01

Conhecendo as funções existentes no Oracle

Transcrição

Bem-vindos ao nosso curso sobre a Certificação Oracle. No curso anterior, aprendemos a selecionar os nossos dados de várias formas, aplicando filtros. Neste curso nós iremos aprender algumas funções para manipular os dados que nós recebemos através do SELECT. Para isto, vamos disponibilizar um arquivo para trabalharmos com alguns dados.

Vamos abrir o terminal, vou até o `Desktop` e usarei o comando `dir` .

```
C:\Users\Alura\Desktop>dir
 Volume in drive C is BOOTCAMP
 Volume Serial Number is CE¨)-34C1

 Directory of C:\Users\Alura\Desktop

22/04/2016  10:57    <DIR>          .
22/04/2016  10:57    <DIR>          ..
10/04/2016  08:16    <DIR>          oracle
22/04/2016  10:27             1.529 sql-oracle-3.sql
20/04/2016  09:41         5.262.916 testel.trec
               2 File(s)      5.264.445 bytes
               3 Dir(s)  358.086.397.952 bytes fres
```

Ele irá mostrar o arquivo `sql-oracle-3.sql` , que iremos utilizar agora.

Vamos entrar no console com `sqlplus` e depois, logaremos com o usuário `alura` .

```
C:\Users\Alura\Desktop>sqlplus

SQL \*Plus: Release 12.1.0.2.0 Production on Sex Abr 22 11:12:41 2016

Copyright (c) 1982, 2014, Oracle. All rights reserved.

Informe o nome do usuario: alura
Informe a senha:
Horario do ultimo log-in bem-sucedido: Sex Abr 22 2016 10:26:36 -03:00

Conectado a:
Oracle Database 12c Enterprise Edition Release 12.1.0.2.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Tasting options
```

No curso anterior, nós criamos a tabela `funcionarios` . O nosso arquivo já irá removê-la.

Par a iniciar o arquivo, nós iremos usar o comando `start` e o nome do mesmo.

```
SQL> start sql-oracle-J.sql
```

Ao executarmos, a tabela será criada, com várias linhas. Mas, qual tabela foi criada? Com um novo SELECT podemos verificar o nome da tabela.

```
SQL> select table_name from user_tables;

TABLE_NAME
------------------------------------------------------------------------------------------------------------------------
FUNCIONARIOS

SQL>
```

E para enxergarmos a estrutura da tabela, podemos usar o comando `desc funcionario` .

```
SQL> desc funcionarios;
 Nome                          Nulo?    Tipo
 ----------------------------- -------- ------------------
 ID                            NOT NULL NUMBER(11)
 NOME                          NOT NULL VARCHAR2(20)
 SOBRENOME                     NOT NULL VARCHAR2(20)
 SALARIO                                NUMBER(10,2)
 PC_COMISSAO                            NUMBER(10,2)

 SQL>
```

A estrutura inclui: `id` , `Nome` , `Sobrenome` , `Salario` e `Porcentagem de Comissão` . Para termos acesso aos dados da tabela, usaremos a *query*:

```
SQL> select * from funcionarios;

        ID NOME                 SOBRENOME          SALARIO PCT_COMISSAO
---------- -------------------- --------------- --------- ------------
         1 Kauan                Silva                1100           ,3
         2 Jose                 Souza                1200           ,3
         3 Eduarda              Oliveira             1200           ,2
         4 Leonardo             Amaral               3700
         5 Thais                Silveira             6700
         6 Nicole               Moraes               1700           ,3
         7 Leticia              Moreira                             ,9
         8 Joao                 Silva                6700           ,3
         9 Igor                 Pires                1500            0
        10 Diogo                Paizano              1500           ,2
        11 Karlos               Pereira              1500

        ID NOME                 SOBRENOME          SALARIO PCT_COMISSAO
---------- -------------------- --------------- --------- ------------
        12 Stevan               Rodrigues            1500
        13 Otavio               Martez               1500
```

Vemos listados vários funcionários, os salários e as porcentagens de comissão recebida por cada um deles. Mas observe que alguns não temum salário definido e recebem apenas comissão. Este campos vazios no Oracle, costumamos dizer que é `null` , referente a **nulo**. Trabalhar com um valor é um pouco mais complicado. Veremos o porquê.

Imagine que queremos gerar um relatório, e precisamos multiplicar o salário pela porcentagem da comissão. O nosso SELECT ficaria assim:

```
SQL> select salario * pct_comissao`
```

Nós iremos também adicionar um **alias** (um apelido), `salarioFinal` , além do nome dos funcionários e da tabela.

```
SQL> select nome,salario + salario * pct_comissao` as salarioFinal from funcionarios;

NOME                 SALARIOFINAL
-------------------- ------------
Kauan                         330
Jose                          360
Eduarda                       240
Leonardo
Thais
Nicole                        510
Leticia
Joao
Igor                            0
Diogo                         300
Karlos

NOME                 SALARIOFINAL
-------------------- ------------
Stevan
Otavio

13 linhas selecionadas.
```

No caso, ele fez apenas o cálculo do valor da comissão. Para termos o valor total do que é recebido pelos funcionários, teremos que somar o salário, com o valor da comissão.

```
SQL> select nome, salario * pct_comissao as salarioFinal from funcionarios;

NOME                 SALARIOFINAL
-------------------- ------------
Kauan                        1430
Jose                         1560
Eduarda                      1440
Leonardo
Thais
Nicole                       2210
Leticia
Igor                         6700
Diogo                        1800
Karlos                       1500

NOME                 SALARIOFINAL
-------------------- ------------
Stevan
Otavio
```

```
13 linhas selecionadas.
```

Agora, temos os valores corretos do `salarioFinal`. Mas ainda vemos algumas pessoas com um valor nulo. Isto acontece, porque eles têm a porcentagem da comissão especificada como nula na tabela. Nós queremos que, ainda que o valor seja nulo, ele retorne o valor do salário. Vale ressaltar que nulo não é igual a 0, ele indica que não temos valor nenhum. Se nulo fosse igual a 0, com o nosso SELECT conseguiríamos calcular o valor correto. Precisamos resolver isto.

Quando eu fiz o exeme da certificação, duas questões falaram sobre NULL. Este é um tema recorrente na prova, porque o NULL tem alguns comportamentos inesperados, e é fácil cometer erros. Mas temos **funções** que nos permitem trabalhar com os valores nulos. A primeira é a `nvl()` (de *null values*), que permite substituir o valor de NULL por outro. Quando a porcentagem da comissão for nula, queremos que seja substituída por zero. Mas, quando a porcentagem estiver definida, queremos que o valor especificado seja utilizado.

Então, vamos usar a função `nvl()` que irá receber dois parâmetros: a coluna `pct_comissao` e o valor susbtitutivo, `0`.

```
SQL> select nome, salario * nvl(pct_comissao,0) as salarioFinal from funcionarios;

NOME                 SALARIOFINAL
-------------------- ------------
Kauan                        1430
Jose                         1560
Eduarda                      1440
Leonardo                     3700
Thais                        6700
Nicole                       2210
Leticia
Joao
Igor                         6700
Diogo                        1800
Karlos                       1500

NOME                 SALARIOFINAL
-------------------- ------------
Stevan
Otavio

13 linhas selecionadas.
```

Minha consulta listou todos os funcionário, menos a Letícia e o João, porque os dois tem os valores da coluna `salario` nulos. Iremos adicionar o `nvl()`, que inclua como parâmetro `salario` e o valor mínimo, R$800. Também precisarei incluir `salario` dentro do segundo `nvl`.

```
SQL> select nome,nvl(salario, 800) + nvl(salario * pct_comissao,0) as salarioFinal from funcionarios

NOME                 SALARIOFINAL
-------------------- ------------
Kauan                        1430
Jose                         1560
Eduarda                      1440
Leonardo                     3700
Thais                        6700
Nicole                       2210
```

```
Leticia                         800
Joao                            800
Igor                           6700
Diogo                          1800
Karlos                         1500

NOME                 SALARIOFINAL
-------------------- ------------
Stevan                         1500
Otavio                         1500

13 linhas selecionadas.
```

Agora, todos os campos foram preenchidos. As duas pessoas que tinham um valor nulo no salário, passaram a receber R$800.

Nós aprendemos a trabalhar com valores nulos. Mas, e se quisermos incluir um bônus de 10% na comissão, para quem a recebe além do salário. Como podemos fazer isto? Observe que até aqui, definimos que quando o valor de uma comissão for nulo, ele será substituído por zero. Agora, se o valor não for nulo, deveremos aumentar 10%. Para esta consulta, iremos usar o `nvl2()`, que é bastante semelhante ao `nvl` anterior, mas receberá três argumentos. O primeiro argumento é a expressão `salario * pct_comissao`. O segundo, será referente ao caso de que a comissão não seja nula, e logo, receberá o bônus, `salario * pct_comissao * 1.1`. O terceiro argumento será no caso de que a comissão seja nula, então, será substituída por 0. Agora, podemos customizar o valor, caso a expressão não seja nula.

```
SQL> select nome, nvl(salario,800) + nvl2(salario * pct_comissao,salario * pct_comissao * 1.1,0);

NOME                 SALARIOFINAL
-------------------- ------------
Kauan                          1463
Jose                           1596
Eduarda                        1464
Leonardo                       3700
Thais                          6700
Nicole                         2261
Leticia                         800
Joao                            800
Igor                           6700
Diogo                          1830
Karlos                         1500

NOME                 SALARIOFINAL
-------------------- ------------
Stevan                         1500
Otavio                         1500

13 linhas selecionadas.
```

O que é `nvl()` e `nvl2()`? Eles são ***Single-Row Functions***, o que traduzido, significa que são funções de linha única. Graças a isto, quando executamos o `nvl()`, a função será aplicada para cada registro da tabela.

Mais adiante, iremos conversar sobre funções de agrupamento e iremos esclarecer as diferenças de funções. Nós já aprendemos a trabalhar com um pouco de valores nulos e também aprendemos o que é uma *single row function*.