

## Funções INSTR e LENGTH

### Transcrição

Outra tarefa que costumamos fazer é pesquisar se uma *string* possui caracteres específicos dentro. Por exemplo, se quero pesquisar se um nome possui a letra "A". Para isto, podemos usar uma função com nome facilmente relacionável: `instr`. Então, na nossa consulta, iremos verificar se dentro do nome `Rogério` encontramos a substring `a` - também seria válido pesquisar por um número maior de caracteres. Vamos executá-la na tabela `dual`.

```
SQL> select instr('Rogério','a') from dual;
```

```
INSTR('ROGERIO','A')
-----
0
```

O SQL \*Plus retornou que a palavra `Rogério` possui 0 ocorrências da letra `a`. Teríamos um resultado diferente se pesquisássemos pela letra `o`:

```
SQL> select instr('Rogério','o') from dual;
```

```
INSTR('ROGERIO','O')
-----
2
```

Agora, ele retornou que na posição 2 encontramos a primeira ocorrência da letra `o`. Se pesquisássemos por `ge`, ele informaria que na posição 3 encontraríamos os caracteres.

```
SQL> select instr('Rogério','ge') from dual;
```

```
INSTR('ROGERIO','GE')
-----
3
```

No entanto, o que acontece em casos no quais o caractere pesquisado possui mais de uma ocorrência. Vamos fazer o teste, pesquisando pela letra `a` no nome `amanda`.

```
SQL> select instr('amanda','a') from dual;
```

```
INSTR('AMANDA','A')
-----
1
```

Ele nos informou que encontrou a letra "a" na posição 1. Para que ele indique outra ocorrência da letra "a", podemos indicar um outro parâmetro: a posição pela qual a busca deve começar. No caso do nome `amanda`, podemos indicar que a busca inicie a partir da letra `m`.

```
SQL> select instr('amanda','a',2) from dual;
```

```
INSTR('AMANDA','A',2)
-----
3
```

O primeiro `a` encontrado, a partir da posição `2`, está localizado na posição `3`. Ainda que a busca tenha sido iniciada a partir da posição `2`, ele sempre será iniciada pelo primeiro caractere da esquerda. Neste tipo de pesquisa, ele também irá diferenciar letras maiúsculas de minúsculas.

Se fizermos uma busca da letra `a` dentro da palavra `amAnda`, começando pela segunda posição, o retorno será:

```
SQL> select instr('amAnda','a',2) from dual;
```

```
INSTR('AMANDA','A',2)
-----
6
```

Ele indicará que `a` está localizado na posição `6`. Com a função `instr()` ele irá retornar `0` se não encontrar nenhuma ocorrência e caso encontre, irá indicar a posição da primeira ocorrência. Esta função também tem um quarto parâmetro opcional que é a indicação de qual ocorrência nós estamos procurando.

Por exemplo, podemos pedir que ele imprima a segunda ocorrência da letra `a`.

```
SQL> select instr('amanda','a',2,2) from dual;
```

```
INSTR('AMANDA','A',2,2)
-----
6
```

Como começamos pela posição `2`, ele ignorou a primeira ocorrência da letra `a` e retornou que a segunda está na posição `6`. Caso não existisse uma segunda ocorrência, o retorno seria `0`. Faremos o teste com o nome `amando`.

```
SQL> select instr('amando','a',2,2) from dual;
```

```
INSTR('AMANDA','O',2,2)
-----
0
```

Ele não encontrou uma segunda aparição da letra `a`.

Outro uso interessante para a função é com nomes de arquivos. Se queremos buscar um caractere como `.` (ponto), por exemplo, no arquivo `desktop/compras.txt`. Vamos buscar desde a primeira posição e que ele indique a primeira ocorrência.

```
SQL> select instr('desktop/compras.txt','.',1,2) from dual;
```

```
INSTR('DESKTOP/COMPRAS.TXT','.',1,1)
-----
16
```

Ele informou que na posição 16 foi encontrado um . .

Agora, podemos buscar qual é o . antes da extensão do arquivo. Porém, e se o arquivo tivesse duas ocorrências de ponto? Então, uma solução possível é começarmos a busca pelo fim da *string*. Para isto, iremos usar como terceiro parâmetro -1 , para que a busca comece pela última letra.

```
SQL> select instr('desktop/compras.txt', '.', -1, 1) from dual;
```

```
INSTR('DESKTOP/COMPRAS.TXT', '.', -1, 1)
-----
20
```

Ainda que a busca seja iniciada pelo fim da *string*, a contagem da posição será feita da esquerda para direita.

Mais adiante, iremos ver como pesquisar por trechos de uma *string*.

A função `instr()` possui muitas variações e nem todas serão cobradas na certificação. Quando eu fiz o exame, foi cobrada uma questão de como pesquisar usando parâmetros negativos, como -1 . Além do uso padrão da função e qual será o retorno quando for feito uma consulta que a inclua.

Outra possível questão é identificar o tamanho da *string*. Para isto, usaremos a função `length()` , juntamente com o texto que iremos pesquisar.

```
SQL> select length('renan') from dual;
```

```
LENGTH('RENAN')
-----
5
```

Ela basicamente contou quantos caracteres o termo de pesquisa tem. Um variação da função é `lengthb` , que informa quanto bytes o termo de pesquisa tem. No caso, do nome `renan` , teremos a mesma a quantidade.

```
SQL> select lengthb('renan') from dual;
```

```
LENGTHB('RENAN')
-----
5
```

A variação `instrb()` também irá retornar o resultado em bytes, porém, a posição.

```
SQL> select instrb('desktop/compras.txt', '.', -1, 1) from dual;
```

```
INSTR('DESKTOP/COMPRAS.TXT', '.', -1, 1)
-----
20
```

Como cada caractere costuma ocupar um byte, o resultado foi semelhante a outra função.

E vamos ver o que acontece quando passamos um valor `null` para a função `lengthb()` ?

```
SQL> select length(null) from dual;
```

```
LENGTHB(NULL)  
-----
```

Os dois últimos parâmetros também devem ser numéricos. Caso seja usado letras ou caracteres ele irá retornar uma mensagem de `identificador invalido`.

```
SQL> select instrb('desktop/compras.mes.txt','.', 'a', 'a') from dual;  
select instrb('desktop/compras.mes.txt','.', 'a', 'a') from dual
```

```
ERRO na linha 1:  
ORA-01722:numero invalido
```

Se um dos parâmetros da busca for `null`, o retorno também será.

```
SQL> select instrb(null, '.', 'a', 'a') from dual;
```

```
INSTRB(NULL, '.', 'A', 'A')  
-----
```

Conhecemos mais algumas funções do Oracle, mais adiante veremos como transformar todos os caracteres em caixa alta ou caixa baixa.

