02

Agrupando dados e fazendo consultas mais inteligentes

Exibindo a soma dos valores de todas as compras

Vamos supor que precisamos calcular a soma de todas as compras realizadas até hoje. Precisaríamos fazer algo parecido com um *loop*, pegando todos os valores retornados, e somando um a um. A SQL nos permite fazer consultas que agrupam valores da forma que queremos.

Nesse caso, queremos somar (`SUM`) todos os elementos da coluna `VALOR` :

```
SELECT SUM(VALOR) FROM COMPRAS;
```

Por ser um comando `SELECT` , podemos colocar condições, por exemplo, calcular a soma de todas as compras feitas antes de 01/01/2010:

```
SELECT SUM(VALOR) FROM COMPRAS WHERE DATA < '01-JAN-2010';
```

Mas eu quero a média dos valores das compras, como exibi-la?

Podemos calcular informações ainda mais interessantes, tal como a média (`AVG`) das compras feitas antes de 01/01/2010:

```
SELECT AVG(VALOR) FROM COMPRAS WHERE DATA < '01-JAN-2010';
```

Podemos inclusive juntar as duas, dando nomes mais bonitos para as colunas geradas, usando o comando `AS` , visto nas aulas anteriores:

```
SELECT AVG(VALOR) AS MEDIA, SUM(VALOR) AS SOMA FROM COMPRAS WHERE DATA < '01-JAN-2010';
```

E a contagem de compras?

Podemos contar (`COUNT`) o número de compras da mesma forma:

```
SELECT COUNT(VALOR) FROM COMPRAS WHERE DATA < '01-JAN-2010';
```

Percebemos até então que o comando `SELECT` unido às funções de soma, média e contagem pode gerar dados muito úteis. Mas e se quisermos a soma de todas as compras recebidas e não recebidas? Podemos fazer duas consultas, por exemplo:

```
SELECT SUM(VALOR) FROM COMPRAS WHERE RECEBIDO = '1';
SELECT SUM(VALOR) FROM COMPRAS WHERE RECEBIDO = '0';
```

Agrupando compras de acordo com um valor específico

E se quisermos todo esse resultado em apenas uma única consulta? Afinal, executar uma a uma manualmente pode gerar muito trabalho. Imagine se quiséssemos a soma das compras por forma de pagamento, seriam 3 consultas diferentes, uma para BOLETO, uma para CARTAO, uma para DINHEIRO.

Para isso, precisamos informar ao Oracle que queremos agrupar os dados. No nosso caso, queremos agrupar compras de acordo com o valor da coluna `RECEBIDO`. Usamos então o comando `GROUP BY`, junto ao resto da SQL. Veja o exemplo:

```
SELECT SUM(VALOR) FROM COMPRAS GROUP BY RECEBIDO;
```

Veja que agora ele agrupou e exibiu dois resultados, muito provavelmente a soma de todas as compras na qual recebido vale 1, e as compras na qual recebido vale 0. O problema é que não sabemos qual valor corresponde a cada uma das condições de `RECEBIDO`. Para isso, podemos exibir a coluna `RECEBIDO` na consulta também:

```
SELECT RECEBIDO, SUM(VALOR) FROM COMPRAS GROUP BY RECEBIDO;
```

Agora sabemos a soma para cada grupo! Podemos juntar também os valores desse agrupamento para exibir a soma e a quantidade de itens em cada grupo, por exemplo:

```
SELECT RECEBIDO, SUM(VALOR) AS SOMA, COUNT(VALOR) AS TOTAL FROM COMPRAS GROUP BY RECEBIDO;
```

Mas eu percebi que o resultado está desordenado, como ordená-lo?

Podemos refinar ainda mais esse resultado, ordenando, por exemplo, pelo grupo na qual a soma seja maior (ou seja, de maneira decrescente). Para isso, usamos o comando `ORDER BY`, informando o campo que queremos ordenar no final da nossa consulta SQL:

```
SELECT RECEBIDO, SUM(VALOR) AS SOMA, COUNT(VALOR) AS TOTAL FROM COMPRAS GROUP BY RECEBIDO ORDER
```

Repare a palavra `DESC`. Ela indica que queremos ordenar de maneira decrescente. Para ordenar de maneira crescente, fazemos uso da palavra `ASC`:

```
SELECT RECEBIDO, SUM(VALOR) AS SOMA, COUNT(VALOR) AS TOTAL FROM COMPRAS GROUP BY RECEBIDO ORDER
```

Consultas agrupadas são um recurso muito interessante e permite-nos extrair muitas informações dos nossos dados!