Linguagem SQL

Aula 04 – CREATE TABLE

Sand Onofre

Sand.Onofre@FaculdadeImpacta.com.br

Sumário

- Data Definition Language – DDL
  - Create Table:
    - Colunas e Tipos de Dados
    - Null e Not Null
    - Identity
    - Primary Key, Unique Key, Foreign Key
  - Alter Table
    - Add
    - Alter Column
    - Drop Column
  - Drop Table
- Exercícios

Data Definition Table

Para persistir os dados em um banco de dados, precisamos armazená-los em objetos do tipo tabela. Comandos de criação, alteração ou eliminação de objetos, fazem parte da categoria de comandos DDL (Data Definition Language).

Nesta aula aprenderemos a sintaxe básica para criação, alteração e eliminação de tabelas e nas próximas aulas iremos aprofundar o assunto, atrelando propriedades nas tabelas, de forma que desempenhem determinadas funcionalidades como verificação de valores, valores padrão e relacionamento entre outras tabelas.

Para iniciar a criação de tabelas, precisaremos definir adequadamente suas colunas de forma que permitam a movimentação de informações segundo seus tipos de dados e obrigatoriedade ou não de informação.

Também veremos como programar para que determinadas colunas sejam preenchidas automaticamente com valores que determinaremos.

Data Definition Table

Como o tipo de dados é fundamental na definição das colunas de uma tabela, faremos uma breve revisão sobre os principais tipos de dados utilizados, lembrando que o conteúdo mais completo foi passado na aula 4, Tipos de Dados.

Tipos de dados determinam quais os tipos de valores serão permitidos no armazenamento e os principais tipos são agrupados em categorias conforme mostrado abaixo:

Categorias dos Tipos de Dados	
Numéricos Exatos	Data e Hora
Strings de Caractere	

Data Definition Table

- Numéricos Exatos

Data type	Range	Storage (bytes)
tinyint	0 to 255	1
smallint	-32,768 to 32,767	2
int	2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)	4
bit	1, 0 or NULL	1
decimal/numeric	- 10^38 +1 through 10^38 – 1 when maximum precision is used	5-17

Data Definition Table

- Dados Caractere Non-Unicode

Data Type	Range	Storage
CHAR(n)	1-8000 characters	n bytes, padded
VARCHAR(n)	1-8000 characters	n+2 bytes
VARCHAR(MAX)	1-2^31-1 characters	Actual length + 2

Data Definition Table

- Data e Hora

Data Type	Storage (bytes)	Date Range	Accuracy	Recommended Entry Format
DATETIME	8	January 1, 1753 to December 31, 9999	3-1/3 milliseconds	'YYMMDD hh:mm:ss:nnn'
SMALLDATETIME	4	January 1, 1900 to June 6, 2079	1 minute	'YYMMDD hh:mm:ss:nnn'
DATE	3	January 1, 0001 to December 31, 9999	1 day	'YYYY-MM-DD'

Data Definition Table

Revisado os principais tipos de dados, veremos agora a sintaxe básica para o comando de criação de tabelas.

Uma atenção especial deve ser dada em que banco de dados estamos criando a estrutura de dados, portanto é importante verificar EM QUAL DATABASE estamos conectados antes de executar o comando requerido:

```
CREATE TABLE <nome da tabela>
    (
    <nome coluna 1> <tipo da coluna> (<tamanho da coluna>) [NOT NULL]
    , <nome coluna 2> <tipo da coluna> (<tamanho da coluna>) [NOT NULL]
    , ...
    );
```

Data Definition Table

O comando anterior é uma simplificação da cláusula CREATE TABLE, sendo que a mesma cláusula aceita vários parâmetros e objetos associados. Como exemplo do comando mostrado, a cláusula abaixo cria a tabela Aluno com 4 atributos:

```
CREATE TABLE Aluno
    (
    Matricula int
    , Nome varchar(20)
    , MeioNome varchar(20)
    , SobreNome varchar(20)
    );
```

Data Definition Table

A cláusula NULL e NOT NULL define se o preenchimento de determinada coluna é ou não obrigatório. A cláusula NULL é a padrão, ou seja, quando não mencionamos nada como na criação da tabela do slide anterior, os campos PERMITIRÃO o NULL (ausência de valor).

Se quisermos OBRIGATORIAMENTE que um campo seja preenchido, utilizamos o NOT NULL imediatamente após a definição do tipo de dados da coluna e esta passa a requerer algum valor para a entrada de registros na tabela. Caso não seja fornecido algum valor para a coluna, será gerado ERRO e não aceitará a entrada de dados:

```
CREATE TABLE Veiculo
    (
    Placa char(8) NOT NULL
    , Marca varchar(20)  NOT NULL
    , NomeProprietario varchar(60)
    );
```

Data Definition Table

Podemos precisar que uma determinada coluna GERE números automaticamente, como por exemplo, número de matrícula.

Colunas de tabelas possuem propriedades específicas para geração de números e podem ser associadas a uma coluna. O database permite que qualquer tabela tenha UMA e SOMENTE UMA das colunas com autonumeração, lembrando que é opcional a escolha ou não desta autonumeração, assim como será feita essa série de numeração:

IDENTITY (Seed, Increment)

O parâmetro SEED representará o primeiro número a ser gerado pela série e o INCREMENT significa de quanto em quanto os próximos números serão incrementados.

Data Definition Table

Perceba que dependendo do tipo de dados, o IDENTITY não se encaixará. Veja alguns exemplos da função:

IDENTITY (Seed, Increment)

IDENTITY ou IDENTITY (1, 1) → É o padrão da função (qualquer tipo numérico)

IDENTITY (0, 1) → Qualquer tipo numérico. Inicia no 0 e incrementa de 1 em 1

IDENTITY (-32768, 1) → Smallint ou maior. Inicia no -32768, -32767, ..., 0, 1, 2, ...

IDENTITY (255, -1) → Tinyint ou maior. Inicia no 255, 254, 253, ...

IDENTITY (0, 10) → Tinyint ou maior. Inicia no 0, 10, 20, ...

Data Definition Table

Note que por definição, uma coluna autonumerada será automaticamente colocada como NOT NULL, mesmo não escrevendo explicitamente esta cláusula. O exemplo abaixo mostra a aplicação do IDENTITY:

```
CREATE TABLE Veiculo
    (
    idVeiculo INT NOT NULL IDENTITY
    , Placa char(8) NOT NULL
    , Marca varchar(20)  NOT NULL
    );
CREATE TABLE Aluno
    (
    Matricula int IDENTITY (500, 1)
    , Nome varchar(20)
    , MeioNome varchar(20)
    , SobreNome varchar(20)
    );
```

Data Definition Table

PRIMARY KEY - Podemos ter uma e apenas uma chave primária em cada tabela. Por definição chave primária possui um número único para todos os registros ou seja, dado um valor correspondente a chave primária de uma tabela, o retorno máximo de registros é uma linha.

A chave primária pode ser simples (apenas uma coluna) ou composta (mais de uma coluna).

A definição da chave primária (primary key) segue o seguinte modelo:

CONSTRAINT <nome da primary key> PRIMARY KEY (coluna1, coluna2, ...)

Data Definition Table

Exemplo:

```
CREATE TABLE Aluno
    (
    Matricula int IDENTITY (500, 1)
    , Nome varchar(20)
    , MeioNome varchar(20)
    , SobreNome varchar(20)
    , CONSTRAINT pkAluno PRIMARY KEY (Matricula)
    );
```

Data Definition Table

FOREIGN KEY – Chave Estrangeira faz o relacionamento entre uma ou mais colunas de uma tabela com a chave primária ou única de outra tabela. O formato do comando deve REFERIR ÀS COLUNAS DE AMBAS AS TABELAS (a tabela dom a FOREIGN KEY e a tabela com a PRIMARY KEY).

Uma tabela pode ter várias chaves estrangeiras para outras tabelas, representando o relacionamento que possui com cada uma das outras ENTIDADES.

Para uma tabela poder receber dados na coluna com a FOREIGN KEY é necessário que esse dado já exista na tabela de referência (PRIMARY KEY), caso contrário, o banco de dados irá gerar ERRO não permitindo a inserção do registro.

CONSTRAINT <nome da foreign key> FOREIGN KEY (coluna1, coluna2, ...)

REFERENCES <tabela da primary key> (coluna1, coluna2, ...)

Data Definition Table

Exemplo:

```
CREATE TABLE Aluno
      (
      Matricula int IDENTITY (500, 1)
      , Nome varchar(20)
      , CONSTRAINT pkAluno PRIMARY KEY (Matricula)
      );

CREATE TABLE Prova
      (
      idProva int IDENTITY (1, 1)
      , Matricula int NOT NULL
      , Nota decimal(4,2) NOT NULL
      , CONSTRAINT pkProva PRIMARY KEY (idProva)
      , CONSTRAINT fkProva_Matricula FOREIGN KEY (Matricula)
        REFERENCES Aluno(Matricula)
      );
```

Data Definition Table

UNIQUE KEY – Chave Única é semelhante a chave primária, fazendo com que o campo envolvido seja único na tabela. Podemos ter várias chaves únicas em uma tabela, diferentemente da chave primária, onde só podemos ter uma.

Por exemplo, numa tabela de Cliente, podemos ter um colunas com o número do cliente, CPF e RG. Todos os três campos não permitem repetição na tabela, ou seja, esses atributos representam CHAVES ALTERNATIVAS. Podemos eleger qualquer um desses campos como chave primária, por exemplo: número do cliente. Neste caso o CPF e o RG poderiam ter chaves únicas, já que a tabela só permite uma chave primária.

CONSTRAINT <nome da unique key> UNIQUE (coluna1, coluna2, ...)

Data Definition Table

Exemplo:

```
CREATE TABLE Cliente
    (
    NumCliente int IDENTITY (1, 1)
    , CPF int NOT NULL
    , RG int NOT NULL
    , CONSTRAINT pkCliente PRIMARY KEY (NumCliente)
    , CONSTRAINT uqCliente_CPF UNIQUE (CPF)
    , CONSTRAINT uqCliente_RG UNIQUE (RG)
    );
```

Data Definition Table

Posteriormente podemos alterar tabelas depois de criadas, adicionando novos campos, removendo campos existentes ou alterando tipos de dados e configurações de colunas existentes.

O comando básico para isso é o ALTER TABLE.

Para adicionar uma coluna, utilizamos a claúsula ADD. Para eliminar, DROP COLUMN e para alterar, ALTER COLUMN.

ALTER TABLE <nome da tabela> ADD <nome da coluna> <tipo de dados>

ALTER TABLE <nome da tabela> ALTER COLUMN <nome da coluna> <tipo de dados>

ALTER TABLE <nome da tabela> DROP COLUMN <nome da coluna>

Data Definition Table

Exemplos:

```
ALTER TABLE Cliente ADD Nome varchar(30) NOT NULL
ALTER TABLE Cliente ADD Sobrenome varchar(30) NOT NULL
```

```
ALTER TABLE Cliente ALTER COLUMN Nome varchar(50) NULL
```

```
ALTER TABLE Cliente DROP COLUMN Sobrenome
```

Data Definition Table

Para eliminar uma tabela, usamos o comando DROP TABLE. Note que após esse comando, a tabela e seus dados serão DESCARTADOS e a princípio, não terão recuperação.

DROP TABLE <nome da tabela1>, <nome da tabela2>, ..., <nome da tabela n>

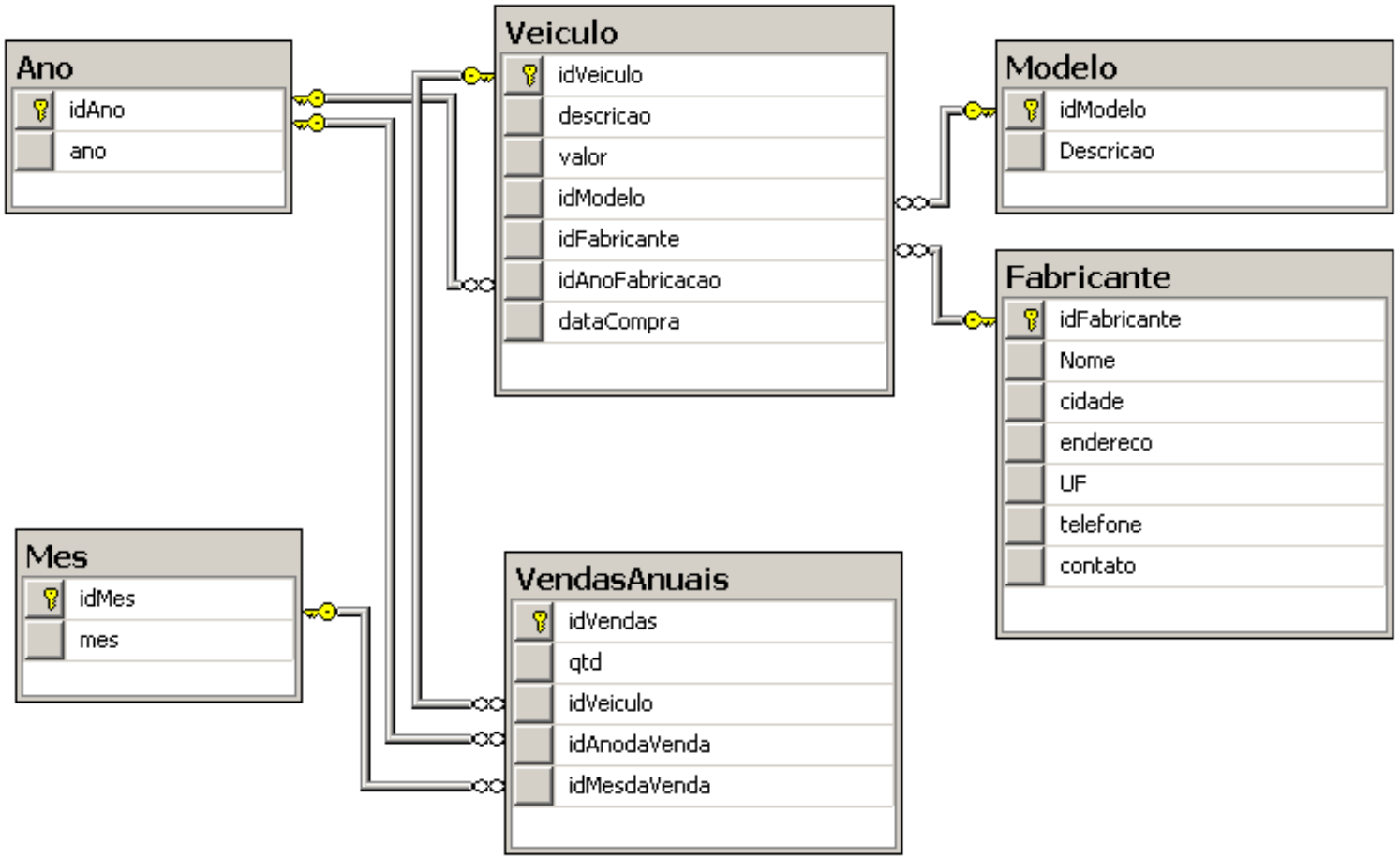
Note que se uma tabela possui uma chave primária e esta chave participa de relacionamentos com outras tabelas como chave estrangeira, não será permitida a eliminação da tabela que contém a chave primária, ou seja, neste caso, precisamos eliminar as tabelas que mencionam essa chave primária, para posteriormente conseguirmos eliminar a tabela com chave primária.

Como exemplo, onde temos a tabela Aluno (chave primária), relacionada com a tabela Prova (chave estrangeira), se quisermos eliminar a tabela Aluno, precisamos primeiro, eliminar a tabela Prova, ou retirar as contraints destes objetos, liberando a “amarração” entre os objetos.

DROP TABLE Prova, Aluno

Exercícios

O modelo de dados abaixo se refere a uma Concessionária. Crie as tabelas, colunas e chaves de acordo com o modelo proposto:



Exercícios

Posicionados no banco de dados ***TEMP***, executem os comandos DDL que montem o modelo proposto:

1. As colunas IDs são todas auto-numeradas, iniciando com 1 e de incremento 1, com exceção da tabela mês que iniciará no número 15 com incremento 3. Todas as colunas deverão possuir os menores tipos de dados possíveis para:

Tabelas AnoFabricacao, modelo, mês e fabricante não passarão de 200 registros

Tabelas Veículo e Vendas anuais não ultrapassarão 10.000 registros

2. Colunas de texto Nome, Descrição, Contato e Cidade serão obrigatórias, variáveis armazenando até 50 bytes, endereço variável até 100 bytes, telefone variável de 20 bytes e UF fixa com 2 bytes.

3. Os atributos Ano e Mes serão numéricos, obrigatórios, com tipos apropriados para seus armazenamentos convencionais.

4. Campo Valor, obrigatório, terá duas casas decimais chegando a casa dos R$ 200.000,00 e qtd, obrigatório, inteiro de 1 e 5.000.

5. O atributo DataCompra, tem precisão de dia e é obrigatório.

6. Construa todas as constraints Primary Key e Foreign Key conforme indicação no modelo.

Obrigado !

Sand Onofre
Sand.Onofre@FaculdadeImpacta.com.br