INSTITUTO FEDERAL
Rio Grande do Norte | Campus Parnamirim

Modelo Físico
Linguagem SQL – Parte 01

Prof. Fábio Procópio

Tecnologia em Sistemas para Internet – Banco de Dados

Relembrando...

- Na aula passada, nós vimos que a **Álgebra Relacional** é uma coleção de operadores que utilizam várias relações para produzir uma nova relação;

- Vimos também os operadores:
  1. União
  2. Intersecção
  3. Diferença
  4. Produto cartesiano
  5. Seleção e Projeção
  6. Junção
  7. Divisão
  8. Renomeação e Atribuição

Introdução

- A SQL (*Structure Query Language*) foi projetada pela IBM, em 1970;
- É uma linguagem padrão para lidar com bancos de dados relacionais e é interpretada por quase todos os produtos existentes no mercado;
- Versões da SQL:
  - SQL-86, SQL-89, SQL-92, SQL-1999, SQL-2003, SQL-2006, SQL-2008, SQL-2011, SQL-2016;
- Embora seja padrão, SQL não é idêntica em todos os SGBDs:
  - O SGBD Oracle 12c (PL-SQL), por exemplo, implementa funções diferentes do SGBD Microsoft SQL Server (T-SQL);
- O uso de instruções fora dos padrões da SQL podem não ser interpretadas por todos SGBDs.

Abrindo parênteses (...

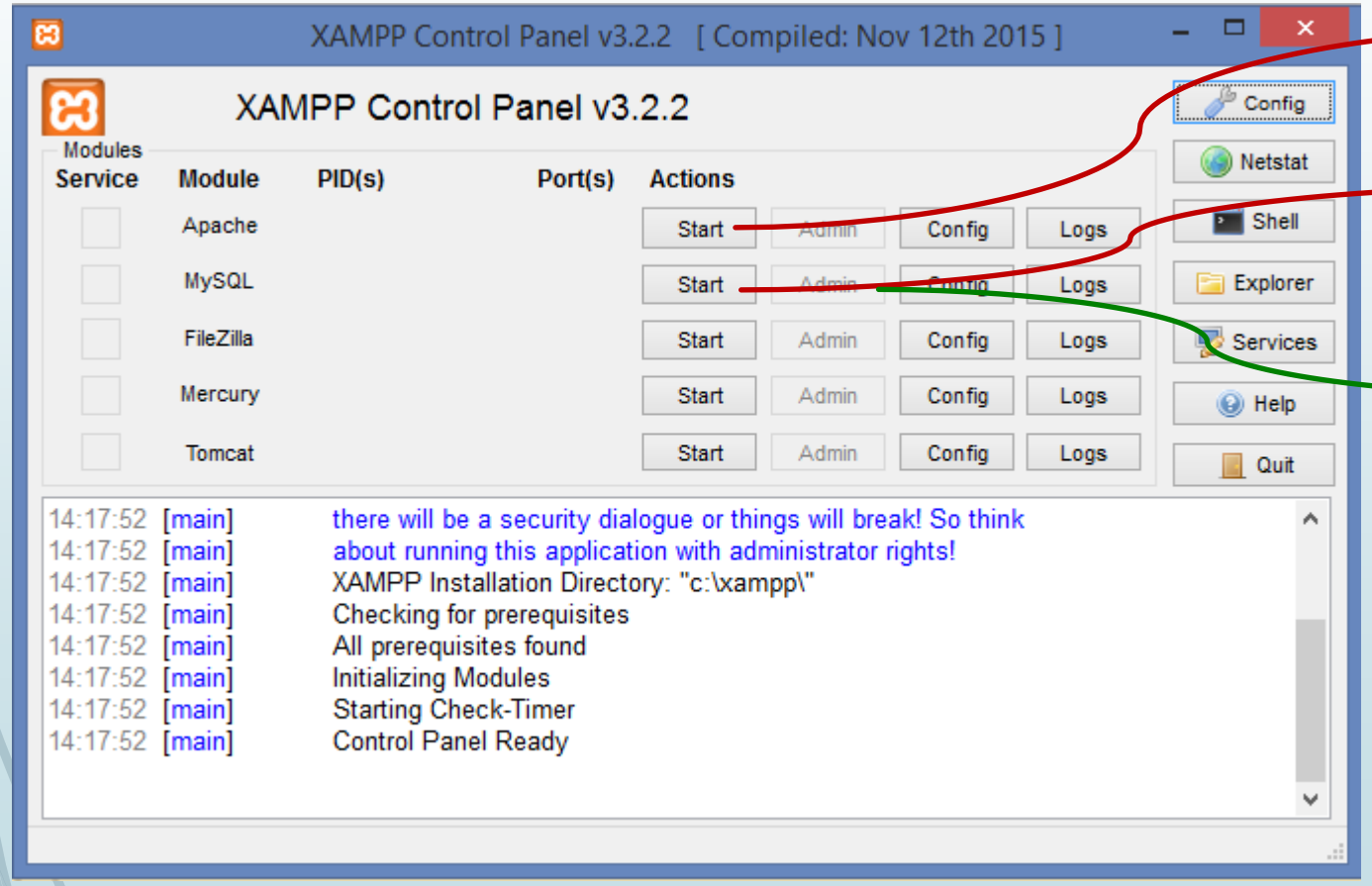
- Antes de começarmos as práticas com a linguagem SQL, nós precisamos preparar e configurar um ambiente de trabalho. Para isto, é necessário:
  1) Um servidor de banco de dados
  2) Um servidor web

- A fim de facilitar a preparação do ambiente, usaremos o XAMPP que oferece um:
  1) Servidor independente da plataforma seja ela Windows, Linux ou OS X
  2) Serviço de gerenciamento de banco de dados para o MariaDB
  3) Servidor web chamado Apache
  4) Interpretador de scripts PHP e Perl

- Para baixar o XAMPP, acesse https://www.apachefriends.org/pt_br/index.html
- Caso tenha dúvidas de como instalá-lo, acesse https://youtu.be/h6DEDm7C37A

Painel de controle do XAMPP

- Acesse o painel de controle do XAMPP e inicialize os serviços Apache e MySQL:

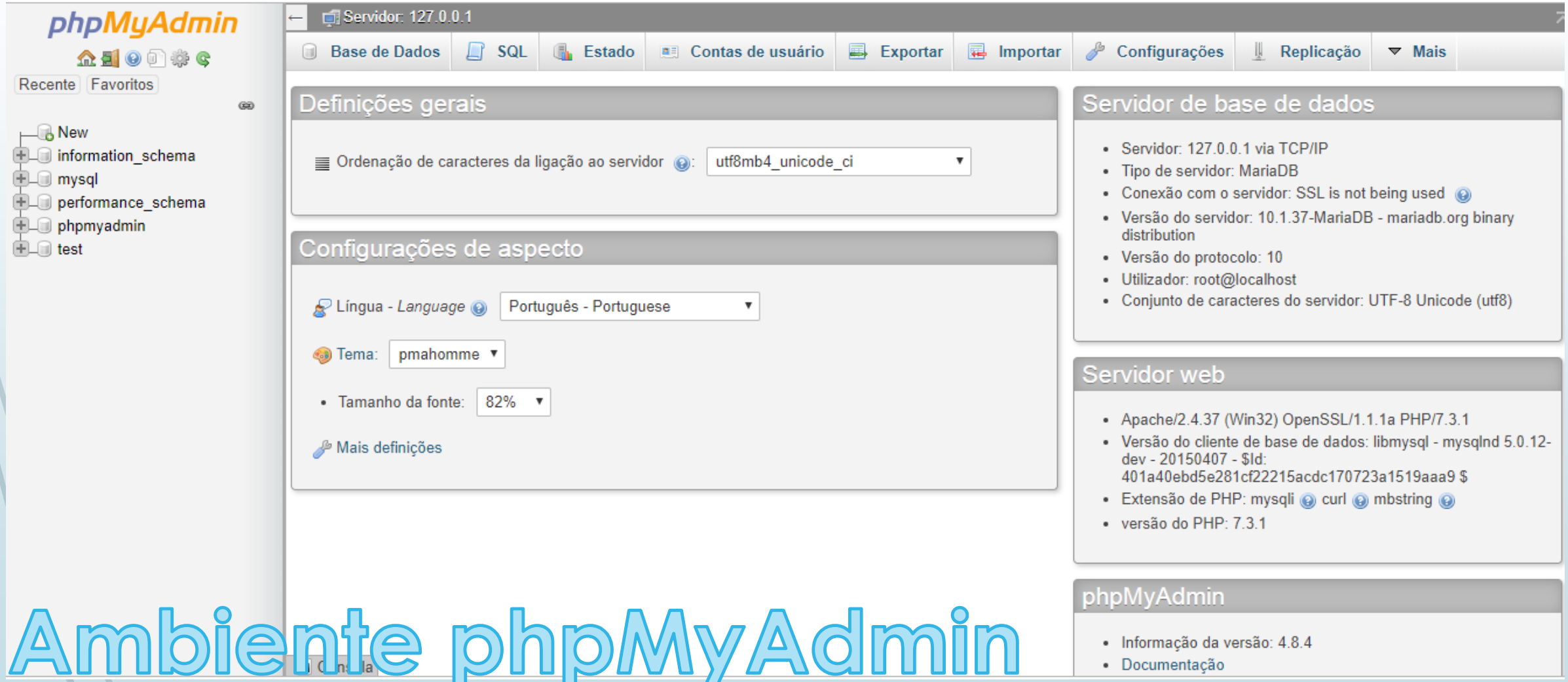


Clique neste botão para iniciar o servidor web Apache.

Clique neste botão para iniciar o servidor de banco de dados MySQL.

Após iniciar os 2 servidores, clique no botão **Admin** (do MySQL) para abrir o phpMyAdmin.

Fechando parênteses)...



Ambiente phpMyAdmin

Os subgrupos da SQL – 1 de 2

- **DDL – *Data Definition Language***
  - Define, modifica e exclui esquemas de relações do banco de dados
- **DML – *Data Manipulation Language***
  - Linguagem de atualização de dados, baseada na álgebra e no cálculo relacional
- **DQL – *Data Query Language***
  - Usada para consultar dados
- **DCL – *Data Control Language***
  - Trata dos aspectos relacionados à segurança, com instruções para controlar a autorização e direitos de acesso ao BD
- **DTL – *Data Transaction Language***
  - Usada para gerenciar as diferentes transações de um BD.

Os subgrupos da SQL – 2 de 2

- **DDL – *Data Definition Language***
  - Comandos: CREATE, ALTER e DROP
- **DML – *Data Manipulation Language***
  - Comandos: INSERT, UPDATE e DELETE
- **DQL – *Data Query Language***
  - Comando: SELECT
- **DCL – *Data Control Language***
  - Comandos: GRANT e REVOKE
- **DTL – *Data Transaction Language***
  - Comandos: COMMIT, ROLLBACK e SAVEPOINT

Comandos DDL

- **DDL – *Data Definition Language***
  - Comandos: CREATE, ALTER e DROP
- **DML – *Data Manipulation Language***
  - Comandos: INSERT, UPDATE e DELETE
- **DQL – *Data Query Language***
  - Comando: SELECT
- **DCL – *Data Control Language***
  - Comandos: GRANT e REVOKE
- **DTL – *Data Transaction Language***
  - Comandos: COMMIT, ROLLBACK e SAVEPOINT

CREATE/DROP DATABASE – 1 de 2

- Criação simplificada de um banco de dados:

```
CREATE DATABASE nome_bd;
```

- Destruição de um banco de dados:

```
DROP DATABASE nome_bd;
```

CREATE/DROP DATABASE – 2 de 2

- Criação simplificada do banco de dados dbTeste:

```
CREATE DATABASE dbTeste
```

- Destruição do banco de dados dbTeste:

```
DROP DATABASE dbTeste
```

Para escrever os comandos SQL usados nas aulas, use o **phpMyAdmin**.

Para executar os comandos SQL, pressione o botão **Executar**.

- Para acompanhar a sequência da aula, vamos criar o banco de dados dbAulaSQL:

```
CREATE DATABASE dbAulaSQL
```

CREATE TABLE – 1de 2

- Criação simplificada de tabelas em um banco de dados:

```
CREATE TABLE nome_tabela
(
  nome_campo_1  tipo_1,
  nome_campo_2  tipo_2,
  ...
  nome_campo_N  tipo_N,
  PRIMARY KEY (nome_campo_x, ...)
);
```

- Para determinar que um campo deve ter preenchimento obrigatório, usa-se **NOT NULL**;
- Para definir que um campo deve ter auto-incremento, usa-se **AUTO_INCREMENT**;
- Para definir um valor padrão para um campo, usa-se **DEFAULT** seguido do respectivo valor.

CREATE TABLE – 2 de 2

- Para acessar um determinado banco de dados, usa-se **USE** nome_bd;

- Criação da tabela TbProduto:

```
USE dbAulaSQL;

CREATE TABLE TbProduto
(
  CoProduto  INT AUTO_INCREMENT,
  NoProduto VARCHAR(50) NOT NULL,
  QtEstoque TINYINT DEFAULT 0,
  PRIMARY KEY (CoProduto)
);
```

Palavra-chave usada para definir campos auto-incremento .

ALTER TABLE – 1de 4

- Depois que uma tabela é criada, é possível alterar a sua estrutura. As modificações possíveis são:
  - Adicionar uma nova coluna
  - Remover uma coluna
  - Alterar o nome de uma coluna
  - Alterar o tipo de dados de uma coluna

- Para alterar a estrutura de uma tabela, usa-se:

```
ALTER TABLE nome_tabela <modificação>;
```

ALTER TABLE – 2 de 3

- Para **adicionar** uma coluna a uma tabela existente, usa-se:

```
ALTER TABLE nome_tabela ADD <nome_coluna> <tipo_coluna>;
```

- Para **alterar o nome** de uma coluna, usa-se:

```
ALTER TABLE nome_tabela CHANGE <nome_antigo> <nome_novo> <tipo_coluna>;
```

- Para **alterar o tipo de dado** de uma coluna, usa-se:

```
ALTER TABLE nome_tabela CHANGE <nome_coluna> < nome_coluna> <novo_tipo>;
```

- Para **remover** uma coluna, usa-se:

```
ALTER TABLE nome_tabela DROP <nome_coluna>;
```

ALTER TABLE – 3 de 3

- Adicionando a coluna Cadastro na tabela TbProduto:

```
ALTER TABLE TbProduto ADD Cadastro DATE;
```

- Alterando o nome da coluna Cadastro para DaCadastro:

```
ALTER TABLE TbProduto CHANGE Cadastro DaCadastro DATE;
```

- Alterado o tipo de dados da coluna QtEstoque:

```
ALTER TABLE TbProduto CHANGE QtEstoque QtEstoque INT;
```

- Para **remover** uma coluna, usa-se:

```
ALTER TABLE TbProduto DROP DaCadastro;
```

DROP TABLE

- Para destruir uma tabela, usa-se o seguinte comando:

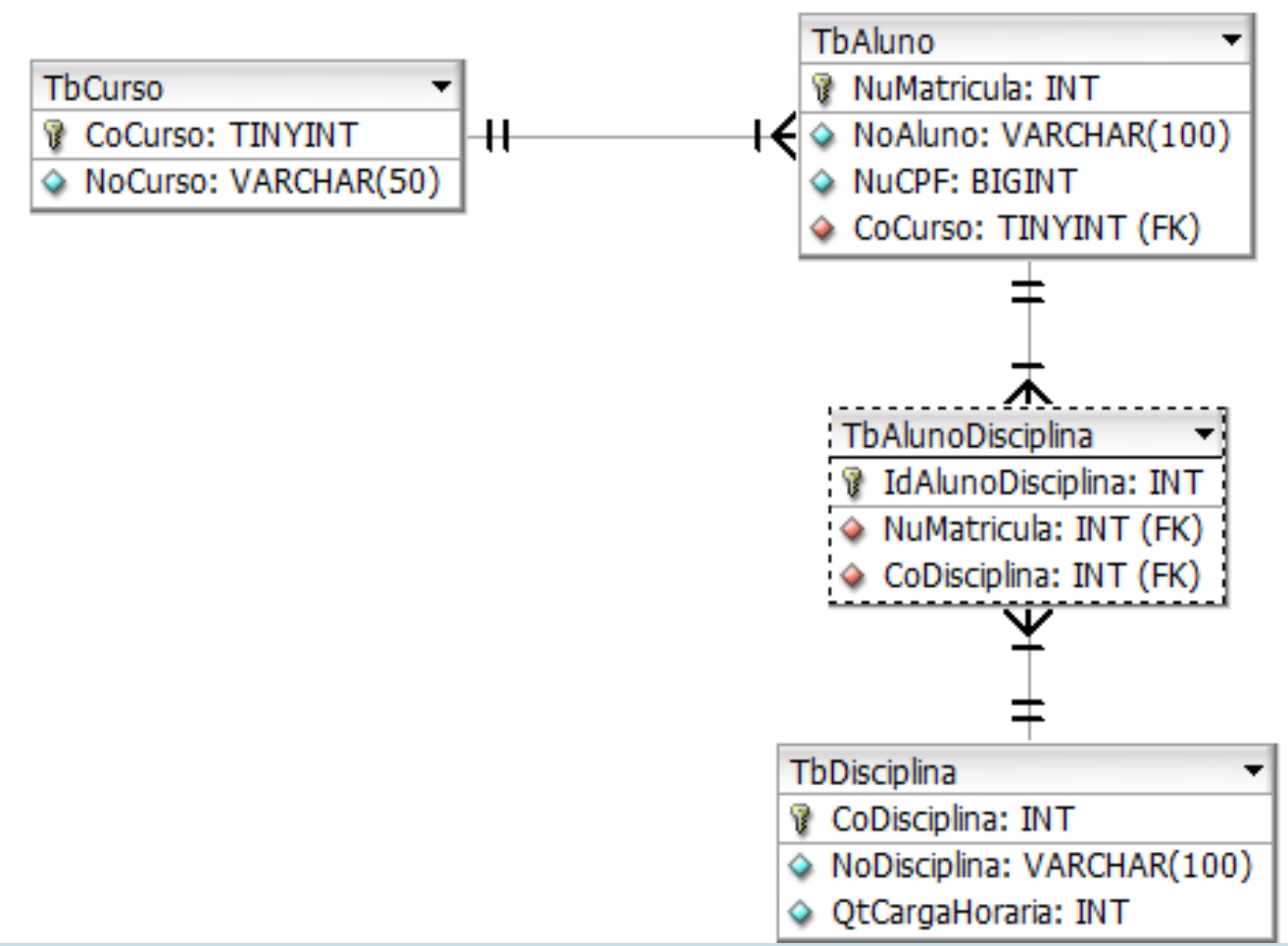
```
DROP TABLE nome_tabela;
```

- Destruindo a tabela TbProduto:

```
DROP TABLE TbProduto;
```

Exercício de Fixação – 1 de 3

- Baseado no modelo lógico abaixo, crie o respectivo modelo físico em **dbAulaSQL**:



Exercício de Fixação – 2 de 3

```
CREATE TABLE TbCurso
(
  CoCurso  TINYINT AUTO_INCREMENT PRIMARY KEY,
  NoCurso VARCHAR(50) NOT NULL
);
```

```
CREATE TABLE TbAluno
(
  NuMatricula  INT AUTO_INCREMENT PRIMARY KEY,
  NoAluno VARCHAR(100) NOT NULL,
  NuCPF BIGINT UNIQUE,
  CoCurso TINYINT NOT NULL,
  CONSTRAINT FK_CursoAluno FOREIGN KEY (CoCurso)
  REFERENCES TbCurso (CoCurso)
);
```

Determina que o valor do campo é único. Pode ser usado para definir chaves candidatas.

Exercício de Fixação – 3 de 3

```
CREATE TABLE TbDisciplina
(
  CoDisciplina  INT AUTO_INCREMENT PRIMARY KEY,
  NoDisciplina VARCHAR(100) NOT NULL,
  QtCargaHoraria INT NOT NULL
);
```

```
CREATE TABLE TbAlunoDisciplina
(
  IdAlunoDisciplina INT AUTO_INCREMENT PRIMARY KEY,
  NuMatricula  INT NOT NULL,
  CoDisciplina INT NOT NULL,
  CONSTRAINT FK_Aluno FOREIGN KEY (NuMatricula)
  REFERENCES TbAluno (NuMatricula),
  CONSTRAINT FK_Disciplina FOREIGN KEY (CoDisciplina)
  REFERENCES TbDisciplina (CoDisciplina)
);
```

Principais Referências

1) 1KEYDATA. **TUTORIAL DE SQL**. Disponível em: https://www.1keydata.com/pt/sql/. Acessado em: 12 mai 2018.

2) TechOnTheNet. ***SQL Server: ALTER TABLE Statement***. Disponível em: https://www.techonthenet.com/sql_server/tables/alter_table.php. Acessado em: 12 mai 2018.

3) WIKIVERSIDADE. **Introdução ao SQL/Criando Tabelas**. Disponível em: https://pt.wikiversity.org/wiki/Introdu%C3%A7%C3%A3o_ao_SQL/Criando_Tabelas. Acessado em: 12 mai 2018.