SESI SENAI

PELO FUTURO DO TRABALHO

04- Seleção de dados básica

Professor Hermano Roepke

A estrutura do comando

O comando SELECT apresenta seis cláusulas, sendo que duas delas são obrigatórias, ou seja, sempre deverão estar presentes quando você utilizar o comando. Conheça a estrutura:

```
SELECT
   { *, coluna [apelido], coluna [apelido], ... }
FROM
   { tabela [apelido], tabela [apelido], ... }
WHERE
   { condição, condição, ... }
GROUP BY
   { coluna_agrupamento, coluna_agrupamento, ... }
HAVING
   { condição_função_agrupamento, condição_função_agrupamento, ... }
ORDER BY
   { coluna, coluna, ... }
```

As cláusulas SELECT e FROM

A primeira cláusula (SELECT) que dá nome ao comando é utilizada para que você informe qual ou quais são as colunas que você deseja recuperar. A(s) coluna(s) deve(m) pertencer à tabela presente da cláusula FROM. Se você deseja selecionar todas as colunas da tabela poderá utilizar o asterisco (*). Veja um exemplo:

```
SELECT *
FROM empregado;
```

As cláusulas SELECT e FROM

Já o próximo comando especifica as colunas que serão recuperadas/ apresentadas. Quando Você faz isto está utilizando de um conceito denominado de projeção de dados. Veja o exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado;
```

É importante ressaltar que os exemplos acima não restringem os registros recuperados, ou seja, em ambos, todos os registros da tabela serão recuperados.

Atribuindo apelido à coluna

Um recurso usual na seleção de dados é a atribuição de um apelido para a(s) coluna(s). Veja como isso pode ser feito:

```
SELECT nm_empregado as NOME, funcao as FUNÇÃO, salario as SALÁRIO
FROM empregado;
```

Em alguns SGBDS, como o MySQL, não é necessário utilizar a palavra reservada "as" antes do apelido atribuído. Caso se deseje atribuir nomes compostos ao apelido, será necessário utilizar aspas ("apelido composto"). Veja um exemplo:

```
SELECT nm_empregado as NOME, funcao as FUNÇÃO, salario as "SALÁRIO FIXO"
FROM empregado;
```

Em alguns SGBDs o apelido atribuído pode ser utilizado na cláusula ORDER BY, e somente nesta.

Utilizando a cláusula WHERE

Raramente um comando SELECT é executado sem a presença da cláusula WHERE. Através da cláusula WHERE é possível restringir os registros que você deseja recuperar, além de estabelecer a relação entre duas ou mais tabelas. Vamos conhecer um exemplo de restrição (condição) feita através da cláusula WHERE:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE id_empregado = 7788;
```

Para adicionar mais de uma restrição (condição), é necessário utilizar um operador lógico entre elas. Vamos a um exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE id_departamento = 10 AND salario > 1000;
```

Principais operadores

Os operadores mais utilizados na cláusula WHERE são classificados em relacionais (ou de comparação) e lógicos (booleanos):

Operadores Relacionais

Operador	Significado
=	Igual a
>	Maior do que
>=	Maior do que ou igual a
<	Menor do que
<=	Menor do que ou igual a
<>	Diferente de

Operadores Lógicos

Operador	Significado
OR	Uma das condições for verdade
AND	Ambas as condições devem ser verdade
NOT	Negação ou inverso da variável

As condições segue as seguintes **regras de precedência:**

1ª Operadores Relacionais
2ª Operador NOT
3ª Operador AND
4ª Operador OR

Para ajustar a precedência, caso seja necessário, utilize parênteses.

Operador alfanumérico

Pode-se afirmar que o principal operador utilizado para recuperação dos dados alfanuméricos (textuais) é o LIKE. O operador LIKE busca valores alfanuméricos incompletos a partir de um ou mais caracteres. Para tanto, basta utilizar os seguintes caracteres especiais:

- "**%**" corresponde a uma sequência qualquer de zero ou mais caracteres
- "_" corresponde a qualquer caractere.
- Veja alguns exemplos:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE nm_empregado LIKE 'S%';
```

No exemplo, todos os nomes que iniciam com a letra "S" serão recuperados.

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE nm_empregado LIKE 'M_LLER';
```

No exemplo, nomes como MULLER, MILLER E MYLLER serão recuperados.

Operadores IS e IN

O operador IS é utilizado quando é necessário avaliar se uma coluna apresentar ou não valor nulo. Exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE comissao IS NULL;
```

Neste exemplo os registros que apresentarem ausência de um valor válido, ou seja, que forem nulos, serão recuperados.

Já o operador IN é utilizado para testar a coluna em uma lista de valores. Exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE id_departamento IN (10,30);
```

Neste exemplo serão recuperados registros onde o departamento for igual a 10 ou 30.

Comando equivalente

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE id_departamento = 10 OR
id_departamento = 30;
```

Operador BETWEEN

O operador BETWEEN é utilizado quando para testar um intervalo de valores. Exemplos:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE salario BETWEEN 800 AND 1200;
```

Neste exemplo os registros onde o salário entre 800 e 1200 (incluindo estes) serão recuperados.

Comando equivalente

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE salario >= 800 AND SALARIO <=1200;
```

É importante destacar que os valores das extremidades também compõe a lista a ser analisada. Neste, assim como nos demais operadores, é possível utilizar a negação da condição através do operador lógico NOT(Ex.: ... WHERE SALARIO NOT BETWEEN 800 AND 1200).

Cláusula ORDER BY

A cláusula ORDER BY é opcional no comando SELECT. Ela permite especificar a ordem em que os registros serão apresentados no resultado. Veja um exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
ORDER BY nm_empregado;
```

No exemplo acima, o resultado será ordenado pela coluna "nm_empregado", do menor para o maior. Essa é a opção padrão da cláusula (ASC), ou seja, ordenar sempre do menor para o maior quando não for especificado o critério de ordenação. Porém é possível indicar que a sequência seja ordenada do maior para o menor. Para tanto é necessário utilizar a palavra reservada "DESC", conforme este exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
ORDER BY salario DESC;
```

Cláusula ORDER BY

Ainda sobre a cláusula ORDER BY, é importante destacar que é possível utilizar mais de uma coluna – as quais podem apresentar critérios diferentes de ordenação. Veja exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
ORDER BY funcao DESC, nm_empregado ASC;
```

No exemplo acima, o resultado será ordenado pela coluna "SALARIO" do maior para o menor e, caso haja mais de um registro com o mesmo salário, será apresentado o registro cujo nome for o menor (segundo o conjunto alfanumérico). Observe ainda que foi acrescida a palavra reservada "ASC" (opcional) após a coluna "NM_EMPREGADO".

Cláusula ORDER BY

Além do que foi apresentado, é possível utilizar a posição da coluna presente na cláusula SELECT. Veja este exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
ORDER BY 3 DESC, 1 ASC;
```

Os exemplos apresentados são equivalentes, porém com diferenças na cláusula ORDER BY. Qualquer campo pertencendo à(s) tabela(s) indicada(s) na cláusula for pode ser utilizado para ordenação, mesmo que estão não seja apresentada no resultado.

Funções alfanuméricas

Cada SGBD apresenta um conjunto de funções de apoio a manipulação de dados alfanuméricos. É recomendável que você consulte sempre a documentação do SGBD que você estiver utilizando. Mesmo assim, é importante destacar duas funções muito utilizadas e que são encontradas em todos os SGBDs: A UPPER e a LOWER.

Veja um exemplo:

```
SELECT UPPER(nm_empregado), LOWER(funcao)
FROM empregado;
```

No exemplo dado, o resultado apresentado vai conter o nome (coluna nm_empregado) em letras maiúsculas; e atividade (coluna funcao), em letras minúsculas.

Funções alfanuméricas

Essas funções também podem ser utilizadas em outras cláusulas, como neste exemplo:

```
SELECT nm_empregado, funcao, salario
FROM empregado
WHERE UPPER(NM_EMPREGADO) LIKE 'SMIT%';
```

Resumo

O comando SELECT apresenta 6 cláusulas, sendo que foram apresentados até então 4 delas (SELECT, FROM, WHERE e ORDER BY).

Os tipos e os principais operadores utilizados na cláusula WHERE.

Especificar critérios para a exibição dos dados recuperados através da cláusula ORDER BY.

Algumas funções que podem ajudar na manipulação de dados, em especial os alfanuméricos.

O comando SELECT

Este é o principal comando da linguagem SQL. Estudos mostram que mais de 70% das operações em um SGBD correspondem a consultas aos dados – estas por sua vez, realizadas com o uso deste comando. Com este comando você poderá:

- **selecionar dados de uma ou mais tabelas;**
- **restringir e projetar dados de uma ou mais tabelas;**
- **agrupar e selecionar dados de uma ou mais tabelas;**
- **gerar diferentes visões sobre os mesmos dados.**

SESI SENAI

PELO FUTURO DO TRABALHO

sesisc.org.br sc.senai.br

0800 048 1212

Rodovia Admar Gonzaga, 2765 - Itacorubi - 88034-001 - Florianópolis, SC