**Geração de Scripts**

Com o Modelo Relacional completo e validado pelas regras de normalização, partimos para a próxima etapa de desenvolvimento, a formação dos *scripts* para construção do Banco de Dados.

Os scripts que estamos falando se tratam de comandos escritos em linguagem SQL (Structured Query Language) utilizados para dizer ao SGDB (Sistema Gerenciador de Banco de Dados) como criar e manipular o nosso banco de dados, suas tabelas e seus dados.

Para realizar esse processo se utilizamos de uma das funcionalidades da ferramenta Visual Paradigm, a geração de scripts de criação a partir de um modelo relacional já criado. Exportamos os comandos gerados para um bloco de notas e os copiamos para o DBeaver, ferramenta que utilizamos para manipular o Banco de Dados.

**Comandos de Criação do Banco de Dados**

Abaixo você pode conferir os comandos gerados para realizar a criação do Banco de Dados e suas devidas tabelas. Estes mesmos comandos podem ser encontrados em nosso repositório no GitHub junto dos comandos de documentação(comentários) criados para cada tabela e cada coluna.

create table ESTADO (

EST\_COD serial

constraint EST\_COD\_PK primary key,

EST\_SIGLA char(2) not null unique,

EST\_NOME varchar(25) not null

);

**--------------------------------------------------------------**

create table MUNICIPIO (

MUN\_COD serial

constraint MUN\_COD\_PK primary key,

MUN\_NOME varchar(40) not null,

COD\_ESTADO int4 not null

);

alter table MUNICIPIO

add constraint MUN\_COD\_ESTADO\_FK foreign key (COD\_ESTADO)

references ESTADO (EST\_COD);

**--------------------------------------------------------------**

create table CEP (

CEP\_COD varchar(9)

constraint CEP\_COD\_PK primary key,

COD\_MUNICIPIO int4 not null

);

alter table CEP

add constraint CEP\_COD\_MUNICIPIO\_FK foreign key (COD\_MUNICIPIO)

references MUNICIPIO (MUN\_COD);

**--------------------------------------------------------------**

create table CARGO (

CAR\_COD serial

constraint CAR\_COD\_PK primary key,

CAR\_DESCRICAO varchar(50) not null,

CAR\_SALARIO numeric(8, 2) not null

);

**--------------------------------------------------------------**

create table FUNCIONARIO (

FUN\_COD serial

constraint FUN\_COD\_PK primary key,

FUN\_NOME varchar(50) not null,

FUN\_CPF numeric(11, 0) not null unique,

FUN\_RG numeric(7, 0) not null unique,

FUN\_DATA\_NASCIMENTO date not null,

FUN\_GENERO char(1)

constraint FUN\_GENERO\_CK check (FUN\_GENERO in ('F', 'M')),

COD\_CARGO int4 not null

);

alter table FUNCIONARIO

add constraint FUN\_COD\_CARGO\_FK foreign key (COD\_CARGO)

references CARGO (CAR\_COD);

**--------------------------------------------------------------**

create table PESSOA (

PES\_COD serial

constraint PES\_COD\_PK primary key,

PES\_NOME varchar(50) not null,

PES\_TIPO char(1) not null

constraint PES\_TIPO\_CK check (PES\_TIPO in ('F', 'J')),

PES\_CPFCNPJ numeric(14, 0) not null unique,

PES\_IE varchar(9) unique,

PES\_GENERO char(1)

constraint PES\_GENERO\_CK check (PES\_GENERO in ('F', 'M')),

PES\_DATA\_NASCIMENTO date not null,

PES\_DATA\_CADASTRO date not null

);

**--------------------------------------------------------------**create table TIPO\_ENDERECO (

TIE\_COD serial

constraint TIE\_COD\_PK primary key,

TIE\_DESCRICAO varchar(25) not null

);

**--------------------------------------------------------------**

create table ENDERECO\_FUN (

ENF\_ID serial

constraint ENF\_ID\_PK primary key,

ENF\_TIP\_END int4 not null,

ENF\_FUNCIONARIO int4 not null,

constraint ENF\_TPE\_FUM\_UK Unique(ENF\_TIP\_END, ENF\_FUNCIONARIO),

ENF\_BAIRRO varchar(50),

ENF\_RUA varchar(50),

ENF\_NUMERO numeric(6, 0),

ENF\_TELEFONE varchar(9),

ENF\_CELULAR varchar(12),

ENF\_EMAIL varchar(40),

ENF\_CEP varchar(9) not null

);

alter table ENDERECO\_FUN

add constraint ENF\_TIP\_END\_FK foreign key (ENF\_TIP\_END)

references TIPO\_ENDERECO (TIE\_COD);

alter table ENDERECO\_FUN

add constraint ENF\_FUNCIONARIO\_FK foreign key (ENF\_FUNCIONARIO)

references FUNCIONARIO (FUN\_COD);

alter table ENDERECO\_FUN

add constraint ENF\_CEP\_FK foreign key (ENF\_CEP)

references CEP (CEP\_COD);

**--------------------------------------------------------------**

create table ENDERECO\_PES (

ENP\_ID serial

constraint ENP\_ID\_PK primary key,

ENP\_TIP\_END int4 not null,

ENP\_PESSOA int4 not null,

constraint ENP\_TPE\_PES\_UK Unique(ENP\_TIP\_END, ENP\_PESSOA),

ENP\_BAIRRO varchar(50),

ENP\_RUA varchar(50),

ENP\_NUMERO numeric(6, 0),

ENP\_TELEFONE varchar(9),

ENP\_CELULAR varchar(12),

ENP\_EMAIL varchar(40),

ENP\_CEP varchar(9) not null

);

alter table ENDERECO\_PES

add constraint ENP\_TIP\_END\_FK foreign key (ENP\_TIP\_END)

references TIPO\_ENDERECO (TIE\_COD);

alter table ENDERECO\_PES

add constraint ENP\_PESSOA\_FK foreign key (ENP\_PESSOA)

references PESSOA (PES\_COD);

alter table ENDERECO\_PES

add constraint ENP\_CEP\_FK foreign key (ENP\_CEP)

references CEP (CEP\_COD);

**--------------------------------------------------------------**create table PRINCIPIO\_ATIVO (

PIA\_COD serial

constraint PIA\_COD\_PK primary key,

PIA\_DESCRICAO varchar(50) not null

);

**--------------------------------------------------------------**

create table TIPO\_MEDICAMENTO (

TIM\_COD serial

constraint TIM\_COD\_PK primary key,

TIM\_DESCRICAO varchar(25) not null

);

**--------------------------------------------------------------**

create table TIPO\_TARJA (

TIT\_COD serial

constraint TIT\_COD\_PK primary key,

TIT\_DESCRICAO varchar(25) not null

);

**--------------------------------------------------------------**

create table UNIDADE\_MEDIDA (

UNM\_COD serial

constraint UNM\_COD\_PK primary key,

UNM\_DESCRICAO varchar(25) not null

);

**--------------------------------------------------------------**

create table PRODUTO (

PRO\_COD serial

constraint PRO\_COD\_PK primary key,

PRO\_NOME varchar(40) not null,

PRO\_DESCRICAO varchar(60),

PRO\_PRECO numeric(8, 2) not null,

PRO\_QUANTIDADE numeric(8, 2) not null,

PRO\_PESO numeric(7, 3),

PRO\_PRI\_ATIVO int4 not null,

PRO\_TIP\_MED int4 not null,

PRO\_TIP\_TAR int4 not null,

PRO\_UNI\_MED int4 not null,

PRO\_ESTOQUE\_MINIMO numeric(8, 2) not null

);

alter table PRODUTO

add constraint PRO\_PRI\_ATIVO\_FK foreign key (PRO\_PRI\_ATIVO)

references PRINCIPIO\_ATIVO (PIA\_COD);

alter table PRODUTO

add constraint PRO\_UNI\_MED\_FK foreign key (PRO\_UNI\_MED)

references UNIDADE\_MEDIDA (UNM\_COD);

alter table PRODUTO

add constraint PRO\_TIP\_TAR\_FK foreign key (PRO\_TIP\_TAR)

references TIPO\_TARJA (TIT\_COD);

alter table PRODUTO

add constraint PRO\_TIP\_MED\_FK foreign key (PRO\_TIP\_MED)

references TIPO\_MEDICAMENTO (TIM\_COD);

**--------------------------------------------------------------**

create table TIPO\_PAGAMENTO (

TPG\_COD serial

constraint TPG\_COD\_PK primary key,

TPG\_DESCRICAO varchar(40) not null

);

**--------------------------------------------------------------**

create table VENDA (

VEN\_NUMERO serial

constraint VEN\_NUMERO\_PK primary key,

COD\_PESSOA int4 not null,

VEN\_DATA timestamp not null,

COD\_PAGAMENTO int4 not null,

COD\_FUNCIONARIO int4 not null,

VEN\_RECEITA bytea

);

alter table VENDA

add constraint VEN\_COD\_PESSOA\_FK foreign key (COD\_PESSOA)

references PESSOA (PES\_COD);

alter table VENDA

add constraint VEN\_COD\_PAGAMENTO\_FK foreign key (COD\_PAGAMENTO)

references TIPO\_PAGAMENTO (TPG\_COD);

alter table VENDA

add constraint VEN\_COD\_FUNCIONARIO\_FK foreign key (COD\_FUNCIONARIO)

references FUNCIONARIO (FUN\_COD);

**--------------------------------------------------------------**

create table VENDA\_ITEM (

VEI\_ID serial

constraint VEI\_ID\_PK primary key,

VEI\_VEN\_NUMERO int4 not null,

VEI\_PRO\_COD int4 not null,

VEI\_QTD\_ITEM numeric(8, 2) not null,

VEI\_PRECO numeric(8, 2) not null,

VEI\_TOTAL\_ITEM numeric(8, 2) not null

);

alter table VENDA\_ITEM

add constraint VEI\_VEN\_NUMERO\_FK foreign key (VEI\_VEN\_NUMERO)

references VENDA (VEN\_NUMERO);

alter table VENDA\_ITEM

add constraint VEI\_PRO\_COD\_FK foreign key (VEI\_PRO\_COD)

references PRODUTO (PRO\_COD);

**--------------------------------------------------------------**

create table COMPRA (

COM\_NUMERO serial

constraint COM\_NUMERO\_PK primary key,

COD\_PESSOA int4 not null,

COM\_DATA timestamp not null,

COD\_PAGAMENTO int4 not null,

COD\_FUNCIONARIO int4 not null

);

alter table COMPRA

add constraint COM\_COD\_PESSOA\_FK foreign key (COD\_PESSOA)

references PESSOA (PES\_COD);

alter table COMPRA

add constraint COM\_COD\_PAGAMENTO\_FK foreign key (COD\_PAGAMENTO)

references TIPO\_PAGAMENTO (TPG\_COD);

alter table COMPRA

add constraint COM\_COD\_FUNCIONARIO\_FK foreign key (COD\_FUNCIONARIO)

references FUNCIONARIO (FUN\_COD);

**--------------------------------------------------------------**

create table COMPRA\_ITEM (

COI\_ID serial

constraint COI\_ID\_PK primary key,

COI\_COM\_NUMERO int4 not null,

COI\_PRO\_COD int4 not null,

COI\_QTD\_ITEM numeric(8, 2) not null,

COI\_UNITARIO\_ITEM numeric(8, 2) not null,

COI\_TOTAL\_ITEM numeric(8, 2) not null

);

alter table COMPRA\_ITEM

add constraint COI\_PRO\_COD\_FK foreign key (COI\_PRO\_COD)

references PRODUTO (PRO\_COD);

alter table COMPRA\_ITEM

add constraint COI\_COM\_NUMERO\_FK foreign key (COI\_COM\_NUMERO)

references COMPRA (COM\_NUMERO);

**--------------------------------------------------------------**

create table PRODUTO\_MOVIMENTO (

PRM\_ID serial

constraint PRM\_ID\_PK primary key,

PRM\_DATA timestamp not null,

PRM\_PRODUTO int4 not null,

PRM\_PESSOA int4 not null,

PRM\_QTD\_PRODUTO numeric(8, 2) not null

);

alter table PRODUTO\_MOVIMENTO add constraint PRM\_PESSOA\_FK foreign key (PRM\_PESSOA)

references PESSOA (PES\_COD);

alter table PRODUTO\_MOVIMENTO add constraint PRM\_PRODUTO\_FK foreign key (PRM\_PRODUTO)

references PRODUTO (PRO\_COD);

**--------------------------------------------------------------**

**Comandos de Inserção de Dados**

Após o Banco de Dados ter sido criado, precisamos alimentá-lo com informações. Para este processo nos utilizamos de um gerador de dados da internet (*https://generatedata.com/generator*) que através de parâmetros informados em seu site é capaz de gerar dados aleatoriamente.

Os dados gerados foram inseridos no Banco de Dados se utilizando dos comandos de SQL abaixo:

**COMANDOS**

**Comandos de Consulta (Relatórios)**

A linguagem SQL é muito poderosa quando o assunto é geração de relatórios, pois podemos nos utilizar de comandos de consulta que retornam os dados conforme nossas necessidades. Através dos comandos de consulta (Conhecidos como Select’s) podemos retornar os dados de acordo com certas condições a serem atendidas, ordenar, agrupar, contar e até ligar as tabelas entre si para retornar dados de outra tabelas.

Abaixo você confere 4 exemplos de comandos de consulta para geração de relatórios do Projeto Gaia:

* Relacionar o código, nome e tipo de todos os medicamentos. Ordene o relatório de forma ascendente pelo nome:
* Relacionar o nome do medicamento e o nome do fornecedor para todos os medicamentos.Filtre somente produtos com estoque maior que 10. Ordene o relatório de forma descendente pelo nome do fornecedor;
* Relacionar o código do medicamento, nome do medicamento, mês da venda, quantidade total de vendas no mês para meses entre anos de 2020 e 2021. Ordene o relatório do produto com mais vendas(em termos de quantidade) para o produto com menos vendas;
* Relacionar o cpf do cliente, nome do cliente e o total gasto em medicamentos. Filtrar somente clientes do sexo masculino, com idades entre 20 e 40 anos e que realizaram compras em meses pares de 2021. Ordene o relatório do cliente com mais gastos para o cliente com menos gastos.