SESI SENAI

PELO FUTURO DO TRABALHO

03- Manipulação de dados DML

Professor Hermano Roepke

Adicionando uma nova linha

Com os comandos DML você poderá:

- adicionar novas linhas a uma tabela
- modificar linhas existentes em uma tabela
- remover linhas existentes em uma tabela

Lembre-se de que "linha" é o sinônimo de "registro" em uma tabela de dados.

Em alguns SGBDS, como é o caso do Oracle, por default, quando um dos comandos acima é executado, automaticamente uma transação é criada.

Uma transação é uma unidade lógica de trabalho que contempla um ou mais comandos DML.

Adicionando uma nova linha

A representação gráfica ilustra a inserção de uma nova linha na tabela DEPARTAMENTO.

NOVA LINHA

DESENVOLVIMENTO	DETROIT	50

DEPARTAMENTO

ID_DEPARTAMENTO	NM_DEPARTAMENTO	LOCALIZACAO
10	CONTÁBIL	NEW YORK
20	PESQUISA E DESENVOLVIMENTO	DALLAS
30	VENDAS	CHICAGO
40	OPERAÇÕES	BOSTON

DEPARTAMENTO

ID_DEPARTAMENTO	NM_DEPARTAMENTO	LOCALIZACAO
10	CONTÁBIL	NEW YORK
20	PESQUISA E DESENVOLVIMENTO	DALLAS
30	VENDAS	CHICAGO
40	OPERAÇÕES	BOSTON
50	DESENVOLVIMENTO	DETROIT

```
INSERT INTO departamento(id_departamento,
nm_departamento, localizacao) VALUES (50,
'DESENVOLVIMENTO', 'DETROIT');
```

O comando INSERT

Adicione novas linhas em uma tabela usando o comando INSERT. Veja a sintaxe:

```
INSERT INTO tabela [(coluna1[, coluna...])]
VALUES (valor [,valor...]);
```

Somente uma linha é inserida por vez com está sintaxe.

Cada coluna deve apresentar um valor correspondente na cláusula VALUES.

Inserindo novas linhas

- Insira uma nova linha contendo valores para cada coluna.
- Liste valores na ordem default das colunas na tabela.
- Liste opcionalmente as colunas na cláusula INSERT.
- Coloque os valores de data e caracteres entre apóstrofos (também conhecidos como aspas simples).

Exemplo:

```
INSERT INTO departamento(id_departamento, nm_departamento, localizacao)
VALUES (50, 'DESENVOLVIMENTO', 'DETROIT');
```

Inserindo linhas com valores nulos

- **Método implícito:** omita a coluna cujo valor você não deseja inserir na lista de colunas. Isso fará com que a coluna receba a informação de que está nula.

```
INSERT INTO departamento(id_departamento, nm_departamento)
VALUES (60, 'RECURSOS HUMANOS');
```

- **Método explícito:** especifique a palavra-chave NULL para o campo que você deseja ignorar (não informar um valor).

```
INSERT INTO departamento
VALUES (70, 'FINANÇAS', NULL);
```

Obs.: Para que você consiga omitir o preenchimento de uma coluna, ela não pode estar definida como NOT NULL ou fazer parte da chave primária da tabela.

Copiando linhas a partir de outra tabela

Crie a instrução INSERT com uma subconsulta. Veja um exemplo:

- Não use a cláusula VALUES.
- Faça a correspondência do número de colunas na cláusula INSERT com o número de colunas na subconsulta.

```
INSERT INTO GERENTES (id, nome, salario, data_contratacao)
        SELECT id_empregado, nm_empregado, salario, data_contratacao
        FROM empregado
        WHERE FUNCAO = 'GERENTE';
```

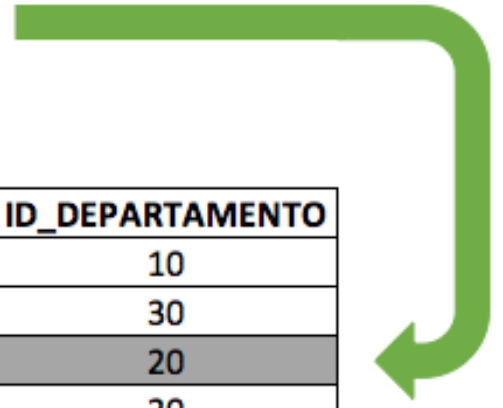
Alterando os dados em uma tabela

A representação gráfica a seguir ilustra a alteração do valor da coluna ID_DEPARTAMENTO no registro funcionário "CLARK". Veja:

EMPREGADO

ID_EMPREGADO	NM_EMPREGADO	FUNCAO	...	COMISSAO	ID_DEPARTAMENTO
7839	KING	PRESIDENTE			10
7698	BLAKE	GERENTE			30
7782	CLARK	GERENTE			10
7566	JONES	GERENTE			20

... atualize uma linha em uma tabela



EMPREGADO

ID_EMPREGADO	NM_EMPREGADO	FUNCAO	...	COMISSAO	ID_DEPARTAMENTO
7839	KING	PRESIDENTE			10
7698	BLAKE	GERENTE			30
7782	CLARK	GERENTE			20
7566	JONES	GERENTE			20

O comando UPDATE

Modifique linhas existentes com o comando. Conheça a sintaxe do comando:

```
UPDATE tabela
SET coluna = valor [, coluna = valor, ...]
[WHERE condição];
```

Atualize mais de uma linha por vez, se necessário, mas fique atento(a) à condição da cláusula WHERE, pois a sua ausência significa que todos os registros serão afetados.

Na maioria das vezes o(s) campo(s) que compõe(m) a chave primária da tabela é (são) utilizado(s) na cláusula WHERE.

Atualizando linhas em uma tabela

Uma linha ou linhas específicas são modificadas quando você especifica a cláusula WHERE.

```
UPDATE empregado
SET id_departamento = 20
WHERE id_empregado = 7782;
```

Todas as linhas na tabela são modificadas quando você omite a cláusula WHERE.

```
UPDATE empregado
SET id_departamento = 20;
```

Atualizando linhas: erro de restrição de integridade

Observe no exemplo a seguir a tentativa de alterar um registro da tabela EMPREGADO :

```
UPDATE empregado
SET id_departamento = 55
WHERE id_departamento = 10;
```

A execução deste comando deverá gerar a mensagem abaixo. Isso corre porque não existe o departamento 55 na tabela DEPARTAMENTO, e como há uma restrição de integridade referencial entre as tabelas EMPREGADO e DEPARTAMENTO, o SGBD não permitirá a alteração, visto que somente valores existentes da tabela DEPARTAMENTO poderão ser utilizados da tabela EMPREGADO.

```
Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails
(`hermanoempregadoresa`.`empregado`, CONSTRAINT `empregado_ibfk_1` FOREIGN KEY
(`ID_DEPARTAMENTO`) REFERENCES `departamento` (`ID_DEPARTAMENTO`))
```

Removendo uma linha de uma tabela

A representação gráfica ilustra a remoção do departamento "DESENVOLVIMENTO" da tabela DEPARTAMENTO. Veja:

DEPARTAMENTO

ID_DEPARTAMENTO	NM_DEPARTAMENTO	LOCALIZACAO
10	CONTÁBIL	NEW YORK
20	PESQUISA E DESENVOLVIMENTO	DALLAS
30	VENDAS	CHICAGO
40	OPERAÇÕES	BOSTON
50	DESENVOLVIMENTO	DETROIT
60	FINANÇAS	
...		

... remova uma linha em uma tabela

DEPARTAMENTO

ID_DEPARTAMENTO	NM_DEPARTAMENTO	LOCALIZACAO
10	CONTÁBIL	NEW YORK
20	PESQUISA E DESENVOLVIMENTO	DALLAS
30	VENDAS	CHICAGO
40	OPERAÇÕES	BOSTON
60	FINANÇAS	
...		

O comando DELETE

Você pode remover linhas existentes de uma tabela usando o comando DELETE. Conheça a sintaxe do comando:

```
DELETE [FROM] tabela
[WHERE condição];
```

É possível remover mais de uma linha por vez, se necessário, mas fique atento(a) à condição da cláusula WHERE, pois a sua ausência significa que todos os registros serão afetados (neste caso, removidos).

Na maioria das vezes o(s) campo(s) que compõe(m) a chave primária da tabela é (são) utilizado(s) na cláusula WHERE.

Deletando linhas de uma tabela

Linhas específicas são removidas quando você utiliza a cláusula WHERE com uma condição válida. Veja o exemplo a seguir:

```
DELETE FROM departamento
WHERE id_departamento = 50;
```

Já no exemplo a seguir, todas as linhas na tabela serão removidas, pois foi omitida a cláusula WHERE.

```
DELETE FROM departamento;
```

Obs.: Ao executar o segundo exemplo apresentado, você vai perceber que o SGBD não permitirá a exclusão dos registros, pois alguns registros afetados estão sendo referenciados na tabela EMPREGADO. Este é mais um exemplo da ação da integridade referencial.

Deletando linhas: erro de restrição de integridade

Observe no exemplo a seguir a tentativa de remoção de um registro da tabela DEPARTAMENTO:

```
DELETE FROM departamento
WHERE id_departamento = 10;
```

A execução deste comando deverá gerar a mensagem abaixo. Isso ocorre porque você não pode remover um registro que está sendo referenciado como chave estrangeira em outra tabela. No contexto, a tabela EMPREGADO apresenta uma chave estrangeira para tabela DEPARTAMENTO, lembra-se?

```
Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails
(`hermanoempregadoresa`.`empregado`, CONSTRAINT `empregado_ibfk_1` FOREIGN
KEY (`ID_DEPARTAMENTO`) REFERENCES `departamento` (`ID_DEPARTAMENTO`))
```

Transações de banco de dados

“Uma transação é uma unidade lógica de trabalho que contempla um ou mais comandos DML”.

E qual a importância da transação para os SGBDS?

Talvez você não se lembre, mas uma das funções de um SGBD é garantir a consistência dos dados e, principalmente, possibilitar que o desenvolvedor utilize esse recurso para agrupar comandos DML (inserção, alteração e remoção de registros) de tal forma que atenda o propósito em sua totalidade.

Transações de banco de dados

Considere um o cenário de uma instituição financeira (banco). Quando o cliente de um banco transfere dinheiro de uma conta de poupança para uma conta corrente, a transação consiste em três operações separadas: diminuir a conta poupança, aumentar a conta corrente e registrar a operação no lançamento da transação.

O SGBD deve garantir que todas as três instruções SQL sejam executadas para manter as contas com um saldo apropriado. Quando algo impedir que um dos comandos da transação seja executado, os outros comandos da transação deverão ser desfeitos.

Transações de banco de dados

Começa quando for executado o primeiro comando SQL executável (INSERT, UPDATE, DELETE).

Termina com um dos seguintes eventos:

- COMMIT ou ROLLBACK é emitida;
- o usuário fecha a sessão com o SGBD;
- o sistema é fechado.

Vantagens do uso dos comandos COMMIT e ROLLBACK:

- garantir consistência dos dados;
- visualizar alterações nos dados antes de fazer as alterações permanentemente;
- Agrupar operações relacionadas logicamente.

Estado dos dados

Antes da execução do comando COMMIT ou ROLLBACK:

- o estado anterior dos dados manipulados podem ser recuperado com a execução do comando ROLLBACK;
- o usuário atual pode revisar os resultados das operações DML usando o comando SELECT e assim garantir que o propósito foi contemplado;
- outros usuários não poderão ver os resultados dos comandos DML do usuário atual até que ele execute o comando COMMIT;
- os registros afetados são bloqueados: assim, outros usuários não poderem alterar os dados dentro dos registros afetados.

Estado dos dados

Depois da execução do comando COMMIT OU ROLLBACK:

- as alterações nos dados são feitas permanentemente no banco de dados;
- o estado anterior dos dados é perdido permanentemente.
- todos os usuários podem ver os resultados;
- os registros são desbloqueados e ficam disponíveis para serem manipulados por outros usuários.

Submetendo dados a COMMIT

Observe a sequência de dados abaixo. O primeiro deles realiza a modificação em um registro da tabela EMPREGADO.

```
UPDATE empregado
SET id_departamento = 10
WHERE id_empregado = 7782;
```

O segundo comando faz a efetivação (confirmação da alteração).

```
COMMIT;
```

Submetendo dados a ROLLBACK

Se você deseja descartar todas as alterações pendentes, é necessário executar o comando ROLLBACK. Após essa ação, as alterações nos dados são desfeitas, o estado anterior dos dados é restaurado e as linhas afetadas são desbloqueadas.

Veja um exemplo onde o comando delete foi executado de forma equivocada, uma vez que não se pretendia remover todos os registros da tabela EMPREGADO. No contexto, logo em seguida foi submetido o comando ROLLBACK:

```
DELETE FROM empregado;
ROLLBACK;
```

Resumo

Instrução	Descrição
INSERT	Adiciona uma nova linha à tabela
UPDATE	Modifica linhas existentes na tabela
DELETE	Remove linhas existentes da tabela
COMMIT	Torna permanentes todas as alterações pendentes
ROLLBACK	Descarta todas as alterações nos dados pendentes

SESI SENAI

PELO FUTURO DO TRABALHO