



# Restrições de Integridade

*Miriã Corrêa*

Análise e Desenvolvimento de Sistemas

[miriacoeelho@gmail.com](mailto:miriacoeelho@gmail.com)



# Sumário

Descrição das restrições de integridade

NOT NULL e DEFAULT

Chaves

PRIMARY KEY

UNIQUE

FOREIGN KEY

# Descrição

- As restrições básicas de integridade podem ser definidas no comando SQL como parte da criação de uma tabela.
- Elas podem ser usadas para impor regras no nível da tabela, sempre que uma operação de incluir uma nova linha, remover ou modificar uma linha existente for executada.
- Possibilitam, ainda, impedir que uma tabela seja removida se houver dependências de outras tabelas.



# Restrições básicas de integridade

- As restrições básicas de integridade de dados são:
  - NOT NULL;
  - PRIMARY KEY;
  - UNIQUE;
  - FOREIGN KEY.



# NOT NULL

- Como a SQL permite NULLs como valores de atributo, uma restrição NOT NULL pode ser especificada se o valor NULL não for permitido para determinado atributo.
- Isso sempre é especificado de maneira implícita para os atributos que fazem parte da chave primária de cada relação.



# NOT NULL

Esta coluna permite  
valores nulos



	matricula	nome_aluno	cod_curso	endereco
1	111	Joaquim	MAT	
2	123	Fulano	CC	
3	234	Beltrano	MAT	
4	456	Ciclano	FIS	
5	789	Maria	HIS	



# NOT NULL - Declaração

- É possível definir o NOT NULL na definição de atributo durante a criação da tabela ou após a criação da tabela.
- Declaração do NOT NULL na definição do atributo durante a criação da tabela.
  - Sintaxe:

**<nome coluna> <tipo coluna> NOT NULL;**



# NOT NULL

```
create table aluno(matricula decimal(4) NOT NULL,  
                    nome varchar(30),  
                    tipo_aluno integer(1),  
                    curso varchar(3),  
                    endereco varchar(40) NOT NULL);
```

Endereço e matrícula não podem conter o valores NULL.

```
matricula decimal(4) NOT NULL  
endereco varchar(40) NOT NULL
```





# NOT NULL

- Declaração do NOT NULL após a criação da tabela.
  - Sintaxe:

```
ALTER TABLE <nome_da_tabela> MODIFY <nome_da_coluna>  
<tipo_da_coluna> NOT NULL;
```

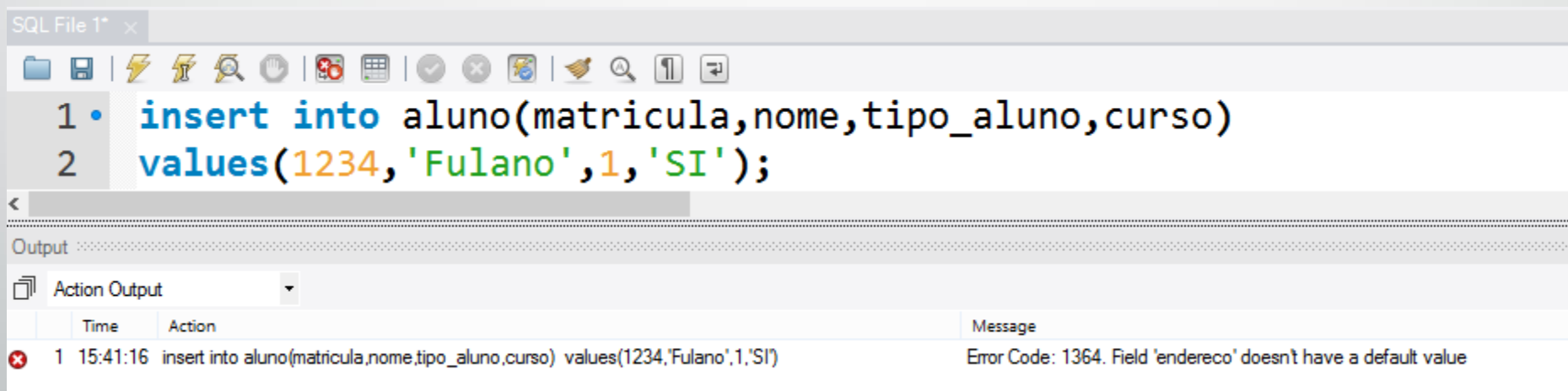
- Exemplo:

```
alter table aluno modify endereco varchar(40) not null;
```



# NOT NULL

- Se tentássemos inserir um novo aluno sem o campo endereço obteríamos a seguinte mensagem de erro:



```
1 • insert into aluno(matricula,nome,tipo_aluno,curso)
2 values(1234,'Fulano',1,'SI');
```

Output

	Time	Action	Message
✖	1 15:41:16	insert into aluno(matricula,nome,tipo_aluno,curso) values(1234,'Fulano',1,'SI')	Error Code: 1364. Field 'endereco' doesn't have a default value

# DEFAULT

- É possível definir um valor padrão para um atributo, para isto basta adicionar à cláusula **DEFAULT** <valor> a uma definição de atributo.
- Se nenhuma cláusula default for especificada, o valor padrão será NULL para atributos que não possuem a restrição NOT NULL.

– Sintaxe:

```
<nome coluna> <tipo coluna>[restrições de atributo] DEFAULT <valor>;
```

# DEFAULT

```
create table aluno(matricula decimal(4) NOT NULL,  
                    nome varchar(30),  
                    tipo_aluno integer(1) default 1,  
                    curso varchar(3));
```

- Se nenhum valor for especificado durante a inserção de um aluno a coluna tipo\_aluno receberá o valor 1.

```
tipo_aluno integer(1) default 1;
```






# DEFAULT

- Ao inserirmos um novo aluno sem especificar o valor do campo `tipo_aluno` ele automaticamente receberá o valor default que é 1.

```
7 • insert into aluno(matricula,nome,curso,endereco)
8   values(1234,'Fulano','SI','Rua A');
9
10 • select * from aluno;
11
```

<

Result Grid |  Filter Rows:  | Export:  | Wrap Cell Content: 

	matricula	nome	tipo_aluno	curso	endereco
	1234	Fulano	1	SI	Rua A

# DEFAULT

- Declaração do DEFAULT após a criação da tabela.

– Sintaxe:

```
ALTER TABLE <nome_da_tabela> MODIFY <nome_da_coluna>  
<tipo_da_coluna> DEFAULT <valor> ;
```

– Exemplo:

```
alter table aluno modify endereco varchar(40) default '?';
```

# DEFAULT

Esta coluna possui o  
valor default '?'



	matricula	nome_aluno	cod_curso	endereco
1	111	Joaquim	MAT	?
2	123	Fulano	CC	?
3	234	Beltrano	MAT	?
4	456	Ciclano	FIS	?
5	789	Maria	HIS	?



# Restrições de chave

- Como chaves e restrições de integridade referencial são muito importantes, existem cláusulas especiais dentro da instrução CREATE TABLE para especificá-las.
- A cláusula PRIMARY KEY especifica um ou mais atributos que compõem a chave primária de uma relação.





# PRIMARY KEY (chave primária)

- Chave primária: é uma chave escolhida dentre as chaves candidatas.
- A coluna que contém a chave primária não pode conter valores nulos e nem valores repetidos. Os valores contidos nesta coluna devem ser capazes de representar univocamente as tuplas contidas na tabela.



# PRIMARY KEY (chave primária)

- Exemplos:

Chave primária



Chave candidata



	matricula	nome	endereço	cidade	cpf
1	200001	Marcos Alfredo	Rua Margarida, 32	Juiz de Fora	11122233344
2	200002	Periceles de Abreu	Rua Leopoldo, 12	Juiz de Fora	22233344455
3	200003	Antônia de Souza	Rua Itamar, 563	Juiz de Fora	33344455566
4	200004	Raimundo Silva	Rua Pereira, 456	Juiz de Fora	44455566677
5	200005	Escobar Peres	Rua Moraes, 451	Juiz de Fora	55566677788
6	200006	Daniela Araújo	Rua Catarina, 14	Juiz de Fora	66677788899
7	200007	Daniel Duarte	Rua Pedro Aquino,	Juiz de Fora	77788899900
8	200008	Gabriela Caldas	Rua Jose, 23	Juiz de Fora	88899900011
9	200009	Fernando Henrique	Rua Tadeu, 458	Juiz de Fora	99900011122

# PRIMARY KEY

- Exemplos:

Chave primária



	num_peca	pnome	cor	peso	cidade	preco
1	1	Porca	Vermelho	12	Londres	2.2
2	2	Pino	Verde	17	Paris	1.3
3	3	Parafuso	Azul	17	Oslo	1.2
4	4	Parafuso	Vermelho	14	Londres	1.21
5	5	Came	Azul	12	Paris	1.32
6	6	Tubo	Vermelho	19	Londres	3.85
7	7	Prego	Preto	12	Paris	3

# PRIMARY KEY- Declaração

- É possível definir a PRIMARY KEY durante a criação da tabela ou após a criação da tabela.
- Declaração de PRIMARY KEY durante a criação da tabela.
  - Sintaxe:

**CONSTRAINT**<nome constraint> **PRIMARY KEY** (<nome coluna>);



# PRIMARY KEY

- Se uma chave primária tiver um único atributo, a cláusula pode acompanhar o atributo diretamente.
- Por exemplo:

```
create table aluno(matricula decimal(4) NOT NULL primary key,  
                    nome varchar(30),  
                    tipo_aluno integer,  
                    curso varchar(3));
```

- Ou pode ser colocada ao final da declaração da tabela.

```
create table aluno(matricula decimal(4) NOT NULL ,  
                    nome varchar(30),  
                    tipo_aluno integer,  
                    curso varchar(3),  
                    constraint pk_aluno primary key(matricula));
```

# PRIMARY KEY

- Se a tabela já tiver sido criada a inserção da chave primária é feita da seguinte forma:

- Sintaxe:

```
ALTER TABLE <nome tabela> ADD CONSTRAINT<nome constraint>  
PRIMARY KEY (<nome coluna>;
```

- Por exemplo:

```
alter table aluno add constraint pk_aluno  
primary key(matricula);
```



# Nomeando as restrições

- Os nomes de todas as restrições dentro de um esquema em particular precisam ser exclusivos.
- Um nome de restrição é usado para identificar a restrição em particular caso ela deva ser removida mais tarde e substituída por outra.

```
constraint pk_aluno primary key(matricula)
```






# UNIQUE

- Usada para especificar as chaves candidatas.

Chave candidata

Chave primária



	matricula	nome	endereco	cidade	cpf
1	200001	Marcos Alfredo	Rua Margarida, 32	Juiz de Fora	11122233344
2	200002	Periceles de Abreu	Rua Leopoldo, 12	Juiz de Fora	22233344455
3	200003	Antônia de Souza	Rua Itamar, 563	Juiz de Fora	33344455566
4	200004	Raimundo Silva	Rua Pereira, 456	Juiz de Fora	44455566677
5	200005	Escobar Peres	Rua Moraes, 451	Juiz de Fora	55566677788
6	200006	Daniela Araújo	Rua Catarina, 14	Juiz de Fora	66677788899
7	200007	Daniel Duarte	Rua Pedro Aquino,	Juiz de Fora	77788899900
8	200008	Gabriela Caldas	Rua Jose, 23	Juiz de Fora	88899900011
9	200009	Fernando Henrique	Rua Tadeu, 458	Juiz de Fora	99900011122



# UNIQUE - Declaração

- É possível definir UNIQUE durante a criação da tabela ou após a criação da tabela.
- Declaração de UNIQUE durante a criação da tabela.
  - Sintaxe:

**CONSTRAINT** <nome constraint> **UNIQUE** <nome coluna>;



# UNIQUE

- A cláusula **UNIQUE** especifica chaves alternativas (candidatas), como:

```
create table aluno(matricula decimal(4) NOT NULL,  
                    nome varchar(30),  
                    cpf decimal(11) NOT NULL,  
                    tipo_aluno integer,  
                    curso varchar(3),  
                    primary key(matricula),  
                    constraint ck unique(cpf));
```

```
constraint ck unique(cpf)
```

# UNIQUE

- Esta cláusula também pode ser especificada diretamente para uma chave candidata se esta for um único atributo, como no exemplo a seguir:

```
create table aluno(matricula decimal(4) NOT NULL,  
                    nome varchar(30),  
                    cpf decimal(11) NOT NULL UNIQUE,  
                    tipo_aluno integer,  
                    curso varchar(3),  
                    primary key(matricula));
```

```
cpf decimal(11) NOT NULL UNIQUE
```



# UNIQUE

- Se a tabela já tiver sido criada a inserção da chave candidata é feita da seguinte forma:
- Sintaxe:

```
ALTER TABLE <nome tabela> ADD CONSTRAINT<nome constraint>  
UNIQUE (<nome coluna>);
```

- Por exemplo:

```
alter table aluno add constraint ck unique(cpf);
```




# FOREIGN KEY (chave estrangeira)

- Exemplo:

Chave primária

Chave estrangeira

**Aluno**



	matricula	nome_aluno	curso
1	111	Joaquim	MAT
2	123	Fulano	CC
3	234	Beltrano	MAT
4	456	Ciclano	FIS
5	789	Maria	MED

Chave primária

**Curso**



	codigo	nome
1	CC	Ciência da Computação
2	FIS	Física
3	GEO	Geografia
4	HIS	História
5	MAT	Matemática Aplicada
6	MED	Medicina

# FOREIGN KEY

- É possível definir a FOREIGN KEY durante a criação da tabela ou após a criação da tabela.
- Declaração de FOREIGN KEY durante a criação da tabela.
  - Sintaxe:

```
CONSTRAINT<nome constraint> FOREIGN KEY (<nome coluna>)  
REFERENCES <nome tabela>(<nome coluna>;
```



# Integridade referencial

- A integridade referencial é especificada por meio da cláusula **FOREIGN KEY**.

```
create table aluno(matricula decimal(4) NOT NULL,  
                   nome varchar(30),  
                   cpf decimal(11) NOT NULL UNIQUE,  
                   tipo_aluno integer,  
                   curso varchar(3),  
                   primary key (matricula),  
                   CONSTRAINT fk_curso foreign key(curso)  
                   references curso(codigo));
```

```
CONSTRAINT fk_curso foreign key(curso)  
references curso(codigo));
```

# Integridade referencial

- Se a tabela já tiver sido criada a inserção da chave estrangeira é feita da seguinte forma:
- Sintaxe:

```
ALTER TABLE <nome tabela> ADD CONSTRAINT<nome constraint>  
FOREIGN KEY (<nome coluna>)REFERENCES <nome tabela> (<nome coluna>)
```

- Por exemplo:

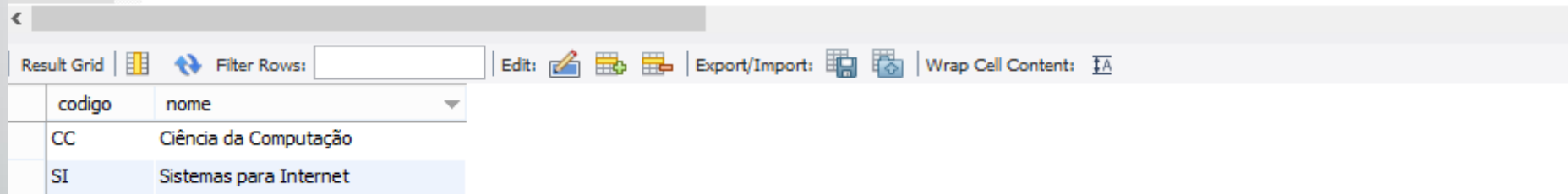
```
alter table aluno add constraint fk_curso  
foreign key (curso) references curso(codigo);
```



# FOREIGN KEY (chave estrangeira)

- Exemplo:

```
1 • INSERT INTO curso (codigo,nome) values('SI', 'Sistemas para Internet'),  
2   ('CC', 'Ciência da Computação');  
3 • SELECT * FROM universidade.curso;
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

codigo	nome
CC	Ciência da Computação
SI	Sistemas para Internet

# FOREIGN KEY (chave estrangeira)

- Caso tentássemos inserir um aluno que com o valor de curso que não está cadastrado na tabela curso, obteríamos a seguinte mensagem de erro:

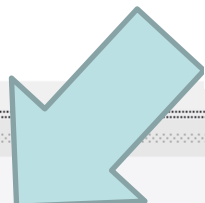
```
1 • insert into aluno(matricula,nome,curso)
2 values (431, 'Ciclano','ENG')
```

**Violação de restrição de integridade referencial**

Output


Action Output

	Time	Action	Message
✖	1 17:45:29	insert into aluno(matricula,nome,curso) values (431, 'Ciclano','ENG')	Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('universidade'. ...



# FOREIGN KEY (chave estrangeira)

```
1 • insert into aluno(matricula,nome,curso)
2 values (431, 'Ciclano','SI')
```



Output

Action Output

	Time	Action	Message
✓ 1	17:46:33	insert into aluno(matricula,nome,curso) values (431, 'Ciclano','SI')	1 row(s) affected



# Integridade referencial

- Uma restrição de integridade pode ser violada quando tuplas são inseridas, atualizadas ou excluídas.
- A ação default que a SQL toma para uma violação de integridade é **rejeitar** a operação de atualização que causará uma violação, o que é conhecido como opção RESTRICT.



# Integridade referencial

- Porém o projetista de BD pode especificar uma ação alternativa para ser tomada conectando uma cláusula de **ação de disparo referencial** a qualquer restrição de chave estrangeira.
- As opções incluem SET NULL, CASCADE e RESTRICT.
- Uma opção deve ser qualificada com ON DELETE ou ON UPDATE.



# RESTRICT

- O comando RESTRICT rejeita a operação de exclusão ou atualização para tabela que é referenciada por outras tabelas.
- Por exemplo: A tabela curso é referenciada na tabela aluno, logo se declararmos a chave estrangeira na tabela aluno com a opção restrict e tentarmos eliminar uma linha da tabela curso a operação não será efetuada.



# RESTRICT

- Exemplo:

Chave primária

Chave estrangeira

Chave primária

**Aluno**

	matricula	nome_aluno	curso
1	111	Joaquim	MAT
2	123	Fulano	CC
3	234	Beltrano	MAT
4	456	Ciclano	FIS
5	789	Maria	MED

**Curso**

	codigo	nome
1	CC	Ciência da Computação
2	FIS	Física
3	GEO	Geografia
4	HIS	História
5	MAT	Matemática Aplicada
6	MED	Medicina

# RESTRICT

- Exemplo:

```
alter table aluno add constraint fk_curso  
foreign key(curso) references curso(codigo)  
on delete restrict;
```

```
1 • delete from curso where codigo='FIS';
```

Output

Action Output

	Time	Action	Message
✖	1 17:56:26	delete from curso where codigo='FIS'	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails ('universida...



# CASCADE

- Elimina ou atualiza a linha da tabela que é referenciada por outras tabelas e apaga automaticamente ou atualiza as linhas correspondentes na tabela que a referenciou.
- Por exemplo: A tabela curso é referenciada na tabela aluno, logo se declararmos a chave estrangeira na tabela aluno com a opção cascade e tentarmos eliminar uma linha da tabela curso a operação será efetuada e a linha que continha o código na tabela aluno será apagada.



# CASCADE

```
alter table aluno add constraint fk_curso foreign key (curso)  
references curso(codigo) on delete cascade
```

- Exemplo:

	matricula	nome	tipo_aluno	curso	endereco
	456	Cidano	1	FIS	?

```
1 • delete from curso where codigo='FIS';
```

Output

Action Output

	Time	Action
✓	1 18:48:11	delete from curso where codigo='FIS'

```
1 • SELECT * FROM aluno;
```

Result Grid | Filter Rows: | Edit: |

	matricula	nome	tipo_aluno	curso	endereco
*	NULL	NULL	NULL	NULL	NULL

# Integridade referencial

```
create table aluno(matricula decimal(4) NOT NULL primary key,  
    nome varchar(30),  
    cpf decimal(11),  
    tipo_aluno integer,  
    curso varchar(3),  
    constraint fk_curso foreign key(curso)  
    references curso(codigo) on delete cascade  
    on update cascade);
```

```
alter table aluno add constraint fk_curso foreign key(curso)  
    references curso(codigo) on delete cascade  
    on update restrict;
```

```
alter table aluno add constraint fk_curso foreign key(curso)  
    references curso(codigo) on delete cascade  
    on update set null;
```



# Dúvidas?

