

TLBD II

Continuação

As **expressões aritméticas** podem conter **nomes de colunas, valores numéricos** constantes e os **operadores aritméticos**:

OPERADOR	DESCRIÇÃO
+	Adição
-	Subtração
*	Multiplificação
/	Divisão

Prioridade na execução dos operadores: *, /, +, -. Em caso de igualdade é da esquerda para a direita. O uso de parênteses altera as prioridades.

Exemplo: `select nome, (sal+250)*12 from empregados;`

Expressões Aritméticas

Um valor nulo significa:

- Um valor não atribuído;
- Um valor que não está disponível;
- Um valor desconhecido;
- Um valor que não é aplicável;

A utilização de nulos cria os seguintes problemas:

- Se numa **expressão** um dos valores for nulo, o resultado é nulo;
- As funções estatísticas não contam com os valores nulos;
- Um valor nulo não é indexado por um índice e portanto uma pesquisa por valores nulos implica *"full table scan"* (não tira partido do índice);

Valores Nulos

Crie a seguinte tabela **empregados**:

Id [int(11); primary key; auto_increm ent]	Nome [varchar (255)]	Salario [float(11)]	Comissao [float(11)]
1	André	800	2000
2	Alberto	725	(NULL)
3	Vanessa	1500	(NULL)
4	Leandro	900	1000

Exercício - IFNULL()

O query abaixo pretende determinar salário anual de cada empregado:

```
SELECT nome, salario*14+comissao "Salário Anual"  
FROM empregados;
```

Resultando:

nome	Salário Anual
André	13200
Alberto	(NULL)
Vanessa	(NULL)
Leandro	13600

Exercício - IFNULL()

Os empregados que não são vendedores têm valor nulo na comissão e por isso o resultado da expressão é nulo. Para obter um resultado é necessário converter os nulos em zero. Para isso podemos usar a função **IFNULL()** que tem dois argumentos:

- Uma expressão (origem do valor nulo);
- Um valor não nulo para substituição;

Experimente:

```
SELECT nome, salario*14+IFNULL(comissao,0) "Salário Anual"  
FROM empregados;
```

E veja o resultado.

Exercício - IFNULL()

Descreve um objeto/tabela:

Experimente:

DESC empregados;

O comando DESCRIBE (DESC)

A cláusula **WHERE** corresponde ao **operador restrição** da Álgebra Relacional. Contém uma condição que os registros têm de satisfazer para que sejam visualizados.

A condição tem a seguinte estrutura:

Nome de uma coluna	Operadores de comparação / Operadores de SQL	Nome de uma coluna / constante / lista de valores
-----------------------	--	--

A cláusula WHERE

OPERADOR	SIGNIFICADO
=	Igual a
!= ^= <>	Diferente
>	Maior que
>=	Maior ou igual a
<	Menor que
<=	Menor ou igual a

Operadores de Comparação

OPERADOR	SIGNIFICADO
BETWEEN NOT BETWEEN	Intervalo fechado compreendido entre os dois valores val1 e val2
IN (val1,val2,val3,val4) NOT IN (val1,val2,val3,val4)	Pertence à lista de valores val1, val2, val3, val4
LIKE NOT LIKE	Cadeia de caracteres que satisfaz a condição. Utilizar % e _
IS NULL IS NOT NULL	É um valor NULL

Operadores do SQL

Utilizando a tabela do exercício anterior,
execute:

```
SELECT nome, salario  
FROM empregados  
WHERE salario BETWEEN 900 AND 1500
```

```
SELECT nome, salario  
FROM empregados  
WHERE salario NOT BETWEEN 900 AND 1500
```

BETWEEN

Execute:

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE comissao IN (2000,1000)
```

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE comissao NOT IN (2000,1000)
```

Por que no último exemplo nada foi exibido?

IN

Execute:

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE nome LIKE 'a%'
```

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE nome NOT LIKE 'a%'
```

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE nome LIKE '____e____'
```

LIKE

Execute:

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE comissao IS NULL
```

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE comissao IS NOT NULL
```

OBS: Se o valor NULL for requerido numa comparação, então devem ser utilizados os operadores **IS NULL** ou **IS NOT NULL**. Se nenhum deles for utilizado, então o resultado da comparação **dará sempre falso** porque um valor NULL não pode ser igual nem diferente a outro valor.

IS NULL

Os operadores **AND** e **OR** podem ser utilizados para construir expressões lógicas compostas.

AND será *True* quando ambas condições forem *True*.

OR será *True* quando uma das condições for *True*.

Ainda na tabela anterior, execute:

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE salario > 800 AND comissao IS NULL
```

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE salario > 800 OR comissao IS NULL
```

Operadores Lógicos

AND tem maior prioridade que **OR**, por exemplo:

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE salario < 900 AND nome LIKE 'a%' OR comissao IS NULL
```

Serão selecionados os empregados que começam com a letra "A" e tem salário menor que 900 OU os que não possuem comissão.

Se quisermos selecionar os empregados que têm salário menor que 900 E os que começam com a letra "A" OU não possuem comissão deveríamos usar parênteses:

```
SELECT nome, salario, comissao  
FROM empregados  
WHERE salario < 900 AND (nome LIKE 'a%' OR comissao IS NULL)
```

Operadores Lógicos

EXERCÍCIOS