SQL

segunda-feira, 13 de janeiro de 2025 14:19

★ Structured Query Language (SQL)
↳ Projetada pela IBM como interface p/ o sistema de banco de dados relacional System R.

• Além de consulta, SQL tem recursos p/:
- definição das estruturas de dados;
- atualização de dados;
- definição de visões;
- 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, asserções 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 (metadados) p/ cada elemento do esquema (ex: tabelas, domínios, tipos, etc).

-> CREATE SCHEMA - comando p/ criar um esquema.
Exemplo:

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

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

-> DROP SCHEMA - comando p/ excluir um esquema.
↳ pode ser CASCADE (remove tudo associado ao esquema) ou RESTRICT (apenas exclui se não existir nada associado ao esquema).

Exemplo:

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

modelo E-R
modelo relacional

-> 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 no comando CREATE TABLE ou ALTER TABLE.

Exemplos:

↳ Nome da tabela

```
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,
 idDepto INT NOT NULL,
 PRIMARY KEY (idFunc)
 );
```

atributos

Restrições: Nome, Tipo

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

chave secundária

chave estrangeira ↳ relação c/ "Funcionario"

* Tipos de dados

↳ Numéricos:
- Inteiro: INTEGER ou INT, SMALLINT
- Real: FLOAT ou REAL, DOUBLE PRECISION
- C/ 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

↳ 45,633 (i=6, j=3)

l: número de caracteres

↳ Cadeias de caracteres:
- Tamanho fixo: CHAR(l) ou CHARACTER(l)
- Tamanho variável: VARCHAR(n) ou CHAR VARYING(n) ou CHARACTER VARYING(n)

n: número máximo de caracteres

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

m: tamanho em K, M ou G bytes

↳ Datas:
- Data: DATE(aaaa-mm-dd) (2025-01-20)
- Hora: TIME(hh:mm:ss) (12:20:00)

↳ Booleano: BOOLEAN: TRUE, FALSE ou NULL

↳ Sequência de bits — l: número de bits
- Tamanho fixo: BIT(l)
- Tamanho variável: BIT VARYING(n)
n: número máximo de bits
- Objetos grandes (ex: imagens): BINARY LARGE OBJECT(m) ou 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 p/ especificar o tipo de um atributo "CPF".

→ DROP DOMAIN — Exclui um domínio.

↳ Especificação de Restrições
- Valor nulo/não nulo: NULL e NOT NULL
↳ permite valor nulo (padrão)
↳ não permite valor nulo

- Valor padrão: DEFAULT
↳ atributo recebe um valor padrão, caso um específico não tenha sido definido.

- Limite de valores: CHECK (condição)
↳ limita os valores dos atributos de acordo c/ a condição especificada.

• Exemplo:
```
CREATE TABLE Funcionario (
 sexo CHAR(1) NOT NULL CHECK (sexo='M' or sexo='F'),
 idSuperv INT NULL DEFAULT 2,
 idDepto INT NOT NULL  DEFAULT 1
 -- Demais atributos omitidos
);
```
↳ checa se o valor do atr. atende à condição
↳ Tem que ser "M" ou "F"
↳ Se nenhum valor p/ esse atr. for explicitado na inserção de uma nova tupla, o valor dele será 1

- Chave primária: PRIMARY KEY
↳ especifica qual(is) atributo(s) define(m) a chave primária da tabela.

- Chave secundária: UNIQUE
↳ especifica qual(is) atributo(s) define(m) a chave secundária da tabela

- Chave estrangeira: FOREIGN KEY
↳ especifica qual(is) atributo(s) define(m) a chave estrangeira da tabela

chave estrangeira: FOREIGN KEY

↳ especifico qual(is) atributo(s) definem a chave estrangeira da tabela

↳ opções de exclusão/atualização:
- CASCADE (propagação),
- RESTRICT (restrição),
- SET NULL (substituição por nulos),
- SET DEFAULT (substituição por valor padrão)

caso a opção não seja especificada usamos RESTRICT por padrão

```sql
CREATE TABLE Funcionario (
 idFunc INT NOT NULL PRIMARY,
 nomeFunc VARCHAR(50) NOT NULL,

 -- ...

 idSuperv INT NULL,
 idDepto INT NOT NULL,

 CONSTRAINT fk_superv FOREIGN KEY
(idSuperv)
 REFERENCES Funcionario(idFunc)
 ON DELETE SET NULL
 ON UPDATE CASCADE,

 CONSTRAINT fk_depto FOREIGN KEY
(idDepto)
 REFERENCES Departamento(idDepto)
 ON DELETE RESTRICT
 ON UPDATE CASCADE
);
```