**Pesquisa em Múltiplas Tabelas**

O funcionamento do SQL é baseado na **teoria de conjuntos**, ou seja, as tabelas são consideradas conjuntos de dados e o resultado das pesquisas são subconjuntos que podem ser obtidos, por exemplo, através da interseção ou união dos conjuntos originais.

Tendo isso em mente, é fácil usar as cláusulas SQL para realizar pesquisas mais complexas. Para isso, o SQL disponibiliza também comandos que permitem pesquisar dados de várias tabelas ou combinar dados de tabelas diferentes.

**SQL: Utilizando o Operador JOIN**Para mostrar como o comando JOIN funciona, vamos utilizar um exemplo bem básico, mas, que pode ser facilmente extrapolado para situações mais complexas, desde que se tenha o conceito em mente.

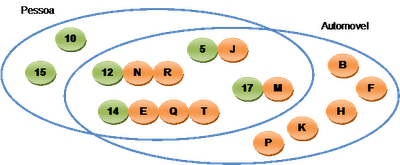
Imagine um banco de dados que possua apenas duas tabelas, cujos layout e conteúdo estão detalhados abaixo:

* **Modelo de dados e conteúdo das tabelas**



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Pessoa** | | | **codPessoa** | **nomePessoa** | | 5 | João | | 10 | Pedro | | 12 | Maria | | 14 | José | | 15 | Carlos | | 17 | Regina | |  | |  |  |  | | --- | --- | --- | | **Automovel** | | | | **codAuto** | **codPessoa** | **descricaoAuto** | | B | (null) | Fusca branco | | E | 14 | Opala bege | | F | (null) | Passat amarelo | | H | (null) | Fusca verde | | J | 5 | Opala marrom | | K | (null) | Brasília branca | | M | 17 | Passat cinza | | N | 12 | Belina verde | | P | (null) | Fiat 147 | | Q | 14 | Fusca azul | | R | 12 | Brasília preta | | T | 14 | Opala vermelho | |

Se representarmos o conteúdo deste banco de dados em forma de conjunto, ele ficaria assim:



Feito isso, nós podemos entender melhor o efeito do JOIN quando aplicado a este conjunto de dados.

Veja os exemplos abaixo:

* **INNER JOIN**

O INNER JOIN retorna o sub-conjunto interseção dos conjuntos envolvidos na pesquisa.

SELECT Pessoa.codPessoa, Pessoa.nomePessoa, Automovel.codAuto, Automovel.idPessoa, Automovel.descricaoAuto

FROM **Pessoa**

INNER JOIN **Automovel** ON Pessoa.codPessoa = Automovel.codPessoa;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pessoa.codPessoa** | **nomePessoa** | **codAuto** | **Automovel.codPessoa** | **descricaoAuto** |
| 5 | João | J | 5 | Opala marrom |
| 12 | Maria | N | 12 | Belina verde |
| 12 | Maria | R | 12 | Brasília preta |
| 14 | José | E | 14 | Opala bege |
| 14 | José | Q | 14 | Fusca azul |
| 14 | José | T | 14 | Opala vermelho |
| 17 | Regina | M | 17 | Passat cinza |

Só serão exibidos elementos que estão nos dois conjuntos. Esta é a utilização mais comum do JOIN nas pesquisas que envolvem mais de uma tabela. O INNER JOIN pode ser escrito também no formato abaixo (chamado de implícito) e o resultado é exatamente o mesmo:

SELECT Pessoa.codPessoa, Pessoa.nomePessoa, Automovel.codAuto, Automovel.codPessoa, Automovel.descricaoAuto FROM Pessoa, Automovel  
WHERE Pessoa.codPessoa = Automovel.codPessoa;

* **LEFT JOIN**

O LEFT JOIN retorna todos os elementos que estão no conjunto da esquerda (a primeira tabela), mais os elementos que estão na interseção.

SELECT Pessoa.codPessoa, Pessoa.nomePessoa, Automovel.codAuto, Automovel.codPessoa, Automovel.descricaoAuto FROM Pessoa

LEFT JOIN Automovel ON Pessoa.codPessoa = Automovel.codPessoa;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pessoa.codPessoa** | **nomePessoa** | **codAuto** | **Automovel.codPessoa** | **descricaoAuto** |
| 5 | João | J | 5 | Opala marrom |
| 10 | Pedro | (null) | (null) | (null) |
| 12 | Maria | N | 12 | Belina verde |
| 12 | Maria | R | 12 | Brasília preta |
| 14 | José | E | 14 | Opala bege |
| 14 | José | Q | 14 | Fusca azul |
| 14 | José | T | 14 | Opala vermelho |
| 15 | Carlos | (null) | (null) | (null) |
| 17 | Regina | M | 17 | Passat cinza |

* **RIGHT JOIN**

O RIGHT JOIN, ao contrário, retorna todos os elementos que estão no conjunto da direita (a segunda tabela), mais os elementos que estão na interseção.

SELECT Pessoa.codPessoa, Pessoa.nomePessoa, Automovel.codAuto, Automovel.codPessoa, Automovel.descricaoAuto FROM Pessoa

RIGHT JOIN Automovel ON Pessoa.codPessoa = Automovel.codPessoa;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pessoa.codPessoa** | **nomePessoa** | **codAuto** | **Automovel.codPessoa** | **descricaoAuto** |
| (null) | (null) | B | (null) | Fusca branco |
| (null) | (null) | F | (null) | Passat amarelo |
| (null) | (null) | H | (null) | Fusca verde |
| (null) | (null) | K | (null) | Brasília branca |
| (null) | (null) | P | (null) | Fiat 147 |
| 5 | João | J | 5 | Opala marrom |
| 12 | Maria | N | 12 | Belina verde |
| 12 | Maria | R | 12 | Brasília preta |
| 14 | José | E | 14 | Opala bege |
| 14 | José | Q | 14 | Fusca azul |
| 14 | José | T | 14 | Opala vermelho |
| 17 | Regina | M | 17 | Passat cinza |

Os exemplos acima, apesar de muito simples, podem ser facilmente extrapolados para a maioria das pesquisas onde é necessária a pesquisa em mais de uma tabela, pois, apesar de ser possível existir mais complexidade nas cláusulas, o raciocínio básico é exatamente o mesmo.

**SQL: Utilizando o Operador UNION e UNION ALL**

O operador UNION combina os resultados de duas ou mais queries em um único resultado, retornando todas as linhas pertencentes a todas as queries envolvidas na execução. Para utilizar o UNION, o número e a ordem das colunas precisam ser idênticos em todas as queries e os datatypes precisam ser compatíveis.  
  
Existem dois tipos de operador UNION, sendo eles UNION e UNION ALL.

**UNION**

O operador UNION combina o resultado de execução das duas queries e então executa um SELECT DISTINCT a fim de eliminar as linhas duplicadas. Este processo é executado mesmo que não hajam registros duplicados.

Exemplo:

SELECT codigo, nome FROM clientes;

**UNION**

SELECT codigo, nome FROM fornecedores;

**Resultado:**

Codigo Nome

---------------------------------------------------------  
01 Ana Maria

03 Carlos

04 Maria Luiza

02 Bernardo

**UNION ALL**

O operador UNION ALL tem a mesma funcionalidade do UNION, porém, não executa o SELECT DISTINCT e apresentam todas as linhas, inclusive as ***linhas duplicadas.***

**RECOMENDAÇÕES**

1) Se não existe a possibilidade de haver registros duplicados em suas tabelas ou se não houver problemas para a aplicação apresente duplicações, utilize o operador **UNION ALL**. A vantagem é que este operador não executa a função **SELECT DISTINCT**, utiliza menos recursos do SQL Server e como consequência, melhora a performance da aplicação.

2) Não utilize o operador **UNION** em conjunto com a função **SELECT DISTINCT** pois o resultado final será exatamente o mesmo, porém, o SQL Server estará executando a mesma operação duas vezes, causando queda de desempenho.

3) Uma query com uma ou mais cláusula **OR** pode ser reescrita utilizando o operador **UNION ALL**.