



UNIPAC - CENTRO UNIVERSITÁRIO PRESIDENTE ANTÔNIO CARLOS
CAMPUS BARBACENA

Bacharelado em Ciência da Computação



Banco de Dados

Material de Apoio

Parte X – Restrições de Integridade

Prof. José Osvano da Silva, PMP, PSM I
joseosvano@unipac.br

1º sem / 2022

Sumário

- Restrições de Integridade
- Restrições de Vazio
- Restrições de Domínio
- Restrições de Chave – Primária
- Restrições de Chave – Chave Candidata
- Restrição de Integridade Referencial
- Restrições Semânticas
- Gatilhos
- Asserções

Restrições de Integridade

- Objetivo primordial de um SGBD
 - garantir a **integridade de dados**
- Para garantir a integridade de um banco de dados
 - SGBDs oferecem o mecanismo de restrições de integridade
- *Uma restrição de integridade é uma regra de consistência de dados que é garantida pelo próprio SGBD*
 - *Precisa ser testada quando um registro é incluído, alterado ou excluído do BD*

Restrições de Integridade (RI)

- RI garantem que mudanças feitas no banco de dados por usuários autorizados não resultem na perda da consistência dos dados

Restrições de Integridade Básicas

- Restrição de vazio
- Restrições de domínio
- Restrições de chave primária
- Integridade referencial

Garantidas pelo
SGBD

O programador
Não se preocupa
Com estas restrições

- Check constraints
- Gatilhos
- Asserções

Restrições de Integridade Semântica

- Há muitas restrições de integridade que não se encaixam nas categorias básicas
 - Essas restrições são chamadas de restrições semânticas (ou regras de negócio)
- Exemplos de restrições semânticas:
 - Um empregado do departamento “Financeiro” não pode ter a categoria funcional “Engenheiro”.
 - Um empregado não pode ter um salário maior que seu superior imediato.
- Também chamadas de regras de negócio.


The background of the slide is a dark grey surface with faint, light grey chalk-like drawings of various school and office supplies. These include a globe on the left, a stack of books at the top left, a ruler and a compass in the upper left, and a microscope on the right side. The title 'Restrições de Vazio' is centered in a large, bold, black sans-serif font.

Restrições de Vazio

Restrições de Valor Vazio

- O cliente 548 não tem nome
- Esta tupla se refere a um cliente anônimo, o que não tem muito sentido no BD
- Este pode ser um caso em que se deseja proibir valores vazios, restringindo o domínio do atributo nome para *not null*.

Matricula	Nome	endereco
548		Rua Carvalho 615
549	Pedro	Rua Pedro Chaves 22
...		



Restrições de Valor Vazio

- Um valor de campo pode assumir o valor vazio (“null” em inglês)
 - Colunas nas quais **não são admitidos valores vazios** → chamadas de colunas **obrigatórias**
 - Colunas nas quais **podem aparecer valores vazios** → chamadas de **colunas opcionais**
- Abodagem relacional
 - todas colunas que compõem a **chave primária devem ser obrigatórias**
 - demais chaves podem conter colunas opcionais

Restrições de Valor Vazio

- Regra “Nulo”

- **Permite, ou não,** que um atributo de uma tabela tenha valor nulo (ausência de valor)

- Exemplo em SQL:

- Create table funcionario
(matricula integer **not null**,
nome varchar(30) **not null**,
telefone varchar(20))

- Insert into funcionario values (568, "", '48-33542519')

- postgresQL Erro: Null value in column “nome” violates not-null constraint



Restrições de Domínio

Restrições de Domínio

- Refere-se ao domínio de um atributo
 - Conjunto de valores que podem aparecer em uma coluna (atributo)
 - Domínio de valores válidos para um atributo
- Restrições de domínio são as mais elementares
 - Facilmente verificadas pelo sistema

Restrições de Domínio

- Similar aos tipos de variáveis em linguagens de programação
- Vários atributos podem ter o mesmo domínio, mas tem casos em que não faz sentido
- Exemplo: o atributo **idade** é numérico, precisa ser de domínio **inteiro**, e não do tipo character, como é o caso do atributo **nome**

```
Create table funcionario  
(matricula integer not null,  
nome varchar(30) not null,  
idade integer,  
endereco varchar(35))
```

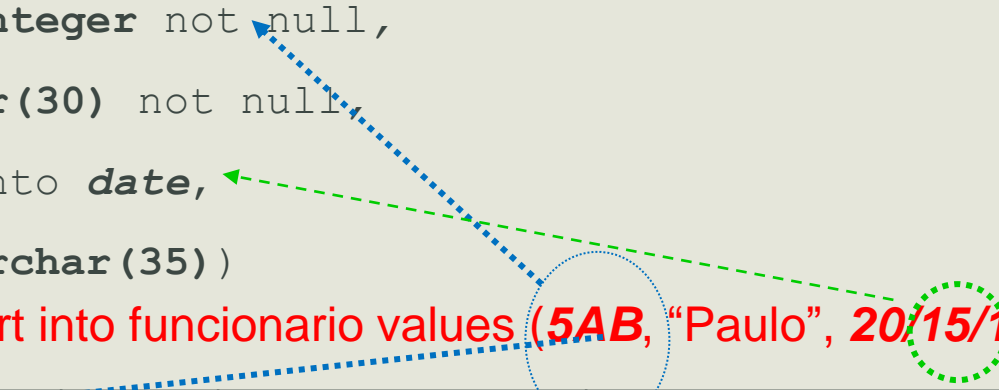
Restrições de Domínio

- O padrão SQL suporta um conjunto restrito de tipos de domínio:
 - Cadeia com comprimento de caracteres fixo, com comprimento especificado pelo usuário
 - Número de casas decimais
 - Inteiro (conjunto finito de números inteiros)
 - Data
 - ...

Create table funcionario

```
(matricula integer not null,  
nome varchar(30) not null,  
dataNascimento date,  
endereco varchar(35) )
```

Insert into funcionario values (**548**, "Paulo", **20/15/1999**, "Av Ipiranga 1900")



Matricula	Nome	dataNascimento	endereco
548	Maria	25/02/1973	Rua Carvalho 615
549	Pedro	14/06/1990	Rua Pedro Chaves 22

The background of the slide is a dark grey chalkboard with various school supplies drawn in white chalk. On the left, there is a globe showing the continents. Above it, there are two circular protractors and a ruler. In the top left corner, there is a stack of books. On the right side, there is a microscope. At the bottom, there are more books and a ruler. The title 'Restrições de Chave' is written in a large, bold, black font in the center of the slide. Below the title, there is a large yellow rectangular area.

Restrições de Chave

Restrições de Chave - Primária

- Regra “chave primária”
 - restringe que cada linha de uma tabela deve ser identificada por um valor único
- Pode ser simples ou composta

Chave simples

```
Create table medico
(codigoM integer not null,
 nome varchar(30) not null,
 endereco varchar(35),
 PRIMARY KEY (codigoM))
```

chave composta

```
Create table consulta
(codigoMedico integer not null,
 codigoPaciente integer not null,
 data date not null,
 PRIMARY KEY (codigoMedico,
 codigoPaciente, data))
```


Restrições de Chave – Chave Candidata

- Restrições **Unique** garantem que os dados contidos em uma coluna ou um grupo de colunas é único em relação a todas as linhas da tabela

- **Sintaxe: quando escrita como uma restrição de coluna**

```
CREATE TABLE produto (nroProduto integer UNIQUE,  
                        nome varchar(30), preco real);
```

- **Sintaxe: quando escrita como uma restrição de tabela**

```
CREATE TABLE produto (nroProduto integer,  
                        nome varchar(30), preco real,  
                        UNIQUE (product_no));
```



Restrições de Integridade Referencial

Restrições de Integridade Referencial

- Uma das restrições mais importantes em BD
- **Definição:** é a garantia de que um valor que aparece em uma relação R1, para um conjunto de atributos, deve obrigatoriamente corresponder a valores de um conjunto de atributos em uma relação R2; OU
 - valores de atributos que são “**chave estrangeira**” em uma relação R1 possuem valores correspondentes em **chaves primárias** da tabela referenciada R2

Restrições de Integridade Referencial

Exemplo

```
CREATE TABLE cidade
(codigoCidade integer NOT NULL,
 descricao varchar(40) NOT NULL,
 estado char(2),
 PRIMARY KEY (codigoCidade))
```

```
CREATE TABLE cliente
(codigoCliente integer NOT NULL,
 nome varchar(30) NOT NULL,
 codCidade integer,
 PRIMARY KEY (codigoCliente),
 FOREIGN KEY (codCidade) REFERENCES Cidade (codigoCidade))
```

Cliente

codigoCliente	Nome	endereco	codigoCidade
548	Maria	Rua Carvalho 615	1
549	Pedro	Rua Pedro Chaves 22	5

Viola a restrição → cidade 5 não existe

Cidade

codigoCidade	Descricao	Estado
1	Florianópolis	SC
2	São José	SC

A restrição garante que não irá existir CLIENTE que more numa cidade que não exista na tabela CIDADE

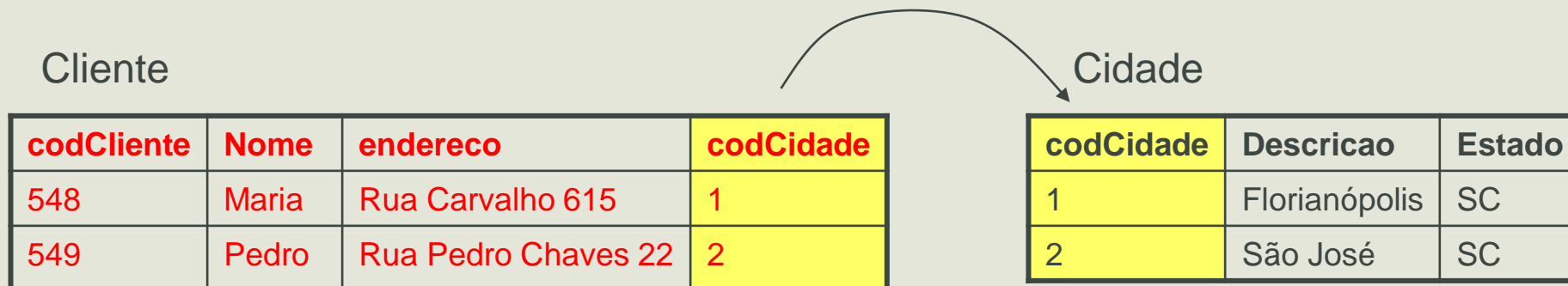
Restrições de Integridade Referencial

- A restrição de integridade é testada quando:
 - **Inclusão:** se uma tupla t_2 é inserida em uma relação r_2 , o sistema precisa assegurar que existe uma tupla t_1 em uma relação r_1 tal que $t_1[r_1] = t_2[r_2]$
 - Ex: inclui novo cliente, testa se a cidade existe
 - **Exclusão:** Uma chave primária referenciada **é removida**
 - ON DELETE
 - Ex: Remove uma cidade referenciada por algum cliente
 - **Alteração:** Uma chave primária referenciada é alterada
 - ON UPDATE
 - Ex: Altera a chave primaria da cidade referenciada em cliente

Restrições de Integridade Referencial - INCLUSÃO

- **Inclusão**: ao inserir um novo cliente, é preciso garantir que o código da cidade na tabela **cliente** EXISTA na tabela **cidade**
- Para garantir isso cria-se a tabela de cliente com a chave estrangeira **codCidade**

```
CREATE TABLE cliente  
(codigoCliente integer NOT NULL,  
nome varchar(30) NOT NULL,  
codCidade integer,  
PRIMARY KEY (codigoCliente),  
FOREIGN KEY (codCidade) REFERENCES Cidade (codigoCidade))
```



Restrições de Integridade Referencial - EXCLUSÃO

■ AÇÕES:

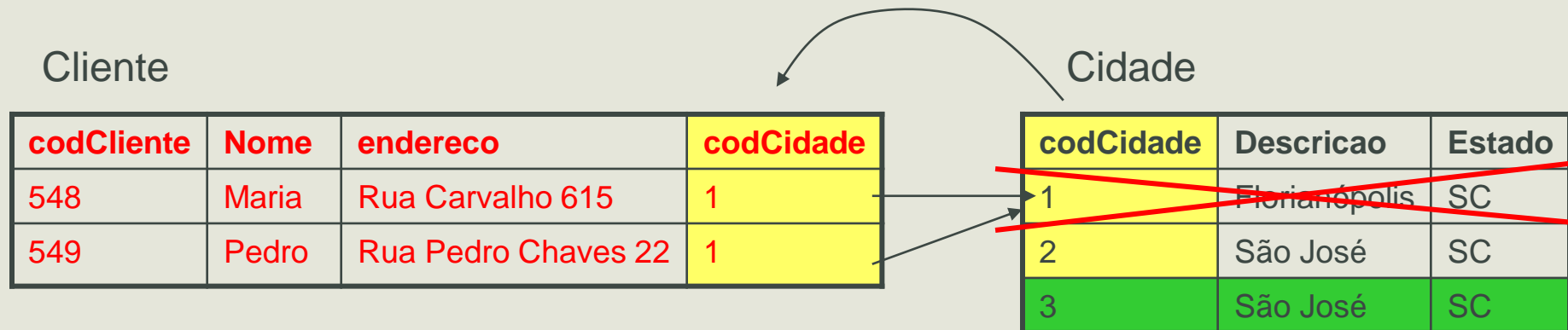
- (opção 1) NÃO permite alteração ou exclusão (**NO ACTION (default) ou Restrict**);
 - não permite a exclusão/alteração enquanto houver dependência;
 - Ex: só permite excluir a cidade quando nenhum cliente referenciar esta cidade
- **(opção 2) SET DEFAULT** : se houver um valor default para a coluna da chave estrangeira, ela recebe este valor
 - Ao excluir uma cidade que tenha dependência;
 - Ex: seta valor nulo para a coluna codCidade.
- **(opção 3) CASCADE** : propaga a exclusão/alteração;



Restrições de Integridade Referencial - EXCLUSÃO

(opção 1) NÃO permitir a exclusão da cidade 1 porque tem clientes morando nesta cidade. A cidade 3 pode ser excluída

```
CREATE TABLE cliente  
(codCliente integer NOT NULL, nome varchar(40) NOT NULL, endereco varchar(40), codCidade  
integer,  
PRIMARY KEY (codCliente),  
FOREIGN KEY (codCidade) REFERENCES Cidade (codCidade) ON DELETE RESTRICT))
```

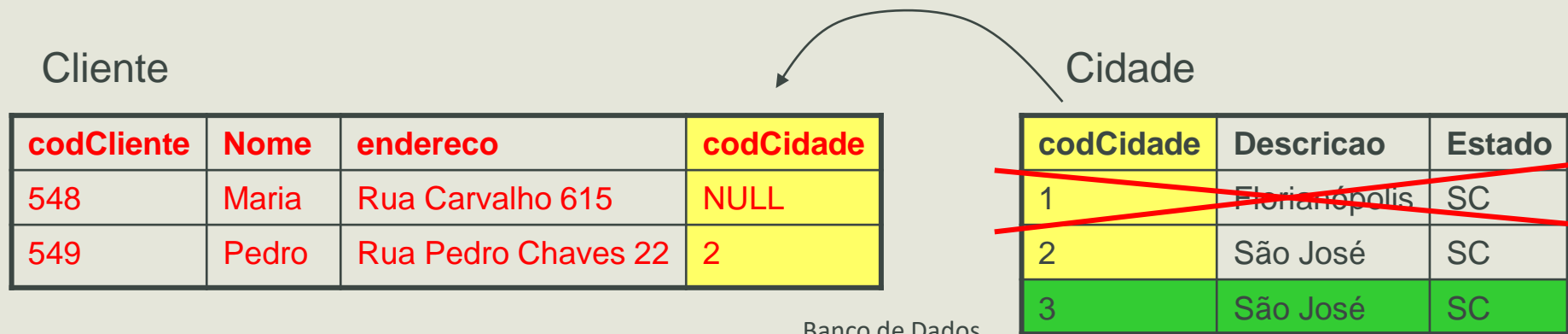


Restrições de Integridade Referencial - EXCLUSÃO

- **(opção 2)** assumir um valor default. Ao excluir uma cidade da tabela **CIDADE**, o SGBD precisa garantir que não exista nenhum cliente na tabela **CLIENTE** referenciando esta cidade

```
CREATE TABLE cliente  
(codigoCliente integer NOT NULL, nome varchar(40) NOT NULL, codigoCidade integer,  
PRIMARY KEY (codigoCliente),  
FOREIGN KEY (codigoCidade) REFERENCES Cidade (codigoCidade) ON DELETE SET DEFAULT)
```

OBS: Não muito recomendado/usado



Restrições de Integridade Referencial - EXCLUSÃO

(opção 3) remove as referencias (remove a cidade e todos os clientes da cidade)

```
CREATE TABLE cliente
(codCliente integer NOT NULL, nome varchar(40) NOT NULL, endereco varchar (40),
codCidade integer,
PRIMARY KEY (codCliente),
FOREIGN KEY (codCidade) REFERENCES Cidade (codigoCidade) ON DELETE CASCADE))
```



ISTO NÃO É permitido neste contexto: significa que AO REMOVER A CIDADE,
REMOVA TAMBEM O CLIENTE

Cliente

codCliente	Nome	endereco	codCidade
548	Maria	Rua Carvalho 615	1
549	Pedro	Rua Chaves 22	1

Cidade

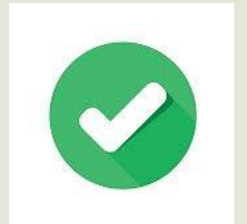
codCidade	Descricao	Estado
1	Florianópolis	SC
2	São José	SC



Restrições de Integridade Referencial - EXCLUSÃO

- **Exclusão:** on delete cascade é útil quando ao remover a tupla de uma tabela queremos excluir todas as tuplas relacionadas de outra tabela

```
CREATE TABLE Preco_Produto  
(codProduto integer NOT NULL, data Date, preco float,  
PRIMARY KEY (codProduto, data),  
FOREIGN KEY (codProduto) REFERENCES Produto (codProduto) ON DELETE RESTRICT),  
FOREIGN KEY (codProduto) REFERENCES Preco (codProduto) ON DELETE CASCADE))
```



Produto

codProduto	Nome
548	Pao
549	Leite

Preco

codProduto	data	preco
548	01/12/2016	2,00
548	06/06/2017	5,00
549	01/02/2017	3,00



Restrições Semânticas

Restrições Semânticas

```
CREATE TABLE products ( product_no integer, name text, price integer CHECK (price > 2 ) ;
```

```
Teste: insert into products values (1,'teste',0)
```

```
CREATE TABLE products ( product_no integer,  
                           name text,  
  
                           price numeric CONSTRAINT positive_price CHECK (price > 0) );
```

```
CREATE TABLE products  
  product_no integer,  
  name text,  
  price numeric CHECK (price > 2),  
  discounted_price numeric  
                      CHECK (discounted_price > 0),  
  CHECK (price > discounted_price) );
```

```
Teste1: insert into products values (1,'teste',0,0)
```

```
Teste2: insert into products values (1,'teste',100,200)
```

```
Teste3: insert into products values (1,'teste',100,50)
```

A dark grey background featuring a collage of white, hand-drawn educational icons. These include a globe, a stack of books, a microscope, a test tube, a percentage sign, and various geometric shapes like triangles and rectangles.

Gatilhos

Gatilhos

- Interessante quando a inclusão, alteração ou exclusão de um atributo em alguma tabela tiver algum efeito sobre um atributo de outra tabela
- Exemplo: quando o saldo da conta do cliente for negativo, insira automaticamente um registro na tabela empréstimo e faça o valor do saldo na conta receber o valor zero

Gatilhos

- Não são o modo recomendado para implementar restrições de integridade
- RI são normalmente suportadas pelos SGBD atuais
- Gatilhos podem ser disparados ANTES ou DEPOIS do evento (insert, delete, update) especificado.

Gatilhos(triggers)

Define trigger saldo-negativo

On update of deposito t

(**if new** t.saldo<0 **then**

insert into emprestimo **values**(t.nomeAgencia,
t.numeroConta, t.nomeCliente, **-new** t.saldo)

update deposito s

set s.saldo=0 **where** s.numeroConta=t.numeroConta))

Gatilhos (triggers)

```
1. CREATE TRIGGER [central].[desativarVeiculos] ON [central].[organizacao] WITH EXECUTE AS CALLER AFTER UPDATE
2. AS
3. declare @vAtivo bit;
4. declare @vAtivoOld bit;
5. declare @vIdOrganizacao int;
6. BEGIN
7.     SET NOCOUNT ON;
8.     select @vAtivo = ativo, @vIdOrganizacao = idOrganizacao
9.     from inserted;

10.    select @vAtivoOld = ativo
11.    from deleted;
12.
13.    if ((@vAtivo = 0) and (@vAtivoOld = 1))
14.    begin
15.        update central.veiculo
16.        set desativacao = GETDATE(), ativo = 0
17.        where idOrganizacao = @vIdOrganizacao and ativo = 1;

18.        update central.motorista
19.        set desativacao = GETDATE()
20.        where idOrganizacao = @vIdOrganizacao and desativacao is null;
21.    end
22.    SET NOCOUNT OFF;
23. END
```

Asserções

- Predicado que expressa uma condição que deve ser sempre satisfeita no BD
 - Exemplo
 - Um aluno só pode se matricular se não tiver débitos pendentes na biblioteca
- sintaxe
 - `create assertion <nome_restrição> check predicado`
 - `drop assertion <nome_restrição>`

Asserções

- Como o SQL não suporta PARA TODO, precisamos usar o NOT EXISTS
- Quando uma asserção é criada
 - O sistema verifica sua validade
 - Se as asserções são válidas
- Qualquer modificação posterior no banco será permitida somente quando a asserção não for violada
- A Verificação de asserções pode gerar um aumento significativo no tempo de processamento

Asserções

- Exemplo: o número da conta do cliente precisa estar entre 10.000 e 99.999

```
create assertion restricaoConta check
```

```
(not exists (select * from conta
```

```
        where  numero < 10.000 OR numero > 99.999) )
```

Exercício de Fixação 12

- 1) No Banco de Dados “BANCO” do Exercício 08;
- 2) Reveja toda a estrutura do Banco de Dados deixando todas as colunas como obrigatórias;
- 3) Confira se todas as chaves estrangeiras estão criadas e se não tiver inclua em seu script.

Dúvidas



José Osvano da Silva
joseosvano@unipac.br