

# Linguagem SQL

O que é SQL?

## SQL

- SQL (Structured Query Language)
- Linguagem de Consulta Estruturada.
- Conjunto de comandos de manipulação de banco de dados, além de incluir, excluir, modificar e pesquisar informações na tabela.
- Não é uma linguagem de programação.

## SQL

- Criada pelo Departamento de Pesquisa da IBM no início dos anos 70.
- Em 1986 o American National Standard Institute (ANSI) publicou o SQL e estabeleceu como linguagem padrão de Banco de Dados Relacional.

## SQL

- Devido ao sucesso desta nova forma de consulta e manipulação de dados, dentro de um ambiente de banco de dados, a utilização da SQL foi se tornando cada vez maior.
- Com isso uma grande quantidade de SGBD's (Sistemas Gerenciadores de Banco de Dados), como o SQL-Server da Microsoft, o ORACLE, SyBase e muitos outros, foi tendo a SQL como sua linguagem básica.
- A SQL se tornou um padrão de fato, no mundo dos ambientes de banco de dados relacionados.
- Uma das razões da popularidade dos sistemas relacionais é a sua facilidade de manipulação e entendimento.

A linguagem SQL é dividida nos seguintes componentes:

- DDL (Data Definition Language): permite a criação dos componentes do banco de dados, como tabelas, índices etc.
- Principais comandos DDL:  
CREATE TABLE  
CREATE DATABASE  
ALTER TABLE  
DROP TABLE  
CREATE INDEX  
ALTER INDEX  
DROP INDEX

## Componentes SQL

- DML (Data Manipulation Language): permite a manipulação dos dados armazenados no banco de dados.
- Comandos DML:  
INSERT  
DELETE  
UPDATE

## Componentes SQL

- DQL (permite extrair dados do banco de dados.
- Comando:

SELECT

## Componentes SQL

- DCL (Data Control Language): provê a segurança interna do banco de dados.

- Comandos DCL:

CREATE USER

ALTER USER

GRANT (autoriza ao usuário executar ou setar operações)

REVOKE (remove ou restringe a capacidade de um usuário de executar operações)

CREATE SCHEMA



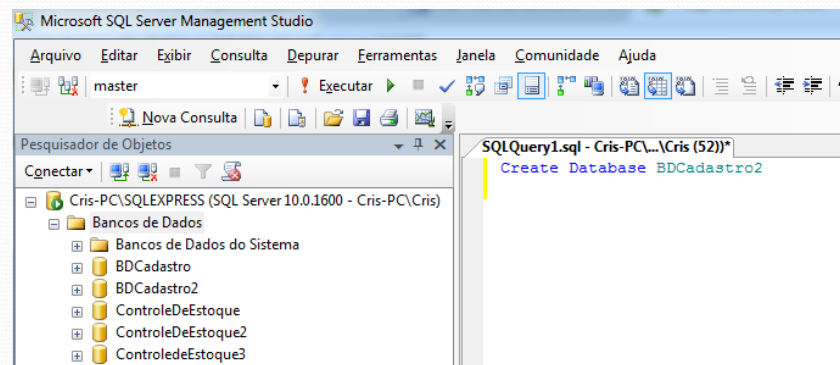
# Instruções DDL (Data Definition Language)

Linguagem de Definição de Dados, define a estrutura dos dados e tabelas

## CREATE DATABASE

- Criando um banco de dados
- Instruções:
  - CREATE DATABASE NomeDoBanco;
- EXEMPLO:
  - Criar uma base de dados chamada Empresa:
  - Create Database Empresa;

## Criar Banco de Dados

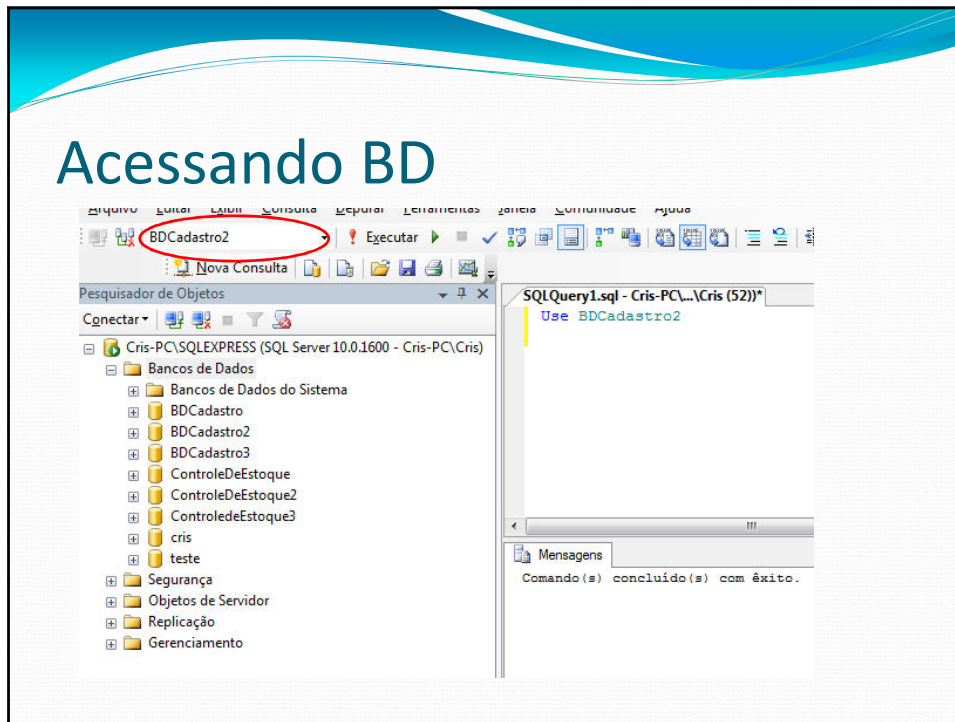


Pressione F5 ou clique em Executar!, para executar os comandos digitados

## Acesso a base de Dados

- Use <nome do banco>
- Exemplos:
  - Use Empresa;
  - Use BDCadastro2;

## Acessando BD



## Excluir um banco de dados

- Exemplo:
- Sintaxe:
- DROP DATABASE <nome do banco>;
- Supondo que você queira excluir o Banco de Dados Empresa
- DROP DATABASE EMPRESA;
- Atenção: O comando DROP DATABASE não pode ser desfeito!!!

## CREATE TABLE

```
CREATE TABLE <nome_tabela>
(<descrição das colunas>,
<descrição das chaves>);
```

onde:

**<nome\_tabela>** deve ser substituído pelo nome da tabela a ser criada.

**<descrição das colunas>** deve ser substituída pela relação das colunas da tabela e seus respectivos tipos de dados (por exemplo, smallint, char, varchar, integer, number, float e etc).

**<descrição das chaves>** deve ser substituída pela lista das colunas que são tratadas como chaves estrangeiras.

## Comando – Create table



SQLQuery1.sql - Cris-PC\...\Cris (52))\*

```
Create table Produtos(
cod_produto int primary key not null,
nome varchar (30),
descricao text,
quantidade int,
data_fabricacao date,
valorcompra money
)
```



## Checando a criação da tabela

- Select \* from <nome da tabela>
- Select \* from Produtos

## Tipos de dados no SQL Server

- **Binary** - Contém até 8.000 bytes de dados binários de tamanho fixo. Tamanho fixo significa que o campo irá sempre utilizar o espaço que você definiu.
- **Bit** - Cria um campo true/false (verdadeiro/falso). Este tipo de dado armazena apenas os valores de 0 e 1 e não pode conter valores nulos.
- **Char** - Contém até 8.000 bytes de uma string de tamanho fixo. Use este campo para dados que irão sempre ser do mesmo tamanho.
- **Datetime** - Armazena a data e hora com precisão de milissegundos em 8 bytes.
- **Decimal** - Contém dados numéricos com um inteiro à esquerda do ponto decimal e uma parte fracionária a direita. Este campo irá ter um número variável de bytes, baseado na escala e precisão que você especificar. Escala é o número máximo de dígitos à direita do ponto decimal e precisão é o número total de dígitos em ambos os lados do ponto decimal.

## Tipos de dados no SQL Server

- **Float** - Um tipo de dados numérico aproximado para armazenamento de longos números em ponto-flutuante, com a precisão de até 15 dígitos em 8 bytes. O SQL Server geralmente arredonda esses números para cima.
- **Image** - Contém até 2.147.483.647 bytes (aproximadamente 2Gbytes) de dados binários e costuma ser utilizado para armazenar grandes quantidades de dados como imagens ou arquivos de som. Campos definidos com este tipo de dado não podem ser indexadas, pesquisadas, agrupadas ou ordenadas.
- **Int** - Armazena um dado numérico preciso para todo número armazenado em 4 bytes. Pode conter números de  $-2^{1031}$  a  $2^{1031}$ .
- **Money** - Contém dados monetários definidos com precisão de 4 dígitos. Ele é representado como um inteiro de precisão dupla e consome 8 bytes de armazenamento. Valores podem estar entre -992.337.293.685.477,5808 e 922.337.203.685.447,5807.

## Tipos de dados no SQL Server

- **Nchar** - Contém até 4.000 caracteres Unicode. Tipo de dado de largura fixa, sendo armazenado no dobro de bytes (caracteres Unicode precisam de 2 bytes por caractere).
- **Ntext** - Armazena caracteres Unicode até 1.073.741.823 posições, armazenados no dobro dos bytes declarados (caracteres Unicode precisam de 2 bytes por caractere).
- **Numeric** - Veja Decimal.
- **Nvarchar** - Contém até 4.000 caracteres Unicode. Tipo de dado de largura variável, sendo armazenado no dobro de bytes (caracteres Unicode precisam de 2 bytes por caractere).
- **Real** - Similar ao tipo de dado float, usando apenas 4 bytes de armazenamento com precisão de até 7 dígitos.
- **Smalldatetime** - Campo data menos preciso, que armazena data e hora em precisão de minutos, utilizando 4 bytes.

## Tipos de dados no SQL Server

- **Smallint** - Armazena dados numéricos preciso até a quantidade de números armazenados em 2 bytes. Menor que o int, ele pode conter números de -32.768 até 32.767.
- **Smallmoney** - Contém dados monetários decimais precisos até o 4 dígito. Usa 4 bytes de armazenamento e pode conter valores de - 214.748,3648 até 214.748,3647.
- **Text** -Contém uma string de caracteres não-Unicode com tamanho de até 2.147.483.647 caracteres. Similiar ao campo memo encontrado em outros bancos de dados, este tipo de dado armazena grandes quantidades de texto em páginas de 2kb. Colunas definidas com este tipo de dado não podem ser indexadas, pesquisadas, agrupadas ou ordenadas.
- **Timestamp** - Um valor binário automaticamente atualizado cada vez que uma tabela com uma coluna deste tipo é alterada. Ele usa 8 bytes de armazenamento e é único dentro do banco de dados. Uma tabela só pode possui uma coluna deste tipo.

## Tipos de dados no SQL Server

- **Tinyint** - O menor tipo de dado inteiro, ele consome apenas 1 byte de armazenamento e pode conter números de 0 a 255.
- **Uniqueidentifier** -Um número hexadecimal de 16 bytes, também conhecido como GUID (globally unique identification number). Use a função do SQL Server NEWID() para gerar novos identificadores únicos para alimentar uma variável ou uma coluna deste tipo de dado.
- **Varbinary** -Similar ao tipo binário, este pode conter até 8.000 bytes de dados binários de tamanho variável. Tamanho variável significa que o campo irá consumir apenas o espaço necessário para armazenar os dados contidos no mesmo.
- **Varchar** -Contém até 8.000 bytes de tamanho variável para strings. Use este tipo para colunas que irão armazenar valores nulos de dados que variam em tamanho.



## Integridade Referencial - Constraints

- São regras agregadas a colunas ou tabelas.
- Sintaxe básica: Create Table nome-tabela
- Exemplo:

```
CREATE TABLE CD (
Codigo_CD          integer not null,
Codigo_gravadora   integer null,
Nome_CD            varchar (60) not null,
Preco_venda        decimal not null,
Data_lancamento   date default sysdate,
Primary key        (codigo_cd),
Foreign key        (codigo_gravadora),
References Gravadora (codigo_gravadora),
Check (Preco_venda>0)
);
```

## DEFAULT

- Serve para atribuir um conteúdo padrão a uma coluna da tabela.
- Exemplo:

...

Quantidade integer default 1,



## NOT NULL

- Indica que o conteúdo de uma coluna não poderá ser nulo.
- Exemplo:

...

Nome\_cliente varchar(50) not null,

...

## UNIQUE

- Indica que não poderá haver repetição no conteúdo da coluna.
- Exemplo:

...

CPF numeric(11) UNIQUE,

...

## CHECK

- Podemos, ao criar uma coluna, especificar quais os valores que poderão ser utilizados para preencher a coluna.
- Exemplo:

...

```
Sexo char(1) check (upper(sexo) = 'M' or upper(sexo) = 'F'),
```

...

## Exercícios

1 – Criar as seguintes tabelas em SQL:

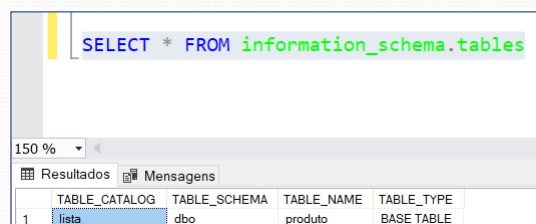
- a) Autor
- b) Professor
- c) Cliente
- d) Fornecedor

## Renomear Tabelas

- Exemplo de utilização:
- `EXEC sp_rename 'NomeTabelaAntigo', 'NomeTabelaNovo';`
- `EXEC sp_rename 'Produtos', 'Tab_Produtos';`

## Mostrar todas as tabelas do banco de dados

- `SELECT * FROM information_schema.tables`



	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE
1	lista	dbo	produto	BASE TABLE

## DROP TABLE

Para excluirmos uma tabela existente devemos usar o comando DROP TABLE. A sua forma geral é:

```
DROP TABLE <nome_tabela>;
```

onde:

<nome\_tabela> dever ser substituído pelo nome da tabela a ser excluída.

### Exemplos

```
drop table item_pedido;  
drop table pedido;  
drop table vendedor;  
drop table produto;  
drop table cliente;
```

## Alteração de estrutura de tabela

- Para alterar a estrutura de uma tabela, utilizamos o comando ALTER TABLE.
- Exemplo: **Acrescentar** novas colunas

```
ALTER TABLE cliente
```

```
ADD email varchar(80) unique
```



## Alteração de estrutura de tabela

- Modificar **tipo** de colunas existente:
- Alter Table produtos
- Alter Column descricao **varchar(80)**

## Alteração de estrutura de tabela

- **Alteração** do nome da coluna:

EXEC sp\_rename 'nome\_da\_tabela.nome\_coluna',  
'novo\_nome\_coluna', 'COLUMN';

Exemplo:

EXEC sp\_rename 'produtos.nome', 'nome\_prod', 'COLUMN'

## Alteração de estrutura de tabela

- Excluindo colunas:
- ALTER TABLE produtos
- Drop COLUMN valor2

## Exercícios

- 1 - Acrescentar na tabela jogadores os campos: filiacao varchar(120), num\_contrato bigint.
- 2 - Acrescentar na tabela times os campos: presidente varchar(80), e-mail varchar(100).
- 3 - Alterar o tamanho do campo nome da tabela times para varchar (150).

## Incluindo, atualizando e excluindo linhas nas tabelas

- Incluindo dados em tabelas:

```
INSERT INTO tabela (coluna, coluna)  
VALUES (conteudo, conteudo)
```

## Exemplos:

- INSERT INTO Autor (codAutor, NmAutor)  
VALUES (1, 'Tom Jobim')

## Exemplo de tabelas com valores decimais

- `create table` produto
- `(codigo int primary key not null,`
- `valor float,`
- `valor1 numeric(10,2),`
- `valor2 real,`
- `valor3 decimal (10,2) )`
  
- `insert into` produto (valor, valor1, valor2, valor3)
- `values`
- `(12.50, 13.50, 14.50, 15.50)`
- OBS: 10,2 para decimal e numeric representam que de 10 dígitos 2 serão casas decimais, caso não seja definido o tamanho por padrão será usado (18,0).
- Logo o número será arredondado. Ex: 13.50 será armazenado 14.

## Atualização de dados na tabela

- Exemplo:

```
UPDATE cd
SET preco_venda = 15
WHERE codigo_cd = 1;
```



## Exemplo 2:

- UPDATE cd  
SET preco\_venda = preco\_venda \* 1.05  
WHERE codigo\_cd = 2;

## Exclusão de dados

- Exemplo  
DELETE FROM Autor  
WHERE codAutor = 1;  
  
DELETE FROM Musica;  
deleta todos os registros de musica, mas não a tabela.

## Exercícios

- 4 – Excluir o jogador de código igual a 1.
- 5 – Excluir o time de código igual a 5.
- 6 – Trocar o nome da tabela Times para Times\_Futebol.