Operar consultas usando a cláusula WHERE

CLÁUSULA WHERE E OPERADORES DA SQL


```
CREATE TABLE ALUNO (
      CODIGOALUNO int NOT NULL,
      NOME varchar(90) NOT NULL,
      SEXO char(1) NOT NULL,
      DTNASCIMENTO date NOT NULL,
      EMAIL varchar(30) NULL,
      CONSTRAINT ALUNO_pk PRIMARY KEY (CODIGOALUNO)
);

INSERT INTO ALUNO (CODIGOALUNO,NOME,SEXO,DTNASCIMENTO,EMAIL) VALUES
(1,'JOSÉ FRANCISCO TERRA','M','28/10/1989','JFT@GMAIL.COM');
INSERT INTO ALUNO (CODIGOALUNO,NOME,SEXO,DTNASCIMENTO,EMAIL) VALUES
(2,'ANDREY COSTA FILHO','M','20/10/1999','ANDREYCF@HOTMAIL.COM');
INSERT INTO ALUNO (CODIGOALUNO,NOME,SEXO,DTNASCIMENTO,EMAIL) VALUES
(3,'PATRÍCIA TORRES LOUREIRO','F','20/10/1980','PTORRES@GMAIL.COM');
INSERT INTO ALUNO (CODIGOALUNO,NOME,SEXO,DTNASCIMENTO,EMAIL) VALUES
(4,'CARLA MARIA MACIEL','F','20/11/1996',NULL);
INSERT INTO ALUNO (CODIGOALUNO,NOME,SEXO,DTNASCIMENTO,EMAIL) VALUES
(5,'LEILA SANTANA COSTA','F','20/11/2001',NULL);
```

RECUPERANDO DADOS COM SELECT/FROM/WHERE/ORDER BY

Sintaxe básica:

```
SELECT COLUNA1 [[AS] APELIDOCOLUNA1],
COLUNA2 [[AS] APELIDOCOLUNA2],
…
COLUNAN [[AS] APELIDOCOLUNAN]
FROM TABELA
```

```
WHERE <CONDIÇÃO>
ORDER BY EXPRESSÃO1[ASC|DESC] [NULLS {FIRST|LAST}],
[EXPRESSÃO2[ASC|DESC] [NULLS {FIRST|LAST}…];
```

O propósito do SELECT é declararmos as colunas da consulta. No FROM, informamos a tabela alvo da consulta. No WHERE, especificamos alguma condição, simples ou composta, para filtrar registros que serão recuperados pelo SGBD. No ORDER BY, declaramos uma ou mais colunas como critério de ordenação, com possibilidade de especificarmos se valores NULL aparecem no início ou no final do resultado.

É importante frisar que a cláusula WHERE realiza a operação de restrição da Álgebra Relacional, também conhecida como seleção – não confundir com o comando SELECT

Operadores relacionais

Operador	Significado
<	menor
<=	menor ou igual a
>	maior
>=	maior ou igual a
=	igual
<> ou !=	> diferente

Operadores lógicos:

Operador	Significado
AND	conjunção
OR	disjunção
NOT	negação

Consulta 01:

```
SELECT NOME, DTNASCIMENTO
FROM ALUNO
WHERE SEXO='F';
```

Consulta 02:

```
SELECT NOME, DTNASCIMENTO
FROM ALUNO
WHERE SEXO='F' AND EXTRACT (MONTH FROM DTNASCIMENTO)=11;
```

RECUPERANDO DADOS COM O USO DO OPERADOR IN

O operador [NOT] IN pode ser utilizado em consultas que envolvam comparações usando uma lista de valores.

Consulta 03:

```
SELECT NOME, DTNASCIMENTO
FROM ALUNO
WHERE EXTRACT (MONTH FROM DTNASCIMENTO) IN (7,8,9,10,11,12);
```

RECUPERANDO DADOS COM O USO DO OPERADOR BETWEEN(ENTRE)

O operador [NOT] BETWEEN verifica se determinado valor encontra-se no intervalo entre dois valores.

Por exemplo, X BETWEEN Y AND Z é equivalente a X>=Y AND X<=Z. De modo semelhante, X NOT BETWEEN Y AND Z é equivalente a X<Y OR X>Z.

Consulta 04:

```
SELECT NOME
FROM ALUNO
WHERE EXTRACT (YEAR FROM DTNASCIMENTO) BETWEEN 1985 AND 2005;
```

Note que a expressão na cláusula WHERE compara o ano de nascimento de cada aluno junto ao intervalo especificado pelo operador BETWEEN. Caso quiséssemos extrair o mesmo resultado sem o uso do BETWEEN, poderíamos programar um comando equivalente, conforme a seguir:

```
SELECT NOME
FROM ALUNO
WHERE EXTRACT (YEAR FROM DTNASCIMENTO) >= 1985 AND EXTRACT (YEAR FROM
DTNASCIMENTO)<= 2005;
```

RECUPERANDO DADOS COM O USO DO OPERADOR LIKE

O uso do [NOT] LIKE permite realizar buscas em uma cadeia de caracteres.

Trata-se de um recurso bastante utilizado em buscas textuais. Você pode utilizar os símbolos especiais a seguir:

_ para ignorar qualquer caractere específico;

% para ignorar qualquer padrão.

-Buscando nome-

Consulta 05:

```
SELECT NOME
FROM ALUNO
WHERE NOME LIKE '%COSTA%';
```

-Buscando com POSIÇÃO DE LETRA de letra-

Consulta 06:

Letra na 2 posição do nome:

```
SELECT NOME
FROM ALUNO
WHERE NOME LIKE '_A%';
```

Note que, para especificar o “A” na segunda posição, o SGBD desprezará qualquer valor na primeira posição da string, não importando o que estiver localizado à direita do “A”.

Consulta 07:

Listar o nome de quem não possua maria no nome;

```
SELECT NOME
FROM ALUNO
WHERE NOME NOT LIKE '%MARIA%';
```

Consulta 08:

Contando quantos alunos tem @gmail.com;

```
SELECT NOME
FROM ALUNO
WHERE NOME NOT LIKE '%MARIA%';
```

RECUPERANDO DADOS COM O USO DO OPERADOR NULL

Quando uma coluna é opcional, significa que existe possibilidade de que algum registro não possua valor cadastrado para a coluna em questão. Nessa hipótese, há entendimento de que o valor da coluna é “desconhecido” ou “não aplicável”.

Para testar se uma coluna possui valor cadastrado, usa-se a expressão COLUNA IS NOT NULL.

Consulta 09:

```
SELECT NOME, DTNASCIMENTO, EMAIL
```

```
FROM ALUNO
WHERE EMAIL IS NOT NULL;
```

O SGBD retorna os registros onde há algum conteúdo cadastrado na coluna EMAIL.

Consulta 10:

```
SELECT NOME
FROM ALUNO
WHERE EMAIL IS NULL;
```

O SGBD retorna os registros sobre os quais não há valor cadastrado na coluna EMAIL.

RECUPERANDO DADOS USANDO ORDENAÇÃO DOS RESULTADOS

Para melhor organizar os resultados de uma consulta, nós podemos especificar critérios de ordenação.

Consulta 11:

Retornar o nome e a data de nascimento dos alunos, ordenando os resultados por nome, de maneira ascendente.

```
SELECT NOME, DTNASCIMENTO
FROM ALUNO
ORDER BY NOME;
```

O SGBD retorna os registros da tabela ALUNO, obedecendo ao critério de ordenação especificado na linha 3 da consulta. O padrão ascendente (ASC) é opcional.

Consulta 12:

Retornar o nome e a data de nascimento dos alunos, ordenando os resultados de modo ascendente pelo mês de nascimento e, em seguida, pelo nome, também de modo ascendente.

```
SELECT NOME, DTNASCIMENTO
FROM ALUNO
ORDER BY EXTRACT (MONTH FROM DTNASCIMENTO), NOME;
```