Revisão / Exercícios

Prof. Márcio Bueno

{bd2tarde,bd2noited}@marciobueno.com

Especificando Campos

- Apelidos de campo

```
SELECT “DEI" + "-UNICAP" as Centro,
Universidade = “UNICAP"
```

- Expressões como campos
  - Podem estar presentes na lista de campos, no ORDER BY ou no GROUP BY

Cláusulas do SELECT

SELECT <lista_de_campos>

FROM tabela1 [,tabela2,tabela3...]

<tipo> JOIN <tabelax> ON <criterio_associacao>

Informa critérios de junção de tabelas

WHERE <condição_booleana>

Especifica critérios para filtragem de dados

GROUP BY <lista_de_campos>

Agrupamento de dados.

ORDER BY <lista_de_campos [ASC/DESC]>

Altera a ordem dos registros

Cláusula WHERE

- Operadores
  - OR, AND, NOT
  - >, <, >=, <=, =, <>
  - | (bitwise OR), & (bitwise AND), ^ (bitwise XOR)
  - LIKE
  - IN
  - EXISTS
  - BETWEEN
- Uso de parênteses

Cláusula WHERE

- Cuidados com NULL e NOT NULL
  - Operador IS NULL
  - NULL = NULL é falso! NULL não é igual a nada.
  - Configurações SET para comportamento com NULL
- Comparando com constantes
  - Inteiro – pode colocar entre parênteses
  - String – entre apóstrofos
  - Data – entre apóstrofos, no formato ‘dd/mm/yyyy hh:nn:ss’
    - Ano pode conter 2 dígitos

WHERE Simples

```
USE Northwind
SELECT
   employeeid,
   name = firstname + ' ' + lastname,
   title
FROM dbo.employees
WHERE employeeid = 5
```

Usando Like

```
USE Northwind
SELECT
   companyname
FROM dbo.customers
WHERE companyname LIKE '%Restaurant%'
   OR companyname LIKE '_en'
   OR companyname LIKE '[CK]%'
   OR companyname LIKE '[S-V]%'
   OR companyname LIKE 'M[^c]%'
```

Faixa de valores

```
USE Northwind
SELECT
  productname,
  unitprice
FROM dbo.products
WHERE unitprice BETWEEN 10 AND 20
```

Lista de valores

```
USE Northwind
SELECT
   companyname,
   country
FROM dbo.suppliers
WHERE country IN
   ('Japan','Italy')
```

- Pode ser outro SELECT retornando 1 campo

Valores desconhecidos

- Usar IS NULL depois do campo
- Ou então IS NOT NULL

```
Use Northwind
SELECT
   companyname,
   fax
FROM dbo.suppliers
WHERE fax IS NULL
```

Ordenando Dados

```
Use Northwind
SELECT
   productid,
   productname,
  categoryid,
  unitprice
FROM dbo.products
ORDER BY categoryid ASC, unitprice DESC
```

Eliminando valores duplicados

```
Use Northwind
SELECT DISTINCT
   country
FROM dbo.suppliers
ORDER BY country
```

Exemplo de SELECT de 1 Tabela

```
USE Northwind
SELECT
convert(VARCHAR(10),productid) + ' - ' +
   productname as 'Nome Produto',
  UnitsInStock as 'Em estoque',
  ReorderLevel as 'Nível Crítico',
  Diferenca = (ReorderLevel - UnitsInStock)
FROM dbo.products
WHERE UnitsInStock < Reorderlevel
ORDER BY (ReorderLevel - UnitsInStock) DESC
```

Exercício

- Retorne os nomes dos produtos que começam com as letras a, d, de m a z, ordenados por nome.

```
CREATE TABLE Products(
    ProductID int IDENTITY(1,1) NOT NULL,
    ProductName nvarchar(40) NOT NULL,
    SupplierID int NULL,
    CategoryID int NULL,
    QuantityPerUnit nvarchar(20) NULL,
    UnitPrice money NULL,
    UnitsInStock smallint NULL,
    UnitsOnOrder smallint NULL,
    ReorderLevel smallint NULL,
    Discontinued bit NOT NULL,
    PRIMARY KEY (ProductID ASC)
)
```

Exercício

- Retorne os nomes dos fornecedores que estão sem homepage informada na tabela.

```
CREATE TABLE Suppliers (
    SupplierID int IDENTITY(1,1) NOT NULL,
    CompanyName nvarchar(40) NOT NULL,
    ContactName nvarchar(30) NULL,
    ContactTitle nvarchar(30) NULL,
    Address nvarchar(60) NULL,
    City nvarchar(15) NULL,
    Region nvarchar(15) NULL,
    PostalCode nvarchar(10) NULL,
    Country nvarchar(15) NULL,
    Phone nvarchar(24) NULL,
    Fax nvarchar(24) NULL,
    HomePage ntext NULL,
    PRIMARY KEY (SupplierID)
)
```

Exercício

- Mostre a lista de países dos clientes da empresa (tabela Customers), sem repetições

```
CREATE TABLE Customers(
    CustomerID nchar(5) NOT NULL,
    CompanyName nvarchar(40) NOT NULL,
    ContactName nvarchar(30) NULL,
    ContactTitle nvarchar(30) NULL,
    Address nvarchar(60) NULL,
    City nvarchar(15) NULL,
    Region nvarchar(15) NULL,
    PostalCode nvarchar(10) NULL,
    Country nvarchar(15) NULL,
    Phone nvarchar(24) NULL,
    Fax nvarchar(24) NULL,
    PRIMARY KEY (CustomerID)
)
```

Exercício

- Mostre os nomes dos clientes que não fizeram pedidos.

```
CREATE TABLE Orders(
    OrderID int IDENTITY(1,1) NOT NULL,
    CustomerID nchar(5) NULL,
    EmployeeID int NULL,
    OrderDate datetime NULL,
    RequiredDate datetime NULL,
    ShippedDate datetime NULL,
    ShipVia int NULL,
    Freight money NULL,
    ShipName nvarchar(40) NULL,
    ShipAddress nvarchar(60) NULL,
    ShipCity nvarchar(15) NULL,
    ShipRegion nvarchar(15) NULL,
    ShipPostalCode nvarchar(10) NULL,
    ShipCountry nvarchar(15) NULL,
    PRIMARY KEY (OrderID)
)
```

Consultas em 2 ou mais tabelas

- No SQL Server: cláusulas JOIN.
- Podem ser INNER ou OUTER.
- Importante uso de apelidos de tabelas.
- Normalmente os JOINs unem FK's com PK's.

Sintaxe Parcial do JOIN

- SELECT <lista_de_campos>
- FROM tabela1 [,tabela2,tabela3...]
- <tipo> JOIN <tabelax> ON <criterio_associacao>
  - Pode existir mais de um JOIN no comando

Exemplo com duas tabelas

```
USE Northwind
SELECT cs.companyname,
   cs.customerid,
   os.orderdate
FROM dbo.customers cs
JOIN dbo.orders os
   ON  os.customerid =
   cs.customerid
WHERE os.orderdate BETWEEN
   '1/1/1997' AND '1/3/1997'
```

Exemplo com mais de duas tabelas

```
USE Northwind
SELECT cliente = cs.companyname, empregado =
   em.firstname + ' ' + em.lastname,
   'Data Pedido' = os.orderdate,
  Via = sh.companyname
FROM dbo.customers cs
JOIN dbo.orders os
  ON  os.customerid = cs.customerid
JOIN dbo.employees em
  ON em.employeeid = os.employeeid
JOIN dbo.shippers sh
  ON sh.shipperId = os.shipVia
WHERE os.orderdate BETWEEN '01/01/1997' AND
   '01/03/1997'
```

OUTER JOIN

- Pode ser RIGHT, LEFT ou FULL
  - RIGHT e LEFT são equivalentes, só muda a direção.
- FULL mostra das duas tabelas
- Aplicações clássicas
  - Mostrar lista dos clientes que não têm pedidos no Sistema
  - Mostrar lista dos empregados com os respectivos pedidos realizados (mostrar TODOS os empregados, mesmo os que não efetuaram pedidos).

Exemplo de OUTER JOIN

```
USE Northwind
SELECT cs.companyname,
   cs.customerid,
  os.orderdate
FROM dbo.customers cs
LEFT JOIN dbo.orders os
  ON  os.customerid = cs.customerid
WHERE os.orderdate between '01/01/1997' AND '01/3/1997'
   OR orderdate is null
ORDER BY os.orderdate
```

- Mostra todos os clientes com respectivas datas de pedido. Dois deles (FISSA e Paris Spécialités) não fizeram pedidos.

CROSS JOIN

- Basta não especificar o critério de associação
- Produto cartesiano de registros

```
SELECT te.TerritoryDescription,
  re.RegionDescription
FROM territories te
CROSS JOIN region re
```

UNION

- Usado para juntar resultados de diferentes SELECTs

```
SELECT name = (em.firstname + ' ' + em.lastname),
  em.city,em.postalcode
FROM employees em
UNION
SELECT cs.companyname,cs.city,cs.postalcode
FROM customers cs
```

Sub-Consultas

- Consiste em introduzir em determinados pontos da consulta uma outra consulta entre parênteses
  - Técnica poderosa e sofisticada
- Sub-consulta como um campo
  - Só pode retornar um campo e um valor
- Sub-consulta após o FROM ou JOIN
  - Importante uso de apelidos
- Sub-consulta no WHERE
  - Só pode retornar um campo e um valor

Exercício

- Listar sem repetições os nomes de empregados que fizeram pedidos no mês de Janeiro/1997

```
CREATE TABLE Orders(
   OrderID int IDENTITY(1,1),
   CustomerID nchar(5),
   EmployeeID int,
   OrderDate datetime,
   ... )

CREATE TABLE  Employees (
   EmployeeID int IDENTITY(1,1),
   LastName nvarchar(20),
   FirstName nvarchar(10),
   ... )
```

Exercício

- Listar os nomes das empresas clientes atendidas por Nancy Davolio, sem repetições

```
CREATE TABLE Customers (
    CustomerID nchar(5),
    CompanyName
       nvarchar(40),
    ...)
```

```
CREATE TABLE Orders(
   OrderID int IDENTITY(1,1) ,
   CustomerID nchar(5),
   EmployeeID int,
   OrderDate datetime,
   ... )
```

```
CREATE TABLE  Employees (
   EmployeeID int IDENTITY(1,1),
   LastName nvarchar(20),
   FirstName nvarchar(10),
   ... )
```

Exercício

- Mostrar os nomes e telefones de clientes e empregados.

```
CREATE TABLE Customers (
    CustomerID nchar(5),
    CompanyName nvarchar(40),
    Phone nvarchar(24),
    ...)

CREATE TABLE Employees (
    EmployeeID int IDENTITY(1,1),
    LastName nvarchar(20),
    FirstName nvarchar(10),
    HomePhone nvarchar(24),
    ...)
```

Agrupamento de Dados

- Pode conter a cláusula GROUP BY e HAVING
- Funções de agrupamento
  - AVG, COUNT, MAX, MIN, SUM, STDEV, STDEVP, VAR, VARP

Agrupamento de Dados – Sem GROUP BY

```
SELECT COUNT(*)
FROM orders
```

```
SELECT COUNT
  (orderdate)
FROM orders
```

```
SELECT
  MAX(orderdate)
FROM orders
```

```
SELECT
  SUM(freight)
FROM orders
```

Agrupamento de Dados

- Regras do GROUP BY
  - Um item da lista de campos do SELECT pode ser um campo de tabela, uma expressão ou uma aplicação de função de agrupamento.
  - Todo campo ou expressão contido na lista de campos do SELECT que não for aplicação de uma função de agrupamento deve estar na lista de campos do GROUP BY
    - A inversa não é verdadeira – os campos do GROUP BY não necessariamente precisam estar contidos na lista de campos
  - Para filtrar registros baseado no resultado de uma função de agrupamento, usar HAVING.

Exemplo de GROUP BY

- Total de quantidade pedida de cada produto

```
SELECT pr.productname,
  Count(od.quantity) as QuantidadeTotal
FROM products pr
JOIN [order details] od
  ON od.productid = pr.productid
GROUP BY productname
ORDER BY count(od.quantity) desc
```

Exemplo de GROUP BY

- Contagem de pedidos feitos por empregado

```
SELECT empregado = (em.firstname + ' ' +
   em.lastname),
  Count (os.orderid) as Pedidos
FROM orders os
JOIN employees em
  ON em.employeeid = os.employeeid
GROUP BY em.firstname + ' ' + em.lastname
ORDER BY Pedidos Desc
```

HAVING

- Filtro sobre resultado de função de agrupamento

```
SELECT empregado = (em.firstname + ' ' +
   em.lastname), Count(os.orderid) as Pedidos
FROM orders os
JOIN employees em ON em.employeeid =
   os.employeeid
GROUP BY em.firstname + ' ' + em.lastname
HAVING Count(os.orderid) > 50
ORDER BY Pedidos Desc
```

TOP

- Útil para listar os primeiros n valores de uma consulta.
  - Pode listar os primeiros n% registros.
- Faz sentido em conjunto com ORDER BY.
- Pode listar também os empates na última colocação
  - WITH TIES
- Específico do SQL Server.

```
SELECT TOP 10 (...)
SELECT TOP 10 PERCENT (...)
SELECT TOP 10 WITH TIES
```

Exercício

- Mostre a menor e maior datas de pedidos realizados.

```
CREATE TABLE Orders(
    OrderID int IDENTITY(1,1),
    OrderDate datetime,
    ...)
```

Exercício

- Mostrar quantos produtos existem em cada categoria. Mostrar o nome da categoria (tabelas Products e Categories)

```
CREATE TABLE Categories (
    CategoryID int IDENTITY(1,1),
    CategoryName nvarchar(15),
    ...)

CREATE TABLE Products (
    ProductID int IDENTITY(1,1),
    ProductName nvarchar(40),
    CategoryID int,
    UnitsInStock smallint NULL,
    ...)
```

Exercício

- Listar a quantidade de pedidos feitos por cada cliente entre Janeiro/97 e Junho/97. Mostrar o nome da empresa cliente.

```
CREATE TABLE Customers (
    CustomerID nchar(5),
    CompanyName nvarchar(40),
    ...)

CREATE TABLE Orders(
    OrderID int IDENTITY(1,1),
    CustomerID nchar(5),
    OrderDate datetime,
    ...)
```

Exercício

- Listar a venda total realizada por empregado no mês de Abril/1997

```
CREATE TABLE  Employees (
   EmployeeID int IDENTITY(1,1),
   LastName nvarchar(20),
   FirstName nvarchar(10),
   ... )

CREATE TABLE Orders(
      OrderID int IDENTITY(1,1),
      EmployeeID int,
      OrderDate datetime,
      ...)
```

Exercício

- Listar o nome e a soma dos valores de pedidos feitos para cada produto no mês de Março/1998 dos 10 maiores

```
CREATE TABLE Products (
    ProductID int IDENTITY(1,1),
    ProductName nvarchar(40),
    ...)
```

```
CREATE TABLE Orders(
    OrderID int IDENTITY(1,1),
    ...)
```

```
CREATE TABLE Order Details (
    OrderID int,
    ProductID int,
    Quantity smallint,
    UnitPrice money
    ...)
```

Exercício

- Listar os 5 fornecedores que mais foram acionados no ano de 1997, por total de vendas. Mostrar o nome do fornecedor e o total da venda.

Exercício

- Listar o total de vendas realizadas mês a mês. Mostrar o mês/ano e o total de vendas realizadas naquele mês/ano.