MySQL – INSERT, TRANSACTION, UPDATE e DELETE

Ewerton J. Silva, BingChar e CHAT GPT

INSERT

O comando insert no mysql é usado para inserir dados em uma tabela.

A sintaxe básica é:

INSERT INTO nome_da_tabela (campo1, campo2, …) VALUES (valor1, valor2, …);

INSERT

Você pode especificar os nomes dos campos e os valores que deseja inserir na tabela. Por exemplo:

INSERT INTO clientes (nome, idade, estado_civil) VALUES (“José”, 39, “casado”);

Esse comando insere um registro na tabela clientes com os valores informados

INSERT

Para inserir um registro na tabela produtos com os campos id, descricao, preco e categoria sem especificar os nomes das colunas, você pode usar o seguinte comando:

INSERT INTO produtos VALUES (4, “Resma Ofício c/500 folhas”, 17.50, 2);

INSERT

Um exemplo de comando insert onde vários registros são inseridos de uma vez é o seguinte:

INSERT INTO nome_da_tabela (Campo1, Campo2, Campo3) VALUES (‘valor1’,‘valor2’,‘valor3’), (‘valor4’,‘valor5’,‘valor6’), (‘valor7’,‘valor8’,‘valor9’);

Esse comando insere 3 registros na tabela nome_da_tabela com os valores especificados para cada campo. Você pode inserir mais registros separando-os por vírgulas dentro dos parênteses.

TRANSACTION

Para iniciar uma transação no MySQL, você pode usar o comando START TRANSACTION ou BEGIN. Depois de executar os comandos que fazem parte da transação (como INSERT, UPDATE ou DELETE), você pode usar o comando COMMIT para confirmar as alterações no banco de dados. Se algo der errado durante a transação e você quiser desfazer as alterações, você pode usar o comando ROLLBACK para voltar ao estado anterior do banco de dados.

As operações ACID são um conjunto de propriedades que garantem que as transações sejam executadas de forma confiável em um banco de dados. As quatro propriedades são:

- Atomicidade: Se as operações forem confirmadas, elas acontecerão todas sem exceção. Não ocorre a execução de apenas parte das operações contidas na transação.
- Consistência: As alterações no banco de dados somente acontecem caso o novo estado do sistema for válido.
- Isolamento: A transação não é visível a outros processos até que ela seja confirmada e persistida no banco.
- Durabilidade: Uma vez que uma transação é confirmada, ela permanecerá no banco de dados mesmo em caso de falha do sistema.

No MySQL, as transações aplicam as quatro propriedades ACID às operações no banco de dados. Neste caso ou a transação acontece por inteiro ou não acontece nada. Ex: transferência valores em conta – sai de uma e aparece em outra ou retorna caso ocorra um problema.

TRANSACTION

O comando TRANSACTION em MySQL permite executar várias instruções SQL como uma unidade lógica e controlar quando os dados são confirmados ou revertidos.

Para iniciar uma transação, use a instrução START TRANSACTION. Por exemplo:

```
START TRANSACTION;

INSERT INTO users (name, email) VALUES ('John Doe', 'johndoe@example.com');

UPDATE accounts SET balance = balance + 100 WHERE name = 'John Doe';
```

TRANSACTION

Para confirmar as alterações feitas na transação, use a instrução COMMIT. Por exemplo:

START TRANSACTION;

INSERT INTO users (name, email) VALUES ('Jane Doe', 'janedoe@example.com');

UPDATE accounts SET balance = balance + 200 WHERE name = 'Jane Doe';

COMMIT;

TRANSACTION

Para reverter as alterações feitas na transação, use a instrução ROLLBACK. Por exemplo:

START TRANSACTION;

UPDATE bankaccounts SET funds = funds - 100 WHERE accountno = 'ACC1';

UPDATE bankaccounts SET funds = funds + 100 WHERE accountno = 'ACC2';

ROLLBACK;

COMMIT e ROLLBACK

Os comandos COMMIT e ROLLBACK são usados para controlar as transações no MySQL. Uma transação é um conjunto de operações que devem ser executadas como uma unidade, ou seja, ou todas são feitas ou nenhuma é feita. Por exemplo, se você quer transferir dinheiro de uma conta para outra, você precisa garantir que o valor seja debitado de uma conta e creditado na outra, sem perder ou duplicar o dinheiro.

COMMIT e ROLLBACK

Você deve usar COMMIT ou ROLLBACK dependendo do resultado das suas instruções SQL na transação. Se você quiser salvar as alterações feitas na transação no banco de dados, use COMMIT. Se você quiser descartar as alterações feitas na transação e restaurar os valores originais, use ROLLBACK.

Por padrão, o MySQL inicia a sessão para cada nova conexão com o modo autocommit habilitado, o que significa que o MySQL faz um commit após cada instrução SQL se essa instrução não retornar um erro2. Se você quiser desabilitar o modo autocommit e controlar quando fazer commit ou rollback das suas transações, você pode usar a instrução SET autocommit = 0.

COMMIT e ROLLBACK

Se você não alterou nada na transação, você pode usar tanto COMMIT quanto ROLLBACK. Qualquer um dos dois irá liberar os bloqueios de leitura que você adquiriu e como você não fez nenhuma outra alteração, eles serão equivalentes. No entanto, alguns wrappers de banco de dados podem considerar um rollback em uma transação interna como um sinal de que ocorreu um erro e fazer rollback de tudo na transação externa, independentemente de a transação externa ter feito commit ou rollback. Portanto, um COMMIT é a forma segura, quando você não pode descartar que seu componente seja usado por algum módulo de software

COMMIT e ROLLBACK

Se alguma das suas instruções SQL falhar na transação, o MySQL continuará executando as próximas instruções (dentro da transação) apenas para serem revertidas posteriormente. Isso significa que você deve verificar se houve algum erro após cada instrução e definir uma variável para indicar se deve fazer commit ou rollback da transação.

Veja alguns exemplos de uso do COMMIT e ROLLBACK no MySQL:

Transferir 100 reais da conta 1 para a conta 2:

```
START TRANSACTION;
UPDATE contas SET saldo = saldo - 100 WHERE id = 1;
UPDATE contas SET saldo = saldo + 100 WHERE id = 2;
COMMIT;
```

Veja alguns exemplos de uso do COMMIT e ROLLBACK no MySQL:

Inserir um novo cliente na tabela clientes e um novo pedido na tabela pedidos. Se houver algum erro na inserção do pedido, cancelar a inserção do cliente também:

```
START TRANSACTION;
INSERT INTO clientes (nome, email) VALUES ('João', 'joao@gmail.com');
INSERT INTO pedidos (cliente_id, produto_id, quantidade) VALUES (LAST_INSERT_ID(), 10, 2);
IF @@error THEN
    ROLLBACK;
ELSE
    COMMIT;
END IF;
```

Veja alguns exemplos de uso do COMMIT e ROLLBACK no MySQL:

Deletar todos os registros da tabela produtos que tenham preço menor que 10 reais. Se não houver nenhum registro deletado, desfazer a operação:

```
START TRANSACTION;
DELETE FROM produtos WHERE preco < 10;
IF @@rowcount = 0 THEN
   ROLLBACK;
ELSE
   COMMIT;
END IF;
```

A diferença entre COMMIT e ROLLBACK em MySQL é que o comando COMMIT confirma a transação atual, tornando suas alterações permanentes. Por outro lado, o comando ROLLBACK desfaz a transação atual, cancelando suas alterações.

Por exemplo, se você executar as seguintes instruções:

START TRANSACTION;
UPDATE bankaccounts SET funds = funds - 100 WHERE accountno = 'ACC1';
UPDATE bankaccounts SET funds = funds + 100 WHERE accountno = 'ACC2';
E depois usar o comando COMMIT:

COMMIT;
As alterações nos saldos das contas serão salvas no banco de dados. Mas se você usar o comando ROLLBACK:

ROLLBACK;
As alterações nos saldos das contas serão descartadas e os valores originais serão restaurados

Comando INSERT com SELECT permite criar uma tabela nova com uma cópia dos dados existentes nas colunas que fazem parte do SELECT.

Exemplo 012.

```
CREATE TABLE so_jogadores
SELECT jog_id, jog_nome, jog_posicao FROM jogadores;

SELECT * FROM so_jogadores;
```

UPDATE

O comando update é usado para modificar registros em uma tabela do MySQL. A sintaxe básica é a seguinte:

UPDATE nome_tabela SET campo1 = “novo_valor” WHERE condição

Você pode atualizar um ou mais campos de uma vez, separando-os por vírgulas depois do SET. Você também pode usar operadores lógicos como AND e OR na condição para especificar quais registros serão atualizados.

UPDATE

UPDATE clientes SET nome = ‘Rafael’, email = ‘contato@rlsystem.com.br’ WHERE id = 1;

- Esse exemplo atualiza o nome e o email do cliente que tem o id igual a 1 na tabela clientes. O nome é alterado para ‘Rafael’ e o email para ‘contato@rlsystem.com.br’.

UPDATE produtos SET descricao = ‘Resma de ofício com 500 folhas’, preco = 18.50 WHERE id = 1 OR preco = 17.50;

- Esse exemplo atualiza a descricao e o preco dos produtos que têm o id igual a 1 ou o preco igual a 17.50 na tabela produtos. A descricao é alterada para ‘Resma de ofício com 500 folhas’ e o preco para 18.50.

UPDATE tb01_alunos SET tb01_nome_aluno = ‘Katia Fushita’, tb01_rg_aluno = 21912854 WHERE tb01_rm_aluno = 201720003;

- Esse exemplo atualiza o tb01_nome_aluno e o tb01_rg_aluno do aluno que tem o tb01_rm_aluno igual a 201720003 na tabela tb01_alunos. O tb01_nome_aluno é alterado para ‘Katia Fushita’ e o tb01_rg_aluno para 21912854.

DELETE

O comando delete é usado para remover registros em uma tabela do MySQL. A sintaxe básica é a seguinte1:

DELETE FROM nome_tabela WHERE condição

Você pode usar operadores lógicos como AND e OR na condição para especificar quais registros serão removidos.

DELETE

DELETE FROM funcionarios WHERE id = 3;

- Esse exemplo remove o registro da tabela funcionarios que tem o id igual a 3.

DELETE FROM produtos WHERE preco < 10 OR categoria = ‘Eletrônicos’;

- Esse exemplo remove os registros da tabela produtos que têm o preco menor que 10 ou a categoria igual a ‘Eletrônicos’.

DELETE FROM tb01_alunos WHERE tb01_nome_aluno LIKE ‘%a%’;

- Esse exemplo remove os registros da tabela tb01_alunos que têm o tb01_nome_aluno contendo a letra ‘a’.

TRUNCATE

O comando TRUNCATE TABLE permite remover todas as linhas de uma tabela em uma única operação, sem registrar as exclusões de linhas individuais. O TRUNCATE TABLE equivale a executar a instrução DELETE, porém sem usar a cláusula WHERE. Portanto, é usada para apagar completamente o conteúdo de uma tabela no MySQL.

TRUNCATE TABLE nome_da_tabela;

O comando TRUNCATE é usado para remover todos os dados de uma tabela sem afetar sua estrutura. No entanto, o comando TRUNCATE não pode ser usado em tabelas com relacionamentos de chave estrangeira. Se a tabela de destino possuir uma referência para uma Foreign Key (chave estrangeira), mesmo se a tabela de referência estiver vazia e a FK desabilitada, a única maneira de permitir o uso do Truncate nesse caso é cancelando as FK's que fazem referência à tabela.

Se você estiver enfrentando problemas com o comando TRUNCATE no MySQL, pode ser que sua tabela possua uma chave estrangeira que está impedindo o uso do comando. Nesse caso, você pode cancelar as FK's que fazem referência à tabela ou usar o comando DELETE.

O comando TRUNCATE TABLE é uma operação DDL e não DML como o DELETE. Isso significa que o TRUNCATE TABLE causará um COMMIT implícito no meio de um bloco de transação. Portanto, use DELETE FROM em uma tabela que você precisa esvaziar em vez de TRUNCATE TABLE.

Desfazer as alterações em uma tabela MySQL

- Se você já salvou as alterações na tabela, mas quer voltar ao estado anterior, você pode usar o comando ALTER TABLE novamente para reverter as modificações que você fez. Por exemplo, se você adicionou uma coluna e quer excluí-la, use o comando DROP COLUMN. Se você modificou o tipo de dado de uma coluna e quer restaurá-lo, use o comando MODIFY COLUMN. Se você renomeou a tabela e quer voltar ao nome original, use o comando RENAME TO.
- Se você alterou os dados da tabela usando os comandos INSERT, UPDATE ou DELETE e quer restaurá-los ao estado anterior, você pode usar o comando ROLLBACK para desfazer a última transação que afetou a tabela. Esse comando só funciona se a tabela estiver em um banco de dados que suporte transações (como InnoDB) e se a transação não tiver sido confirmada com o comando COMMIT.