



DISCIPLINA: Banco de Dados 1

Prof. **GIOVANI** Volnei Meinerz

Aula 13 – SQL

Objetivos da Aula

- Aprender comandos de definição de dados em SQL

SQL: O que é?

Linguagem de Consulta SQL

SQL (*STRUCTURED QUERY LANGUAGE*)

(Linguagem de Consulta Estruturada)

Linguagem de banco de dados relacionais, composta de comandos que permitem aos usuários

- (i) a criação de bancos de dados e estruturas de tabela,
- (ii) a manipulação dos dados para extrair informações, e
- (iii) a administração de dados

História da SQL

- Concebida originalmente em 1974 pela IBM, como parte do projeto **System R**
- Objetivo foi demonstrar a viabilidade da implementação do modelo relacional proposto por *Codd*, e desenvolver um método padrão para acessar e manipular dados em um BD relacional
- O projeto **System R** foi concluído em 1979, e resultou no desenvolvimento do primeiro SGBDR da IBM (1981)
- Inicialmente chamada de **SEQUEL** – *Structured English QUery Language* (Linguagem de Consulta em Inglês Estruturado)
- Em 1980 foi renomeada para SQL, para evitar confusões e problemas legais (marca comercial - **SEQUEL** - já registrada)

História da SQL (cont.)

- Principais SGBDRs com suporte à SQL disponibilizados por diferentes fornecedores
 - Em 1979 a *Relational Software, Inc* (atualmente *Oracle Corporation*) disponibilizou o *Oracle V2*
 - Em 1981 a IBM disponibilizou o *SQL/DS*
 - Em 1983, a IBM disponibilizou o *DB2*
 - Em 1984, a *Data General Corporation* disponibilizou o *DG/SQL*
 - Em 1986, a *Sybase* disponibilizou o *INGRES*
- Tornou-se linguagem padrão de BDs relacionais
 - No entanto, tal expansão fez surgir vários “dialetos”

História da SQL (cont.)

- ✈ Em resposta a proliferação dos dialetos SQL, a ANSI publicou a primeira versão padronizada da SQL em 1986



- ✈ A ISO aprovou o padrão em 1987



Versões SQL

- ➔ Desde então, várias revisões foram realizadas para adicionar novos atributos e incorporar novos comandos e capacidades à linguagem

ANO	NOME	ALGUMAS DAS MELHORIAS INCORPORADAS
1986	SQL-86	Primeira versão formalizada pela ANSI
1989	SQL-89	Definição de restrições de integridade, ...
1992	SQL-92	Novas operações sobre conjuntos (<i>union, natural join, difference...</i>)
1999	SQL:1999	Suporte a SQL embutido em Java; <i>Triggers</i> , ...
2003	SQL:2003	Adiciona tipo XML; permite geração de sequencias padronizadas, ...
2006	SQL:2006	Importar e armazenar dados XML em um BD relacional via SQL, ...
2008	SQL:2008	Adiciona cláusulas <i>ORDER BY</i> e <i>TRUNCATE</i> , ...
2011	SQL:2011	Melhoria na definição e manipulação de dados temporais, ...
2016	SQL:2016	Funções para criação de documentos JSON, ...

Partes da SQL

→ Linguagem de Definição de Dados (DDL)

- DDL da SQL fornece comandos para definir esquemas de relação, excluir relações e modificar esquemas

→ Linguagem de Manipulação de Dados (DML)

- DML da SQL oferece a capacidade de consultar informações do banco de dados e inserir, excluir e modificar tuplas

→ Integridade

- DDL da SQL permite especificar restrições de integridade às quais os dados precisam satisfazer

Partes da SQL (cont.)

→ Definição de visão (*view*)

- DDL da SQL inclui comandos para definir visões

→ Controle de Transação

- DDL da SQL inclui comandos para especificar o início e o fim das transações

→ SQL embutida e SQL dinâmica

- Definem como instruções SQL podem ser incorporadas dentro de linguagens de programação

→ Autorização

- DDL da SQL inclui comandos para especificar direitos de acesso para relações e *views*

Comandos DDL da SQL

- Comandos que permitem criar, excluir e modificar a estrutura do banco de dados

COMANDO	DESCRIÇÃO
CREATE DATABASE	Cria um banco de dados
DROP DATABASE	Exclui um determinado banco de dados
CREATE TABLE	Cria nova tabela no banco de dados
DROP TABLE	Exclui uma tabela (e seus dados) de forma permanente
ALTER TABLE	Modifica a definição de uma tabela (adiciona, modifica ou exclui atributos ou restrições)

CREATE DATABASE

→ Sintaxe SQL

```
CREATE DATABASE nome_do_banco_de_dados;
```

→ Exemplo

→ Escreva a sentença SQL que cria o banco de dados “bd1”

```
1  -- Sentença SQL que cria o BD "bd1"
2  ● CREATE DATABASE bd1;
3  -- ou
4  ● CREATE SCHEMA bd1;
5
6  -- Sentença alternativa...
7  -- Se o BD ainda não existir, crie-o
8  -- Do contrário, não faça nada
9  ● CREATE DATABASE IF NOT EXISTS bd1;
10 -- ou
11 ● CREATE SCHEMA IF NOT EXISTS bd1;
```

Restrições de Integridade

- Conjunto de regras que garantem a consistência dos dados do BD
- SGBDRs oferecem mecanismo para implementá-las
- Podem ser declaradas como parte do comando **CREATE TABLE**
- Também podem ser adicionadas, modificadas ou excluídas de tabelas existentes usando o comando **ALTER TABLE**

Restrições de Integridade (cont.)

- Restrições sobre os atributos de uma tabela
 - Integridade de Domínio (**TIPO DO DADO**)
 - Integridade de Vazio (**NULL / NOT NULL**)
 - Integridade de Chave (**PRIMARY KEY**)
 - Integridade de Unicidade (**UNIQUE**)
 - De validação de dados (**CHECK**)
- Restrição de integridade referencial
 - Integridade Referencial (**FOREIGN KEY**)

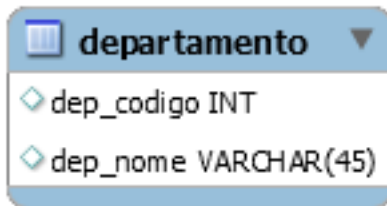
CREATE TABLE

→ Sintaxe SQL

```
CREATE TABLE nome_da_tabela (  
    atributo1 tipo [restrição],  
    atributon tipo [restrição],  
    PRIMARY KEY (atributo1 [, atributon]),  
    FOREIGN KEY (atributo1 [, atributon]) REFERENCES tabela_referenciada  
    (atributo_tabela_referenciada)  
);
```

→ Exemplo – restrição de integridade de domínio

→ Escreva a sentença SQL que cria a tabela referente à entidade a seguir




```
1  -- Indicar em qual banco de dados as sentenças deverão ser executadas  
2  • USE bd1;  
3  
4  • CREATE TABLE departamento (  
5      dep_codigo INT NULL,  
6      dep_nome VARCHAR(45) NULL  
7  );  
8  
9  -- alternativamente...  
10 • CREATE TABLE IF NOT EXISTS bd1.departamento (  
11     dep_codigo INT NULL,  
12     dep_nome VARCHAR(45) NULL  
13 );
```




CREATE TABLE (cont.)

- Exemplo – vide restrições a seguir
 - Baseado na entidade do DER a seguir, escreva a sentença SQL que cria a tabela, contendo as restrições de: **chave**, **vazio**, **domínio**, **unicidade** (além da CHECK(condição_de_validação))

departamento	
dep_codigo	INT
dep_nome	VARCHAR(45)
dep_ramal	INT

departamento - Table

 Table Name:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 dep_codigo	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 dep_nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
 dep_ramal	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

```
1 • USE bd1;
2
3 • CREATE TABLE departamento (
4     dep_codigo INT NOT NULL,
5     dep_nome VARCHAR(45) NOT NULL,
6     dep_ramal INT NOT NULL UNIQUE CHECK(dep_ramal BETWEEN 1111 AND 9999),
7     PRIMARY KEY (dep_codigo)
8     -- UNIQUE dep_ramal_UNIQUE (dep_ramal)
9 );
```

CREATE TABLE (cont.)

- Exemplo de **restrição de integridade referencial**
 - Escreva a sentença SQL que cria a tabela referente à entidade CURSO, e que referencia a entidade DEPARTAMENTO, por meio de sua chave primária DEP_CODIGO



Table: **departamento**

Columns:

<u>dep_codigo</u>	int(11) PK
dep_nome	varchar(45)
dep_ramal	int(11)

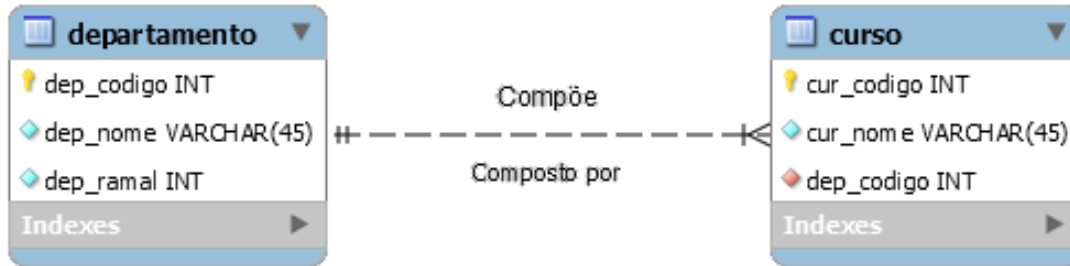
23 • describe departamento;

Result Grid

	Field	Type	Null	Key
	dep_codigo	int(11)	NO	PRI
	dep_nome	varchar(45)	NO	
	dep_ramal	int(11)	NO	UNI

CREATE TABLE (cont.)

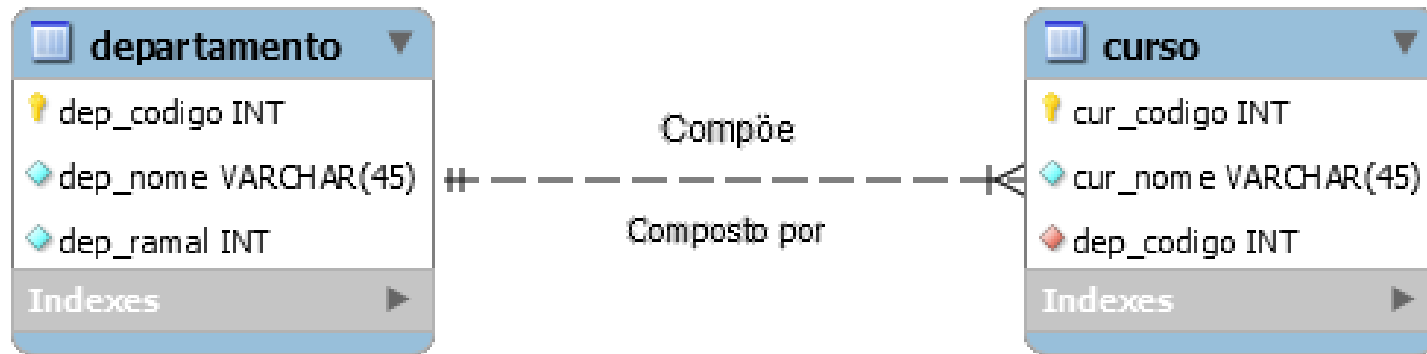
➤ Exemplo de **restrição de integridade referencial** (cont.)



```
1 • USE bd1;
2
3 • CREATE TABLE curso (
4     cur_codigo INT NOT NULL,
5     cur_nome VARCHAR(45) NOT NULL,
6     dep_codigo INT NOT NULL,
7     PRIMARY KEY (cur_codigo),
8     FOREIGN KEY (dep_codigo) REFERENCES departamento (dep_codigo)
9 );
10
11 -- Alternativamente...
12 • CREATE TABLE IF NOT EXISTS bd1.curso (
13     cur_codigo INT NOT NULL,
14     cur_nome VARCHAR(45) NOT NULL,
15     dep_codigo INT NOT NULL,
16     PRIMARY KEY (cur_codigo),
17     CONSTRAINT fk_curso_departamento
18     FOREIGN KEY (dep_codigo)
19     REFERENCES bd1.departamento (dep_codigo)
20 );
```

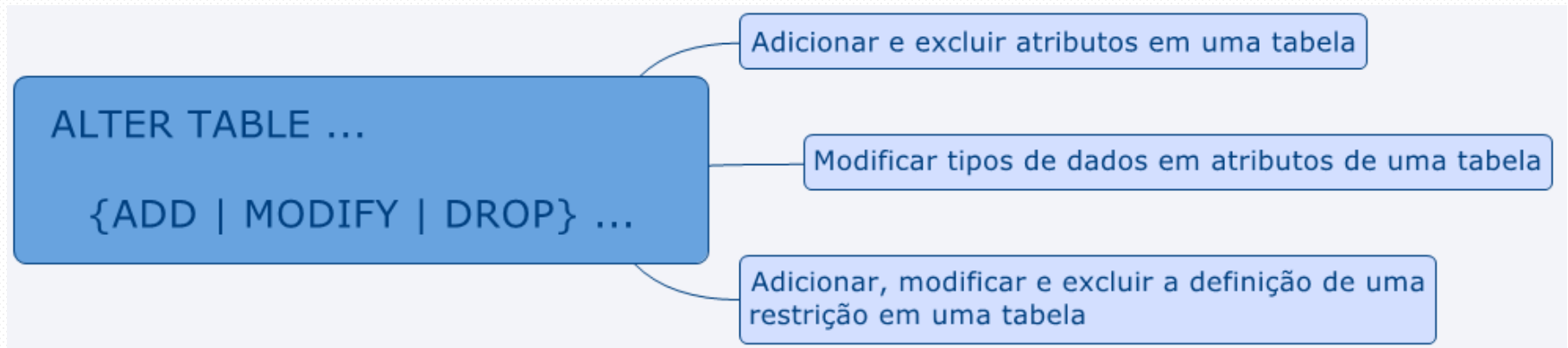
CREATE TABLE (cont.)

→ Exemplo de **restrição de integridade referencial** (cont.)



bd1.curso x							
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants	DDL
Name	Schema	Table	Column	Referenced Schema	Referenced Table	Referenced Column	
fk_curso_departamento	bd1	curso	dep_codigo	bd1	departamento	dep_codigo	

ALTER TABLE



ALTER TABLE (cont.)

➔ Adicionando um atributo – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
    ADD atributo tipo;
```

Table: **curso**

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
dep_codigo	int(11)

```
1  -- Adicionar atributo 'email' à tabela 'curso'  
2  • ALTER TABLE curso  
3      ADD cur_email VARCHAR(50);  
4  
5  -- Alternativamente...  
6  • ALTER TABLE curso  
7      ADD COLUMN cur_email VARCHAR(50)  
8      AFTER cur_nome;
```

Table: **curso**

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
cur_email	varchar(50)
dep_codigo	int(11)

ALTER TABLE (cont.)

→ Adicionando múltiplos atributos – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
    ADD (atributo1 tipo,  
        atributon tipo);
```

Table: departamento

Columns:

<u>dep_codigo</u>	int(11) PK
dep_nome	varchar(45)
dep_ramal	int(11)

```
1 -- Adicionar atributos 'email' e 'fax'  
2 -- à tabela departamento  
3 • ALTER TABLE departamento  
4   ADD (dep_email VARCHAR(50),  
5       dep_fax VARCHAR(50));
```

Table: departamento

Columns:

<u>dep_codigo</u>	int(11) PK
dep_nome	varchar(45)
<u>dep_ramal</u>	int(11)
dep_email	varchar(50)
dep_fax	varchar(50)

ALTER TABLE (cont.)

- Modificando o domínio de **um atributo** – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
MODIFY atributo tipo;
```

Table: **curso**

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
cur_email	varchar(50)
dep_codigo	int(11)

```
1 -- Modificar o tipo de dado do atributo  
2 -- 'email' da tabela curso  
3 • ALTER TABLE curso  
4   MODIFY cur_email CHAR(30);
```

Table: **curso**

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
cur_email	char(30)
dep_codigo	int(11)

ALTER TABLE (cont.)

→ Modificando o domínio de **múltiplos atributos** – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
  MODIFY (atributo1 tipo,  
           atributon tipo);
```

Table: **departamento**

Columns:

<u>dep_codigo</u>	int(11) PK
dep_nome	varchar(45)
dep_ramal	int(11)
dep_email	varchar(50)
dep_fax	varchar(50)

→ Sintaxe MySQL **difere** da sintaxe SQL

```
1  -- Modificar o tipo de dado de múltiplos  
2  -- atributos ('email' e 'fax') da tabela 'departamento'  
3  • ALTER TABLE departamento  
4    MODIFY dep_email CHAR(70),  
5    MODIFY dep_fax INT;
```

Table: **departamento**

Columns:

<u>dep_codigo</u>	int(11) PK
dep_nome	varchar(45)
dep_ramal	int(11)
dep_email	char(70)
dep_fax	int(11)

ALTER TABLE (cont.)

→ Modificar restrição de vazio a **um atributo** – Sintaxe SQL

ALTER TABLE nome_da_tabela
MODIFY atributo tipo **NOT NULL;**

1 • DESCRIBE departamento;

<

Result Grid | Filter Rows:

	Field	Type	Null	Key
	dep codiao	int(11)	NO	PRI
	dep nome	varchar(45)	NO	
	dep ramal	int(11)	NO	UNI
	dep email	char(70)	YES	
	dep fax	int(11)	YES	

```
1 -- Modificar restrição de vazio do atributo FAX,  
2 -- de forma que ele não seja vazio  
3 • ALTER TABLE departamento  
4     MODIFY dep_fax INT NOT NULL;
```

1 • DESCRIBE departamento;

<

Result Grid | Filter Rows:

	Field	Type	Null	Key
	dep codiao	int(11)	NO	PRI
	dep nome	varchar(45)	NO	
	dep ramal	int(11)	NO	UNI
	dep email	char(70)	YES	
	dep fax	int(11)	NO	

ALTER TABLE (cont.)

→ Excluindo um atributo – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
DROP atributo;
```

Table: curso

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
cur_email	char(30)
dep_codigo	int(11)

```
1 -- Excluir o atributo 'email' da tabela curso  
2 • ALTER TABLE curso DROP cur_email;
```

Table: curso

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
dep_codigo	int(11)

ALTER TABLE (cont.)

→ Excluindo múltiplos atributos – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
DROP (atributo1, atributo2);
```

Table: curso

Columns:

<u>cur_codigo</u>	int(11) PK
cur_nome	varchar(45)
cur_email	char(30)
dep_codigo	int(11)

→ Sintaxe MySQL difere da sintaxe SQL

```
1 -- Excluir múltiplos atributos (nome e email)  
2 -- da tabela curso  
3 • ALTER TABLE curso  
4     DROP cur_nome,  
5     DROP cur_email;
```

Table: curso

Columns:

<u>cur_codigo</u>	int(11) PK
dep_codigo	int(11)

ALTER TABLE (cont.)

→ Adicionar restrição UNIQUE a **um atributo** – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
ADD CONSTRAINT nome_restricao UNIQUE (atributo);
```

1 • DESCRIBE departamento;

Result Grid | Filter Rows:

Field	Type	Null	Key
dep_codigo	int(11)	NO	PRI
dep_nome	varchar(45)	NO	
dep_ramal	int(11)	NO	UNI
dep_email	char(70)	YES	
dep_fax	int(11)	NO	

```
1 -- Adicionar restrição de unicidade ao  
2 -- atributo fax da tabela departamento  
3 • ALTER TABLE departamento  
4 ADD CONSTRAINT uq_dep_fax UNIQUE(dep_fax);
```

Column Name	Datatype	PK	NN	UQ
dep_codigo	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dep_nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dep_ramal	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
dep_email	CHAR(70)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
dep_fax	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

ALTER TABLE (cont.)

- Remover restrição UNIQUE de **um atributo** – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
DROP CONSTRAINT nome_restricao;
```

Column Name	Datatype	PK	NN	UQ
dep_codigo	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dep_nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dep_ramal	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
dep_email	CHAR(70)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
dep_fax	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

```
1 -- Remover restrição de unicidade do  
2 -- atributo fax da tabela departamento  
3 • ALTER TABLE departamento  
4   DROP INDEX uq_dep_fax;
```

- Sintaxe MySQL **difere da sintaxe SQL**

Column Name	Datatype	PK	NN	UQ
dep_codigo	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dep_nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dep_ramal	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
dep_email	CHAR(70)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
dep_fax	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

ALTER TABLE (cont.)

→ Adicionar restrição PRIMARY KEY – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
ADD PRIMARY KEY (atributo1, atributon);
```

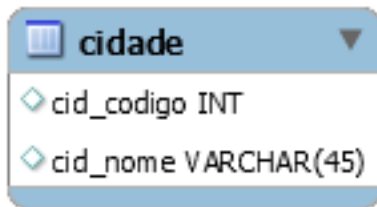


Diagram of the 'cidade' table structure:

- cid_codigo INT
- cid_nome VARCHAR(45)

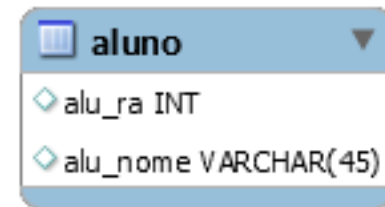


Diagram of the 'aluno' table structure:

- alu_ra INT
- alu_nome VARCHAR(45)

```
1 CREATE TABLE IF NOT EXISTS bd1.cidade (  
2   cid_codigo INT NULL,  
3   cid_nome VARCHAR(45) NULL  
4 );  
5  
6 ALTER TABLE cidade  
7   ADD PRIMARY KEY (cid_codigo);  
8  
9 DESCRIBE cidade;
```

Field	Type	Null	Key	Default	Extra
cid_codigo	int(11)	NO	PRI	NULL	
cid_nome	varchar(45)	YES		NULL	

```
1 CREATE TABLE IF NOT EXISTS bd1.aluno (  
2   alu_ra INT NULL,  
3   alu_nome VARCHAR(45) NULL  
4 );  
5  
6 ALTER TABLE aluno  
7   ADD PRIMARY KEY (alu_ra);  
8  
9 DESCRIBE aluno;
```

Field	Type	Null	Key	Default	Extra
alu_ra	int(11)	NO	PRI	NULL	
alu_nome	varchar(45)	YES		NULL	

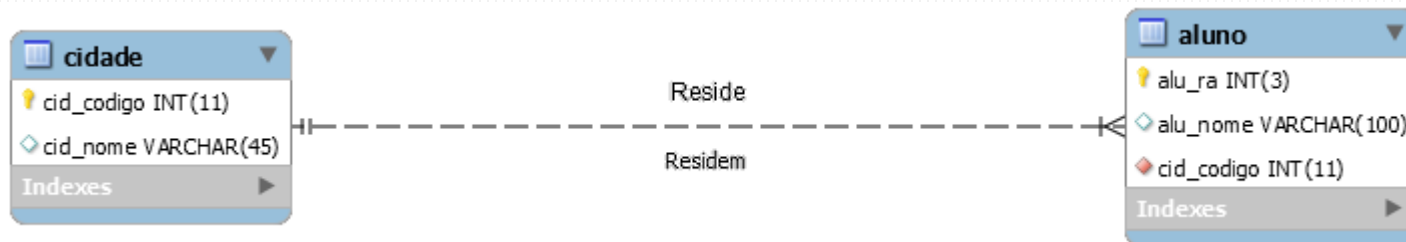
ALTER TABLE (cont.)

➔ Adicionar restrição FOREIGN KEY – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
ADD CONSTRAINT fk_to_td FOREIGN KEY (atributo1 [, atributon])  
REFERENCES tabela_referenciada (atributo_tabela_referenciada);
```

```
1  -- Estabelecer um relacionamento entre ALUNO e CIDADE  
2  -- ALUNO referencia CIDADE por meio de CID_CODIGO  
3  
4  -- Primeiro passo é adicionar em ALUNO o atributo comum  
5  • ALTER TABLE aluno  
6    ADD cid_codigo INT;  
7  
8  -- Após, adicionar a restrição...  
9  • ALTER TABLE aluno  
10 ADD CONSTRAINT fk_aluno_cidade FOREIGN KEY (cid_codigo) REFERENCES cidade (cid_codigo);
```

bd1.aluno x							
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants	DDL
Name	Schema	Table	Column	Referenced Schema	Referenced Table	Referenced Column	
fk_aluno_cidade	bd1	aluno	cid_codigo	bd1	cidade	cid_codigo	



ALTER TABLE (cont.)

→ Remover restrição FOREIGN KEY – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
DROP FOREIGN KEY nome_da_FK;
```

bd1.aluno x							
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants	DDL
Name	Schema	Table	Column	Referenced Schema	Referenced Table	Referenced Column	
fk_aluno_cidade	bd1	aluno	cid_codigo	bd1	cidade	cid_codigo	

```
1 -- Remover relacionamento entre ALUNO e CIDADE, adicionar a restrição...  
2 • ALTER TABLE aluno  
3     DROP FOREIGN KEY fk_aluno_cidade ;
```

bd1.aluno x Query 1							
Info	Columns	Indexes	Triggers	Foreign keys	Partitions	Grants	
Name	Schema	Table	Column	Referenced Schema	Referenced Table	Referenced Column	

ALTER TABLE (cont.)

➤ Remover restrição PRIMARY KEY – Sintaxe SQL

```
ALTER TABLE nome_da_tabela  
DROP PRIMARY KEY;
```

aluno - Table x

Table Name: aluno

Column Name	Datatype	PK	NN	UQ
alu_ra	INT(3)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
alu_nome	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
cid_codigo	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

```
1 -- Remover PK de ALUNO  
2 ALTER TABLE aluno  
3 DROP PRIMARY KEY;
```

aluno - Table x

Table Name: aluno

Column Name	Datatype	PK	NN	UQ
alu_ra	INT(3)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
alu_nome	VARCHAR(100)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
cid_codigo	INT(11)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

DROP TABLE


→ Excluindo uma tabela – Sintaxe SQL

```
DROP TABLE nome_da_tabela;
```

→ Exemplo

```
1  -- Excluir a tabela ALUNO
2  ● DROP TABLE aluno;
3
4  ● SHOW TABLES;
```

<

Result Grid |  Filter Rows:

	Tables_in_bd1
	cidade
	curso
	departamento

DROP DATABASE

→ Sintaxe SQL

```
DROP DATABASE nome_do_banco_de_dados;
```

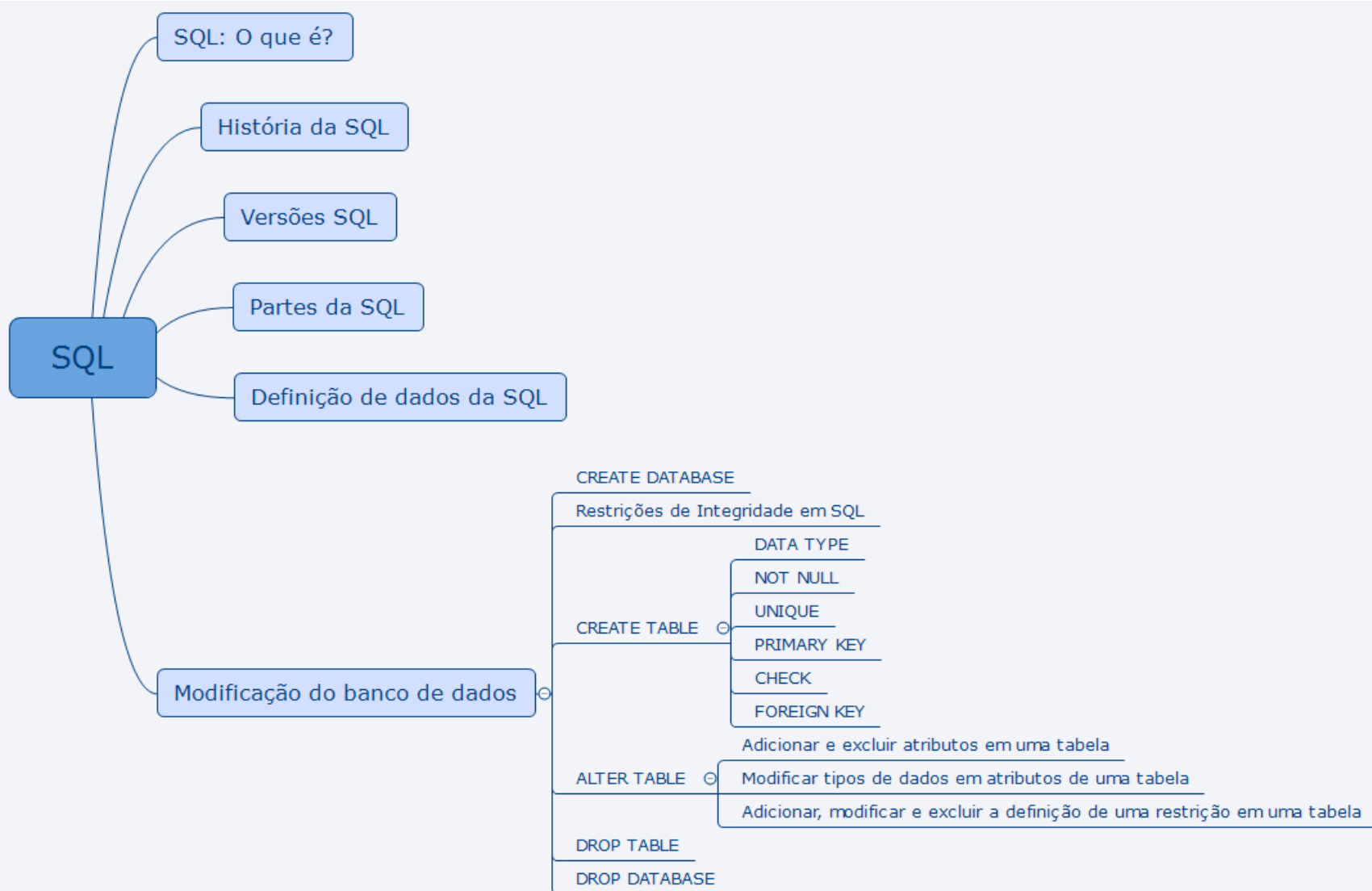
→ Exemplo

```
1  -- Excluir a tabela ALUNO
2  DROP DATABASE bd1;
3
4  SHOW DATABASES;
```

< Result Grid | Filter Rows: | Exp

Database
information schema
efmr01
efmr02
mysql
performance schema
sakila
svs
world

Resumo da Aula





DISCIPLINA: Banco de Dados 1

Prof. **GIOVANI** Volnei Meinerz

Aula 13 – SQL