

## \* Structured Query Language (SQL)

↳ Criada pela IBM como interface p/ o sistema de banco de dados relacional System R.

- Além da consulta, SQL tem recursos p/:
  - definição das estruturas de dados;
  - atualização de dados;
  - definição de regras;
  - especificação de segurança e autorização;
  - restrições de integridade;
  - gerenciamento de transações.

## \* Definição de dados

SQL utiliza os termos **tabela**, **linha** e **coluna** p/ definir relações, tuplos e atributos.

- CREATE - comando p/ definição de dados, usado p/ criar: esquemas, tabelas, tipos/domínios, visões, assertões e triggers.

- **Esquema**: Também chamado de banco de dados, trata-se de um conceito que abarca todas as tabelas e objetos que pertencem à mesma aplicação no SGBD.

↳ Identificamos um esquema por seu nome.

↳ Inclui identificação de autorização p/ identificar o usuário ou a conta proprietária do esquema.

↳ Inclui descritores (métodos) p/ cada elemento do esquema (ex: tabelas, domínios, tipos, etc).

→ CREATE SCHEMA - comando p/ criar um esquema.

Exemplo:

```
1 CREATE SCHEMA Biblioteca
2 AUTHORIZATION gabrafo;
```

↳ comandos são case insensitive (aceitam letras maiúsculas e minúsculas) e terminam com ponto e vírgula.

→ DROP SCHEMA - comando p/ excluir um esquema.

↳ pode ser CASCADE (remove tudo associado ao esquema) ou RESTRICT (apenas exclui se não tiver nada associado ao esquema).

Exemplo:

```
1 DROP SCHEMA Biblioteca CASCADE;
```

→ CREATE TABLE - cria uma nova tabela (relação / entidade).

↳ quanto aos atributos, é necessário especificar seus nomes, tipo (ex: INTEGER) e restrições (ex: NOT NULL).

↳ restrições de chave ou integridade referencial podem ser especificadas no comando CREATE TABLE ou ALTER

restrições de chave ou integridade referencial podem ser especificadas na comando CREATE TABLE ou ALTER TABLE.

Exemplos:

atributos

```
CREATE TABLE Funcionario (
  idFunc INT NOT NULL,
  nomeFunc VARCHAR(50) NOT NULL,
  sexo CHAR(1) NOT NULL,
  endereco VARCHAR(100) NOT NULL,
  salario DECIMAL(8,2) NOT NULL,
  idSuperv INT NULL,
  idDeppto INT NOT NULL,
  PRIMARY KEY (idFunc)
);
```

Restrições  
Tipo

Nome

```
CREATE TABLE Departamento (
  idDeppto INT NOT NULL,
  nomeDeppto VARCHAR(30) NOT NULL,
  idGerente INT NOT NULL,
  PRIMARY KEY (idDeppto),
  UNIQUE (nomeDeppto),
  FOREIGN KEY (idGerente)
  REFERENCES Funcionario (idFunc)
);
```

chave primária  
chave estrangeira

releção c/ "funcionario"

## Tipos de dados

### Númericos:

- Inteiro: INTEGER ou INT, SMALLINT
- Real: FLOAT ou REAL, DOUBLE PRECISION
- Formatação: DECIMAL(i, j) ou DEC(i, j) ou NUMERIC(i, j)

i: número total de dígitos

j: número de dígitos após o ponto decimal

445.633

i=6  
j=3

l: número de caracteres

### Códigos de caracteres:

- Tamanho fixo: CHAR(l) ou CHARACTER(l)
- Tamanho variável: VARCHAR(m) ou CHAR VARYING(m) ou CHARACTER VARYING(m)

m: número máximo de caracteres

- Documentos grandes: CHARACTER LARGE OBJECT(m) ou CLOB(m)

m: tamanho em K, M ou G bytes

### Datos:

- Data: DATE(aaaa-mm-dd)
- Hora: TIME(hh:mm:ss)

### Booleano: BOOLEAN: TRUE, FALSE ou NULL

- Sequência de bits
- Tamanho fixo:  $\text{BIT}(l)$   $l$ : número de bits
  - Tamanho variável:  $\text{BIT VARYING}(n)$   
 $n$ : número máximo de bits
  - Objetos grandes (ex: imagens):  $\text{BINARY LARGE OBJECT}(m)$  ou  $\text{BLOB}(m)$   
 $m$ : tamanho em K, M ou G bits

### Domínio:

→ **CREATE DOMAIN** - Cria um domínio que pode ser usado por diversos atributos em um esquema.

Exemplo:

```
CREATE DOMAIN TipoCPF AS CHAR(11);
```

↳ poderá ser usado na criação de uma tabela para especificar o tipo de um atributo "CPF".

→ **DROP DOMAIN** - Exclui um domínio.