

Bases de Dados

Módulo 7: Definição de esquemas com SQL

Prof. André Bruno de Oliveira

Data 18/07/24

Modelo Relacional – Definição de esquemas com SQL

Objetivo: Aula prática para apresentar a ferramenta SQLite Studio e aprender uma forma básica de como definir esquema e seus elementos com SQL.

TÓPICOS

- Aprender a usar o SQLite Studio
- Criar um banco dados
- Aprender SQL
 - Criar esquemas das tabelas
- Exercício sobre criação de BD para um pesquisa de satisfação






USO DO SQLITE STUDIO

ETAPA 1: Entrar no SQLite Studio

(SQLite Studio é um gerenciador de banco de dados do SQLite).

- Localize a pasta onde está armazenado o executável do SQLite Studio e dê dois-cliques no programa “SQLiteStudio.exe” (a extensão “.exe” pode não estar aparecendo – observe então o desenho do ícone destacado na Figura 1).

Figura 1 – Pasta com o executável do SQLiteStudio

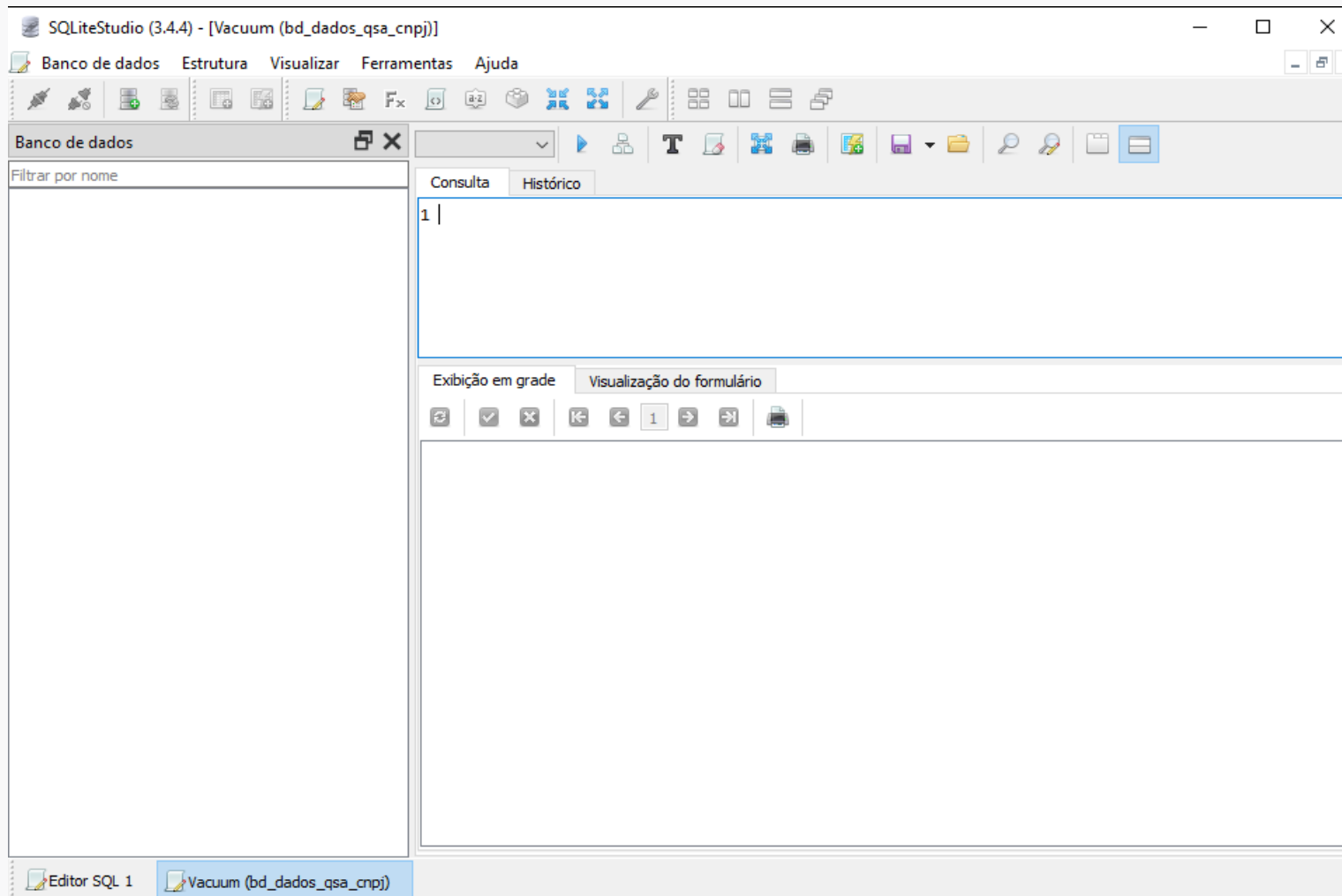
	sqlite3.dll	06/04/2023 19:54	Extensão de aplica...	1.014 KB
	SQLiteStudio.exe	06/04/2023 19:54	Aplicativo	450 KB
	sqlitestudiocli.exe	06/04/2023 19:54	Aplicativo	711 KB
	tcl86.dll	06/04/2023 19:54	Extensão de aplica...	2.277 KB
	zlib1.dll	06/04/2023 19:54	Extensão de aplica...	114 KB

USO DO SQLITE STUDIO

ETAPA 1: Entrar no SQLite Studio

- Uma tela similar abaixo é apresentada.

Figura 2 – Tela inicial do SQLite Studio

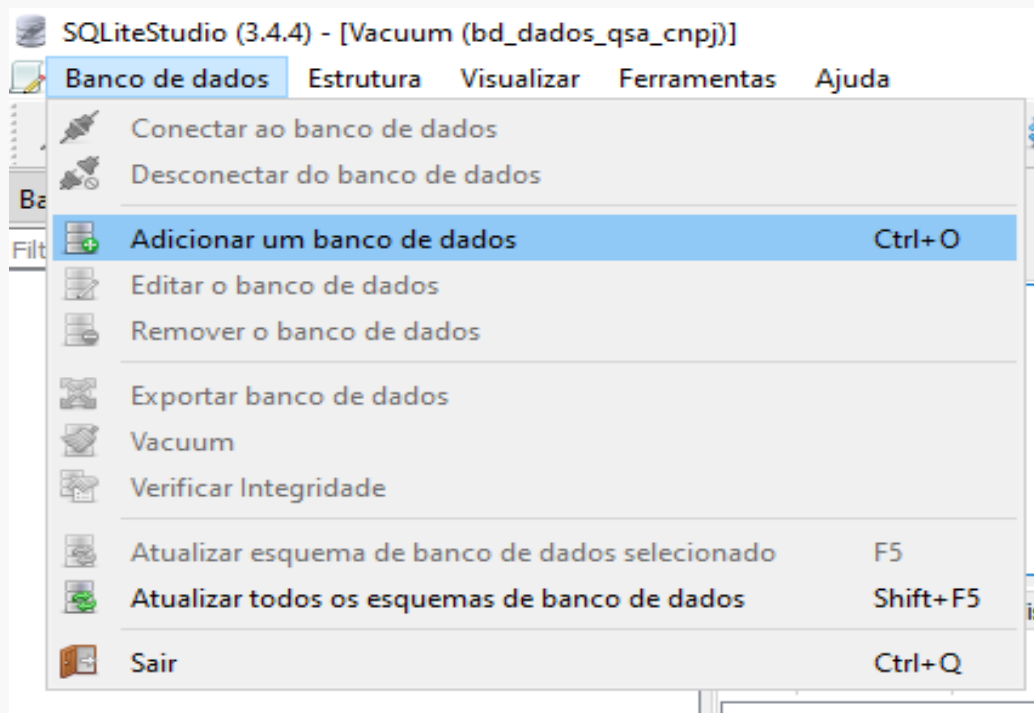


USO DO SQLITE STUDIO

ETAPA 2: Criar um Banco de Dados

- Acesse a opção de menu “Banco de dados”, como mostra a Figura 3. Selecione “Adicionar um banco de dados”.

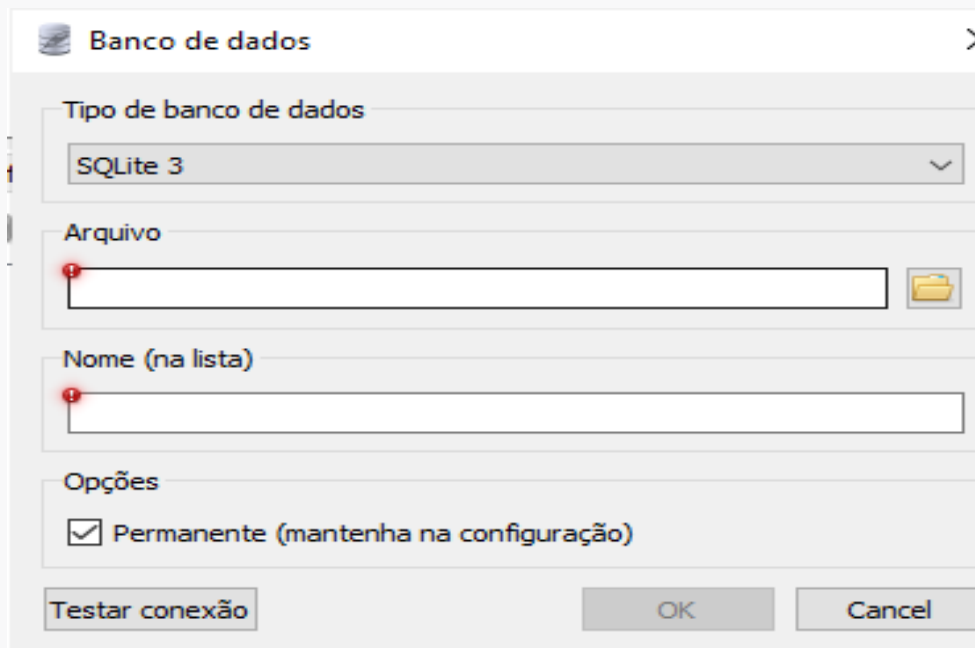
Figura 3 – Menu para adicionar um banco de dados



USO DO SQLITE STUDIO

- **ETAPA 2: Criar um Banco de Dados**
- Na nova janela que aberta (Figura 4) selecione o ícone de pasta para selecionar o local onde deseja criar o banco de dados.
- **Atenção:** Você pode escolher uma pasta qualquer, tanto em seu computador como em um pen drive, para armazenar o BD “Exemplo1.db”.

Figura 4 – Informar parâmetros de criação do banco de dados



Banco de dados

Tipo de banco de dados

SQLite 3

Arquivo

Nome (na lista)

Opções

☒ Permanente (mantenha na configuração)

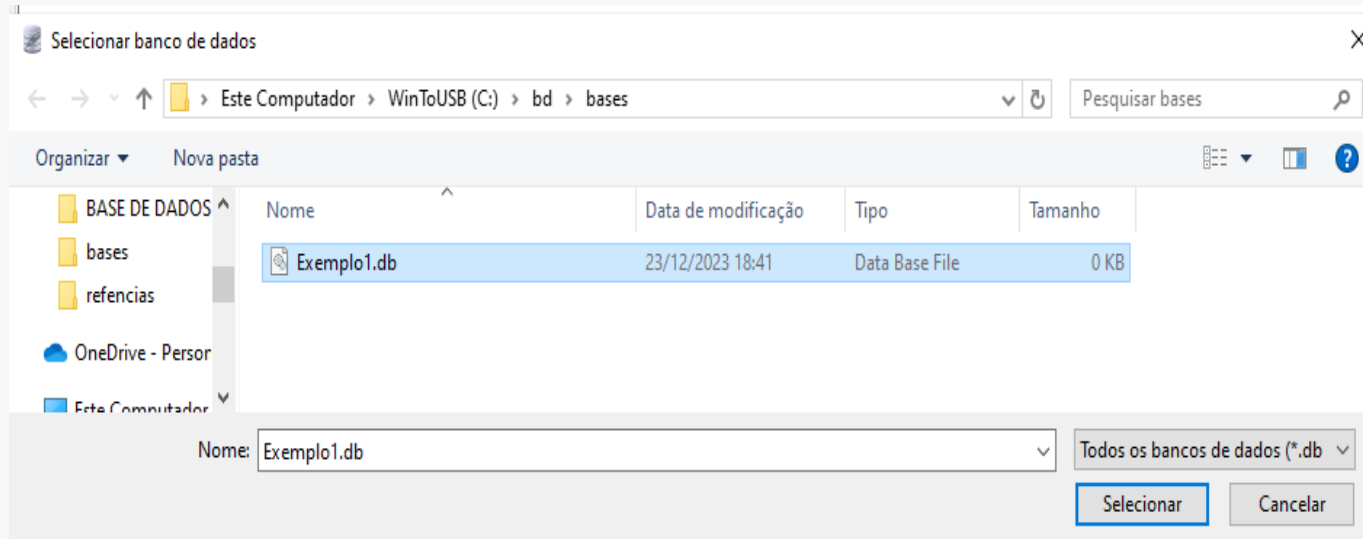
Testar conexão OK Cancel

USO DO SQLITE STUDIO

ETAPA 2: Criar um Banco de Dados

• Digite “Exemplo1.db” para definir o nome do arquivo de BD. Clique na opção “Selecionar” para confirmar. Como não existe um arquivo com este nome nesta pasta, o SQLite Studio compreende que precisa ser criado um banco de dados novo.

Figura 5 – Criando um banco de dados

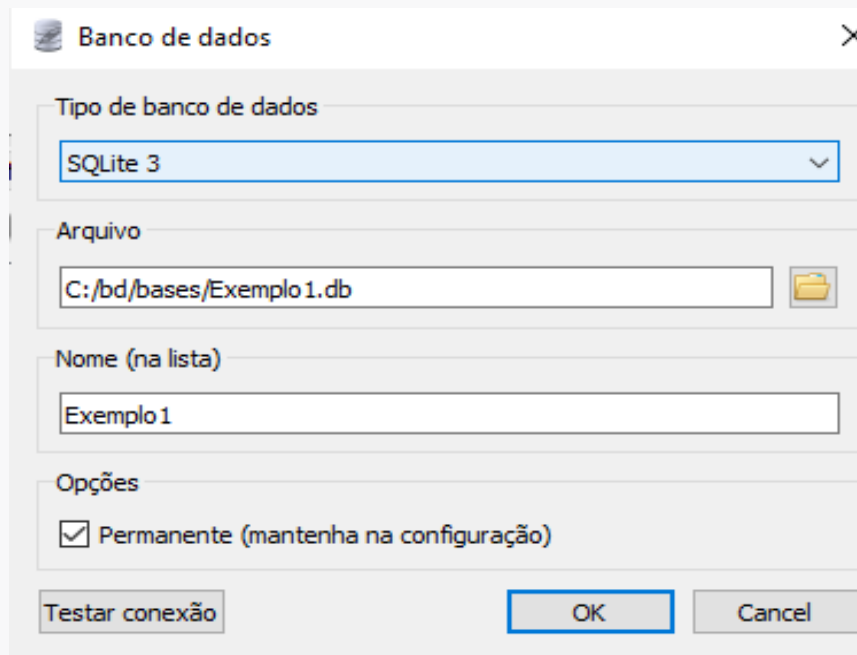


USO DO SQLITE STUDIO

ETAPA 2: Criar um Banco de Dados

- A opção nome exibe o nome fantasia para a conexão com o BD, mantenha este nome Exemplo1 para ficar tudo padronizado. Mantenha a opção “Permanente (*keep it in configuration*)” marcada. Selecione “OK” (Figura 6) para confirmar adição da conexão com o banco de dados.

Figura 6 – Confirmar adição do banco de dados



USO DO SQLITE STUDIO

ETAPA 3: Efetuar a conexão com o BD criado

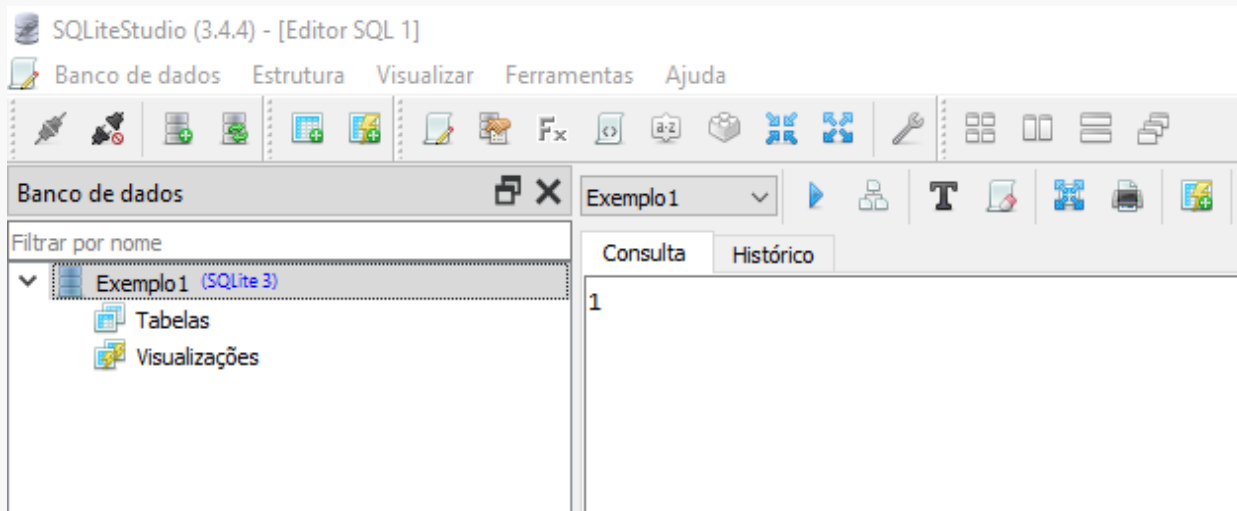
- O SQLite é um BD recomendado para ser usado em dispositivos móveis. Sua arquitetura opera de forma independente sem a necessidade um servidor. Não oferece recurso de autenticação com usuário e senha. Seu funcionamento adota as regras do modelo relacional para manipular e definir as estruturas de tabelas e visões. Usa o SQL como linguagem de manipulação e definição de dados. O SQLite não é um SGBD real, mas sim uma biblioteca que emula um SGBD.

USO DO SQLITE STUDIO

ETAPA 3: Efetuar a conexão com o BD criado

- Nesta etapa ocorre a tarefa de atuar como projetista de BD (como criar tabelas e outros objetos), assim é preciso realizar a conexão com o banco “Exemplo1.db”.
- Para efetuar a conexão, selecione o BD “Exemplo1” e clique no ícone de conexão (primeiro ícone da barra de tarefas na Figura 7). Também é possível fazer a conexão acessando o menu Banco de dados =>Conectar ao banco de dados.

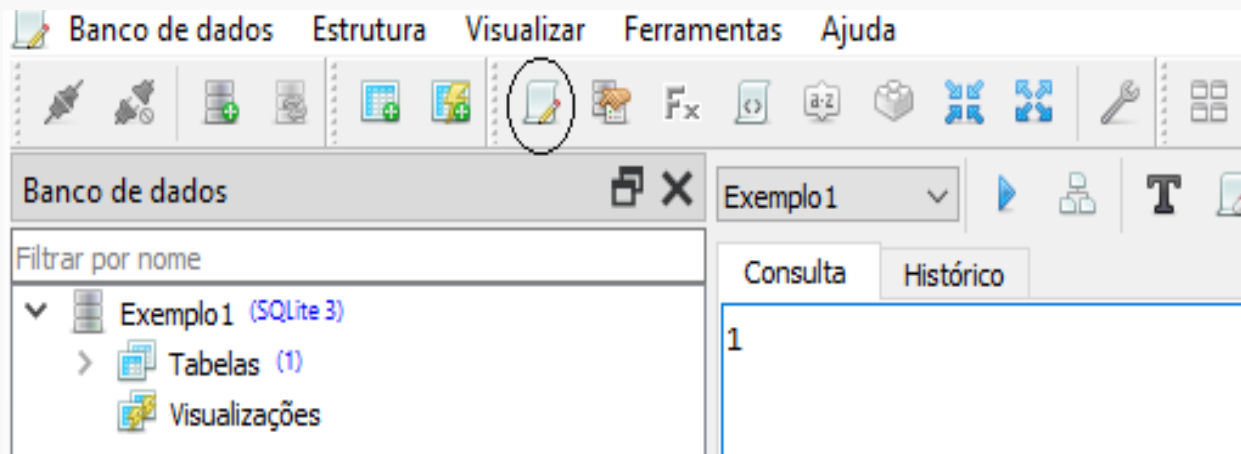
Figura 7 – Conectar com o banco de dados



DEFINIR OS ELEMENTOS DO ESQUEMA

- O BD “Exemplo1” ainda está completamente vazio. É necessário criar suas tabelas “dentro” dele usando a linguagem SQL. A aba de consulta pode ser adicionada selecionando o ícone em destaque na figura 8.

Figura 8 – Aba de consulta



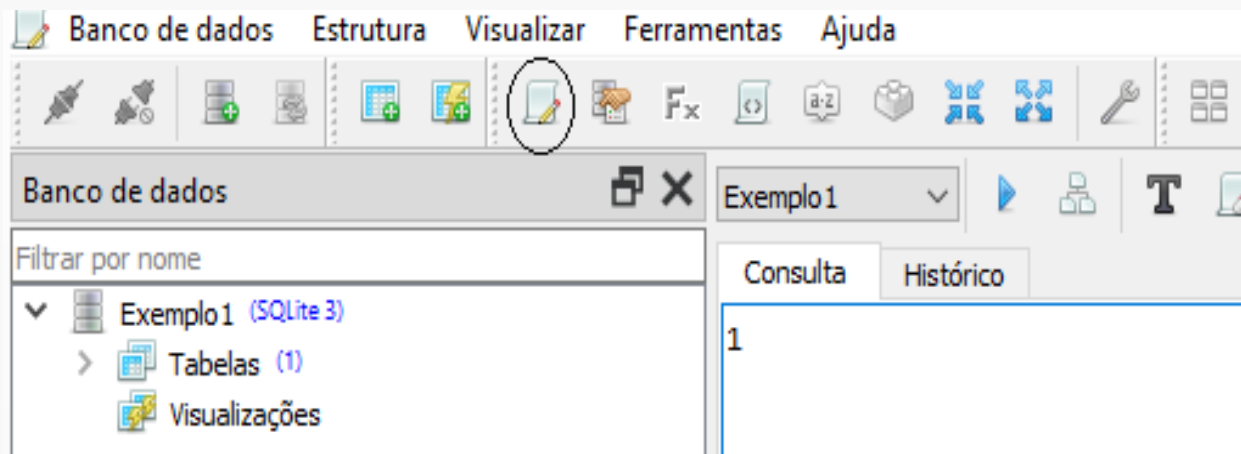
DEFINIR OS ELEMENTOS DO ESQUEMA

- Na área de edição (aba Consulta), você digitará e executará a sua primeira instrução SQL. Esta instrução será responsável por definir o esquema abaixo dentro do BD “Exemplo1”:

- Escola*

(co_entidade :inteiro, no_entidade: alfanumérico, cod_regiao: inteiro, cod_uf: inteiro, cod_municipio: inteiro, tp_dependencia: inteiro, tp_localizacao: inteiro, qt_mat_bas_fem: inteiro, qt_mat_bas_masc: inteiro, qt_mat_bas: inteiro, qt_doc_bas: inteiro)

Figura 8 – Aba de consulta



DEFINIR OS ELEMENTOS DO ESQUEMA

- **CREATE TABLE** é a instrução SQL utilizada para criar tabelas (ou relações). Em sua forma básica, a instrução é utilizada da seguinte maneira:
 - Começa com a expressão **CREATE TABLE** seguida pelo nome da tabela a ser criada;
 - Segue com uma lista de atributos e seus respectivos tipos. Esta especificação deve ser feita entre parênteses, com cada definição de atributo separada por vírgula; No SQLite os tipos básicos são **INT** (inteiro), **NUM** (real) e **TEXT** (alfanumérico), só que há outros tipos associados a estes três tipos.
 - Os atributos que não poderão aceitar valores nulos devem ser especificados com o uso da expressão **NOT NULL** após a indicação de seu tipo.
 - Os atributos que compõem a chave primária são indicados com o uso da expressão **PRIMARY KEY**, no final da especificação do comando. Para criar a tabela *Escola* no SQLite, a seguinte declaração pode ser utilizada:

```
CREATE TABLE Escola (  
  co_entidade integer, no_entidade VARCHAR (200),  
  cod_regiao NUMERIC (1),cod_uf NUMERIC (2),  
  cod_municipio INTEGER, tp_dependencia INTEGER,  
  tp_localizacao INTEGER, qt_mat_bas_fem INTEGER,  
  qt_mat_bas_masc INTEGER, qt_mat_bas INTEGER,  
  qt_doc_bas INTEGER, PRIMARY KEY(co_entidade)  
);
```

DEFINIR OS ELEMENTOS DO ESQUEMA

- Veja que o comando para criar a tabela *Escola* inclui 11 atributos e 2 tipos. A definição de chave primária (*Primary Key PK*) ou identificação única é criada usando somente um atributo. Pode haver situações que uma PK tenha uma chave composta por mais de um atributo. Neste caso, é só incluir na lista de atributos separando-os por vírgula.
- O Tipo INTERGER corresponde ao tipo INT e tipo VARCHAR(n) corresponde ao tipo TEXT.

```
CREATE TABLE Escola (  
  co_entidade integer, no_entidade VARCHAR (200),  
  cod_regiao NUMERIC (1),cod_uf NUMERIC (2),  
  cod_municipio INTEGER, tp_dependencia INTEGER,  
  tp_localizacao INTEGER, qt_mat_bas_fem INTEGER,  
  qt_mat_bas_masc INTEGER, qt_mat_bas INTEGER,  
  qt_doc_bas INTEGER, PRIMARY KEY(co_entidade)  
);
```

DEFINIR OS ELEMENTOS DO ESQUEMA

- Exemplo de criação tabela com chave composta por lista de atributos:

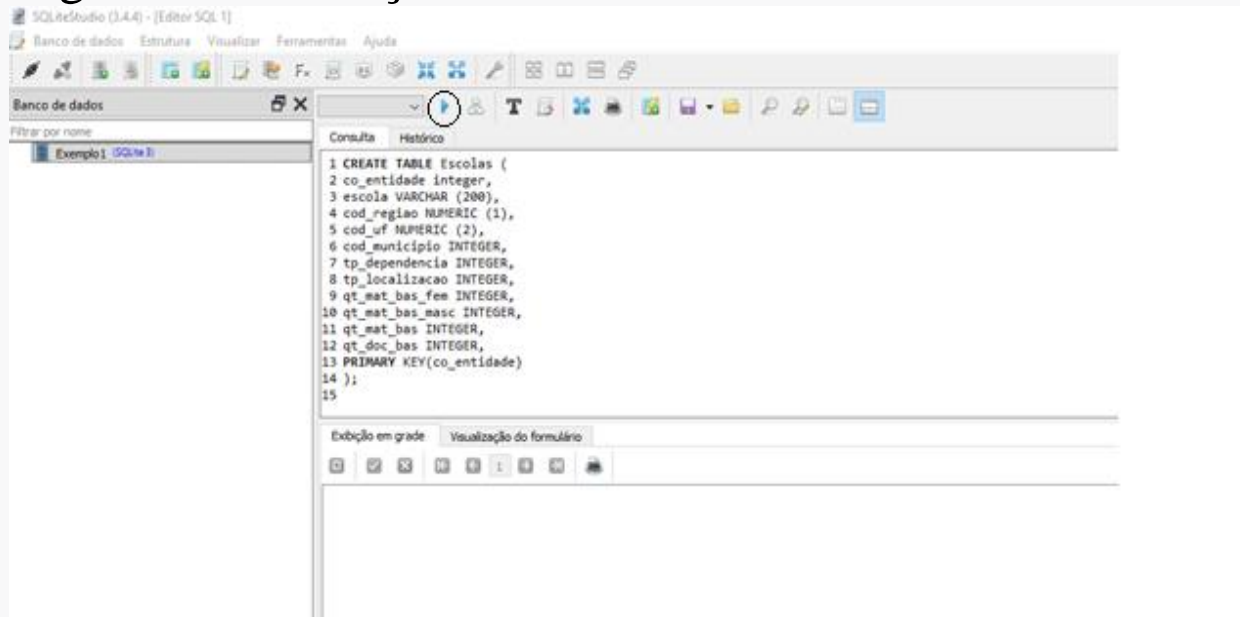
```
CREATE TABLE Pessoa (  
    domicilio NUMERIC,  
    pessoa NUMERIC,  
    idade NUMERIC,  
    parentesco VARCHAR(10),  
    PRIMARY KEY(domicilio, pessoa)  
);
```

- Atenção:** não se preocupe se a escrita dos comandos, nome da tabela e letras estão em letra maiúscula ou minúscula, pois o SQL não faz esta distinção.

DEFINIR OS ELEMENTOS DO ESQUEMA

- Para solicitar a execução de uma consulta SQL no SQLite Studio insira o código na aba de texto Consulta (Figura 9). Clique no botão **Executar Consulta** indicado ou aperte a tecla F9.

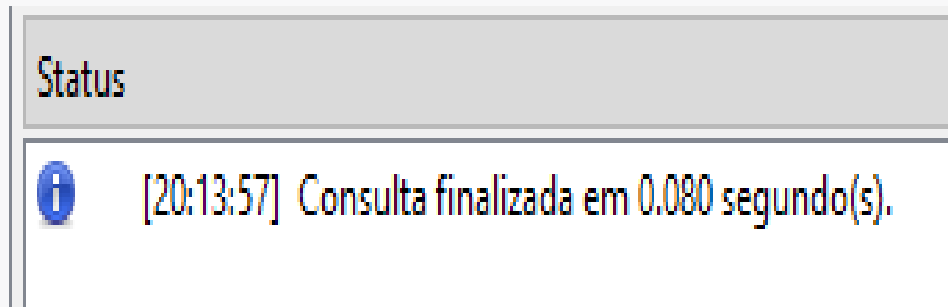
Figura 9 – Execução de consultas



DEFINIR OS ELEMENTOS DO ESQUEMA

- A janela status mostrará se cada operação foi ou não realizada com sucesso. Se tudo tiver corrido bem, uma mensagem similar à apresentada na Figura 10 será apresentada.

Figura 10 – Resultado da consulta executada com sucesso



DEFINIR OS ELEMENTOS DO ESQUEMA

- Agora podemos repetir os mesmos procedimentos para criar as outras tabelas do esquema.

```
CREATE TABLE Regiao (  
  cod_regiao NUMERIC (1),  
  nome_regiao varchar(20),  
  PRIMARY KEY (cod_regiao)  
);  
CREATE TABLE Uf (  
  cod_uf NUMERIC (2),  
  nome_uf varchar(20),  
  sigla_uf VARCHAR(2),  
  PRIMARY KEY (cod_uf)  
);
```

```
CREATE TABLE Municipio (  
  cod_municipio INTEGER,  
  nome_municipio varchar(50));
```

```
CREATE TABLE Tp_dependencia (  
  tp_dependencia INTEGER,  
  dependencia VARCHAR(10),  
  PRIMARY KEY (tp_dependencia)  
);  
CREATE TABLE Tp_localizacao (  
  tp_localizacao INTEGER,  
  localizacao VARCHAR(10),  
  PRIMARY KEY (tp_localizacao)  
);
```

DEFINIR OS ELEMENTOS DO ESQUEMA

- **Só para adiantar**
 - Com o comando INSERT você pode inserir dados em qualquer tabela. E com o comando SELECT você pode consultar os dados de uma tabela. Esses comandos serão apresentados em outras aulas, mas abaixo apresentamos um exemplo envolvendo a tabela Municipio:

--Insere o município de Presidente Kennedy na tabela de Município
INSERT INTO Municipio VALUES (1718402, 'Presidente Kennedy');
--mostra o conteúdo da tabela de Municipio SELECT * FROM
Municipio;

Execute o SQL de INSERT

-- T_MUNICIPIO

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1100205, 'Porto Velho');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1200401, 'Rio Branco');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1302603, 'Manaus');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1400100, 'Boa Vista');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1501402, 'Belém');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1600303, 'Macapá');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (1721000, 'Palmas');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2111300, 'São Luís');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2211001, 'Teresina');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2304400, 'Fortaleza');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2408102, 'Natal');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2507507, 'João Pessoa');

INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2611606, 'Recife');

Execute o SQL de INSERT

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2704302, 'Maceió');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2800308, 'Aracaju');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (2927408, 'Salvador');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (3106200, 'Belo Horizonte');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (3205309, 'Vitória');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (3304557, 'Rio de Janeiro');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (3550308, 'São Paulo');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (4106902, 'Curitiba');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (4205407, 'Florianópolis');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (4314902, 'Porto Alegre');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (5002704, 'Campo Grande');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (5103403, 'Cuiabá');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (5208707, 'Goiânia');
```

```
INSERT INTO Municipio (cod_municipio, nome_municipio) VALUES (5300108, 'Distrito Federal');
```

Execute o SQL de INSERT

-- Tabela: Regiao

```
INSERT INTO Regiao (cod_regiao, nome_regiao) VALUES (1, 'Norte');
```

```
INSERT INTO Regiao (cod_regiao, nome_regiao) VALUES (2, 'Nordeste');
```

```
INSERT INTO Regiao (cod_regiao, nome_regiao) VALUES (3, 'Sudeste');
```

```
INSERT INTO Regiao (cod_regiao, nome_regiao) VALUES (4, 'Sul');
```

```
INSERT INTO Regiao (cod_regiao, nome_regiao) VALUES (5, 'Centro-Oeste');
```

Execute o SQL de INSERT

-- Tabela: UF

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (11, 'Rondônia');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (12, 'Acre');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (13, 'Amazonas');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (14, 'Roraima');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (15, 'Para');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (16, 'Amapá');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (17, 'Tocantins');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (21, 'Maranhão');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (22, 'Piauí');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (23, 'Ceará');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (24, 'Rio Grande do Norte');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (25, 'Paraíba');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (26, 'Pernambuco');
```


Execute o SQL de INSERT

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (27, 'Alagoas');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (28, 'Sergipe');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (29, 'Bahia');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (31, 'Minas Gerais');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (32, 'Espírito Santo');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (33, 'Rio de Janeiro');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (35, 'São Paulo');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (41, 'Paraná');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (42, 'Santa Catarina');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (43, 'Rio Grande do Sul');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (50, 'Mato Grosso do Sul');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (51, 'Mato Grosso');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (52, 'Goiás');
```

```
INSERT INTO UF (cod_uf, nome_uf) VALUES (53, 'Distrito Federal');
```

Execute o SQL de INSERT

- TABELA ESCOLA

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11001100, 'EMEF GEDOCY RUAS WOLFF', 1, 11, 1100205, 3, 2, NULL, NULL, NULL, '');
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11002158, 'EMEF SAO FRANCISCO DE ASSIS', 1, 11, 1100205, 3, 1, 70, 58, 128, 7);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11003391, 'EMEF MANOEL MACIEL NUNES', 1, 11, 1100205, 3, 2, 21, 22, 43, 2);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11003901, 'EMEF ALUIZIO FERREIRA', 1, 11, 1100338, 3, 2, NULL, NULL, NULL, '');
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11004428, 'EEEFM BURITI', 1, 11, 1100452, 2, 1, 337, 293, 630, 22);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11005360, 'EEEFM PAULO FREIRE', 1, 11, 1101104, 2, 1, 500, 524, 1024, 58);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11006722, 'APAE DE ARIQUEMES', 1, 11, 1100023, 4, 1, NULL, NULL, NULL, '');
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11009888, 'EEEFM LAURINDO RABELO', 1, 11, 1100403, 2, 1, 461, 434, 895, 26);
```

Execute o SQL de INSERT

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11017112, 'EMEIEF MARACATIARA', 1, 11, 1100155, 3, 2, 60, 40, 100, 7);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11022230, 'EEEF ALEXANDRE DE GUSMAO', 1, 11, 1100148, 2, 1, 81, 73, 154, 13);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11022256, 'CEEJA GETULIO VARGAS', 1, 11, 1100320, 2, 1, 113, 76, 189, 10);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11031565, 'EMEF ANTONIO CANDIDO SILVEIRA', 1, 11, 1101450, 3, 2, NULL, NULL, NULL, '');
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11037075, 'ESCOLA DE EDUCACAO INFANTIL ENSINO FUNDAMENTAL E MEDIO DIMENSAO', 1, 11, 1100056, 4, 1, 75, 81, 156, 18);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11038012, 'EEEFM MADEIRA MAMORE', 1, 11, 1100205, 2, 2, 50, 461, 511, 25);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11040882, 'EEEF JOAO FRANCISCO CORREIA', 1, 11, 1101104, 2, 1, 138, 157, 295, 10);
```

```
INSERT INTO Escola (co_entidade, no_entidade, cod_regiao, cod_uf, cod_municipio, tp_dependencia, tp_localizacao, qt_mat_bas_fem, qt_mat_bas_masc, qt_mat_bas, qt_doc_bas) VALUES (11040904, 'EMEI RUTH VIEIRA BEXIGA', 1, 11, 1100049, 3, 1, NULL, NULL, NULL, '');
```

DEFINIR OS ELEMENTOS DO ESQUEMA

- A criação das tabelas faz uso do SQL embutido no SQLite para definição de dados e cria os tipos com seus respectivos domínios.
 - O tipo VARCHAR(n) é usado para informar o tamanho do campo alfanumérico
 - O tipo NUMERIC(n) é usado para o informar o número de dígitos que o atributo/campo deve respeitar.
 - Os BD implementam de modo geral suas próprias versões de SQL baseados nos documentos mantidos por entidades americanas e comitês internacionais denominado como padrão ANSI para SQL.
 - A maioria dos SGBD incluem modificações no SQL para atender às suas necessidades, todos baseiam seus programas nesta versão padrão. De modo geral os comandos para modelagem e definição de dados para cada SGBD são parecidos.
 - A SQL passa por revisões periódicas e está na edição ISO/IEC 9075 para SQL (Fonte: <https://blog.ansi.org/sql-standard-iso-iec-9075-2023-ansi-x3-135/#gref>).

DEFINIR OS ELEMENTOS DO ESQUEMA

- O SQLite usa um sistema de tipos dinâmico em que o tipo de um valor está associado ao valor em si e não ao tipo da coluna que o valor deve ser armazenado. É possível usar qualquer nome de tipo de coluna desejado. O Microsoft.Data.Sqlite não aplicará nenhuma semântica adicional a esses nomes.
 - Um problema comum é que o uso de um tipo de coluna de STRING tentará converter valores em INTEGER ou REAL, o que pode levar a resultados inesperados. Recomenda-se usar apenas os quatro nomes de tipo primitivos do SQLite: **INTEGER, REAL, TEXT e BLOB**.
 - O SQLite permite especificar facetas de tipo como comprimento, precisão e escala, mas elas não são impostas pelo mecanismo de banco de dados. **Seu aplicativo é responsável por aplicá-las.**
 - Exemplo VARCHAR(10) tem afinidade para o tipo TEXT. Esta especificação sugere que uma coluna do tipo alfanumérico e espera receber até 10 caracteres. Se por algum motivo o sistema do usuário deixar passar uma cadeia de 20 caracteres, automaticamente o mecanismo do banco de dados vai usar um outro tipo que aceite o formato desejado e nenhum erro vai ocorrer por parte do SQLite.

Fonte: <https://learn.microsoft.com/pt-br/dotnet/standard/data/sqlite/types>

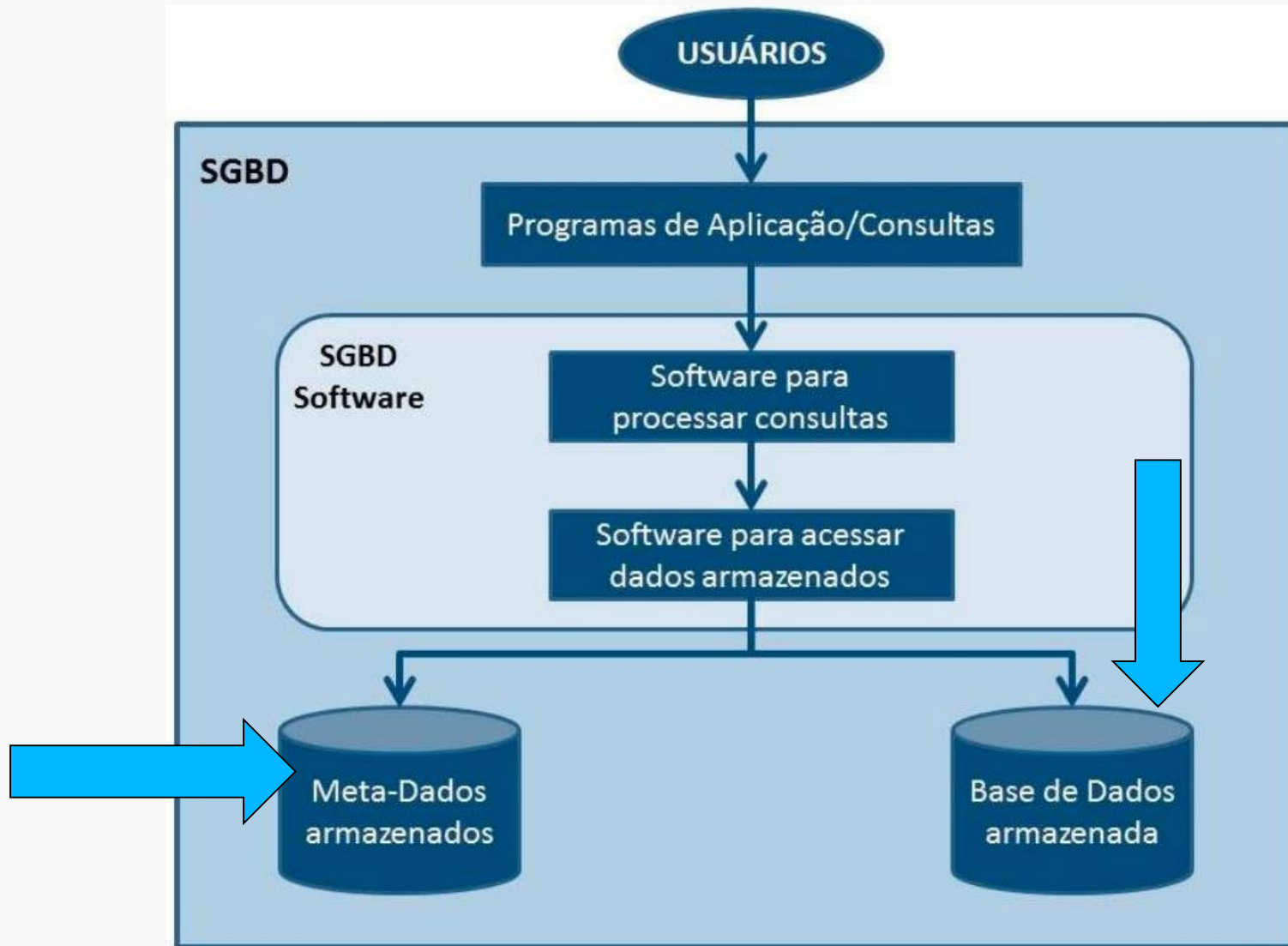
Consultado metados do SQLite

- Nas aulas anteriores foi visto que o BD relacional com SQL mantém os metados (dados que definem dados) dentro do BD para controlar as restrições do banco de dados. O banco de dados relacional usa a sua própria linguagem para delimitar o seu funcionamento.
- Execute a consulta abaixo para você ver os metadados do seu banco de dados.

```
SELECT type, name, tbl_name, sql  
FROM sqlite_master.
```

- Este tipo de controle é marcante no BD relacional, por isso é ditto que este tipo de BD possui definição de esquema. Uma vez definido o esquema, todo seu funcionamento fica baseado no metadado.

Exemplo do modelo de BD Relacional



Exercício proposto 1 Entregar na próxima semana.

- Considere um projeto para atender a pesquisa de satisfação sobre a qualidade do serviço de telefonia móvel. Utilize o SQLite Studio, crie um novo BD chamado “pesq_satisfacao.db”. Após criá-lo, realize a conexão com o mesmo e utilize o comando CREATE TABLE para definir o esquema de **uma única tabela** capaz de armazenar os dados desta pesquisa.

1.1 Crie a tabela de acordo com as suas respostas abaixo:

- a) Qual é a sua idade?
- b) Profissão? Estudante
- c) Qual Unidade Federativa você reside?
- d) Qual Município você reside?
- e) Qual operadora de telefonia celular você usa 1-Claro; 2-Vivo; 3 - Oi ?
- f) Qual tipo de contrato pós-pago (1) ou pré-pago (2)?
- g) Numa escala de zero a 10 o quão satisfeito(a) você está com o seu serviço da operadora?
- f) Identificador único para este registro é 10.

1.2 Faça a inserção da sua resposta na tabela. Utilize o comando INSERT do SQL.

Entregar feito a lápis na próxima aula os scripts de criação das tabelas e de inserção.

Exercício proposto 2:

Faça a importação de um arquivo no formato CSV.

domicilio;pessoa;idade;parentesco

1000;1;45;PAI

1000;2;10;FILHO

1000;3;35;ESPOSA

1000;4;15;FILHO

1000;5;70;SOGRA

i) Crie um banco de dados de nome Familia.db e construa uma tabela **CREATE TABLE Pessoa (familia int, hierarquia int, idade int, parentesco text);**

Para isso, crie um arquivo pessoa.csv pelo editor de bloco de notas do Windows;

ii) Copie os dados abaixo e cole neste arquivo e salve;

iii) Em vá no SQLite Studio e clique com o botão direito do mouse na tabela Pessoa.

iv) Selecione importar para tabela,

v) Veja que a tabela Pessoa está selecionada, clique em *Next* para avançar;

vii) Clique no ícone de pasta para selecionar o local onde você salvou o arquivo pessoa.csv;

viii) Selecione o arquivo;

ix) Clique em abrir, confirme na tela que o separador seja ponto e vírgula (mesmo usado na criação do CSV, clique em *Finish*).

x) Confira se a importação correu com sucesso, use para isso a consulta

```
SELECT * FROM PESSOA
```

Obrigado