Pesquisa para texto, a posição do % significa o resto do texto , %<filtro>% (aparece tudo) | %<filtro>(filtro venha por ultimo) | <filtro>% (filtro venha por antes) | \_ significa ignorar caracteres, por exemplo se quiser que a terceira letra seja F colocar 2 "\_".

/\*

19. Retornando dados que o produto comece com a letra C, e tenha a letra a Z e

termine com qualquer Caractere

\*/

SELECT \* FROM Produto

WHERE Tipo\_Prod LIKE 'C%Z%'

/\*

20.Retornando dados em que no meio do nome do produto Tenha M

\*/

SELECT \* FROM Produto

WHERE Nome\_Prod LIKE '%M%'

/\*

21. Retornando dados que o nome e o tipo do produto tenha a silaba ar

\*/

SELECT \* FROM Produto

WHERE Nome\_Prod LIKE '%ar%'

AND Tipo\_Prod LIKE '%ar%'

/\*

22.Retornando dados em que o nome do produto não comece com a letra P

\*/

SELECT \* FROM Produto

WHERE Nome\_Prod NOT LIKE 'P%'

/\*\*\*\*\*\*\*\*\*Clásula ORDER BY\*\*\*\*\*\*\*\*\*\*\*\*\*/ --pode dar Lentidao

CREATE TABLE Material

(

Cod\_Mat int,

Tipo\_Mat char(1),

Nome\_Mat char(50),

Qtd\_Estq int,

Val\_Unit decimal(10,2)

)

INSERT Material VALUES(1,'A','Caderno Espiral',100,12.50)

INSERT Material VALUES(2,'A','Lápis Preto',500,1.50)

INSERT Material VALUES(3,'A','Lápis de Cor',600,7.50)

INSERT Material VALUES(4,'A','Caneta Azul',800,5.50)

INSERT Material VALUES(5,'B','Caneta Preta',150,5.50)

INSERT Material VALUES(6,'B','Caderno Broxura',200,10.00)

INSERT Material VALUES(7,'B','Borracha Macia',900,5.50)

INSERT Material VALUES(8,'B','Borracha Colorida',1000,1.50)

INSERT Material VALUES(9,'C','Apontador Colorido',2000,5.50)

INSERT Material VALUES(10,'C','Régua de Madeira',3500,9.00)

INSERT Material VALUES(11,'C','Régua de Plástico',2586,10.00)

INSERT Material VALUES(12,'C','Folhas de Papel',2587,5.50)

/\*

23. Escolhendo a ordenação pelo nome da coluna

\*/

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY Nome\_Mat --Para ser decrescente (ORDER BY Nome\_Mat DESC) DESC significa Decrescente.

/\*

24. Escolhendo a ordenação pela posição da coluna no select

\*/

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY 1 -- para mais de uma ordanação ORDER BY 1, 3 DESC

/\*

25. Escolhendo a ordenação pelos nomes das colunas

\*/

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY Nome\_Mat, Val\_Unit -- caso tenha material com o mesmo nome, ele ordena a segunda coluna

/\*

26. Escolhendo a ordenação pela posição da coluna no SELECT

\*/

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY 1,2

/\*

27. Clásula ORDER BY Crescente(ASC) ou Decrescente(DESC)

\*/

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY Nome\_Mat ASC -- ASC é ascendente, se nao colocar nada ASC sera o padrao

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY 3 ASC

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY Tipo\_Mat ASC, Val\_Unit ASC

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY 1 ASC, 2 ASC

SELECT Tipo\_Mat, Val\_Unit, Nome\_Mat FROM Material

ORDER BY 1 ASC, 2 DESC

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Clásula TOP\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SELECT TOP 5 \* FROM Material --Mostra os "x" primeiros (ou ultimos com DESC)

SELECT TOP 5 \* FROM Material

ORDER BY Val\_Unit ASC

-Clásula TOP WITH TIES com ORDER BY

SELECT TOP 5 WITH TIES \* FROM MATERIAL --WITH TIES "quebra" a regra do TOP, caso tenha mais itens que atendam a condição

ORDER BY Val\_Unit ASC

SELECT TOP 3 WITH TIES \* FROM Material

ORDER BY Val\_Unit ASC

SELECT TOP 2 WITH TIES \* FROM Material

ORDER BY Val\_Unit DESC

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*GROUP BY - Totalizando valores\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

Com a Clausula GROUP BY, podemos também utilizar:

\*/

select TOP 1

nome\_cli,

MAX(sal\_cli)

From Cliente

GROUP BY Nome\_Cli -- para agrupamento de registros

order by 2 desc

use BD\_1

select

isnull (CAST (Est\_func as varchar(100)), 'TOTAL') Estado, -- "isnull" bloqueia o nulo "NULL" | CAST - Converte a variavel

Count(Est\_func) Total

From Funcionario

Group by Est\_func

with rollup --faz a soma dos totais

-- Função de agregação

SUM() = Realizar Somas

COUNT() = Contagem de Registros

MAX() = Valor Máximo

MIN() = Valor Minimo

AVG() = Média

use TESTE

DROP TABLE Produto

Go

CREATE TABLE Produto

(

Cod\_Prod INT NOT NULL,

Tipo\_Prod CHAR(1) NOT NULL,

Linha\_Prod TINYINT NOT NULL,

Nome\_Prod VARCHAR(50) NOT NULL,

Qtd\_Prod INT NOT NULL,

Val\_Prod DECIMAL(10,2) NOT NULL

)

Go

INSERT Produto VALUES(1,'A',1,'Mesa',100,500.00)

INSERT Produto VALUES(2,'B',1,'Sofá',200,1500.00)

INSERT Produto VALUES(3,'A',2,'Mesa',100,200.00)

INSERT Produto VALUES(4,'A',2,'Armário',300,200.00)

INSERT Produto VALUES(5,'C',1,'Cama',500,1000.00)

INSERT Produto VALUES(6,'B',1,'Poltrona',100,250.00)

INSERT Produto VALUES(7,'B',2,'Sofá',200,300.00)

INSERT Produto VALUES(8,'C',2,'Cama',100,150.00)

INSERT Produto VALUES(9,'A',1,'Armário',200,800.00)

INSERT Produto VALUES(10,'C',1,'Guarda-Roupa',100,1500.00)

Go

SELECT \* FROM Produto

/\*

28.Agrupando por tipo de Produto, mas Fazendo a Somatória através da coluna Qtd\_Prod

\*/

SELECT

Tipo\_Prod,

SUM (Qtd\_Prod) AS Quantidade

FROM

Produto

GROUP BY Tipo\_Prod

/\*

29. Agrupando por Linha de Produto, mas Fazendo a Somatória através da coluna Qtd\_Prod

\*/

SELECT

Linha\_Prod,

SUM (Qtd\_Prod) AS Quantidade

FROM

Produto

GROUP BY Linha\_Prod

/\*

30.Exibindo A quantidade de Produtos em Estoque, separando-os por tipo e por linha da seguinte forma

\*/

SELECT

Tipo\_Prod,

Linha\_Prod,

SUM(Qtd\_Prod) AS Quantidade

FROM

Produto

GROUP BY Tipo\_Prod, Linha\_Prod -- agrupamento tem que colocar as colunas que nao contem função de agregação

/\*

31.Exibindo o Valor em estoque dos produtos, separando-os por tipo

\*/

SELECT

Tipo\_Prod,

SUM(Qtd\_Prod \* Val\_Prod) AS Quantidade

FROM

Produto

GROUP BY Tipo\_Prod

/\*

32. Exibindo o valor total em estoque dos produtos separando-os por tipo e por Linha, devendo agrupar

os produtos por Tipo e por Linha, em seguida somar a multiplicação da Quantidade de Produtos em estoque

pelo valor de cada Produto

\*/

SELECT

Tipo\_Prod,

Linha\_Prod,

SUM(Qtd\_Prod) AS Quantidade

FROM

Produto

GROUP BY Tipo\_Prod, Linha\_Prod

/\*

33. Exibindo a quantidade e o valor em estoquecdos produtos separando-os os por nome, agrupando

os produtos por nome e somar a quantidade dos mesmo, depois somar a multiplicação da quantidade

pelo valor destes mesmos produtos

\*/

SELECT

Nome\_Prod,

SUM(Qtd\_Prod) as [ Total da Quantidade ],

SUM(Qtd\_Prod \* Val\_Prod) AS Total

FROM

Produto

GROUP BY Nome\_Prod

//\*\*\*\*\*\*\*\*\*\*\*\*\*GROUP BY / HAVING\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

/\*

34.Tem a função de restringirem dados a serem exibidos pelo group by

\*/

SELECT

Tipo\_Prod,

SUM (Qtd\_Prod) AS Quantidade

FROM

Produto

GROUP BY Tipo\_Prod

HAVING SUM (Qtd\_Prod) > 500 -- HAVING (Filtro para ser utilizado nas agregações (junto ao Group By))

SELECT

Tipo\_Prod,

Linha\_Prod,

SUM (Qtd\_Prod) AS Quantidade

FROM

Produto

GROUP BY Tipo\_Prod, Linha\_Prod

HAVING SUM (Qtd\_Prod) > 300

SELECT

Tipo\_Prod,

Linha\_Prod,

SUM (Qtd\_Prod \* Val\_Prod) AS Total

FROM

Produto

GROUP BY Tipo\_Prod, Linha\_Prod

HAVING SUM (Qtd\_Prod \* Val\_Prod) > 80000.00

SELECT

Nome\_Prod,

SUM (Qtd\_Prod) AS Quantidade,

SUM (Qtd\_Prod \* Val\_Prod) AS Total

FROM

Produto

GROUP BY Nome\_Prod

HAVING SUM (Qtd\_Prod) >= 500

AND SUM (Qtd\_Prod \* Val\_Prod) > 220000.00

/\*\*\*\*\*\*\*\*\*\*\*GROUP BY / WITH ROLLUP\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

35.Tem a Função de gerar totais e Sub-Totais.

\*/

SELECT

Tipo\_Prod,

Linha\_Prod,

Nome\_Prod,

SUM (Qtd\_Prod) AS Quantidade,

SUM (Qtd\_Prod \* Val\_Prod) AS Total

FROM

Produto

GROUP BY Tipo\_Prod, Linha\_Prod, Nome\_Prod

WITH ROLLUP -- gera totais e subtotais

SELECT

Tipo\_Prod,

Linha\_Prod,

Nome\_Prod,

SUM (Qtd\_Prod) AS Quantidade,

SUM (Qtd\_Prod \* Val\_Prod) AS Total

FROM

Produto

GROUP BY Tipo\_Prod, Linha\_Prod, Nome\_Prod

WITH CUBE --gera totais e subtotais e mais subtotais

/\*

36.Inserindo um CASE sobre um select

\*/

Select

Nome\_cli,

CASE month(data\_cli)

WHEN 1 THEN 'JANEIRO'

WHEN 2 THEN 'FEVEREIRO'

WHEN 3 THEN 'MARÇO'

WHEN 4 THEN 'ABRIL'

WHEN 5 THEN 'MAIO'

WHEN 6 THEN 'JUNHO'

ELSE 'OUTRO MES'

END [MES REFERENCIA]

from Cliente

USE BD\_1

SELECT \* FROM Funcionario

Select

Nome\_func,

Case est\_func

WHEN 'RJ' THEN 'RIO DE JANEIRO'

WHEN 'SP' THEN 'SAO PAULO'

ELSE 'OUTRO ESTADO'

END ESTADO

FROM FUNCIONARIO

USE BD\_1

SELECT

Produto.Nome\_Prod AS [Produto],

[Tipo de Produto] = CASE TpProduto.Cod\_TpProduto

WHEN 1 THEN 'Produto Bom!!!'

WHEN 2 THEN 'Produto Reaproveitável'

ELSE 'Hummm'

END

FROM Produto INNER JOIN TpProduto

ON Produto.Cod\_TpProduto = TpProduto.Cod\_TpProduto

/\*

37. Funções de Data

\*/

SELECT \* FROM Cliente

WHERE month (DtNasc\_Cli) = 12

Day | year | Month

SELECT \* FROM Cliente

WHERE DtNasc\_Cli > GETDATE() --GETDATE() captura a data atual

/\*\*\*\*\*\*\*\*CONSULTADO MAIS DE UMA TABELA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

1. Selecionar o nome dos Clientes e o número de todos os telefones que cada um possui

\*/

SELECT

Cliente.Nome\_cli,

fone.Num\_fone

FROM Cliente INNER JOIN Fone --liga a tabela principal outra tabela.

ON Cliente.cod\_cli = Fone.cod\_cli --Campos que se relacionam (TABELA.CAMPO)

SELECT -- Esse modelo nao utiliza Inner Join

Cliente.Nome\_Cli,

Fone.Num\_Fone

FROM Cliente , Fone

WHERE Cliente.Cod\_Cli = Fone.Cod\_Cli

SELECT

Cliente.Nome\_Cli,

Fone.Num\_Fone

FROM Cliente LEFT JOIN Fone -- LEFT JOIN Mostra todos os cadastros da tabela do lado esquerdo (mostra clientes que nao tem cadastro na tabela Fone)

ON Cliente.Cod\_Cli = Fone.Cod\_Cli

SELECT

Cliente.Nome\_Cli,

Fone.Num\_Fone

FROM Cliente RIGHT JOIN Fone --RIGHT JOIN Mostra todos os cadastros da tabela do lado Direito

ON Cliente.Cod\_Cli = Fone.Cod\_Cli

SELECT

Cliente.Nome\_Cli,

Fone.Num\_Fone

FROM Cliente FULL JOIN Fone -- FULL JOIN Mostra todos os cadastros das tabelas do lado esquerdo e direito.

ON Cliente.Cod\_Cli = Fone.Cod\_Cli

SELECT

Cliente.Nome\_Cli,

Fone.Num\_Fone

FROM Cliente CROSS JOIN Fone --Cruza as informações para atribuindo todos os dados da tabela 1 para a tabela 2

select \* from funcionario

SELECT

Funcionario.Nome\_func AS FUNCIONARIO,

Dependente.Nome\_Dep AS DEPENDENTE

FROM Funcionario INNER JOIN Dependente

ON Funcionario.Cod\_Func = Dependente.Cod\_Func

--------DESAFIO--------

SELECT \* FROM PEDIDO

SELECT

Cliente.Nome\_cli AS Cliente,

Funcionario.Nome\_func AS Funcionario,

Pedido.ValTotal\_Ped AS Pedido

FROM Cliente INNER JOIN Pedido

ON Cliente.Cod\_Cli = Pedido.Cod\_Cli

INNER JOIN Funcionario

ON Pedido.Cod\_Func = Funcionario.Cod\_Func

/\*

2.Selecionar o nome dos clientes casados e o nome de seus conjuges.

\*/

SELECT Cliente.Nome\_cli,

Conjuge.nome\_conj

FROM Cliente INNER JOIN Conjuge

ON Cliente.cod\_cli = Conjuge.Cod\_cli

/\*

3.Selecionar o nome dos clientes, o número dos telefones e o tipo de telefone que

cada um possui.

\*/

SELECT

Cliente.Cod\_cli,

Cliente.Nome\_cli,

Fone.Num\_Fone,

TpFone.Nome\_TpFone

FROM Cliente INNER JOIN Fone

ON Cliente.Cod\_cli = fone.cod\_Cli

INNER JOIN TpFone

ON Fone.Cod\_TpFone = TpFone.Cod\_TpFone

/\*

4.Selecionar todas os campos da tabela pedido, o nome do cliente que fez o pedido

e o nome do funcionário que atendeu cada pedido.

\*/

SELECT

Pedido.\* ,

Cliente.Nome\_cli AS CLIENTE,

Funcionario.nome\_func FUNCIONARIO

FROM

pedido INNER JOIN Cliente

ON Pedido.Cod\_cli = Cliente.Cod\_cli

INNER JOIN Funcionario

ON Pedido.cod\_func = Funcionario.Cod\_func

/\*

5.Mostrar o nome dos funcionários e o nome de cada dependente que cada um tem

\*/

SELECT

Funcionario.Nome\_func AS [Funcionário],

Dependente.Nome\_Dep AS [Dependente]

FROM

Funcionario

INNER JOIN

Dependente

ON

Funcionario.Cod\_func = Dependente.Cod\_func

/\*

6. Mostrar todas as colunas da tabela pedido e o nome de cada produto vendido na

tabela pedido

\*/

SELECT

Pedido.\* ,

Produto.Nome\_prod

FROM

Pedido

INNER JOIN

Itens

ON

Pedido.Num\_ped = Itens.Num\_ped

INNER JOIN

Produto

ON

Itens.Cod\_prod = produto.cod\_prod

Order by 1

/\*

7.Mostrar o nome de cada produto e o nome do tipo que cada produto tem.

\*/

SELECT

Produto.Nome\_prod,

TpProduto.Nome\_TpProduto

FROM

Produto

INNER JOIN

TpProduto

ON

Produto.Cod\_TpProduto = TpProduto.Cod\_TpProduto

order by 2

/\*

8.Mostrar o nome dos funcionários e o valor total dos prêmios que cada um tem.

\*/

SELECT

Funcionario.Nome\_func AS [Funcionário],

SUM(Premio.Val\_Prem) AS [Total de prêmio]

FROM

Funcionario

INNER JOIN

Premio

ON

Funcionario.cod\_Func = Premio.Cod\_func

GROUP BY

Funcionario.Nome\_func

/\*

9.Mostrar o nome dos Clientes e o nome de cada produto que cada um comprou.

\*/

SELECT

Cliente.Nome\_cli,

Produto.Nome\_prod

FROM

Cliente

INNER JOIN

Pedido

ON

Cliente.Cod\_cli = Pedido.Cod\_cli

INNER JOIN

Itens

ON

Pedido.Num\_ped = Itens.Num\_Ped

INNER JOIN

Produto

ON

Itens.Cod\_prod = Produto.Cod\_prod

/\*

10.Mostrar o nome dos funcionários e o nome de cada produto que cada um vendeu.

\*/

SELECT

Funcionario.Nome\_func,

Produto.Nome\_prod

FROM

Funcionario

INNER JOIN

Pedido

ON

Funcionario.cod\_func = Pedido.Cod\_func

INNER JOIN

Itens

ON

Pedido.Num\_ped = Itens.Num\_ped

INNER JOIN

Produto

ON

Itens.cod\_prod = Produto.Cod\_prod

/\*

11.Mostrar os dados dos clientes que não tem telefone

\*/

SELECT \* FROM Cliente

WHERE

Cliente.Cod\_Cli NOT IN (SELECT Cod\_Cli FROM Fone) --faz o contrario do select entre parenteses

select \* from Funcionario

WHERE

Funcionario.Cod\_Func NOT IN (select Cod\_Func from Pedido)

/\*

12. Mostrar todos os clientes Solteiros

\*/

SELECT \* FROM Cliente

WHERE

Cliente.Cod\_Cli NOT IN (SELECT Cod\_Cli FROM Conjuge)

/\*

13.Mostrar o cliente que tem a maior renda

\*/

SELECT \* FROM Cliente

WHERE Renda\_Cli = (SELECT MAX(Renda\_Cli) FROM Cliente)

/\*

14. Mostrar o nome dos produtos que nunca foram vendidos.

\*/

SELECT Produto.Nome\_Prod

FROM Produto

WHERE Cod\_Prod NOT IN (SELECT Cod\_Prod FROM Itens)

/\*

15.Mostrar os funcionários que não têm dependentes.

\*/

SELECT \* FROM Funcionario

WHERE Cod\_Func NOT IN (SELECT Cod\_Func FROM Dependente)

/\*

16.Mostrar os Funcionários que não têm prêmio.

\*/

SELECT \*FROM Funcionario

WHERE Cod\_Func NOT IN (SELECT Cod\_Func FROM Premio)

/\*

17.Mostrar os Clientes solteiros que nuca fizeram nenhum pedido.

\*/

SELECT \* FROM Cliente

WHERE

Cod\_Cli NOT IN (SELECT Cod\_Cli FROM Conjuge)

AND Cod\_Cli NOT IN (SELECT Cod\_Cli FROM Pedido)

/\*

18.Mostrar os funcionários que não têm dependentes e o que tem o menor salário.

\*/

ALTER VIEW V\_Consulta

WITH ENCRYPTION -- ENCRYPTOGRAFA AS VIEWS

as

SELECT \* FROM Funcionario

WHERE Cod\_Func NOT IN (SELECT Cod\_Func FROM Dependente)

AND Sal\_Func = (SELECT MIN(Sal\_Func) FROM Funcionario)

SELECT \* FROM V\_Consulta

SP\_HELPTEXT 'V\_CONSULTA' -- MOSTRA O FONTE DA VIEW

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*CRIANDO AS VIEWS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

-- Visualizando codigo de Objetos

Select \* from Syscomments

/\*

2. Criar uma view chamada V\_FunSemDep que mostre os Funcionários que não têm dependentes.

\*/

CREATE VIEW V\_FuncSemDep

WITH ENCRYPTION

AS

SELECT Cod\_Func,Nome\_func,DtCad\_Func FROM Funcionario

WHERE Cod\_Func NOT IN (SELECT Cod\_Func FROM Dependente)

SELECT \* FROM V\_FuncSemDep

/\*

3. Criar uma view chamada V\_CliSolteiro que mostre todos os clientes solteiros.

\*/

CREATE VIEW V\_CliSolteiro

WITH ENCRYPTION

AS

SELECT Nome\_cli FROM dbo.cliente

WHERE cod\_cli NOT IN (SELECT Cod\_cli FROM dbo.Conjuge)

SELECT \* FROM V\_CliSolteiro

/\*

4.Criar uma view chamada V\_CliConj que mostre o nome dos clientes e de seus cônjuges

\*/

CREATE VIEW V\_CliConj

AS

SELECT Nome\_cli FROM Cliente

WHERE Cod\_Cli IN (Select Cod\_cli From Conjuge)

select \* from V\_CliConj

/\*

5. Criar uma view chamada V\_CliPedFunc que mostre o código e o nome dos clientes, o código e nome dos funcionários, o número, a data e o valor total dos pedidos.

\*/

CREATE VIEW V\_CliPedFunc

AS

SELECT

Cliente.Cod\_cli AS [Código Cliente],

Cliente.Nome\_cli AS [Nome do Cliente],

Funcionario.Cod\_func AS [Código Funcionário],

Funcionario.Nome\_func AS [Nome do Funcionário],

Pedido.Num\_ped AS [Número],

Pedido.Dt\_Ped AS [Data do pedido],

Pedido.ValTotal\_Ped AS [Total]

FROM Pedido

INNER JOIN

Cliente

ON Pedido.Cod\_cli = Cliente.Cod\_cli

INNER JOIN

Funcionario

ON Pedido.Cod\_Func = Funcionario.cod\_func

SELECT \* FROM V\_CliPedFunc

/\*

6.Criar uma view chamada V\_PremioFunc que mostre o código e o nome do funcionários, a data e o valor dos prêmios que cada um recebeu.

\*/

CREATE VIEW V\_PremioFunc

AS

SELECT

Funcionario.Cod\_func AS [Código],

funcionario.Nome\_func AS [Nome],

Premio.Dt\_Prem AS [Data do recebimento],

Premio.Val\_Prem AS [Valor]

FROM Funcionario

INNER JOIN Premio

ON Funcionario.Cod\_func = Premio.Cod\_func

SELECT \* FROM V\_PremioFunc

where campo\_da\_data BETWEEN now( ) AND DATE\_ADD( now( ) , INTERVAL +7 DAY )