No final tem alguns exemplos, mas não todos, estudem os arquivos feitos em aula também.

Bom Estudo

2 Consultas Básicas

2.1 SELECT...FROM

A clausula *select* é obrigatoriamente seguida de uma clausula *from*. Enquanto a primeira cláusula define quais os dados desejados, a segunda indica de onde serão obtidos.
A fonte de dados pode ser uma tabela, uma view ou uma UDF.
Como exemplo de um comando para seleção de dados, veja a instrução e o resultado:

```
SELECT au_id,au_lname,au_fname,phone,city,state
  FROM Authors

au_id       au_lname                                 au_fname             phone        city                 state
----------- ---------------------------------------- -------------------- ------------ -------------------- -----
172-32-1176 White                                    Johnson              408 496-7223 Menlo Park           CA
213-46-8915 Green                                    Marjorie             415 986-7020 Oakland              CA
238-95-7766 Carson                                   Cheryl               415 548-7723 Berkeley             CA
267-41-2394 O'Leary                                  Michael              408 286-2428 San Jose             CA
274-80-9391 Straight                                 Dean                 415 834-2919 Oakland              CA
341-22-1782 Smith                                    Meander              913 843-0462 Lawrence             KS
...
(23 row(s) affected)
```

O *select* acima não ordenou os dados nem alterou qualquer característica original, mas trouxe apenas as colunas que foram definidas na lista.
Na tabela *authors* da *pubs* (diagrama 10.1) existem outras colunas que não apenas as seis utilizadas nesta consulta, bem como a ordem das colunas na tabela não afetam em nada a ordem em que um *select* solicita o retorno destes dados.
Neste caso utilizamos o database *pubs*, e para isso precisamos definir o database que será utilizado. Podemos definir o database de duas formas, a primeira é utilizando o caminho completo, como a seguir:

```
SELECT au_id,au_lname,au_fname,phone,city,state
  FROM Server.Pubs.dbo.Authors
```

Neste caso definimos o nome do servidor, o database desejado e o esquema ou usuário que criou a tabela. Como padrão o esquema é sempre *dbo* na maior parte dos RDBMS.
A segunda forma de definir o banco de dados é utilizando o comando *Use Pubs*. Não é necessário utilizar este comando em todos os *select* como no exemplo anterior, uma vez que o *use* é um comando de sessão, ou seja, ao abrir a conexão com o servidor enviamos o *use* e não mais precisamos definir o database no *from*.

WHERE

Esta clausula indica as condições da consulta.
As condições melhoram a performance por permitir que o *engine* do banco de dados utilize os índices para organizar a forma de procura, não fazendo método bolha e sim de ponteiro.
Abordaremos índices em detalhes no módulo 9.
Segue a lista operadores, função e exemplo:

Operador	Função	Exemplo
=	Altera valor de variáveis CTS e compara igualdade	Idade = 30
<>	Idade diferente de 30	Idade <> 30
> e >=	Maior (31 para frente) e maior ou igual (inclui o 30)	Idade > 30 Idade >= 30
< e <=	Menor (29 para baixo) e menor ou igual (inclui o 30)	Idade < 30 Idade <= 30
Between	Idade que esteja entre 50 e 90 inclusive	Idade Between 50 And 90
Not	Negação de uma condição	Idade Not Between 50 And 90
Is	Compara igualdade entre tipos	Nome is Null
And	Significa que as comparações devem ser todas verdadeiras	Idade = 30 And Nome = "Fulano"
Or	Significa apenas uma das comparações precisa ser verdadeira	Idade = 30 Or Nome = "Fulano"

Alguns exemplos de *where* podem ser vistos abaixo:

```
--UTILIZAÇÃO DE NUMERICOS
Use Northwind
SELECT *
  FROM PRODUCTS
 WHERE UNITPRICE BETWEEN 10 AND 12
SELECT *
  FROM PRODUCTS
 WHERE UNITPRICE NOT BETWEEN 10 AND 12
SELECT *
```

LIKE (Transact-SQL)

Determina se uma cadeia de caracteres específica corresponde a um padrão especificado. Um padrão pode incluir caracteres normais e curingas. Durante a correspondência de padrões, os caracteres normais devem corresponder exatamente aos caracteres especificados na cadeia de caracteres. No entanto, os caracteres curinga podem ser correspondidos a fragmentos arbitrários da cadeia de caracteres. O uso de caracteres curinga torna o operador LIKE mais flexível que o uso dos operadores de comparação de cadeias de caracteres = (iguais) e NOT LIKE para != (diferentes). Se qualquer um dos argumentos não for do tipo de dados de cadeia de caracteres, o Mecanismo de Banco de Dados do SQL Server o converterá no tipo de dados de cadeia de caracteres, se possível.

Caractere curinga	Descrição	Exemplo
%	Qualquer cadeia de zero ou mais caracteres.	WHERE title LIKE '%computer%' localiza todos os títulos de livro com a palavra 'computer' em qualquer lugar no título do livro.
_ (sublinhado)	Qualquer caractere único.	WHERE au_fname LIKE '_ean' localiza todos os nomes de quatro letras que terminam com ean (Dean, Sean e assim por diante).
[ ]	Qualquer caractere único no intervalo ([a-f]) ou conjunto ([abcdef]) especificado.	WHERE au_lname LIKE '[C-P]arsen' localiza os sobrenomes de autores que terminem com arsen e que comecem com qualquer caractere único entre C e P, por exemplo, Carsen, Larsen, Karsen e assim por diante. Em pesquisas de intervalo, os caracteres incluídos no intervalo podem variar de acordo com as regras de classificação do agrupamento.

Filtragem de Strings

Para filtrar strings temos uma situação e instrução própria, uma vez que números tem um valor absoluto e textos não. Por exemplo, eu procuro pela idade de 31 anos exatamente, mas nem sempre sei o nome exato de uma pessoa.

A instrução like tem como diferencial o uso dos coringas “%” para qualquer coisa na substituição (similar ao “*” do DOS), “[“ e “]” para definir range. Os exemplos abaixo definem bem estas regras:

```
USE NORTHWIND

SELECT *  FROM CUSTOMERS  WHERE CustomerID='ANTON' --Nome Anton é o unico que será encontrado

SELECT *  FROM CUSTOMERS  WHERE COMPANYNAME LIKE '[B-E]%' --Iniciem entre B e E

SELECT *  FROM CUSTOMERS  WHERE COMPANYNAME LIKE '[BMF]%' --Iniciem em B, M ou F

SELECT *  FROM CUSTOMERS  WHERE CONTACTTITLE LIKE '%ETI%' --Contenham ETI em qualquer lugar

SELECT *  FROM CUSTOMERS  WHERE CONTACTTITLE LIKE '%ETI%ASS%' --Contenham ETI e ASS

SELECT *  FROM CUSTOMERS  WHERE CONTACTNAME LIKE 'L[AI]%' --Iniciem em L e a segunda letra A ou I
```

Operador IN

Este operador é uma alternativa a uma extensa lista de or, uma vez que sua utilização é definir uma lista dos valores numéricos ou strings desejados.

Veja abaixo a comparação entre um comando com or e com in:

```
SELECT *  FROM CUSTOMERS  WHERE COUNTRY IN ('BRAZIL','ARGENTINA')

SELECT *  FROM CUSTOMERS  WHERE COUNTRY='BRAZIL' OR COUNTRY='ARGENTINA'
```

A diferença entre os dois operadores fica mais evidente se fossem dez paises, por exemplo, onde o comando com or seria muito maior e de entendimento mais complicado do que o comando com in.

ALIAS

Alias são literalmente apelidos, pois algumas colunas em um select podem ser a soma de outras colunas, resultando em uma nova coluna. Esta nova coluna por padrão recebe o nome Expr_000x e podemos renomear com a instrução as.

```
SELECT COMPANYNAME, REGION+'/'+COUNTRY AS LOCAL  FROM CUSTOMERS
```

```
SELECT COMPANYNAME, LOCAL=REGION+'/'+COUNTRY  FROM CUSTOMERS
```

Nos dois exemplos acima região foi concatenado ao pais para formar uma nova coluna, que chamamos de local. Note que tanto podemos usar o as quanto colocar o nome desejado e o sinal de igual com o conteúdo.

Alias também pode ser utilizado em tabelas, onde após o nome da tabela colocamos o as seguido do nome referencial. É muito utilizado nos joins tratados no módulo 4.

GROUP BY (Transact-SQL)

Agrupa um conjunto de linhas selecionadas em um conjunto de linhas de resumo pelos valores de uma ou mais colunas ou expressões no SQL Server.

Uma linha é retornada para cada grupo. As funções de agregação na lista de <seleção> da cláusula SELECT fornecem informações sobre cada grupo em vez de linhas individuais.

Para compreendermos melhor o uso do **GROUP BY**, considere a tabela **Produtos** (lembrando que os valores associados aos nomes são fictícios e são apenas usados para este exemplo):

Id	Nome	Fabricante	Quantidade	VlUnitario	Tipo
1	Playstation 3	Sony	100.00	1999.00	Console
2	Core 2 Duo 4GB Ram 500GB HD	Dell	200.00	1899.00	Notebook
3	Xbox 360 120GB	Microsoft	350.00	1299.00	Console
4	GT-I6220 Quad Band	Samsung	300.00	499.00	Celular
5	iPhone 4 32GB	Apple	50.00	1499.00	Smartphone
6	Playstation 2	Sony	100.00	399.00	Console
7	Sofá Estofado	Coréia	200.00	499.00	Sofá
8	Armário de Serviço	Aracaju	50.00	129.00	Armário
9	Refrigerador 420L	CCE	200.00	1499.00	Refrigerador
10	Wii 120GB	Nintendo	250.00	999.00	Console

Vamos supor que desejemos obter o número de produtos em estoque, agrupados pelo tipo, para que depois seja feita a soma da quantidade existente em cada um dos grupos. Para isso usamos a função **SUM()** em conjunto com o **GROUP BY**, como a instrução a seguir nos mostra:

```
SELECT Tipo, SUM(Quantidade) AS 'Quantidade em Estoque'
      FROM Produtos
      GROUP BY Tipo
```

Executando a instrução acima temos o seguinte resultado:

Tipo	Quantidade em Estoque
Armário	50.00
Celular	300.00
Console	800.00
Notebook	200.00
Refrigerador	200.00
Smartphone	50.00
Sofá	200.00

Podemos também contar o número de produtos em estoque de acordo com os fabricantes disponíveis. Assim como no exemplo anterior, mais agora levando em conta os fabricantes, os produtos devem ser agrupados por eles, para que depois sejam contabilizados os produtos em relação a essa divisão. Para isso, devemos usar a seguinte instrução:

```
SELECT Fabricante, SUM(Quantidade) AS 'Quantidade em
Estoque'
  FROM Produtos
 GROUP BY Fabricante
```

Assim temos o seguinte resultado:

Fabricante	Quantidade em Estoque
Apple	50.00
Aracaju	50.00
CCE	200.00
Coréia	200.00
Dell	200.00
Microsoft	350.00
Nintendo	250.00
Samsung	300.00
Sony	200.00

Agora vamos somar a quantidade de produtos em estoque de acordo com os tipos e fabricantes disponíveis. Primeiro, será agrupados os produtos de acordo com os tipos e fabricantes, para que depois seja feita a soma de cada um desses grupos (essa é a ordem que o **SQL Server** faz logicamente, mais nossa instrução segue a ordem inversa). Confira como fazer isso a seguir:

```
SELECT Tipo, Fabricante, SUM(Quantidade) AS 'Quantidade em
Estoque'
  FROM Produtos
 GROUP BY Tipo, Fabricante
```

Assim teremos este resultado:

Tipo	Fabricante	Quantidade em Estoque
Smartphone	Apple	50.00
Armário	Aracaju	50.00
Refrigerador	CCE	200.00
Sofá	Coréia	200.00
Notebook	Dell	200.00
Console	Microsoft	350.00
Console	Nintendo	250.00
Celular	Samsung	300.00
Console	Sony	200.00

Podemos também obter o valor total dos produtos em estoque, agrupados por tipo. Veja como:

```
SELECT Tipo, SUM(Quantidade * VlUnitario) AS 'Valor do
Estoque'
  FROM Produtos
 GROUP BY Tipo
```

Temos o seguinte resultado:

Tipo	Valor do Estoque
Armário	6450.0000
Celular	149700.0000
Console	944200.0000
Notebook	379800.0000
Refrigerador	299800.0000
Smartphone	74950.0000
Sofá	99800.0000

Deixo como dica para vocês experimentarem fazer outras combinações com o **GROUP**

BY em conjunto com as **funções** descritas anteriormente, visando assim treinar os conceitos apresentados até aqui.

Outros exemplos com Group By

```
/********************
GROUP BY

o group by serve para agrupar os valores, eh necessario usar uma funcao
de Agregaçãoe caso mostre uma coluna de legenda esta deve ser a mesma
utilizada para o agrupamento.
********************/
select * from products

--selecione os produtos para exibir a quantidade
--de tipos de produtos  agrupando  por codigo
--da categoria
select COUNT(*) from products group by CategoryID
select * from customers
--o problema no exemplo anterior é que esta
--mostrando apenas os valores
select CategoryID,COUNT(*) from
                        products group by CategoryID

select * from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID

select Categoryname,COUNT(*)
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID
                                    group by Categoryname

-- contar a quantidade de clientes por pais
select country,COUNT(*) from Customers
                  group by country

-- contar a quantidade de pedidos de cada  pais
select country,COUNT(*) as 'total'  from Customers as c
                                    inner join  orders as o
                                    on c.CustomerID = o.CustomerID
                  group by country
                  order by total desc

--quero a quantidade de pedidos que cada
-- funcionario vendeu


select FirstName ,title,COUNT(*) as 'total'
                  from orders as o inner join Employees as e
                   on  o.EmployeeID = e.EmployeeID
                  group by FirstName ,title     order by total desc

--contar a quantidade de produtos de cada fornecedor
```

```sql
select CompanyName,COUNT(*)
                  from products as p
                                    inner join Suppliers  as s
                                    on  p.SupplierID  = s.SupplierID
            group by CompanyName

--quantidade de pedidos por clientes
select CompanyName ,COUNT(*) as 'total'
                  from Customers as c
                                    inner join  orders as o
                                    on c.CustomerID = o.CustomerID
                  group by CompanyName
                  order by total desc

select Categoryname,sum(UnitsInStock)
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID
                                    group by Categoryname

--somando a quantidade de produtos
--no estoque por categoria
select Categoryname,sum(UnitsInStock)
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID
                                    group by Categoryname

/* a media dos precos dos
produtos por categoria*/

select Categoryname,avg(Unitprice)  as 'media'
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID

                                    group by Categoryname

/* a media dos precos dos
produtos por categoria
apenas dos produtos com codcategoria maior que 5 */
select Categoryname,avg(Unitprice)  as 'media'
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID
                         where c.CategoryID >5
                                    group by Categoryname

/* a media dos precos dos
produtos por categoria*/
select Categoryname,avg(Unitprice)  as 'media'
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID
                        group by Categoryname
                              having avg(Unitprice)>=30



select avg(Unitprice)  as 'media'
            from  Categories as c  inner join products as p
                        on p.CategoryID = c.CategoryID
```

```
                                where avg(Unitprice)>=30
                            group by Categoryname

/*****************************************
--quantidade de pedidos por funcionarios
--(usando todos os pedidos)
select FirstName ,COUNT(*) as 'total'
                    from Employees  as e
                                        inner join  orders as o
                                        on e.EmployeeID  = o.EmployeeID
                    group by FirstName

--quantidade de pedidos por funcionarios
--(usando os pedidos de 2016)
select FirstName ,COUNT(*) as 'total'
                    from Employees  as e
                                        inner join  orders as o
                                        on e.EmployeeID  = o.EmployeeID
      where OrderDate between '01/01/2016' and '31/12/2016'
                    group by FirstName

--quantidade de pedidos por funcionarios
--(usando os pedidos de 2016) com quantidade de pedidos
--maior ou igual a 50
select FirstName ,COUNT(*) as 'total'
                    from Employees  as e
                                        inner join  orders as o
                                        on e.EmployeeID  = o.EmployeeID
      where OrderDate between '01/01/2016' and '31/12/2016'
                    group by FirstName
            having COUNT(*) >= 50
```

Funções de manipulação de strings:

ASCII(string) = pega o valor em ASCII da string

CHAR(integer) = troca inteiro do ASCII em um caracter

LEN(string) = Identifica o comprimento de uma expressão em caracteres

LOWER(string) = converte uma string uppercase para lowercase.

LTRIM(string) = remove os espaços em branco

REPLICATE(string, integer) = Repete N vezes um caractere especificado

REVERSE(string) = retorna o inverso de uma expressao

RTRIM (string) = remove os espaços em branco à direita de uma string

SPACE(integer) = que retorna o número de espaços em branco informados no parâmetro

SUBSTRING(string texto, posicao_inicial, tamanho) = retorna uma string com o comprimento

definido em "tamanho" extraída da string "texto", a partir da "posicao_inicial"

UPPER(string) = retorna string em maiúsculas

Funções de manipulação de data/hora:

Nomes dos intervalos das partes de data:

```
/* tabela de intevalos

 dd - dia
 mm - mês
 yy - ano

 hh - hora
 mi - minuto
 ss - segundo
 ms - milisegundos
 mcs- microsegundo
 ns - nanosegundos

 qq - trimestre
 dy - num do dia do ano
 dw - num do dia da semana
 ww - número da semana
 */
```

DATEADD (parte, numero, data) = adiciona um valor a parte de uma data

DATEDIFF (parte, data inicial, data final) = subtrai a data inicial da data final, indicando o resultado na unidade definida em "parte"

GETDATE() = retorna a data atual do sistema

DATENAME (parte, data) = retorna o nome da parte de uma data

DATEPART(parte, data) = retorna a parte de uma data

DAY (date)

Retorna um inteiro que representa a parte do dia da date especificada.

MONTH (date)

Retorna um inteiro que representa a parte do mês de uma date especificada.

YEAR (date)

Retorna um inteiro que representa a parte do ano da date especificada.

Funções Matemáticas:

ABS(numero) = retorna o valor absoluto do número

PI() = retorna o valor de PI 3.1415926535897931.

POWER(numero, potencia) = retorna o valor elevado à potencia informada

RANDON(expressao) = um número aleatório entre 0 e 1. Expressão é opcional e será usada como semente da cadeia pseudo-aleatória

SIGN(numero) = retorna sinal positivo, negativo ou zero do numero

SQRT(float) = retorna a raiz quadrada de um número

SQUARE(float) = retorna o quadrado de um número

ROUND(número, precisão, arredonda_ou_trancar) = arredonda ou tranca o número fornecido de acordo com a precisão informada. Se o terceiro parâmetro não for passado para a função, o número é arredondado. Se quiser que o número seja truncado, deve-se fornecer o valor 1

CEILING - Arredonda para baixo retornando o valor inteiro do número e descartando a parte decimal.

FLOOR - Arredonda para cima retornando o valor do próximo inteiro do número e descartando a parte decimal.

Por exemplo, considerando uma expressão numérica 12.9273, CEILING retorna 13 e FLOOR retorna 12. O valor de retorno de FLOOR e CEILING tem o mesmo tipo de dados como a expressão numérica de entrada.

SELECT CEILING(12.9273);

SELECT FLOOR(12.9273);

Abaixo segue alguns exemplos, mas lembrem-se, aqui nao tem nem metade dos exemplos abordados nas aulas, portanto estudem os 6 ou 7 arquivos de aulas.

```
seleta * from Categories      --categorias

select * from products        --produtos

select * from suppliers       --fornecedores

select * from Orders          --pedido

select * from [Order Details] --detalhes do pedido

select * from Customers       --clientes

select * from Employees     --empregados


--seleciona o produto de codigo 20
select * from Products where ProductID=20

--seleciona o produto de codigo 35
select * from products where ProductID=35

--selecionar o produto de nome 'Chai'
select * from products where productname='chai'

--selecionar o produto de preco unitario igual a 20
select * from products where UnitPrice=20

--selecionar o fornecedor de codigo 20
select * from suppliers where SupplierID=20

--exiba os detalhes do pedido, que o codigo do pedido seja 10250
select * from  [order details]   where OrderID=10250

----exiba os clientes, que residem na 'france'
select * from  Customers  where  Country ='france'

----exiba os clientes, que residem na cidade de  'paris'
select * from  Customers  where   city='paris'

----exiba os pedidos , que a data do pedido seja '08/20/1996'
select * from  orders   where  OrderDate = '08/20/1996'

--exiba os pedidos , cuja o codigo do cliente  seja 'anton'
select * from  orders   where  CustomerID = 'anton'

/*
operadores relacionais

=     comparação
<>    diferente
>     maior
```

```
<     menor
>=    maior ou igual que
<=    menor ou igual que

*/

-- selecionar todos os produtos com preco menor ou igual a 20
select * from Products where  UnitPrice <=20

--selecionar todos os produtos com preco menor ou igual a 10
select * from Products where  UnitPrice <=10

--selecionar os produtos com unid. em estoque menor ou igual a 50
select * from Products where  Unitsinstock <=50

--selecionar  os produtos com preco maior que 100
select * from Products where  UnitPrice >100

/*
operadores lógicos
and         --> retorna verdadeiro se todas as condições
                forem verdadeiras
or          --> retorna verdadeiro se ao menos uma condição
                for verdadeira
não     --> inverte o resultado lógico da condição
*/


-- todos os precos de produtos que sejam  maior ou igual a 20 e
-- menor ou igual a 30


select * from Products where  UnitPrice >=20  and unitprice<=30


-- todos os produtos que tenham a quantidade no estoque
--  maior ou igual a 80 e  menor ou igual a 100

select * from Products where  UnitsInStock >=80
                                    and UnitsInStock<=100

select * from orders where  orderdate >='08/01/1997'
                                          and
orderdate<='08/31/1997'

--

select * from  orders   where  ShippedDate > RequiredDate

select * from Products where  UnitsInStock >=80
                                    and UnitsInStock<=100

select * from Products where  UnitsInStock
                              between 80 and 100

select * from orders where  orderdate >='08/01/1997'
                                          and
```

```
orderdate<='08/31/1997'

select * from orders where  orderdate
                            between '08/01/1997'  and  '08/31/1997'

-- todos os empregados que o firstname  comece com a
--letra 'm'
select * from Employees where FirstName like 'm%'

-- todos os empregados que o primeiro nome  termine com a
--letra 't'
select * from Employees where FirstName like '%t'

-- todos os empregados que o primeiro nome  contenha as
--letras 'll'
select * from Employees where lastName like '%ll%'

select * from products where ProductName like 'louisiana%'

-- todos os produtos que o  nome  contenha a
--palavra 'rodney'

select * from products where ProductName like '%rodney%'


-- todos os Clientes  que o  nome da companhia  comece com
--a letra 'A'
select * from Customers where companyname like 'A%'

-- todos os Clientes  que o  nome da companhia  contenha a
--palavra 'gourmet'
select * from Customers where companyname like '%gourmet%'

   luis da silva%

-- todos os Clientes  que o  nome da companhia  contenha a
--palavra 'deli_atessen'
select * from Customers
              where companyname like '%deli_atessen%'

-- todos os Clientes  que o  nome da companhia  comece com as letras
--de "A" até  "D"
select * from Customers
              where companyname like '[a-d]%'


/***************************
           INNER JOIN
****************************/

select * from products as p INNER JOIN Categories as c
                on p.categoryid = c.categoryid

select * from Customers  as cus INNER JOIN orders as o
                on cus.CustomerID  = o.CustomerID
```

```sql
select * from products as p INNER JOIN Suppliers as s
                            on p.SupplierID = s.SupplierID
                          INNER JOIN Categories as c
                            on p.categoryid = c.categoryid
                          INNER JOIN [Order Details]as od
                            on p.ProductID = od.ProductID


--customers --- orders --- employeers --- [order details]



select * from Customers  as cus INNER JOIN orders as o
                            on cus.CustomerID  = o.CustomerID
                          INNER JOIN Employees as e
                            on e.EmployeeID= o.EmployeeID
                          INNER JOIN [Order Details]as od
                            on od.OrderID  = o.OrderID
                          Inner join Products as p
                            on p.ProductID=od.ProductID
                          Inner join Suppliers as s
                            on s.SupplierID=p.SupplierID
                          inner join Categories as c
                            on c.CategoryID=p.CategoryID

/*****************************************
            GROUP BY
*****************************************/
-- quais sao as 5 principais funcoes de agregação?
-- count, avg, sum, max , min
regras gerais
1. Sempre deve ter uma funcao de agregação
2. Só pode usar como legenda a coluna usada
   para o agrupamento
*/

select COUNT(*) from Customers    group by country
select country, COUNT(*) from Customers    group by country

--faca a contagem de clientes por contacttitle (cargo)

select ContactTitle , COUNT(*) from Customers    group by ContactTitle

-- ordernar pela contagem
select ContactTitle , COUNT(*) as 'valor' from Customers
                                    group by ContactTitle
                                    order by valor desc
-- qual a quantidade de pedidos por cliente
-- qntos pedidos foram feitos por cada cliente

select companyname , COUNT(*) as 'quantidade'
                  from orders as o inner join Customers as cus
                    on o.CustomerID=cus.CustomerID
                  group by companyname
```

```sql
select companyname,contactname  , COUNT(*) as 'quantidade'
                  from orders as o inner join Customers as cus
                                on o.CustomerID=cus.CustomerID
                        group by companyname,contactname



select   country from customers
select distinct  country from customers

select   country,city from customers
select distinct  country,city from customers


--qual a quantidade que cada produto foi vendido nos pedidos
--quantas vezes o produto foi vendido
select * from [Order Details]

select COUNT(*) from [Order Details] group by productid

--exibindo o cod do produto como legenda
select productid,COUNT(*) from [Order Details] group by productid

--colocando o nome do produto como legenda
select productname,COUNT(*) from [Order Details] as od
            INNER JOIN Products as p
                  ON  od.ProductID = p.ProductID
                  group by productname

--*****************************************************
--qual a soma da quantidadede cada produto que ja foi vendido
--qual os produtos mais vendidos

select od.productid,productname,SUM(quantity) as 'total' from [Order
Details] as od
                  INNER JOIN Products as p
                        ON  od.ProductID = p.ProductID
                        group by od.productid,productname
                        order by total desc



--*****************************************************
--qual a soma dos preços cada produto que ja foi vendido
--qual o valor das vendas por produto
select productname,SUM(od.unitprice*quantity ) as 'total' from [Order
Details] as od
                  INNER JOIN Products as p
                        ON  od.ProductID = p.ProductID
                        group by productname
                        order by total desc


--*****************************************************
--qual a soma dos preços cada produto que ja foram vendidos em 2017
--qual o valor das vendas por produto vendidos em 2017
```

```sql
select productname,SUM(od.unitprice *quantity ) as 'total' from [Order
Details] as od
                  INNER JOIN Products as p
                        ON  od.ProductID = p.ProductID
                  INNER JOIN orders as o
                      ON od.OrderID= o.OrderID
            where OrderDate between '01/01/2017' and '31/12/2017'
                              group by productname
                              order by total desc


--*****************************************************
--qual a soma dos preços dos produtos  vendidos por categoria
--qual o valor das vendas por categoria

select categoryname,SUM(od.unitprice *quantity ) as 'total' from [Order
Details] as od
                  INNER JOIN Products as p
                        ON  od.ProductID = p.ProductID
                  INNER JOIN Categories  as c
                        ON  c.CategoryID  = p.CategoryID

                              group by categoryname
                              order by total desc



--*****************************************************
--qual a quant de  categoria  vendidos por clientes

select categoryname,CustomerID,COUNT(*) as 'total' from [Order Details]
as od
                        INNER JOIN Products as p
                        ON  od.ProductID = p.ProductID
                        INNER JOIN Categories  as c
                        ON  c.CategoryID  = p.CategoryID
                        INNER JOIN Orders  as o
                        ON  o.OrderID   = od.OrderID

--Opcional para reduzir registros de clientes
                        where CustomerID like 'a%'

                        group by categoryname,CustomerID
                        order by CustomerID asc,total desc


/*Qual o valor das vendas por empregado */
select FirstName,lastname ,SUM(od.unitprice *quantity ) as 'total'
                  ,SUM(od.unitprice *quantity )*0.02 as 'bonus'
            from [Order Details] as od
                  INNER JOIN Orders  as o
                        ON  o.OrderID   = od.OrderID
                  INNER JOIN Employees as e
                         ON e.EmployeeID=o.EmployeeID
                              group by FirstName,lastname
                              order by total desc
```

```
/*
Qual o valor das vendas por clientes
quanto cada cliente gastou na minha loja
*/

select companyname ,SUM(od.unitprice *quantity ) as 'total'
            from [Order Details] as od
                  INNER JOIN Orders  as o
                        ON  o.OrderID   = od.OrderID
                  INNER JOIN Customers as c
                        ON c.CustomerID=o.CustomerID
            group by companyname    order by total desc

--***********************************************
--funcoes matematicas

--retorna o numero sem o sinal
select ABS(5)
select ABS(-5)

--retorna o arcCosseno de um numero
select ACOS(0.5)

--retorna o sinal de um numero,
--sendo: -1 para negativo, 1 para positivo e 0 para zero
select sign(5)
select sign(0)
select sign(-5)

--retorna o valor
select PI()
--calcular a area de um circulo 5 cm
select PI()*5*5

-- calcula a raiz quadrada
select SQRT(400)
-- retorna o quadrado de um numero
select SQUARE(8)
select PI()*square(5)
--retorna um numero elevado a outro
select POWER(2,10)

--retorna valor aleatorio
select RAND()
select floor(RAND()*6+1)

--arredondamento
--floor - arredonda para baixo retornando numero inteiro
select FLOOR(2.2)
select FLOOR(2.456)
select FLOOR(2.00001)
select FLOOR(2.999999999)

--ceiling - arredonda para cima  retornando numero inteiro
select ceiling(2.2)
select ceiling(2.456)
select ceiling(2.00001)
```

```sql
select ceiling(2.999999999)

--round   - arredonda seguindo as regras matematicas ,
--                e pode -se escolher a quantidade de casas decimais

select round(2.823645,5)
select round(2.823645,1)
select round(2.823645,3)
select round(2.8236555,4,1)
select round(2.823645,0,5)

select UnitPrice,
       UnitPrice *1.05 as 'Aumenta 5%',
       ROUND(UnitPrice *1.05,2) as 'Aumenta 5% e arred',
       sqrt(UnitPrice) as 'raiz'

                                    from products

--*****************************************
--funcoes de String
--*****************************************
--retorna quantidade de caracteres
select  LEN('ivan carlos pavao')

retorna o codigo do caracter na tabela ascii
select ASCII('a')
select ASCII('A')
select ASCII('3')

retorna o caracter da tabela ascii de um numero
select CHAR(97)
select CHAR(65)
select CHAR(51)

--retira espacos em branco da esquerda
select LTRIM('       Ivan           ')+'carlos'
--retira espacos em branco da direita
select RTRIM('       Ivan           ')+'carlos'
select LTRIM( RTRIM('       Ivan           '))+'carlos'

--tranforma texto em minuscula
select  lower('Ivan Carlos Pavao')
--tranforma texto em maiuscula
select  upper('Ivan Carlos Pavao')

--acrescenta espacos
select 'Ivan'+space(15)+'carlos'

--replica o texto a quantidade de vezes indicada
select replicate('Ivan ',8)


--recorta da esquerda a quant de caracteres indicadas
--de um texto
select LEFT ('computador',3)

--recorta da direita a quant de caracteres indicadas
```

```
--de um texto
select RIGHT ('computador',3)


--recorta uma frase determinando
--o ponto inicial do corte e a quant a ser cortada
--de um texto
select substring ('computador',4,4)

--substitui um texto por outro
select REPLACE('banana','n','t')
select REPLACE('O garotão é bonito','bonito','lindo')

--retorna o grau de diferença entre as palavras
select DIFFERENCE('abacate','abacate')
select DIFFERENCE('abacate','abaca')
select DIFFERENCE('abacate','abacaxi')
select DIFFERENCE('abacate','oilado')


--inverte o texto
select reverse ('ivan carlos pavao')


--***********************************************
--***********************************************
--Funções de DATA e HORA
--***********************************************

--retorna a data e hora do sistema
select GETDATE()

select day(GETDATE())
select month(GETDATE())
select year(GETDATE())

/*
intervalos

dd    dia
mm    mês
yy    ano
hh    hora
mm    minuto
ss    segundo
mi    milisegundos
ns    nanosegundos
ww  numero da semana
dw  dia da semana
dy  dia do ano
qq  trimestre
*/
select DATEPART(ww,getdate())
select DATEPART(dw,getdate())
select DATEPART(qq,'10/01/2014')
select DATEPART(dy,GETDATE())
```

```
select datename(dw,GETDATE())
select datename(mm,GETDATE())


/***********************************************/
--                      intervalo, quantidade, data
select DATEADD (  DD     ,    40     , GETDATE())
select DATEADD (  DD     ,    80     , GETDATE())
select DATEADD (  DD     ,    120     , GETDATE())


select DATEADD (  mm     ,    9     , GETDATE())


select DATEADD ( yy     ,    5     , GETDATE())


--------------------------------------
--ALTERA O PADRAO DE DATA DO SISTEMA PARA DIA MES E ANO
SET DATEFORMAT DMY

select DATEDIFF(DD,GETDATE(),'21/12/2014')

select DATEDIFF(SS,'21/05/1972','18/09/2014')

SELECT * FROM ORDERS

SELECT ORDERID,CUSTOMERID,ORDERDATE,SHIPPEDDATE,
    DATEDIFF (DD,OrderDate,ShippedDate) AS 'TEMPO DE ENTREGA'
    FROM ORDERS
    ORDER BY [TEMPO DE ENTREGA]    DESC

-----------------------------------------------------
--
/*
EXERCICIOS

1)  LISTAR DA TABELA DE CLIENTES OS QUE SAO DE 'LONDON'
2)  MOSTRAR OS DADOS DA TABELA EMPLOYEES COM
    UM CAMPO CALCULADO PARA EXIBIR A IDADE DO FUNCIONARIO
3)  EXIBA TODOS OS PRODUTOS COM PREÇOS ENTRE 30 E 80
4)  EXIBA TODOS OS CLIENTES QUE POSSUEM 'ANTON'
      NO INICIO DO NOME
5)  EXIBA TODOS OS CLIENTES QUE POSSUEM 'ANTON'
      EM QUALQUER PARTE DO NOME
*/
```