

Atualizando e excluindo dados

Continuaremos com a estrutura de dados utilizada no capítulo anterior. Se você deseja reconstruí-la, baixe o arquivo do projeto que contém alguns INSERTs para criar compras na tabela COMPRAS. Importe o arquivo igual na aula passada.

```
SQL> @DIRETORIO_DO_ARQUIVO_SQL/NOME_DO_ARQUIVO
```

Selecionando dados de um intervalo específico de valores

Agora que já existem várias compras cadastradas, desejamos selecionar apenas as que possuem valor entre R\$200,00 e R\$700,00. Para isso, usaremos a cláusula `WHERE` :

```
SELECT * FROM COMPRAS WHERE valor >= 200 AND valor <= 700;
```

Mas será que não existe algum operador mais simples?

```
SELECT * FROM COMPRAS WHERE valor BETWEEN 200 AND 700;
```

O operador `BETWEEN` serve para filtrar registros de acordo com um intervalo de valores. É possível filtrar apenas as compras realizadas entre 05/01/2010 e 25/06/2010.

```
SELECT * FROM COMPRAS WHERE DATA BETWEEN '05-JAN-2010' AND '25-JUN-2010';
```

Atualizando os dados das nossas compras

Mas o que precisamos fazer é alterar os registros criados durante esse período. Para todos eles vamos colocar a observação "compra emergencial". O comando SQL para realizar alteração em uma tabela é o `UPDATE` . Devemos informar o nome da coluna que será alterada e o novo valor. Por exemplo, se desejamos atualizar todas as compras para emergenciais, gostaríamos de algo como:

Atualize as compras, alterando a observação para 'compra emergencial'.

Podemos fazer um `UPDATE` como o a seguir:

```
UPDATE COMPRAS SET OBSERVACOES = 'compra emergencial';
```

O comando `UPDATE` atualiza os registros já existentes. Ele é seguido pelo nome da tabela, no nosso caso, `COMPRAS` . Também é preciso indicar quais colunas terão seu valor alterado, usando o comando `SET` seguido pelo nome das colunas e pelo novo valor desejado. Portanto, `SET OBSERVACOES = 'compra emergencial'` indica que queremos colocar o valor 'compra emergencial' na coluna `OBSERVACOES` .

No comando acima, em nenhum momento foi dito que essa alteração só deveria ser aplicada para as compras realizadas em um determinado período. Como não havia nenhuma restrição no comando, todos os registros foram

alterados e agora todas as compras têm a mesma observação. Todas as restrições que aprendemos até agora ao usar o comando `WHERE` se aplicam ao comando `UPDATE`. Com o uso do `WHERE`, teríamos evitado a alteração de todos os registros.

Para corrigir, vamos alterar a observação de todas as compras que **não** estão em um determinado intervalo de datas:

Atualize as compras, alterando a observação para 'uma compra comum', somente se a data não estiver entre os dias 05/01/2010 e 25/06/2010.

Traduzindo para sql:

```
UPDATE COMPRAS SET OBSERVACOES = 'uma compra comum' WHERE DATA NOT BETWEEN '05-JAN-2010' AND '25-JUN-2010';
```

No comando acima, o `UPDATE` foi utilizado com o `WHERE` para restringir as compras que serão afetadas pela alteração que desejamos. Nessa condição, apareceu um novo operador, `NOT`, em `WHERE DATA NOT`. Queremos alterar todas as compras que foram feitas em qualquer data **exceto** as de um determinado período; ou seja, queremos fazer uma negação. O `BETWEEN` estabelece esse período e o `NOT` nega essa condição. Podemos ler o trecho `DATA NOT BETWEEN '05-JAN-2010' AND '25-JUN-2010'` como: a data não está no período entre 05/01/2010 e 25/06/2010.

Atualizando as compras utilizando um conjunto de valores

Agora sabemos que é possível usar `WHERE` em um comando `UPDATE` para filtrar os dados da mesma forma que em um `SELECT`. Podemos alterar as compras feitas no dia 25/12/2010 ou no dia 12/10/2010 ou ainda no dia 12/06/2010 para conter a observação "datas festivas". Uma forma seria usar vários `OR`:

```
UPDATE COMPRAS SET OBSERVACOES = 'datas festivas' WHERE DATA = '25-DEC-2010' OR DATA = '12-OCT-2010' OR DATA = '12-JUN-2010';
```

O exemplo está correto, porém o código ficou feio, pois repete-se o nome da coluna `DATA` muitas vezes. Existe uma condição que pode ser usada justamente para melhorar isso:

```
UPDATE COMPRAS SET OBSERVACOES = 'datas festivas' WHERE DATA IN ('25-DEC-2010', '12-OCT-2010', '12-JUN-2010');
```

O operador `IN` foi usado para criar um conjunto de valores que usamos na comparação. O trecho `WHERE DATA IN ('25-DEC-2010', '12-OCT-2010', '12-JUN-2010')` diz que queremos todos os registros com qualquer uma das datas que esteja entre os parênteses, e apenas essas datas (nenhuma compra feita em outra data serve).

Deletando dados da nossa tabela

Na medida em que usamos o banco de dados, muita coisa que está sendo guardada se torna obsoleta. É possível excluir qualquer informação que não seja mais de nosso interesse. Vamos:

Excluir todas as compras efetuadas antes de 2009:

```
DELETE FROM COMPRAS WHERE DATA < '01-JAN-2009';
```

O comando `DELETE` exclui registros de uma tabela. E por isso **sempre** deve ser executado usando o `WHERE`. Se nenhuma condição for usada junto com o comando `DELETE`, então todos os registros da tabela serão excluídos. **Cuidado:** Usamos a condição `WHERE DATA < '01-JAN-2009'` para excluir todas as compras feitas antes de 01/01/2009.

Os comandos `UPDATE` e `DELETE` são essenciais para a manutenção dos registros de nosso banco de dados. Como eles alteram definitivamente os valores dos registros, é importantíssimo sempre usarmos uma condição através do comando `WHERE` para limitar o efeito desses comandos a apenas alguns registros. Uma dica é sempre escrever primeiro o `SELECT` com o `WHERE`, ao usar um `UPDATE` ou `DELETE`. Por exemplo, antes de excluir as compras anteriores a 2009, poderíamos ter executado o seguinte:

```
SELECT * FROM COMPRAS WHERE data < '01-JAN-2009';
```

Assim, poderíamos analisar o resultado e ter certeza que só compras anteriores a 2009 estavam na lista de resultados. Que tal fazer alguns exercícios para se acostumar com o uso do `UPDATE` e do `DELETE` ?