ASEBD / CEBDSQL / PSQL

GRSI / TPSI

atec

Data Definition Language – Criar Tabelas

Na sua forma mais básica é preciso apenas indicar o nome da tabela, os nomes das várias colunas e o tipo de cada uma delas.

create table <nometabela> (

<nomecoluna> <tipocoluna >,

<nomecoluna> <tipocoluna>);

Data Definition Language – Tipos de Dados

Os tipos de dados mais importantes são:

VARCHAR(N) - Cadeia de carateres com tamanho máximo n; e exige que um tamanho seja definido no momento de sua utilização. Seu tamanho máximo é de 8000 caracteres e por possuir comprimento variável todo espaço não utilizado não ocupa espaço na base de dados.

TEXT – Cadeia de carateres com tamanho máximo de 2GB.

BIT- Um bit: 0, 1 ou NULL

INTEGER ou **INT** - Números inteiros (4 bytes); Existe também as variantes tinyint, smallint, bigint)

Data Definition Language – Exemplo

```
create table Aluno (
    numero integer ,
    nome varchar (100),
    bi integer,
    nacionalidade varchar(30),
    mediaCurso decimal (4 ,2),
    media_notasExame decimal(4 ,2),
    datanascimento date,
    cursoid integer);
```

Data Definition Language – Valor por omissão

```
create table Aluno (
    numero integer ,
    nome varchar (100),
    bi integer,
    nacionalidade varchar(30) default 'Portuguesa'
    mediaCurso decimal (4 ,2) default 0,
    media_notasExame decimal (4 ,2),
    datanascimento date default GETDATE(),
    cursoid integer );
```

Data Definition Language – Restrições

Em SQL podem ser definidas restrições de vários tipos.

- **Restrições de integridade**: definem a **chave primária** e a **chave estrangeira** com a tabela e a chave primária referenciadas.
- **Restrições de valor:** definem se os valores nulos não são permitidos, se são necessários valores únicos, e se apenas determinado conjunto de valores são permitidos numa coluna

Uma restrição pode ser criada ao mesmo tempo em que a tabela é criada ou pode ser adicionada à tabela posteriormente.

Existem dois níveis em que uma restrição é definida:

- **Nível da coluna** - refere-se apenas a uma coluna e é descrita em frente à coluna em causa);
- **Nível da tabela-** refere-se a mais do que a uma coluna e fica separada da definição das colunas).

Restrições – Chave Primária

```
create table Aluno (
    numero integer primary key, --restrição a nível de coluna
    nome varchar (100),
    bi integer,
    nacionalidade varchar(30) default 'Portuguesa'
    mediaCurso decimal (4 ,2) default 0,
    media_notasExame decimal (4 ,2),
    datanascimento date default GETDATE(),
    cursoid integer );
```

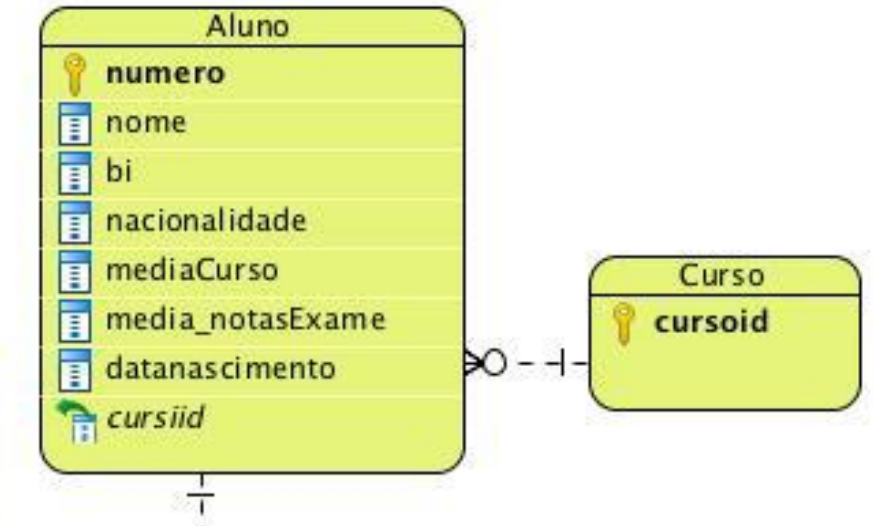
Restrições – Chave Primária composta

```
create table Inscricao(
    ano_letivo decimal(4,0),
    numero integer,
    cod_disc decimal(4,0),
    datainscricao date,
    constraint pk_Inscricao_anoletivo_coddisc primary key
    (ano_letivo,numero,cod_disc));
```

Podemos e devemos sempre dar nomes às restrições para que seja mais fácil identificar a razão pela qual a inserção de dados falha.

Restrições – Chave Estrangeira

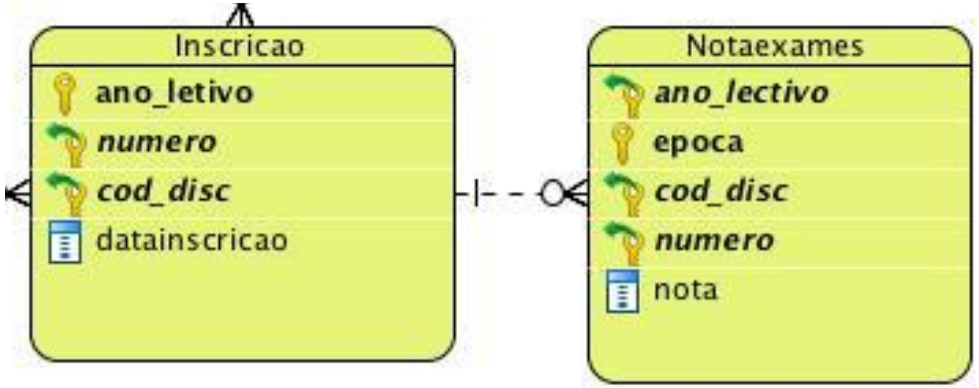
```
create table Aluno (
    numero integer primary key,
    nome varchar (100),
    bi integer,
    nacionalidade varchar (30) ) default 'Portuguesa',
    mediaCurso decimal (4 ,2) default 0,
    media_notasExame decimal (4 ,2),
    datanascimento date default GETDATE(),
    cursoid integer references curso(id) )
```



atributo que é chave primária na tabela curso

Restrições – Chave Estrangeira composta

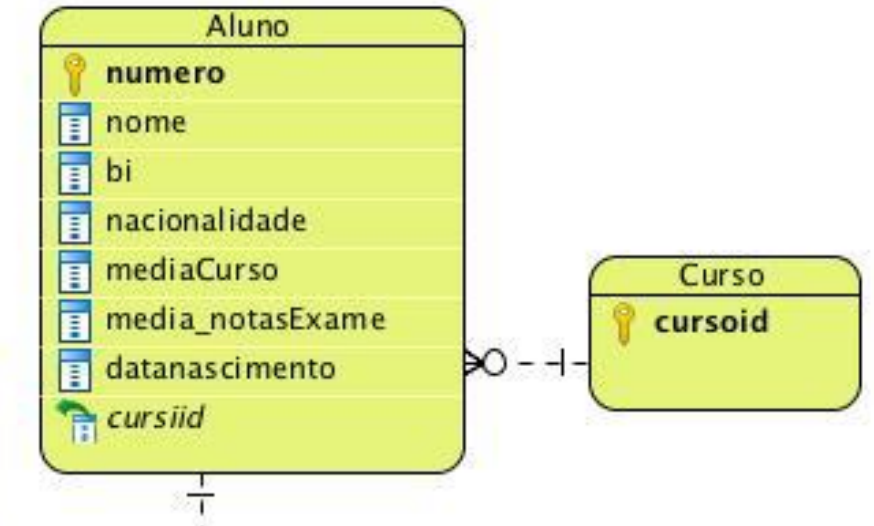
```
create table NotaExame (
    ano_letivo decimal(4,0),
    epoca varchar(20),
    cod_disc decimal(4,0),
    numero integer,
    nota decimal(4,2),
    ..........
    constraint fk_NotaExame_anoletivo_numero_coddisc Foreign Key (ano_letivo,
    numero, cod_disc) references Inscricao (ano_letivo, numero, cod_disc));
```



Restrições – Chave Estrangeira

create table Aluno (

numero integer primary key,
nome varchar (100),
bi integer,
nacionalidade varchar (30)) default 'Portuguesa',
mediaCurso decimal (4 ,2) default 0,
media_notasExame decimal (4 ,2),
datanascimento date default GETDATE(),



- cursoid integer **references curso(id)** **on delete** set null **on update** cascade, ...)

Neste exemplo estamos a definir que no caso de um curso ser apagado os alunos que pertençam a esse curso devem ficar com o curso a null. No caso da chave primária do curso ser modificada, essa modificação deve propagar-se e modificar também o curso da tabela aluno.

Restrições – Check

```
create table Aluno (
    numero integer primary key,
    nome varchar (100),
    bi integer,
    nacionalidade varchar(30) default 'Portuguesa'
    mediaCurso decimal (4 ,2) default 0
        constraint ck_Aluno_mediacurso check (mediaCurso >= 0),
    media_notasExame decimal (4 ,2),
    datanascimento date default GETDATE(),
    cursoid integer
    …. );
```

Restrições – Not Null

```
create table Aluno (
    numero integer primary key,
    nome varchar (100) not null,
    bi integer,
    nacionalidade varchar(30) default 'Portuguesa'
    mediaCurso decimal (4 ,2) default 0
        constraint ck_Aluno_mediacurso check (mediaCurso >= 0),
    media_notasExame decimal (4 ,2),
    datanascimento date default GETDATE(),
    cursoid integer
    …. );
```

Restrições – Valores únicos

```
create table Aluno (
    numero integer primary key,
    nome varchar (100) not null,
    bi integer unique,
    nacionalidade varchar(30) default 'Portuguesa'
    mediaCurso decimal (4 ,2) default 0
        constraint ck_Aluno_mediacurso check (mediaCurso >= 0),
    media_notasExame decimal (4 ,2),
    datanascimento date default GETDATE(),
    cursoid integer
    …. );
```

Modificação de Tabelas

É possível alterar tabelas

- Acrescentar novas colunas, alterar o tipo de dados e eliminar colunas;
- Acrescentar ou retirar *constraints.*

Acrescentar uma coluna

- ALTER TABLE Aluno **ADD** (email varchar(100))

Eliminar uma coluna

- ALTER TABLE Aluno **DROP** column email

Alterar o tipo de dados de uma coluna

- ALTER TABLE Aluno **MODIFY** (email varchar(75))

Modificação de Tabelas

Eliminar uma constraint

- ALTER TABLE Aluno **DROP CONSTRAINT** FK_Nota_id_aluno_Aluno

Acrescentar uma constraint

- ALTER TABLE Aluno **ADD CONSTRAIN**T FK_Nota_id_aluno_Aluno FOREIGN KEY (id_aluno) REFERENCES Aluno(id_aluno)

Considerações:

- A sintaxe dos comandos pode variar de SGBD para SGBD;
- Nem todos os SGBD permitem todas as versões do comando ALTER TABLE;
- Quando se adiciona uma *constraint*, terá sempre de ser no formato de uma table_constraint;
- **A alteração de tipo de dados de uma coluna não pode violar regras de integridade.**

Data Manipulation Language

Data Manipulation Language (DML) é utilizada para efetuar operações de seleção, ordenação, cálculo de informação guardada em tabelas, entre outras.

COMANDOS:

- **INSERT** – inserir dados numa tabela;
- **UPDATE** – atualiza os dados existentes numa tabela;
- **DELETE** – elimina registos numa tabela;
- **SELECT**- recuperar dados da base de dados.

Manipulação de Dados - **INSERT**

insert into *<tabela> <lista de atributos>*
values *<Conjunto de tuplos>*

- O comando insert tem como função permitir inserir registo de dados em tabelas.
- Se forem indicadas o nome das colunas em que queremos inserir os dados, as restantes ficarão com o seu valor por omissão ou nulas.
- Podemos não indicar o nome das colunas. Neste caso somos obrigados a seguir a mesma ordem pela qual criamos a tabela.

Manipulação de Dados - INSERT

- **INSERT INTO aluno (numero, nome, bi) VALUES (12, 'Joao', 3333);**
- **INSERT INTO aluno VALUES (1234 , ' Joao ' , 777777, 'Espanhola', 12.50, 12,4,'12-12-1980',1));**
- **INSERT INTO aluno VALUES (12 , ' Joao ' , 1234, default, 12.6, default, 2);**

Manipulação de Dados - **UPDATE**

update *<tabela>*
set *<Atributo> = <Expressão>, <Atributo> = <Expressão>, ...*
where *<Condição>*

- O comando update permite modificar os dados numa tabela.

Manipulação de Dados - **UPDATE**

```
UPDATE Aluno
SET numero=1234571
WHERE nome='João'

UPDATE Aluno
SET mediacurso=mediacurso * 1.2

UPDATE Aluno
SET mediacurso=mediacurso * 1.2, numero = 123456
WHERE nome='João'
```

Manipulação de Dados - **DELETE**

delete from *<tabela >*
where *<Condição>*

- Usa-se o comando delete para remover registos das tabelas.
- Permite apagar apenas registos inteiros. Não é possível apagar um campo com o Delete.

DELETE FROM aluno WHERE mediacurso > 12;
DELETE FROM aluno WHERE numero = 1234;
DELETE FROM aluno;