

INFORMA, FORMA, TRANSFORMA.

Arquivo Morto

Um elemento muito importante em manutenção de BD é o arquivo morto (tabela morta). Ele deve ser uma tabela com o mesmo formato da usual (tabela ativa), que será utilizada para armazenar os registros removidos da tabela ativa.

Ao projetar um BD e suas tabelas, é muito comum esquecer que muitas vezes os registros não podem ser removidos definitivamente da tabela ativa. É ideal que se mantenha em algum lugar uma cópia do registro removido da tabela ativa.

Por exemplo, na situação de um cadastro de funcionários, não é conveniente remover o registro de um determinado funcionário sem antes inseri-lo no arquivo morto. O registro transferido para o arquivo morto pode ser reativado a qualquer momento na tabela ativa caso o funcionário seja readmitido.


3


Arquivo Morto (Cont.)

Primeiro, crie uma nova tabela (que será a tabela morta), com a mesma estrutura da tabela ativa. Vamos continuar trabalhando com a nossa tabela **funcionario**.

Será criada então a tabela **morto**:

```
CREATE TABLE morto (
    CODFUN      INTEGER PRIMARY KEY,
    NOME        VARCHAR (40),
    DEPTO       CHAR( 2),
    FUNCAO      CHAR(20),
    SALARIO     DECIMAL(10,2),
    ADMISSAO    DATE
);
```

Obs: A estrutura da tabela denominada **morto** é semelhante à estrutura da tabela **funcionario**. A diferença está no fato de não ser necessário utilizar nos campos da tabela **morto** a definição **NOT NULL**, uma vez que o preenchimento dessa tabela não será executado como foi o da **funcionario**. Seu preenchimento ocorre a partir dos registros que já se encontram cadastrados na tabela **funcionario**. A única semelhança é com relação ao campo **CODFUN** que será a chave primária em ambas as tabelas.


4

Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Arquivo Morto (cont.)

Para arquivar um registro em um arquivo morto antes de removê-lo da tabela ativa, é necessário usar a instrução **INSERT INTO** anexa ao comando **SELECT**, conforme a sintaxe indicada em seguida:


```
INSERT INTO <arquivo morto>
  SELECT <campos>
  FROM <tabela ativa>
  WHERE <condição>
;
```

Nome da a tabela que guardará a transferência do registro.

Campos que serão transferidos. Não pode usar * para todos.

Nome da tabela ativa (de onde o registro será retirado)

É a definição do controle de ação para a efetivação do comando.



5

Sistema FIRJAN | **INFORMA, FORMA, TRANSFORMA.**

Arquivo Morto (cont.)


No nosso exemplo de estudo, consideremos que o funcionário cujo **CODFUN = 4** pediu demissão. Então faremos o seguinte:

```
INSERT INTO morto
  SELECT CODFUN, NOME, DEPTO, FUNCAO, SALARIO,
  ADMISSAO
  FROM funcionario
  WHERE CODFUN = 4
;
```

Sempre visualize os registros em ambas as tabelas antes e depois de cada alteração. Faça disto um hábito.

Agora que o registro foi copiado para morto, exclua-o de funcionario com o seguinte comando:

```
DELETE FROM funcionario WHERE CODFUN = 4;
```



6

O Uso de Subconsultas

O **MySQL** disponibiliza um recurso muito útil quando há necessidade de fazer consultas encadeadas em uma tabela de uma base de dados. Trata-se de um recurso denominado subconsulta (*subquery*), que é uma consulta definida após a cláusula **WHERE** do comando **SELECT**.

Por exemplo, considere a necessidade de fazer uma consulta para apresentar os nomes dos registros de todos os funcionários da tabela **funcionario** que possuam seu salário igual aos salários dos funcionários demitidos que estejam cadastrados na tabela **morto**. Utilize a instrução:

```
SELECT NOME FROM funcionario WHERE SALARIO =  
(SELECT SALARIO FROM morto);
```



7

O Uso de Subconsultas (Cont.)



Após a instrução anterior são apresentados os registros da tabela **funcionario** que possuem salário igual ao salário dos registros contidos em **morto**.

Vantagens de se utilizar subconsultas:

- Possibilidade de efetuar consultas que estejam estruturadas, assim é possível isolar cada parte de uma instrução.
- Possibilidade de fornecer modos alternativos para realizar operações que, de outra forma, exigiriam instruções baseadas nos comandos JOIN e UNION (que são comandos complexos).
- O aumento da legibilidade e definição de consultas complexas de forma bem simples.



8


  INFORMA, FORMA, TRANSFORMA.

O Uso de Subconsultas (Cont.)

Apresente uma consulta de todos os funcionários da tabela funcionario que tenham o número de departamento igual ao número de departamento do(s) registro(s) existente(s) na tabela morto.

A instrução então é a seguinte:

```
SELECT * FROM funcionario WHERE DEPTO = (SELECT  
DEPTO FROM morto);
```


 9

  INFORMA, FORMA, TRANSFORMA.


Dúvidas?




10



Bibliografia



MySQL 5.1 Interativo (3ª Edição)
José Augusto N.G. Manzano
Ed. Érica



MySQL – Guia de Bolso (2ª Edição)
George Reese
Ed. Alta Books

11