SQL

Alteração de tabelas e *constraints*

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Nesta webaula você vai ver as instruções SQL para realizar alterações em tabelas do banco de dados, como acrescentar, alterar ou excluir colunas. Além de também acrescentar ou excluir restrições às colunas.

Alteração de tabelas (ALTER TABLE)

ALTER TABLE é um comando que altera a estrutura de uma tabela. Por exemplo, você pode adicionar ou excluir colunas, criar ou destruir índices, alterar o tipo de colunas existentes ou renomear colunas ou a própria tabela. Você também pode alterar características como o mecanismo de armazenamento usado para a tabela ou o comentário da tabela.

```
ALTER TABLE nome_tabela
   [especificação_alteração
      [,especificação_alteração] ...]
```

A seguir veja os principais qualificadores de especificação_alteração que podem ser utilizados.

Qualificadores do comando ALTER TABLE

```
opções_tabela
| ADD [COLUMN] (nome_coluna column_definition,...)
| ADD [COLUMN] nome_coluna column_definition
      [FIRST | AFTER nome_coluna]
| ADD {INDEX|KEY} [índice_nome]
        [índice_tipo] (índice_nome_coluna,...) [índice_opção] ...
| ALTER [COLUMN] nome_coluna {SET DEFAULT literal | DROP DEFAULT}
| ALTER INDEX índice_nome {VISIBLE | INVISIBLE}
| CHANGE [COLUMN] old_nome_coluna new_nome_coluna column_definition
        [FIRST|AFTER nome_coluna]
| [DEFAULT] CHARACTER SET [=] charset_nome [COLLATE [=] collation_nome]
| CONVERT TO CHARACTER SET charset_nome [COLLATE collation_nome]
| {DISABLE|ENABLE} KEYS
| {DISCARD|IMPORT} TABLESPACE
| DROP [COLUMN] nome_coluna
| DROP {INDEX|KEY} índice_nome
| DROP PRIMARY KEY
| MODIFY [COLUMN] nome_coluna column_definition
        [FIRST | AFTER nome_coluna]
| ORDER BY nome_coluna [, nome_coluna] ...
| RENAME COLUMN old_nome_coluna TO new_nome_coluna
| RENAME {INDEX|KEY} old_índice_nome TO new_índice_nome
| RENAME [TO|AS] new _ tbl _ nome
```

A sintaxe para muitas das alterações permitidas é semelhante às cláusulas da instrução CREATE TABLE.

DICAS

■ A palavra COLUMN é opcional e pode ser omitida, exceto RENAME COLUMN (para distinguir uma operação de renomeação de coluna da operação de renomeação de tabela RENAME).

■ Múltiplas cláusulas ADD, ALTER, DROP e CHANGE são permitidas em uma única instrução ALTER TABLE, separadas por vírgulas.

Exemplos:

Destacar várias colunas

Para descartar várias colunas em uma única instrução:

```
ALTER TABLE cliente DROP COLUMN parentesco, DROP COLUMN telefones;
```

Nesta instrução a tabela “cliente” terá as colunas “parentesco” e “telefones” excluídas.

Alterar valor do campo auto incremento

Para alterar o valor do campo auto incremento de uma tabela:

```
ALTER TABLE cliente AUTO_INCREMENT = 13;
```

Nesta instrução, na tabela “cliente”, o campo auto incremento está sendo alterado explicitamente para o valor 13. A próxima inserção de registro nesta tabela terá neste campo automaticamente preenchido para o valor 13 e, assim consecutivamente.

Adicionando e excluindo colunas (ADD e DROP)

Use o comando **ADD** para adicionar novas colunas a uma tabela e o comando **DROP** para remover colunas existentes.

`DROP nome_coluna` é uma extensão do MySQL para o SQL padrão.

Para adicionar uma coluna a uma posição específica dentro de uma linha da tabela, deve-se usar FIRST ou AFTER nome_coluna. O padrão é adicionar a coluna à última posição.

Exemplos:

Adicionar um campo - ADD

Adicionar campo “nome” com o tipo especificado na tabela pessoas:

```
ALTER TABLE pessoas ADD nome VARCHAR(50);
```

Excluir um campo - DROP

Excluir o campo “sobrenome” da mesma tabela:

```
ALTER TABLE pessoas DROP COLUMN sobrenome
```

Renomeando, redefinindo e reordenando colunas (CHANGE, MODIFY, RENAME COLUMN e ALTER)

As cláusulas CHANGE, MODIFY, RENAME COLUMN e ALTER permitem que os nomes e definições de colunas existentes sejam alterados.

Exemplos:

CHANGE

```
ALTER TABLE pessoas CHANGE antigo novo BIGINT NOT NULL;
```

Neste exemplo de alteração, uma coluna está sendo renomeada e especificada como NOT NULL de “antigo” para “novo” e alterando sua definição para usar o tipo de dados BIGINT, mantendo o atributo NOT NULL.

MODIFY

```
ALTER TABLE pessoas MODIFY novo INT NOT NULL;
```

A cláusula MODIFY é mais conveniente para alterar a definição sem alterar o nome porque requer o nome da coluna apenas uma vez.

RENAME

```
ALTER TABLE pessoas RENAME COLUMN novo TO antigo;
```

Altera o nome de uma coluna, mas não sua definição.

Chave primária

Ao adicionar os qualificadores UNIQUE INDEX ou PRIMARY KEY a uma tabela, o MySQL a armazenará antes de qualquer índice não exclusivo para permitir a detecção de chaves duplicadas o mais cedo possível. Você pode identificar numa cláusula da instrução CREATE TABLE ou ALTER TABLE um campo como PRIMARY KEY.

Usando restrições (*CONSTRAINTS*)

Veja a instrução que está especificando o campo “id” como chave primária e sem poder conter valores nulos:

```
CREATE TABLE pessoa(
    id int NOT NULL PRIMARY KEY,
    nome varchar(255) NOT NULL,
    sobrenome varchar(255),
    idade int
);
```

Supondo que deseja-se alterar a chave primária, ela conterá duas colunas e não apenas uma.

1. Primeiro deve-se retirar a chave primária declarada:

```
ALTER TABLE pessoa
   DROP PRIMARY KEY;
```

2. Após isso será necessário a nomeação de uma restrição PRIMARY KEY e para defini-la em várias colunas, usa-se a seguinte sintaxe SQL, que declara na tabela "pessoa" uma chave primária composta, com o nome de PK_pessoa com os dois campos que a compõe:

```
ALTER TABLE pessoa
  ADD CONSTRAINT PK_pessoa PRIMARY KEY (id, sobrenome);
```

Chave estrangeira

O MySQL suporta chaves estrangeiras, que permitem a referência cruzada de dados relacionados entre tabelas e restrições de chaves estrangeiras, o que ajuda a manter consistentes esses dados dispersos. A sintaxe essencial para uma definição de restrição de chave estrangeira em uma instrução CREATE TABLE ou ALTER TABLE.

```
1  ALTER TABLE tbl_nome(
2     ADD [CONSTRAINT [símbolo] FOREIGN KEY
3   [index_nome] (index_col_nome, ...)
4           REFERENCES tbl_nome (index_col_nome, ...)
5           [ON DELETE referências]
6           [ON UPDATE referências]
7
8  referências:
9          RESTRICT | CASCADE | SET NULL | NO ACTION
```

O MySQL cria implicitamente um índice de chave estrangeira que é nomeado de acordo com as seguintes regras:

DICAS

■ Se definido, o valor do símbolo CONSTRAINT é usado. Caso contrário, o valor do index_nome FOREIGN KEY é usado.

■ Se nenhum símbolo CONSTRAINT ou FOREIGN KEY index_nome estiverem definidos, o nome do índice de chave estrangeira será gerado usando o nome da coluna de chave estrangeira de referência.

Relacionamento tabela pai e tabela filha

Os relacionamentos de chave estrangeira envolvem uma tabela pai (que contém os valores de dados centrais) e uma tabela filha (com valores idênticos apontando para seu pai). A cláusula FOREIGN KEY é especificada na tabela filha.

Veja a criação de tabelas pai e filha:

```
CREATE TABLE pai(
   id INT NOT NULL,
 nome VARCHAR (50),
       PRIMARY KEY (ID)
);
CREATE TABLE filha(
   id INT NOT NULL,
 parente_id INT,
       nome VARCHAR(50)
);
```

Com estes qualificadores, as duas tabelas serão criadas e podemos notar um campo de relacionamento entre elas, demonstrado através do campo parente_id na tabela filha.

Integridade referencial

Por exemplo, vamos criar uma integridade referencial entre as tabelas utilizando este relacionamento por meio da seguinte instrução:

```
ALTER TABLE filha
  ADD CONSTRAINT FK_parente
  FOREIGN KEY (parente_id) REFERENCES pai(id);
```

Assim, o MySQL cria uma integridade referencial, uma chave estrangeira referenciando a tabela filha com a tabela pai através dos campos parente_id e id, respectivamente. Com estes comandos, embora não esteja explícito, o MySQL criou um índice para a coluna parente_id automaticamente.

Para garantir a integridade referencial, o MySQL rejeita qualquer operação INSERT ou UPDATE que tente criar um valor de chave estrangeira em uma tabela filha, se não houver um valor de chave candidato correspondente na tabela pai (CARDOSO e CARDOSO, 2013).

Quando uma operação UPDATE ou DELETE afeta um valor de chave na tabela pai que possui linhas correspondentes na tabela filha, o resultado depende da ação referencial especificada usando as subcláusulas ON UPDATE e ON DELETE da cláusula FOREIGN KEY.

O MySQL suporta opções sobre a ação a ser tomada:

CASCADE

Esse qualificador exclui ou atualiza a linha da tabela pai e exclui ou atualiza automaticamente as linhas correspondentes na tabela filha. Tanto o qualificador ON DELETE CASCADE como o ON UPDATE CASCADE são suportados. Entre duas tabelas, não devem ser definidas várias cláusulas ON UPDATE CASCADE que atuam na mesma coluna na tabela pai ou na tabela filha.

SET NULL

Este qualificador exclui ou atualiza a linha da tabela pai e define como NULL a coluna ou colunas de chave estrangeira na tabela filha. Ambas as cláusulas ON DELETE SET NULL e ON UPDATE SET NULL são suportadas. Você deve ter o cuidado de certificar-se que ao especificar uma ação SET NULL, não ter declarado as colunas na tabela filha como NOT NULL.

RESTRICT

Rejeita a operação de exclusão ou atualização da tabela pai. Especificar RESTRICT (ou NO ACTION) é o mesmo que omitir a cláusula ON DELETE ou ON UPDATE.

NO ACTION

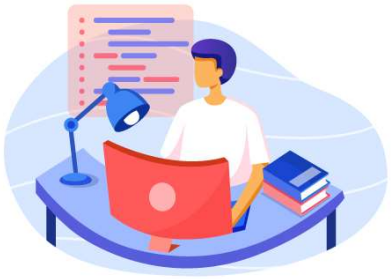
Esta é uma palavra-chave do SQL padrão. No MySQL, que é equivalente a RESTRICT. O servidor MySQL rejeita a operação de exclusão ou atualização para a tabela pai, se houver um valor de chave estrangeira relacionado na tabela referenciada. Alguns sistemas de banco de dados possuem cheques diferidos e NO ACTION é um cheque adiado. No MySQL, as restrições de chave estrangeira são verificadas imediatamente, portanto, NO ACTION é o mesmo que RESTRICT.

Excluindo restrição

Para excluir uma restrição execute a instrução a seguir, em que a chave criada é mantida após esta execução e, somente a restrição será excluída.

```
ALTER TABLE filha DROP FOREIGN KEY FK_parente;
```

Aqui, você viu como são feitas as alterações nas tabelas e também como criar restrições para as mesmas.



Fonte: Shutterstock.