Fundamentos de Banco de Dados
Aula 4

Prof. Me. Marco Aurelio M. Antunes

Chave Primária

A chave primária, ou **primary key**, é o conceito mais básico relacionado à organização em um banco de dados. Cada tabela pode usar somente uma chave primária. Essa chave é utilizada como identificador único da tabela, sendo representada por aquele campo (ou campos) que não receberá valores repetidos.

Existe uma lista de características que deve ser levada em consideração ao definir uma chave primária:

- Chaves primárias não podem ser nulas;
- Normalmente, chaves primárias podem ser incrementadas automaticamente pelo banco de dados, ou seja, não há necessidade de passarmos esse valor em um INSERT.
- São as chaves para o relacionamento entre entidades ou tabelas da base de dados. Assim haverá na tabela relacionada uma referência a essa chave primária (que será, na tabela relacionada, a chave estrangeira).

```
create table pessoa(
id_pessoa       int             identity,
nome            varchar(40)     not null,
fone            varchar(20)     null,
email           varchar(40)     null
primary key(id_pessoa)
);
```

Chave Estrangeira

A chave estrangeira, ou **foreign key**, é um conceito que diz respeito, especificamente, a um relacionamento entre tabelas. De forma sucinta, a chave estrangeira é uma referência em uma tabela a uma chave primária de outra tabela.

Diferentemente da chave primária, a chave estrangeira:

- Pode ser nula (NOT NULL);
- É um campo em uma tabela que faz referência a um campo que é chave primária em outra tabela;
- É possível ter mais de uma (ou nenhuma) em uma tabela.

Um alerta: embora não haja, efetivamente, nenhum problema das chaves estrangeiras aceitarem o valor null, tal característica pode gerar o que é chamado de registro órfão, isto é, um registro sem dados para um determinado relacionamento.


```
create table carro(
id_carro        int             identity,
Marca           varchar(40)     not null,
Modelo          varchar(40)     not null,
id_pessoa       int             not null
primary key(id_carro)
foreign key(id_pessoa) references pessoa(id_pessoa)
);
```

Integridade de Dados

A integridade refere-se à confiabilidade dos dados em todo o seu ciclo de vida. Verificação de erros e validação, por exemplo, são métodos comuns para garantir a integridade dos dados como parte de um processo, e não deve ser confundida com segurança de dados.

Integridade de dados se refere à confiabilidade dos dados, enquanto segurança de dados refere-se à proteção de dados – a qual se concentra em como minimizar o risco de vazamento de propriedade intelectual, documentos comerciais, dados de assistência médica, e-mails, informações comerciais sigilosas e muito mais.

Para garantir a integridade dos dados é necessário a criação/utilização de **constraints** que são restrições que o sistema gerenciador de banco de dados (SGBD) disponibiliza como um meio de manter a integridade dos dados dentro do BD. Podem ser criadas no momento de criação das tabelas ou após a criação das mesmas, como por exemplo:

- NOT NULL / NULL
- PRIMARY KEY
- FOREIGN KEY
- DEFAULT
- UNIQUE
- CHECK

Chave Estrangeira

Exemplo 2

```
-- exemplo de chave estrangeira
create table gravadora
(
cod_gravadora      int              not null,
gravadora          varchar(45)      not null,
telefone           varchar(20)      null
primary key(cod_gravadora)
);

create table cd
(
cod_cd                    int              identity,
cod_gravadora             int              not null,
banda                     varchar(40)      not null,
nome_cd                   varchar(40)      not null,
data_lancamento           datetime         not null
primary key (cod_cd)
foreign key(cod_gravadora) references gravadora (cod_gravadora)
);
```

Chave Estrangeira

```
-- inserindo dados tabela gravadora
-- tabela PAI
insert into gravadora values (1,'som livre','9999-9999');
insert into gravadora values (2,'globo','8888-8888');
insert into gravadora values (3,'warner music','9999-9999');
select * from gravadora;
-- forçando o erro - chave primária
insert into gravadora values (1,'Emi','1111-2222');
-- forçando erro - chave estrangeira
delete from gravadora;
-- forçando erro - chave estrangeira 2
drop table gravadora;


-- inserindo dados tabela cd
-- tabela FILHO
insert into cd values (1,'Legiao urbana','dois','20/12/2019');
insert into cd values (1,'Legiao urbana','quatro estações','10/11/2019');
insert into cd values (2,'Titãs','cabeça dinossauro','02/09/2019');
insert into cd values (2,'Engenheiros do Hawaii','alivio imediato','20/12/2019');
select * from cd;
-- forçando o erro - chave primária
-- NÃO TEM COMO TER ERRO
-- forçando erro chave estrangeira
insert into cd values (4,'Engenheiros do Hawaii','a revolta dos dandis','20/12/2019');
-- arrumando erro
insert into cd values (2,'Engenheiros do Hawaii','a revolta dos dandis','20/12/2019');
select * from cd;
```

Regras – Constraints

DEFAULT

Serve para atribuir um conteúdo-padrão a uma coluna da tabela. Sempre que for incluída uma nova linha na tabela, deve-se especificar a palavra-chave default, seguida do conteúdo-padrão.

Exemplo:

```
create table CD
(
codigo_cd          int           identity,
codigo_gravadora   int           null,
nome_cd            varchar(60)   not null,
data_lancamento    datetime      not null,
quantidade         int           default 1
primary key (codigo_cd)
);
```

```
insert into CD values (1,'Mais do Mesmo','10/30/2012',default);
```

Regras – Constraints

UNIQUE

Indica que não poderá haver repetição no conteúdo da coluna. Isso é diferente do conceito de chave primária. A chave primária, além de não permitir repetição, não pode conter valor nulo. Ao especificarmos que uma coluna deve conter valores únicos, indicamos que todos os valores não nulos devem ser exclusivos.

Exemplo:

```
create table CLIENTES
(
CODIGO_CLI      int             not null,
CPF             varchar(15)     UNIQUE,
NOME_CLI        varchar(40)     not null,
FONE_CLI        varchar(18)     null,
primary key (CODIGO_CLI)
);

insert into CLIENTES values (1,‘111111111-22',‘José',‘98888-8888');
insert into CLIENTES values (2,‘111111111-22',‘Maria',‘97777-7777');
```

CHECK

Um domínio é uma expressão de valores possíveis para o conteúdo de uma coluna. Podemos, ao criarmos a coluna, especificarmos quais os valores que poderão ser utilizados para preencher a coluna, para isso utilizamos a palavra-chave check seguida da condição que validará o conteúdo.

Exemplo:

```
create table CLIENTES
(
codigo_cli     int              not null,
cpf            varchar(15)      UNIQUE,
rg             varchar(15)      null,
nome_cli       varchar(40)      not null,
fone_cli       varchar(18)      null,
sexo           varchar(1) check(upper(sexo)='M' or upper(sexo)='F'),
primary key (codigo_cli)
);
```

```
insert into CLIENTES values (1,'111', '222','José','8888-8888','M');
insert into CLIENTES values (2,'222', '222','Maria','8888-8888','F');
insert into CLIENTES values (3,'333', '333','José','8888-8888','J');
```

Alteração de Estrutura de Tabela

Para alterar a estrutura de uma tabela, utilizaremos o comando ALTER TABLE.

Acrescentar novas colunas

Sintaxe:

ALTER TABLE nome_da_tabela

ADD nome_coluna tipo_dado_colunaN colunaN_constraints;

Exemplo1: Acrescentando campo

```
alter table CLIENTES
add cidade varchar(30) not null;
```

Exemplo2: Acrescentando campo com constraints

```
alter table CLIENTES
add email varchar(30)  unique;
```

Excluir Elementos

Sintaxe: Excluir Banco de Dados

DROP DATABASE nome_do_banco_de_dados

Exemplo:

```
drop database FIB;
```

Sintaxe: Excluir Tabela

drop table nome_da_tabela;

Exemplo:

```
drop table CLIENTES;
```

Sintaxe: Excluir campo

ALTER TABLE nome_da_tabela DROP COLUMN nome_coluna;

Exemplo:

```
alter table CLIENTES drop column cidade;
```

Acrescentar novas constraints

Sintaxe:
ALTER TABLE nome_da_tabela
ADD constraints;

```
create table CLIENTES
(
codigo_cli       int              not null,
nome_cli         varchar(40)      not null,
cod_dependente   int              not null,
);

create table DEPENDENTE
(
cod_dependente   int              not null,
nome             varchar(50)      null
);

alter table CLIENTES
add primary key (codigo_cli);

alter table DEPENDENTE
add primary key (cod_dependente);

alter table DEPENDENTES
add foreign key (cod_dependente)
references CLIENTES(cod_dependente);
```

Acrescentar novas constraints

Sintaxe:

ALTER TABLE nome_da_tabela
ADD constraint NOME_DA_RESTRIÇÃO
Restrição

Exemplo:

```
alter table CLIENTES
add constraint VALIDA_SEXO
check (SEXO='M' or SEXO='F‘);
```

Excluindo constraints

O comando ALTER TABLE também é utilizado para excluir uma restrição existente em uma tabela.

Sintaxe:

ALTER TABLE nome_da_tabela
DROP CONSTRAINT nome_da_restrição

Exemplo:

```
Alter table CLIENTES
drop constraint VALIDA_SEXO;
```

Modificando colunas

Podemos modificar qualquer característica de uma coluna, seja o tipo de dado (alguns banco de dados requerem ausência de conteúdo na coluna para fazer essa alteração), o tamanho(alguns banco de dados só aceitam alterações para valores maiores que o definido) e as constraints.

Sintaxe:

```
ALTER TABLE nome_da_tabela
ALTER COLUMN nome_da_coluna  tipo_dado  constraints;
```

Exemplo1 : -- Modificando tipo de dado e tamanho de campo

```
alter table CLIENTES
alter column RG varchar(20);
```

Exemplo2: -- Modificando nome de campo

```
alter table CLIENTES drop column FONE_CLI;
alter table CLIENTES add TELEFONE_CLI varchar(18) null;
```

- SP_RENAME - Altera o nome de um objeto criado pelo usuário no banco de dados atual. Esse objeto pode ser uma tabela, índice, coluna, tipo de dados de alias ou tipo de dados CLR definido pelo usuário do Microsoft.NET Framework Common Language Runtime.
-
- SINTAXE:
- SP_RENAME `'NOME_DA_TABELA.NOME_DO_CAMPO','NOVO_CAMPO';`
-
- Exemplo1 : `-- Modificando nome de um campo`
-
- `SP_RENAME 'CLIENTES.RG','RG_CLI';`

Cuidado

A alteração de qualquer parte de um nome de objeto pode quebrar scripts e procedimentos armazenados. É recomendável não usar essa instrução para renomear procedimentos armazenados, gatilhos, funções definidas pelo usuário ou exibições; em vez disso, descarte o objeto e recrie-o com o novo nome.

Exercícios - Lista 4

1) Criar e testar as tabelas da aula 4
2) Inserir registros nas tabelas e quando possível forçar erro
3) Testar chave primária forçando o erro
4) Testar chave primária composta forçando o erro
5) Testar chave estrangeira forçando o erro
6) Criar uma tabela de forcenedores (codfor, nome e fone) com a chave primária codfor - PAI
7) Criar uma tabela de produtos (codpro, codfor, produto e preço) com a chave primária codpro e a chave estrangeira codfor - FILHO
8) Inserir 2 registros na tabela fornecedor - PAI
9) Inserir 3 registros na tabela produtos - FILHO
10) Tentar inserir produto sem PAI
11) Incluir fornecedor sem filho
12) Tentar excluir tabela PAI
13) Criar uma tabela que utilize chave primária é unique
14) Criar um relacionamento entre tabela PAI e FILHO utilizando chave primária e unique nas 2 tabelas e chave estrangeira
15) Criar uma tabela que utilize default e check
16) Criar uma tabela que utilize TODAS as constraints
17) Alterar a estrutura de qualquer uma das tabelas acima incluindo 2 campos
18) Alterar a estrutura de qualquer uma das tabelas acima excluindo 2 campos
19) Alterar a estrutura de qualquer uma das tabelas acima aumentando o tamanho de um campo varchar e diminuindo o tamanho de outro
20) Deletar todos os registros de uma tabela
21) Deletar uma tabela de um banco de dados
22) Deletar o banco de dados