

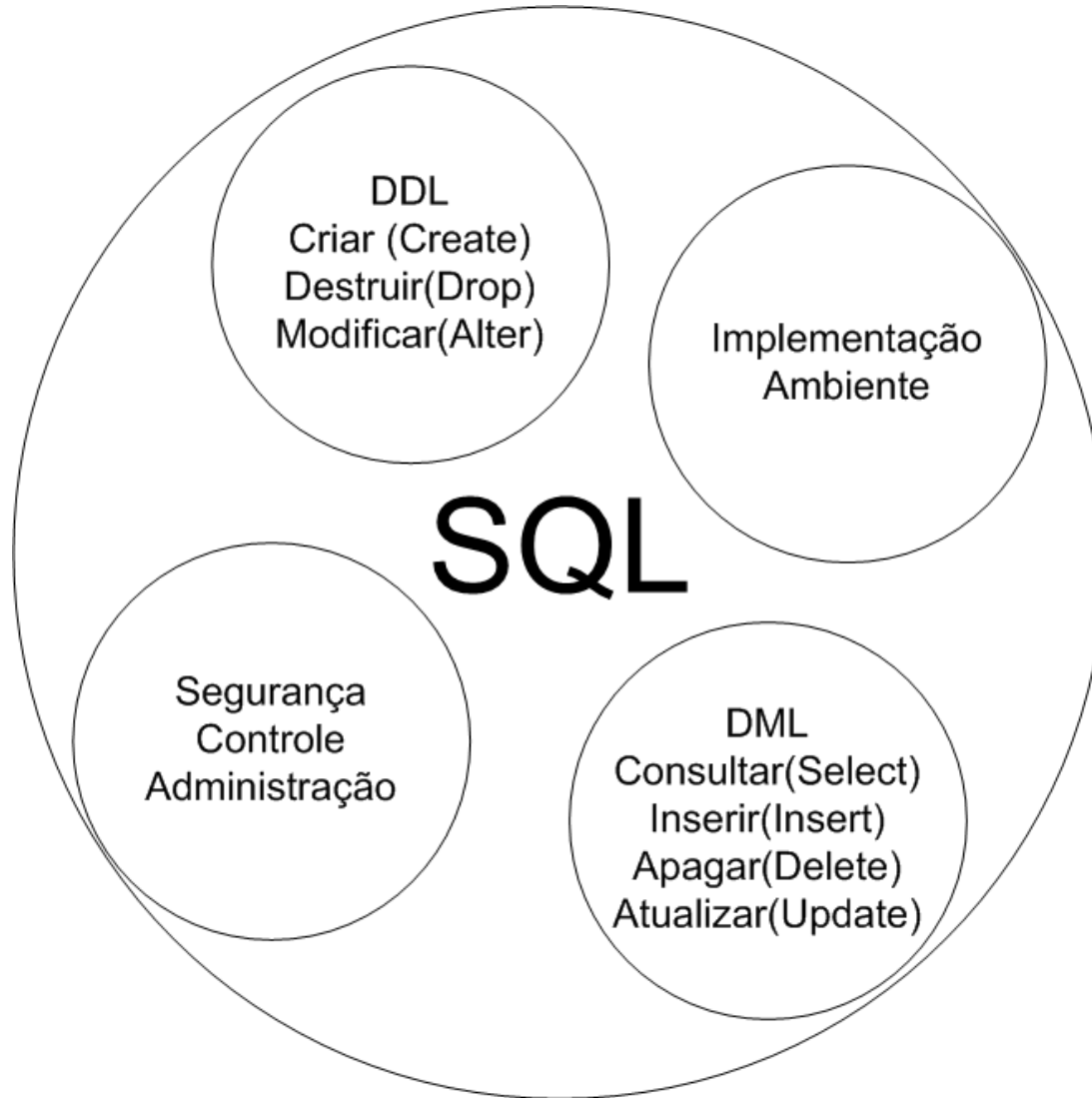
A importância do SQL

- **Linguagem Interativa de consulta** : O usuário cria consultas poderosas sem uso de programas;
- **Linguagem de programação para acesso a B. D. :** Comandos SQL embutidos em programas acessam dados armazenados;
- **Linguagem de administração de banco de dados:** Comandos SQL disponíveis para o administrador do B. D realizar suas tarefas.
- **Linguagem cliente/servidor:** os programas cliente, usam comandos SQL para se comunicarem , através da rede, a um servidor que compartilha seus dados;

A importância do SQL

- **Linguagem para B. D. distribuído:** A SQL auxilia na distribuição dos dados ;
- **Caminho de acesso a outros B. D. em diferentes máquinas :** A SQL auxilia na conversão entre diferentes produtos de B.D. colocados em diferentes máquinas (pequeno porte à grande porte);

Composição da SQL



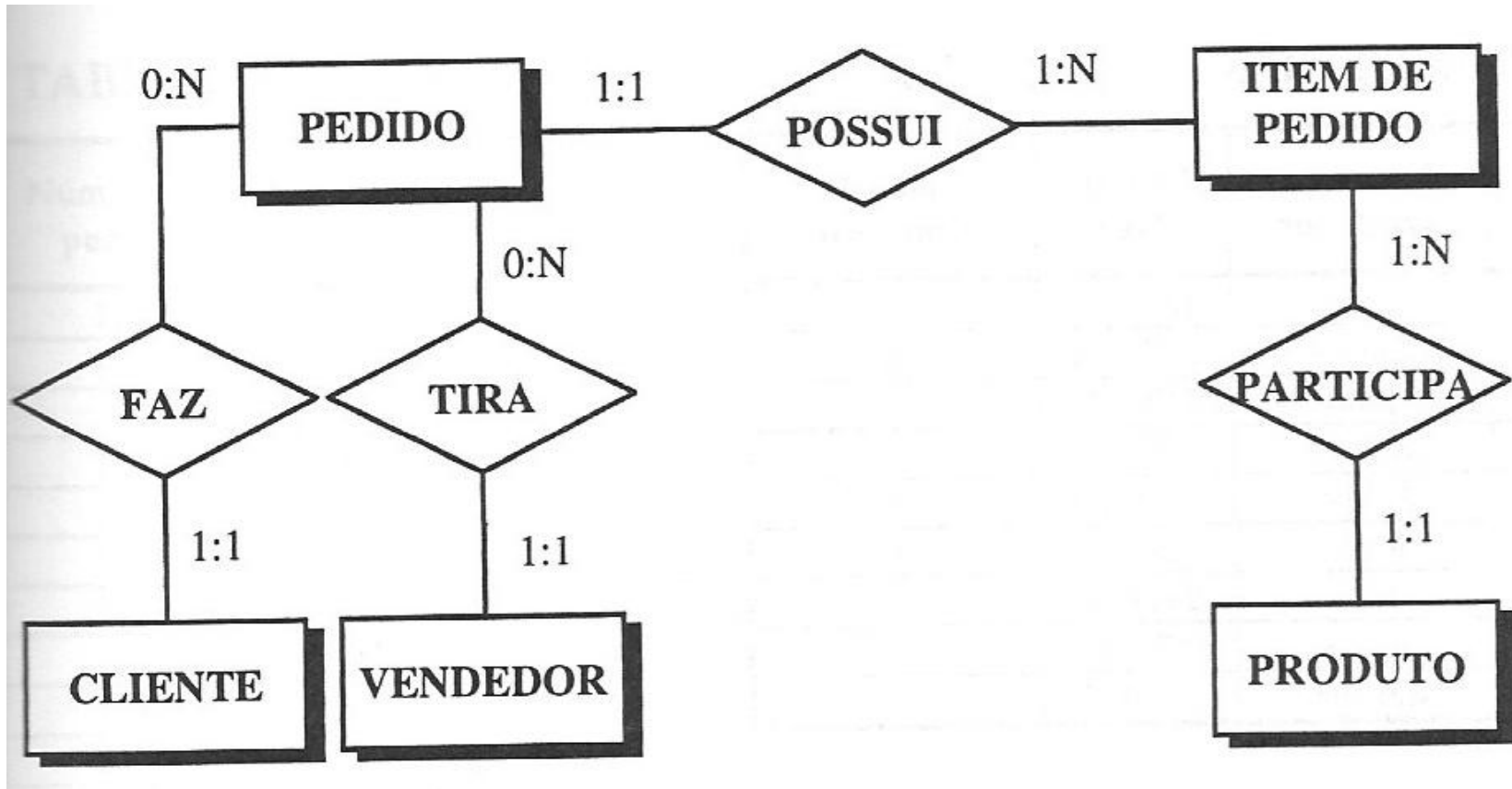
Composição da SQL

- **DDL** : definição da estrutura e organização dos dados armazenados, e seus relacionamentos;
- **DML** : rotinas de inclusão, remoção , seleção ou atualização dos dados armazenados do B. D.;
- **Controle acesso**: protege dados de manipulação não autorizadas;
- **Compartilha dados**: coordena o compartilhamento dos dados por usuários concorrentes;
- **Integridade dos dados**: auxilia no processo de definição da integridade dos dados, protege contra corrupções, inconsistências e falhas.

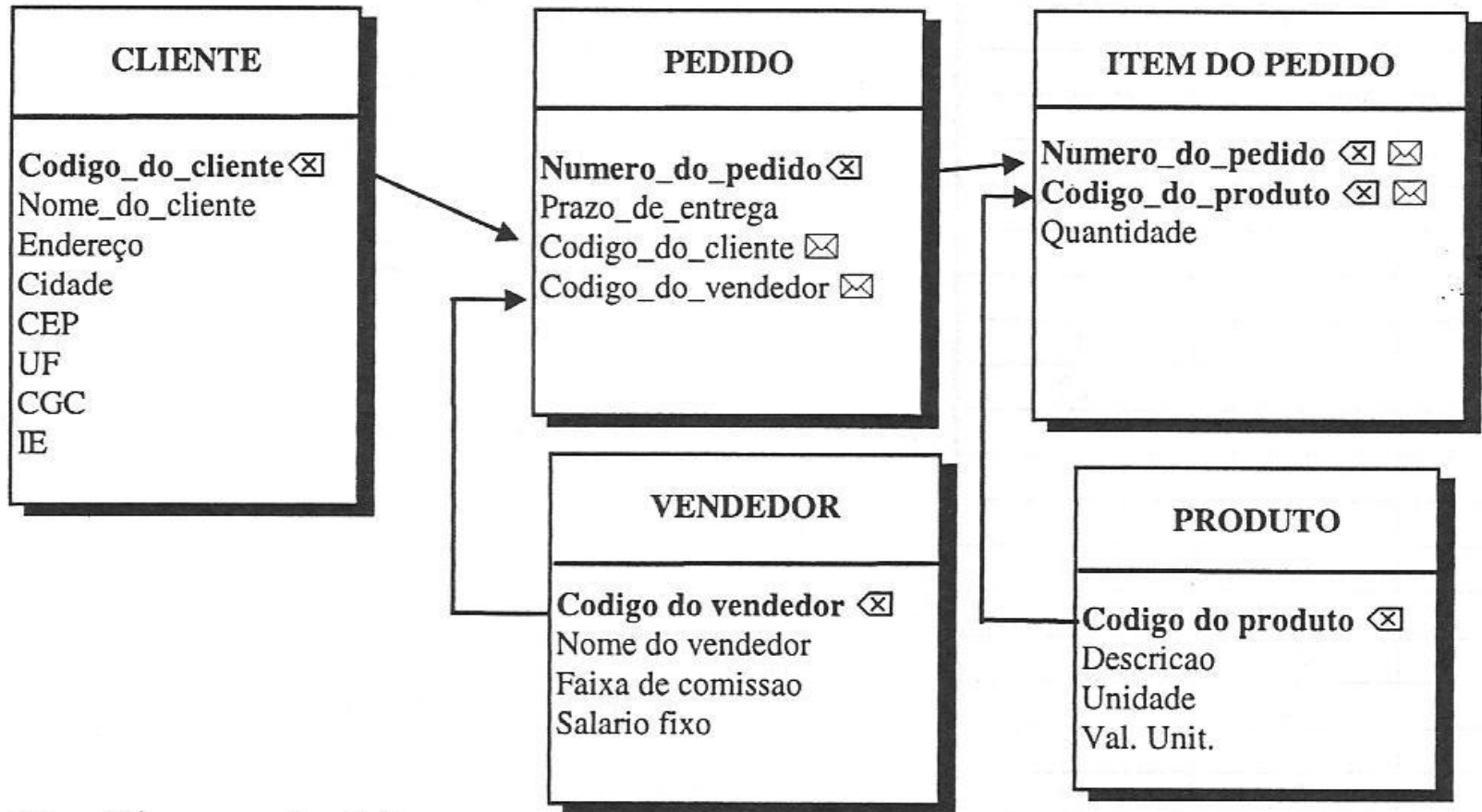
Vantagens

- **Independência de fabricantes** : padronização dos comandos (ANSI);
- **Portabilidade entre computadores**: de computadores pessoais à grande porte;
- **Redução de custos com treinamentos**;
- **Inglês estruturado de alto nível**: conjunto simples de sentenças em inglês;
- **Consulta interativa**: acesso rápido e respostas a consultas complexas;
- **Múltiplas visões dos dados**: criação de diferentes visões dos dados armazenados pelo usuário;
- **Definição dinâmica dos dados** : modificação da estrutura de dados com a flexibilidade;

Modelo conceitual



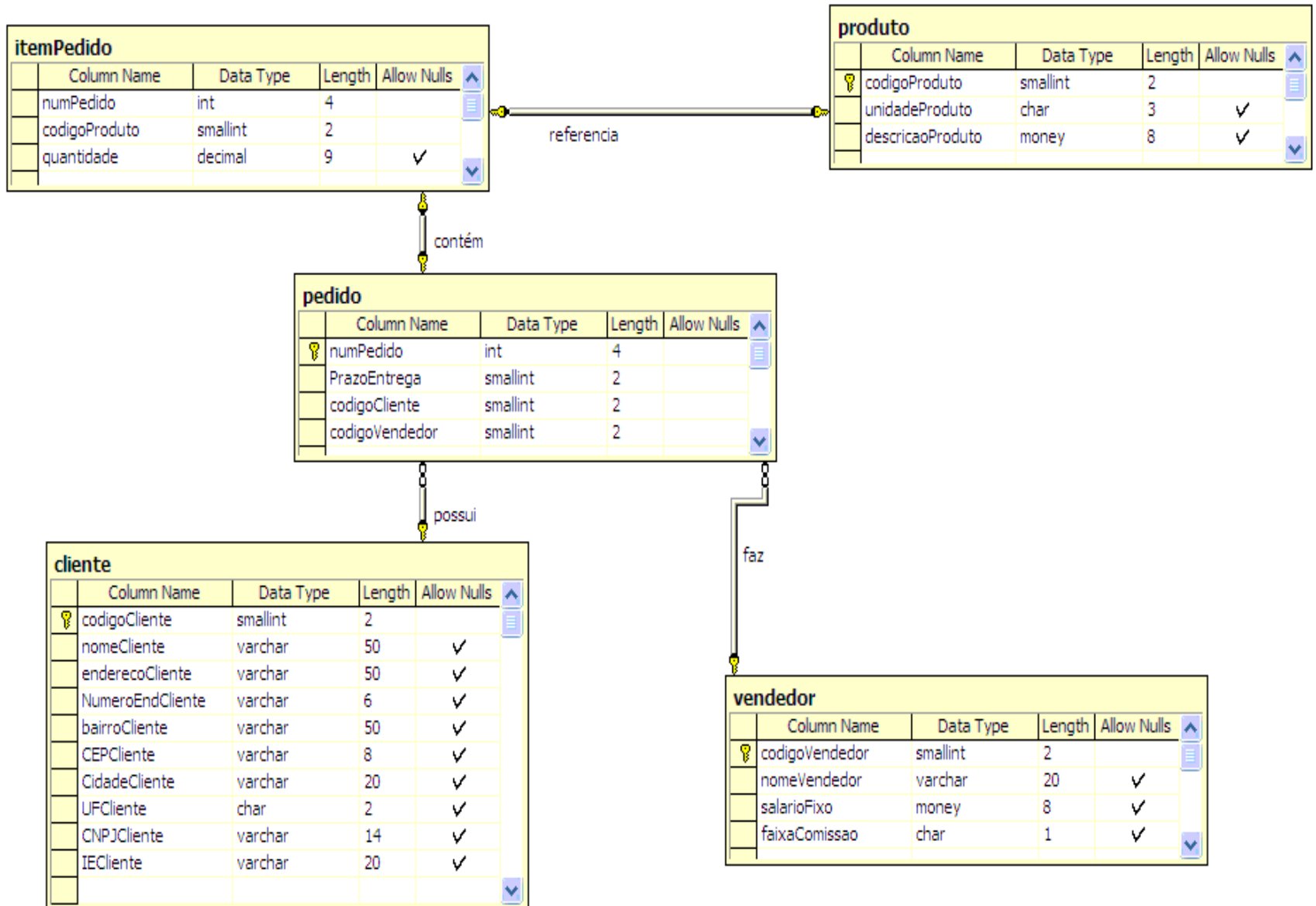
Modelo Lógico



☒ : Chave primária;

☒ : Chave estrangeira.

Modelo físico



DDL - Create Database

```
CREATE DATABASE nome_do_bd  
[ ON PRIMARY ] arquivo1 , arquivo2, ...  
[ LOG ON arquivo3, arquivo4,... ]  
[ FOR RESTORE ]  
[ COLLATE nome_collate ]
```

1. **nome_do_bd**: é o nome do banco de dados (podemos usar até 128 caracteres) ;
2. **ON PRIMARY**: especifica o primeiro arquivo que contém tabelas de sistema e informações internas
 - a. **arquivo1 , arquivo2**: representam as especificações dos arquivos (nome, tamanho, etc...);
3. **LOG ON**: é usada para definir um ou mais arquivos como destino do log de transações do banco de dados;
4. **FOR RESTORE**: usada apenas por questões de compatibilidade com as versões anteriores;
5. **COLLATE**: define um conjunto de regras padrão para o banco de dados;

Exemplos – Create database

- **CREATE DATABASE** vendas (default com tamanho de 2 MB);
- **CREATE DATABASE** vendas **ON** default = 256 (cria a database vendas no device default com 256 MB);
- **CREATE DATABASE** vendas **ON default** = 50, novosdados = 25 (Cria a data base vendas e aloca 50 MB no device default, e 25 MB no device novosdados);
- **CREATE DATABASE** vendas **ON** library_dev1 = 10 **LOG ON** librlog_dev2 = 4 (cria database vendas e aloca 10 MB em library_dev1 e coloca 4 MB para log de transações num device separado chamado librlog_dev2);

DDL – Alter database

Definição : Utilizado para alterar as características do banco de dados (nome banco de dados, nome dos arquivos de log, e arquivo das tabelas);

Sintaxe :

```
ALTER DATABASE database
```

```
{
```

```
ADD FILE < filespec > [ ,...n ] [ TO FILEGROUP filegroup_name ]
```

```
| ADD LOG FILE < filespec > [ ,...n ]
```

```
| REMOVE FILE logical_file_name
```

```
| MODIFY FILE < filespec >
```

```
| MODIFY NAME = new_dbname
```

```
}
```

Exemplos – Alter database

Alter database vendas

Modify name = “vendas2”

Alter database vendas

```
MODIFY FILE (NAME = exemplo1olddata, FILENAME =  
'E:\SQLData\exemplo1newdata.mdf')
```

Alter database exemplo1

```
modify file (NAME = exemplo1oldlog, FILENAME =  
'E:\SQLData\exemplo1newslog.ldf')
```

DDL – Drop database

Definição : Utilizado para apagar uma base de dados específica;

Sintaxe :

Drop database <nome da base de dados>

Exemplos : drop database vendas

DDL – Create table

Sintaxe :

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

Exemplos :

Create table cliente

```
(  
    codigo_cliente      int identity not null unique, (primary key,)  
    Nome_cliente        char(20) ,  
    Endereco            char (20),  
    Cidade               char(15),  
    CEP                  char (8),  
    UF                   char (2),  
    CGC                  char(20),  
    IE                   char (20)  
    Primary key (codigo_cliente)  
);
```

DDL – Create table

```
CREATE TABLE pessoa (  
    identificação INTEGER,  
    nome VARCHAR(30) UNIQUE,  
    morada VARCHAR(30) DEFAULT 'Covilhã',  
    sexo CHAR NOT NULL CHECK (sexo IN ('M','F')));
```

- **UNIQUE** indica que os nomes não se podem repetir nesta tabela
- **DEFAULT** preenche por defeito o campo morada com covilhã
- **NOT NULL** impede que o campo sexo fique em branco
- **CHECK** valida apenas os dados inseridos que respeitem a condição

DDL - Constraints

Definição : As restrições (Constraints) podem ser definidas especificamente através do comando CONSTRAINT. Cada uma é um objeto da base de dados que pode ser criado, modificado ou eliminado independentemente da tabela da qual está associada sendo referenciado pelo nome com que foi definida.

```
CREATE TABLE pessoa (  
    identificação INTEGER,  
    nome VARCHAR(30),  
    cdpostal INTEGER,  
    CONSTRAINT ck_pessoa_identificacao CHECK (identificacao > 0),  
    CONSTRAINT pk_pessoa_identificacao PRIMARY KEY (identificacao),  
    CONSTRAINT uu_pessoa_nome UNIQUE(identificacao),  
    CONSTRAINT fk_pessoa_cdpostal FOREIGN KEY (cdpostal)  
    REFERENCES morada(cdpostal) );
```

Alteração das Tabelas

Sintaxe :

Mostrar o help do Query Analyzer

Exemplos :

```
ALTER TABLE pessoa ADD idade INTEGER
```

```
ALTER TABLE pessoa ADD CONSTRAINT ck_pessoa_idade CHECK  
(idade BETWEEN 0 AND 100)
```

```
ALTER TABLE pessoa ALTER COLUMN nome VARCHAR(40)
```

```
ALTER TABLE pessoa DROP CONSTRAINT ck_pessoa_idade
```

```
ALTER TABLE pessoa DROP idade INTEGER
```

```
ALTER TABLE cliente add campo1 char(5)
```

```
ALTER TABLE cliente add constraint pertence Foreign key  
(idloja) References loja;
```

DDL – Drop table

Definição : Utilizado para apagar uma tabela ;

Sintaxe :

```
drop table <nome da tabela>
```

Exemplo : drop table cliente

DML - Insert

Insert

Descrição : destinado a inserir um registro em uma tabela específica.

Sintaxe :

```
Insert into tabela (campo1,campo2,campo3)  
Values (valor1,valor2,valor3)
```

Exemplo :

```
insert into cliente (codigoCliente,nomeCliente,enderecoCliente)  
values (1,'Mauricio','av antonelo da messina')
```

DML - Update

Update

Descrição : Destinado a alterar um ou um grupo de registros de uma tabela específica;

Sintaxe :

```
Update from <tabela>
```

```
Set campo1 = 'conteúdo a ser alterado'
```

```
Where campo2 = <critério>
```

Exemplo :

```
Update from cliente
```

```
set nomecliente = 'Joao da Silva'
```

```
Where codigocliente = 1
```

DML – Select

Select

Descrição : Destinado a selecionar um ou um grupo de registros em uma ou mais tabelas específicas.

Sintaxe :

```
Select campo1,campo2 ...  
    from <nome da tabela>  
    Where campo1 = <critério de consulta>
```

Exemplo :

```
Select codigocliente,nomecliente  
    From cliente  
    Where codigocliente = '1'
```

DML - Delete

Delete

Descrição : Utilizado para apagar um ou um grupo de registro de uma tabela específica.

Sintaxe :

```
delete from <tabela>
```

```
Where campo = <critério de deleção>
```

Exemplo :

```
delete from cliente
```

```
Where codigocliente = 1
```