02

Alterando e restringindo o formato das nossas tabelas

Não quero inserir uma observação, e agora?

E se não quiséssemos colocar uma observação? Em vez de deixarmos em branco, podemos usar `NULL`, um tipo especial suportado pelos bancos de dados pra representar vazio.

Veja o seguinte comando SQL:

```
INSERT INTO COMPRAS (ID, VALOR, DATA, OBSERVACOES, RECEBIDO) VALUES (ID_SEQ.NEXTVAL, 100.0, '02
```

Repare o `NULL`. Como dito anteriormente, isso indica que não queremos passar uma observação. É muito comum ver o uso de `NULL` nos mais diversos tipos de banco de dados.

Evitando a inserção de valores nulos na nossa tabela

Mas às vezes queremos evitar que isso aconteça, proibindo o usuário de inserir um valor nulo na tabela. Podemos pedir ao banco de dados que rejeite nulos nas colunas. Para isso, precisamos alterar a tabela e dizer que a coluna `OBSERVACOES` não aceitará mais valores nulos:

```
ALTER TABLE COMPRAS MODIFY (OBSERVACOES VARCHAR2(30) NOT NULL);
```

Basicamente, o comando acima altera (`ALTER`) a tabela (`TABLE`) COMPRAS, e modifica (`MODIFY`) a coluna `OBSERVACOES`, mantendo seu tipo (`VARCHAR2(30)`), mas dizendo que ela não deve ser nula (`NOT NULL`).

Se tentarmos agora executar o `INSERT` acima, ele falhará, pois o banco recusará o valor nulo para a coluna `OBSERVACOES`.

Atribuindo um valor padrão para uma coluna específica

Podemos ainda definir mais coisas para as nossas tabelas. Vamos supor que a regra de negócio do nosso sistema diz que todas as compras **nunca** são entregues de primeira. Ou seja, por padrão todas as compras criadas começam como não entregues. Podemos falar ao banco que o valor padrão (`DEFAULT`) da coluna `RECEBIDO` é falso (ou seja, 0).

Para isso, vamos alterar novamente a tabela:

```
ALTER TABLE COMPRAS MODIFY (RECEBIDO CHAR DEFAULT '0' CHECK (RECEBIDO IN (0,1)));
```

Novamente, fizemos um `MODIFY`, mas dessa vez na coluna `RECEBIDO`. Veja que setamos o valor padrão para 0 através do `DEFAULT '0'`. Poderíamos também setar valores padrão para qualquer uma das colunas.

Caso a coluna tenha um valor padrão, você não precisa passá-lo no comando `INSERT`. Por exemplo, agora que `RECEBIDO` possui um valor default, é possível pulá-lo na instrução:

```
INSERT INTO COMPRAS (ID, VALOR, DATA, OBSERVACOES) VALUES (ID_SEQ.NEXTVAL, 189.76, '09-JAN-2009
```

Como não passamos o valor do `RECEBIDO`, o Oracle entende que deve ser inserido o valor padrão, ou seja, 0.

Adicionando novas colunas à tabela

Vamos agora criar uma nova coluna para guardar a forma de pagamento de cada compra. Nosso sistema deve suportar compras por cartão de crédito, boleto bancário e dinheiro. Podemos criar uma coluna do tipo `VARCHAR2` ou do tipo `INT` para guardar essa forma de pagamento.

O problema é que se declararmos essa coluna vai aceitar qualquer valor, inclusive formas de pagamento inexistentes. Podemos avisar ao Oracle que essa coluna só deve aceitar valores como 'CARTAO', 'BOLETO' e 'DINHEIRO', por exemplo. Vamos criar essa coluna:

```
ALTER TABLE COMPRAS ADD (FORMA_PAGT VARCHAR2(15) CHECK (FORMA_PAGT IN ('CARTAO', 'BOLETO', 'DINH
```

Veja a sintaxe: declaramos a forma de pagamento (`FORMA_PAGT`) como varchar2 e checamos a lista de itens válidos para ele. Isso significa que se fizermos um INSERT com algum valor diferente do que estiver listado, o banco irá recusar. Veja a criação de uma compra feita por cartão de crédito:

```
INSERT INTO COMPRAS (ID, VALOR, DATA, OBSERVACOES, FORMA_PAGT) VALUES (ID_SEQ.NEXTVAL, 189.76,
```

Vi que o nome da coluna está errado, como corrigir?

Mas repare que demos um nome estranho à coluna (`FORMA_PAGT`). Vamos renomeá-la (`RENAME COLUMN`) para (`TO`) `FORMA_PAGTO`, usando o comando `ALTER TABLE`:

```
ALTER TABLE COMPRAS RENAME COLUMN FORMA_PAGT TO FORMA_PAGTO;
```

O comando `ALTER TABLE` nos permite alterar a tabela já existente. Ele é bem extenso e você deveria pesquisar mais a respeito!