13

Consolidando o seu conhecimento

Chegou a hora de você pôr em prática o que foi visto na aula. Para isso, execute os passos listados abaixo.

1) Abra o arquivo **SQL_10.sql**, que você baixou no início desta aula, e copie o seu conteúdo.

2) Acesse o **SQL Developer**, crie uma nova consulta, cole o conteúdo do arquivo **SQL_10.sql** e execute-o. As tabelas serão apagadas, recriadas e novos registros incluídos.

3) Verifique o conteúdo das tabelas de produtos e clientes, digitando:

```
SELECT * FROM TB_CLIENTES;
SELECT * FROM TB_PRODUTOS;
```

4) Crie uma nova consulta no **SQL Developer**. Digite:

```
SELECT
    CPF, NOME, ENDERECO1, ENDERECO2,
    BAIRRO, CIDADE, ESTADO, CEP, IDADE,
    SEXO, LIMITE_CREDITO, VOLUME_COMPRA,
    PRIMEIRA_COMPRA, DATA_NASCIMENTO
FROM TB_CLIENTES;
```

Ou seja, você pode usar o `*` para selecionar todos os campos ou discriminar um por um.

5) Você pode selecionar alguns campos apenas:

```
SELECT CPF, NOME, IDADE, DATA_NASCIMENTO
FROM TB_CLIENTES;
```

6) Ou também criar um *alias* para cada campo, por exemplo:

```
SELECT
    CPF AS CPF_DO_CLIENTE,
    NOME AS NOME_DO_CLIENTE,
    IDADE,
    DATA_NASCIMENTO
FROM TB_CLIENTES;
```

7) Os registros podem ser filtrados usando o mesmo tipo de cláusula `WHERE` utilizada no `UPDATE` e `DELETE`:

```
SELECT * FROM TB_PRODUTOS WHERE PRODUTO = '1037797';
```

8) Mas não é somente pela chave primária que você pode filtrar as consultas:

```
SELECT * FROM TB_CLIENTES WHERE CIDADE = 'Rio de Janeiro';
SELECT * FROM TB_PRODUTOS WHERE SABOR = 'Laranja';
```

9) Digite o comando abaixo de `UPDATE`, para fazer uma alteração em diversos registros ao mesmo tempo:

```
UPDATE TB_PRODUTOS SET SABOR = 'Cítricos' WHERE SABOR = 'Limão';
```

10) Você pode fazer consultas utilizando condições baseadas em números (decimais ou inteiros). Abaixo, é uma condição de igualdade:

```
SELECT * FROM TB_CLIENTES WHERE IDADE = 22;
```

11) Mas você usar sinal de maior, menor, maior ou igual, menor ou igual. Olhe alguns exemplos:

```
SELECT * FROM TB_CLIENTES WHERE IDADE > 22;
SELECT * FROM TB_CLIENTES WHERE IDADE < 22;
SELECT * FROM TB_CLIENTES WHERE IDADE <= 22;
```

12) Há também o sinal de diferente, que é expresso como `<>`. Veja abaixo:

```
SELECT * FROM TB_CLIENTES WHERE IDADE <> 22;
```

13) As condições de maior, menor, maior ou igual, menor ou igual, e diferente podem ser aplicadas a textos. O critério será a ordem alfabética:

```
SELECT * FROM TB_CLIENTES WHERE NOME >= 'Fernando Cavalcante';
SELECT * FROM TB_CLIENTES WHERE NOME <> 'Fernando Cavalcante';
```

14) As condições de maior, menor, maior ou igual, menor ou igual, igual e diferente podem ser aplicadas a campos `FLOAT`:

```
SELECT * FROM TB_PRODUTOS WHERE PRECO_LISTA >= 16.008
SELECT * FROM TB_PRODUTOS WHERE PRECO_LISTA = 16.008
SELECT * FROM TB_PRODUTOS WHERE PRECO_LISTA <> 16.008
```

15) Você pode usar datas como filtro. Veja alguns exemplos:

```
SELECT * FROM TB_CLIENTES WHERE DATA_NASCIMENTO = TO_DATE('07/10/1995','DD/MM/YYYY');
SELECT * FROM TB_CLIENTES WHERE DATA_NASCIMENTO > TO_DATE('07/10/1995','DD/MM/YYYY');
SELECT * FROM TB_CLIENTES WHERE DATA_NASCIMENTO < TO_DATE('07/10/1995','DD/MM/YYYY');
```

16) Existem algumas funções de data que podem ser usadas como filtros:

```
SELECT * FROM TB_CLIENTES WHERE TO_CHAR(DATA_NASCIMENTO,'MM') = 9
```

17) Por fim, você pode usar filtros compostos, usando, entre cada teste, os comandos `AND` ou `OR`. Veja abaixo alguns exemplos que podem ser testados no **SQL Developer**:

```
SELECT * FROM TB_PRODUTOS WHERE PRECO_LISTA >= 16.007;
SELECT * FROM TB_PRODUTOS WHERE PRECO_LISTA >= 16 AND PRECO_LISTA <= 21
SELECT * FROM TB_CLIENTES WHERE IDADE > 22 AND SEXO = 'F'
SELECT * FROM TB_CLIENTES WHERE CIDADE = 'Rio de Janeiro' OR BAIRRO = 'Jardins';
SELECT * FROM TB_CLIENTES WHERE (IDADE > 22 AND SEXO = 'F') AND (CIDADE = 'Rio de Janeiro' OR BAIRRO
```