Aula 020

Professores:

Geraldo Xexéo Geraldo Zimbrão

Conteúdo:

Introdução a SQL



Roteiro

- Introdução
- DDL
 - Criação de tabelas
 - Criação de restrições



- Consultas simples
- Junções
- Subconsultas
- Inserções, Alterações e Remoções
- Comandos especiais
 - ─ Índices



<u>Introdução</u>



SQL é a linguagem de definição e consulta padrão para sistemas gerenciadores de bancos de dados relacionais

- Structured Query Language
- Elaborada para ser independente de hardware ou de software
- Desenvolvida pela IBM para ser a linguagem de consulta do SYSTEM R (anos 70)
- Padronizada pelo ANSI em 1986



Abstrai detalhes físicos

- Arquivos, organização dos dados, leitura e escrita



DDL - Data Definition Language



Serve para criar os elementos do banco de dados

─ Tabelas, restrições, índices, visões etc



Possui um conjunto padrão de tipos de dados primitivos

Caracteres, strings, data, números etc

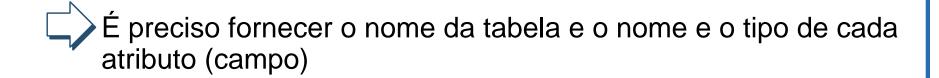


Os comandos são separados por ";"



Comando Create Table





- Opcionalmente fornecemos também
 - Chaves primárias e chaves candidatas
 - Chaves estrangeiras
 - Restrições



Exemplo 1 - Create Table

```
Create table Pais (
nome varchar(80) not null,
capital varchar(80),
habitantes numeric
);
```

- Criamos uma tabela País, com os atributos:
 - Nome, do tipo string com até 80 caracteres, e que não pode ser deixado sem valor (nulo)
 - Capital, do tipo string com até 80 caracteres
 - Habitantes, do tipo numérico



Considerações - 1



Na tabela País, não especificamos a chave primária

- Isso significa que nenhum atributo serve para identificar unicamente um país
- Sabemos no entanto que o nome identifica um país
- Nome pode ser portanto uma chave primária para país
- Não especificamos que os outros atributos devem estar preenchidos
 - ─ É possível deixar a capital do país em branco



Considerações - 2



No modelo relacional:

- Tuplas, linhas e registros significam a mesma coisa, assim como atributo, campo ou coluna
- Uma tabela com uma linha e uma coluna (uma tupla com um único atributo) equivale a um valor
- Embora se diga que uma tabela seja um conjunto de tuplas, é possível haver repetição de tuplas
 - Há um comando específico evitar repetições



Exemplo de Dados na Tabela Pais

Nome	Capital	Habitantes
Brasil	Brasília	170,000,000
Estados Unidos	Washington	300,000,000
Argentina	Buenos Aires	50,000,000
llhas Caymã		900,000



Exemplo 2 - Adicionando a Chave Primária

Alter table Pais add primary key(nome);



Nome foi definido como chave primária da tabela País

- Chaves primárias não podem ter valor nulo
- Seu valor é único, ou seja, não podemos ter dois países com o mesmo nome



Comando Drop Table

Drop table Pais;



Remove uma tabela

- Apaga todo o seu conteúdo
- ─ Índices, restrições e chaves primárias



Esquema Exemplo

```
Create table Continente (
nome varchar(80) not null,
primary key(nome)
);

Create table Pais (
nome varchar(80) not null,
capital varchar(80),
continente varchar(80) not null,
habitantes numeric,
primary key(nome)
);
```

Create table Unidade (
nome varchar(80) not null,
tipo varchar(40),
capital varchar(80),
sigla char(2) not null unique,
pais varchar(80) not null,
habitantes numeric,
primary key(nome)
);



<u>Integridade Referencial - 1</u>



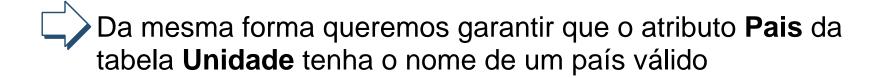
Queremos garantir que o atributo **continente** da tabela **Pais** tenha o nome de um continente válido, já cadastrado na base

- Chamamos isso de integridade referencial
- Para isso, devemos criar uma restrição de integridade

Alter table Pais add foreign key(continente) references Continente(nome);



<u>Integridade Referencial - 2</u>



- Somente podemos criar restrições de integridade sobre chaves primárias ou atributos com valor único (chaves candidatas)
- Não é necessário dizer o nome do campo referenciado se ele for a chave primária

Alter table Unidade add foreign key(pais) references Pais;



Exemplo de Integridade Referencial

Continente

Nome		
América do Sul 🕶		
América do Norte		
América Central		

Nome	Capital	Continente	Habitantes
Brasil	Brasília	América do Sul	170,000,000
Estados Unidos	Washington	América do Norte	300,000,000
Argentina	Buenos Aires	América do Sul	50,000,000
llhas Caymã		América Central	900,000



<u>Variações</u>

```
Create table Unidade (
nome varchar(80) not null primary key,
tipo varchar(40),
capital varchar(80),
sigla char(2) not null unique,
pais varchar(80) not null references Continente,
habitantes numeric
);
```

Alter table Unidade add area numeric not null;

Alter table Pais **drop** habitantes;



DML - Data Manipulation Language



Serve para consultar, inserir, alterar ou remover entradas (tupla ou linhas) em uma tabela



Retorna um conjunto de tuplas (equivale a uma tabela)

INSERT, UPDATE, DELETE

- Insere/altera/remove uma ou mais tuplas em uma tabela
- Não retorna nenhum valor



Comando Select



Usado para retornar as entradas de uma tabela

- Podemos retornar todos os atributos ou escolher alguns
- Podemos retornar todas as entradas ou apenas aquelas que atendam algum critério de seleção
- Podemos realizar operações estatísticas sobre os valores dos atributos das entradas da tabela
 - Somar, calcular a média, encontrar máximos e mínimos
 - Contar o número de entradas retornadas



Sintaxe do comando Select - 1

SELECT Lista de Campos FROM Tabela



Select Nome, Capital from Pais

Nome	Capital
Brasil	Brasília
Estados Unidos	Washington
Argentina	Buenos Aires
llhas Caymã	



Se quisermos todos os campos devemos usar *



Sintaxe do comando Select - 2

SELECT Lista de Campos FROM Tabela
WHERE Restrição



Select Nome, Capital from Pais Where Continente = 'América do Sul'

Nome	Capital	
Brasil	Brasília	
Argentina	Buenos Aires	



Cláusula Where - 1

A *restrição* na cláusula *where* é uma *expressão* composta de **campos**, **operadores** e **funções** que deve retornar um valor lógico (verdadeiro ou falso)

Serão retornadas pela consulta apenas as tuplas para os quais o resultado dessa *expressão* for verdadeiro



Cláusula Where - 2



Principais operadores

Aritméticos, relacionais (comparação) e lógicos

- Expressão Like '%substring%'
 - compara substrings
- Expressão Between X and Y
 - Define intervalos
- Expressão *In* (Valor, Valor, Valor...)
 - Testa se pertence a um conjunto de valores

Exemplo - Like, Between e In

Select Nome, Capital from Pais Where continente **like** '%Sul'

Select Nome, Capital from Pais Where nome **between** 'A' **and** 'C'

Select Nome, Capital from Pais Where nome **In** ('Brasil', 'Argentina')

Nome	Capital	
Brasil	Brasília	
Argentina	Buenos Aires	

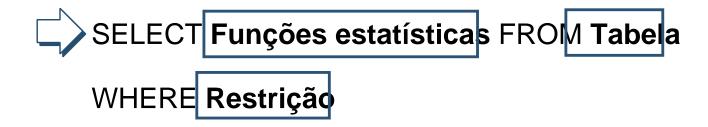


Cláusula Where - 3

- Principais funções
 - Matemáticas
 - Trunc, Round, Abs, Mod, Sqrt ...
 - Strings
 - Strlen ou Length, substr, to_int...
 - Datas
 - Now, Timestamp ...
- As funções variam muito de SGBD para SGBD
- É permitido criar funções novas e utilizá-las
 - A forma varia de SGBD para SGBD



Funções Estatísticas



Função		
Count		
Sum	Soma os valores (numéricos)	
Avg	Calcula a Média simples (sum/count)	
Min	Retorna o menor valor	
Max	Retorna o maior valor	



Exemplo - Funções Estatísticas

Select count(*) from Unidade where Pais = 'Brasil'

Count	
27	

Select sum(habitantes), avg(habitantes) from Unidade where Pais = 'Brasil'

Sum	Avg
170,000,000	6,292,929



As funções estatísticas retornam uma tabela com apenas uma tupla

─ Também são chamadas de funções agregadoras



Agrupamento - Cláusula Group by



- A cláusula **Group by** serve para agrupar as tuplas que tenham o mesmo valor nos atributos listados nela
 - Quando ela é utilizada, apenas os campos que aparecerem na lista de campos dela podem ser retornados pela consulta
 - Além deles, podemos retornar também funções agregadoras sobre os outros campos



Exemplo 1 - Group by

Select pais, sum(habitantes) as "População" from Unidade **Group by** pais

Pais	População	
Brasil	170,000,000	
Estados Unidos	300,000,000	
Argentina	50,000,000	
Ilhas Caymã	900,000	



As tuplas foram agrupadas pelo atributo pais, e as somas foram efetuadas em cada grupo de tuplas



Podemos renomear as colunas da tabela retornada



Agrupamento - Cláusula Having

```
SELECT Lista de Campos FROM Tabela
WHERE Restrição
GROUP BY Lista de Campos
HAVING Restrição sobre agregações
```

- A cláusula **Having** serve para criar restrições sobre as agregações ou sobre os campos usados no agrupamento
 - Apenas as linhas do resultado que satisfizerem a restrição serão retornadas

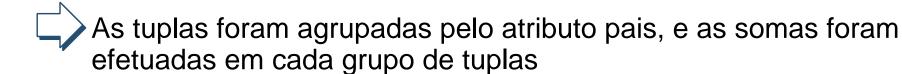


Exemplo 2 - Group by e Having

Select pais, sum(habitantes) as "População" from Unidade Group by pais

Having sum(habitantes) > 100.000.000

Pais	População
Brasil	170,000,000
Estados Unidos	300,000,000



Mas apenas os agrupamentos cuja soma de habitantes é maior do que 100 milhões é retornada



Cláusula Order by

```
SELECT Lista de Campos FROM Tabela
WHERE Restrição
GROUP BY Lista de Campos
HAVING Restrição sobre agregações
ORDER BY Lista de Campos
```

- A cláusula *Order by* serve apenas para ordenar o resultado por determinado(s) campo(s)
 - Não elimina nenhuma linha do resultado da consulta



Exemplo - Order by

Select nome, pais, habitantes from Unidade Where pais like 'Br%' **Order by** habitantes **desc**

Nome	Pais	Habitantes
São Paulo	Brasil	60,000,000
Rio de Janeiro	Brasil	25,000,000
Minas Gerais	Brasil	23,000,000



As tuplas foram ordenadas pelo atributo habitantes em ordem descendente

Se a palavra reservada desc for omitida, o resultado é ordenado em ordem ascendente (asc)

Cláusula Distinct

Select pais from Unidade

Pais	
Brasil	
Argentina	
Brasil	
(0.000) (10.0 0) (10.00)	

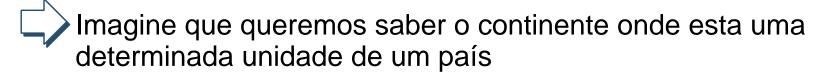


Para evitar tuplas duplicadas na resposta devemos usar a palavra reservada *distinct*

Select distinct pais from Unidade



<u>Junções - Introdução</u>



Por exemplo, onde fica a "Nordrhein-Westfalen"?



- Unidade possui o campo pais
- Pais possui o campo continente
- Mas eles estão em tabelas diferentes
 - Teria de descobrir primeiro o país da unidade, e depois o continente do país
 - Duas consultas separadas, onde segunda depende da primeira



<u>Junções</u>

- Uma **junção** permite que se combine e recupere dados de mais de uma tabela em uma única consulta
 - O resultado continua sendo um conjunto de tuplas
 - Os campos retornados podem vir de mais de uma tabela
 - ─ É uma das operações mais importantes de um banco de dados



<u>Junções - Sintaxe</u>





Select Unidade.nome, Continente from Unidade, Pais Where Unidade.nome = 'Nordrhein-Westfalen' and Pais = Pais.nome

Nome	Continente
Nordrhein-Westfaler	Europa



Junções - Considerações

- No exemplo, as tabelas Unidade e Pais possuem ambas um campo **Nome**
 - Para evitar ambiguidade, devemos qualificar o campo
 - Colocar na frente do campo o nome da tabela a que o mesmo pertence, separado por um ponto
 - A cláusula where não precisa ter, necessariamente, nenhuma outra restrição além da condição de junção
 - Mesmo a condição de junção é opcional
 - Nesse caso o resultado seria bem diferente, não sendo uma junção mas sim um produto cartesiano



Condição de Junção

- De uma forma geral, a condição de junção será na maioria das vezes entre campos que sejam chaves das tabelas envolvidas
 - primária, candidata (unique) ou estrangeira (foreign key)
- Uma junção pode ter quantas tabelas se desejar
 - Para cada tabela extra na junção é necessário uma condição de junção
 - Se tivermos 4 tabelas, em geral serão necessárias três condições de junção
 - Podemos usar qualquer uma das outras cláusulas
 - Podemos renomear as tabelas



Junções - Exemplo

Select Continente, P.nome, count(*) as Total from Unidade **U**, Pais **P** where U.Pais = P.nome group by Continente, P.nome having sum(habitantes) > 5.000.000 order by Continente, Count(*)

Continente	Nome	Total
América do Sul	Brasil	11
América do Sul	Argentina	4
América do Norte	Estados Unidos	31
•••	ner i	



Subconsultas

- Uma consulta pode ser utilizada dentro de uma espressão em SQL
 - Se a consulta retornar um conjunto de tuplas, ela pode ser utilizada em qualquer lugar que um cojunto puder ser utilizado
 - Por exemplo no operador In
- Se a consulta retornar apenas uma linha e uma coluna ela pode ser utilizada em qualquer lugar que um valor puder ser utilizado
 - Por exemplo em uma comparação



<u>Subconsulta - Exemplo</u>

Podemos reescrever a consulta sobre a Nordrhein-Westfalen da seguinte forma:

Select continente from pais where nome = (select pais from unidade where nome = 'Nordrhein-Westfalen')

- É necessário pôr a subconsulta entre parênteses
- Poderíamos usar também o operador **In** no lugar de "="
- Não podemos usar os campos da subconsulta na consulta principal
 - Porém podemos usar campos da consulta principal na subconsulta



<u>Inserções</u>

INSERT INTO Tabela (Lista de Campos)

VALUES (Lista de Valores)

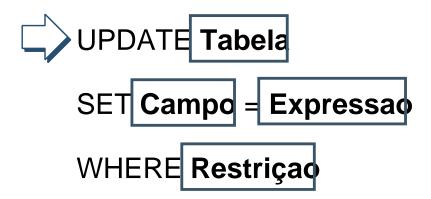
Exemplo:

Insert into Pais (Nome, Capital, Continente) Values ('Espanha', 'Madri', 'Europa')

Esse comando insere mais uma tupla na tabela com os valores especificados para cada campo na respectiva ordem em que aparecem



<u>Alterações</u>





Update Unidade Set habitantes = 16.000.000 Where Nome = 'Rio de Janeiro'

- Altera o valor do atributo nas tuplas que satisfizerem a expressão de restrição
- Podemos substituir a expressão por uma subconsulta
- Se não houver **Restrição**, todas as tuplas serão modificadas

<u>Remoções</u>





delete from Pais where nome = 'Espanha'

- Remove as tuplas que satisfizerem a expressão de restrição
- Se não houver **Restrição**, todas as tuplas serão modificadas
- A restrição pode envolver subconsultas



<u>Índices</u>

Índices são estruturas de dados usadas apenas para acelerar consultas

Árvore B, Hash, Cluster, Bitmap



ON Tabela (Lista de Campos)

Exemplo:

create index pais_idx on unidade(pais, nome);

