
una
O MELHOR
CENTRO UNIVERSITÁRIO
PRIVADO DE BH
Fonte: MEC

una

O MELHOR
CENTRO UNIVERSITÁRIO
PRIVADO DE BH
Fonte: MEC

MySQL®

Professor Leonardo Larback

Engines

MySQL Server possui um conceito chamado de ***Storage Engine*** ou mecanismos de armazenamento, ou ainda, tipos de tabela. Através da engine selecionada, o servidor sabe como deve armazenar os dados e quais recursos estarão disponíveis para aquela tabela. Para ver quais engines estão disponíveis devemos utilizar o comando ***show engines;***

Engines

As principais storages engines são:

MyISAM – Essa é a principal storage engine do MySQL, está habilitada em qualquer configuração do MySQL e é a engine default quando criamos tabelas, ou seja, quando não especificamos uma engine, ela será uma tabela do tipo MyISAM.

Quando criamos uma tabela MyISAM ela é armazenada no disco em três arquivos. Os arquivos tem nomes que começam com o nome da tabela e tem uma extensão para indicar o tipo de arquivo. Arquivos do tipo .frm armazemam o formato da tabela, enquanto os dados ficam em um arquivo com a extensão .MYD, já os índices ficam no arquivo com extensão MYI.

Esta engine tem uma uma alta performance para a busca de dados e foi em grande parte responsável pela popularização do mysql em aplicações web.

Engines

InnoDB – É uma engine rápida e com transações seguras. O InnoDB armazena todas as suas tabelas e índices em um tablespace, o que significa que elas podem ser armazemadas em diversos arquivos, diferente das tabelas MyISAM que são armazenadas em arquivos separados. O mecanismo de armazenamento InnoDB é feito por uma empresa separada, a *InnoBase Oy*. Além de transações seguras o engine InnoDB oferece:

- **Lock de registro** – Isto significa que apenas a linha que estamos usando em uma consulta está indisponível para os outros usuários. A maioria das outras engines usa o lock de tabela, isto é, enquanto um processo estiver atualizando uma tabela, ela não estará disponível para outros processos.
- **Foreing Key** – Para manter a integridade referencial dos dados, tabelas InnoDB suportam chaves estrangeiras.
- **Oracle-style consistent nonlocking reads** – é uma idéia do Oracle de executar leituras consistentes sem locks em SELECTs.

Engines

MERGE – Uma tabela MERGE (MRG_MyISAM) é uma coleção de tabelas MyISAM idênticas que podem ser usada como uma. Só podemos fazer SELECT, DELETE E UPDATE da coleção de tabelas. Se fizermos um DROP na tabela MERGE, estaremos apagando apenas a especificação de MERGE.

As tabelas MERGE são uma maneira inteligente de solucionar os limites do sistema operacional em relação ao tamanho máximo do arquivo. Como os dados cada tabela MyISAM são armazenados em um único arquivo, as tabelas são limitadas pelo tamanho máximo do arquivo do sistema operacional.

Engines

Memory – As tabelas Memory são extremamente rápidas, sendo armazenadas inteiramente na memória. Elas usam um esquema de indexação com hash que é responsável por sua velocidade. Claramente, se algum problema acontecer (falha no mysql, queda de energia, etc) perderemos todos os dados armazenados nesse tipo de tabela. Fica a dica de sempre usar a especificação MAX_ROWS na instrução CREATE, senão passamos a ter o risco de usar toda a memória da máquina.

Chaves estrangeiras

O MySQL, a partir da versão 3.23.43b (***select version();***), passa a incorporar o recurso de criação e manutenção de tabelas do tipo InnoDB. Tabelas do tipo InnoDB suportam restrições por chave estrangeira e o uso de stored procedures. Assim, se você quer modelar/desenvolver bancos implementando o recurso de chaves estrangeiras, suas tabelas terão que ser do tipo InnoDB.

Para criar tabelas do tipo InnoDB, devemos incluir no final da tabela a seguinte sintaxe:

ENGINE = innodb;

Chaves estrangeiras

O MySQL, a partir da versão 3.23.43b (***select version();***), passa a incorporar o recurso de criação e manutenção de tabelas do tipo InnoDB. Tabelas do tipo InnoDB suportam restrições por chave estrangeira e o uso de stored procedures. Assim, se você quer modelar/desenvolver bancos implementando o recurso de chaves estrangeiras, suas tabelas terão que ser do tipo InnoDB.

Para criar tabelas do tipo InnoDB, devemos incluir no final da tabela a seguinte sintaxe:

ENGINE = innodb;

Chaves estrangeiras

Imaginemos um cenário onde desejamos cadastrar times de futebol e seus respectivos jogadores. Cada time pertence defende um estado da federação e cada estado pode possuir vários times. Primeiramente definimos a tabela de estados:

create table uf (sigla char(2) primary key, nome varchar(200)) engine=innodb;

Depois cadastramos alguns estados:

insert into uf values ('MG','Minas Gerais'),('SP','São Paulo'), ('RS','Rio Grande do Sul');

Chaves estrangeiras

Agora, vamos criar a tabela de times, sabendo que o campo estado será uma chave estrangeira, ou seja, não poderemos cadastrar um estado sem que haja uma ocorrência do mesmo na tabela uf:

create table times (codigoCBF int auto_increment primary key, nome varchar(200), dt_fundacao date, presidente varchar(200), estado char(2), constraint fk_estados foreign key(estado) references uf(sigla)) engine=innodb;

Se tentarmos cadastrar um time para um estado que não esteja cadastrado, o mysql não permitirá (mantendo a integridade referencial) e retornará uma mensagem de erro:

insert into times (nome,dt_fundacao,presidente,estado) values ('Flamengo','1901-02-16','Suzana Frey','RJ');

ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails (`una`.`times`, CONSTRAINT `fk_estados` FOREIGN KEY (`estado`) REFERENCES `uf` (`sigla`))

Chaves estrangeiras

Agora, *criemos a tabela* de jogadores sabendo que os jogadores devem estar associados a um time:

create table jogadores (codigo int auto_increment primary key, nome varchar(200), posicao varchar(200), data_nascimento date, time int, constraint fk_times foreign key (time) references times(codigoCBF))engine = innodb;

Se tentarmos cadastrar um jogador para um time que não exista, receberemos a mesma mensagem de violação de integridade referencial.

Chaves estrangeiras

Um outro aspecto importante da integridade referencial é que o mysql não permitirá que um registro pai seja excluído até que todos os seus filhos tenham sido eliminados. Por exemplo, se tentarmos excluir um time que já possui jogadores:

ERROR 1451 (23000): Cannot delete or update a parent row: a foreign key constraint fails (`una`.`jogadores`, CONSTRAINT `fk_times` FOREIGN KEY (`time`) REFERENCES `times` (`codigoCBF`))

Chaves estrangeiras

O InnoDB implementa as restrições de integridade CASCADE, RESTRICT, SET NULL e SET DEFAULT. No primeiro caso, ao se remover um registro da tabela referenciada pela chave estrangeira os registros relacionados àquele removido serão eliminados em todas as tabelas relacionadas. O RESTRICT não permite a remoção de registros que possuam relacionamentos em outras tabelas. Os dois últimos atribuem os valores DEFAULT ou NULL para as chaves estrangeiras cujos registros relacionados foram excluídos. O exemplo abaixo ilustra algumas tabelas que utilizam regras de integridade:

Chaves estrangeiras

```
CREATE TABLE aluno (
        id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  nome CHAR(30) NOT NULL
) engine=InnoDB;
CREATE TABLE cursos (
        id INT NOT NULL AUTO_INCREMENT PRIMARY KEY, nome CHAR(30) NOT NULL
) engine=InnoDB;
CREATE TABLE notas (
        aluno_id INT NOT NULL,
        cursos_id INT NOT NULL,  date DATE NOT NULL,  nota DOUBLE NOT NULL,
        PRIMARY KEY(aluno_id, cursos_id, date),
        INDEX i2 (cursos_id),
        FOREIGN KEY (aluno_id) REFERENCES aluno(id) ON DELETE CASCADE,
        FOREIGN KEY (cursos_id) REFERENCES cursos(id) ON DELETE RESTRICT
) engine=InnoDB;
```

Chaves estrangeiras

Para testar, cadastre alguns alunos, alguns cursos e algumas notas. Depois, tente excluir cursos que possuem notas lançadas. Você verá que não é possível (repare na restrição imposta no relacionamento ***ON DELETE RESTRICT***). Depois tente excluir alunos que possuem notas lançadas, você verá que as notas também serão excluídas (repare na restrição imposta ao relacionamento ***ON DELETE CASCADE***).