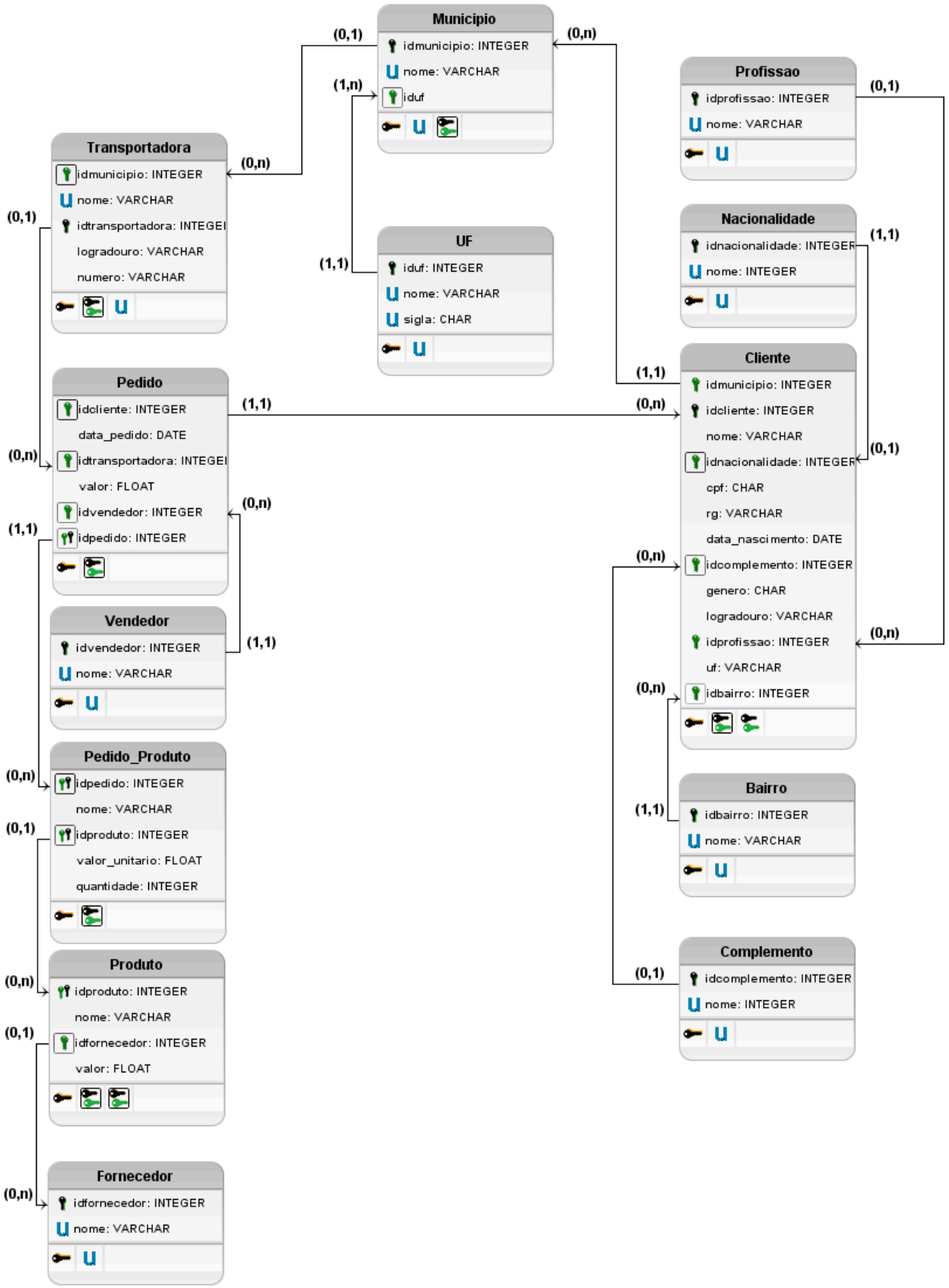
Nome: Gabriel Portelinha Rico
RA: 1136215
Data: 21/05/2025



ATENÇÃO: Não execute os comandos de consulta a partir da linha 630 durante a execução inicial do script.

```
/*********************************************
*   SCRIPT DE CRIAÇÃO DO PROJETO (PROVA PRÁTICA)
*
*   Descrição: [Projeto de banco de dados
*     relacional para gerenciar informações de
*     clientes, produtos, fornecedores e
*     vendas, com tabelas normalizadas e consultas
*     para análise comercial.]
*
*   Aluno: [Gabriel Portelinha Rico]
*   Data: [20/05/2025]
*   Versão: 1.0
*********************************************/

create table cliente (
      idcliente integer not null,
      nome varchar(50) not null,
      cpf char(11),
      rg varchar(15),
      data_nascimento date,
      genero char(1),
      profissao varchar(30),
      nacionalidade varchar(30),
      logradouro varchar(30),
      numero varchar(10),
      complemento varchar(30),
      bairro varchar(30),
      municipio varchar(30),
      uf varchar(30),
      observacoes text,

      -- primary key PK
      constraint pk_cln_idcliente primary key (idcliente)
);

-- Inserção de 15 regristros iniciais com perfis variados
insert into cliente (idcliente, nome, cpf, rg, data_nascimento, genero,
profissao, nacionalidade, logradouro, numero, complemento, bairro,
municipio, uf) values

-- Cliente 1: Estudante masculino com dados completos
(1, 'Manoel', '88828383821', '32323', '2001-01-30', 'M', 'Estudante',
'Brasileira', 'Rua Joaquim Nabuco', '23', 'Casa', 'Cidade Nova', 'Porto
```

```
Uniao', 'SC'),

-- Cliente 2: Engenheiro com endereço em apartamento
(2, 'Geraldo', '12343299929', '56565', '1987-01-04', 'M', 'Engenheiro',
'Brasileira', 'Rua das Limas', '200', 'Ap', 'Centro', 'Poro Uniao',
'SC'),

-- Cliente 3: Pedreiro com endereço abreviado
(3, 'Carlos', '87732323227', '55463', '1967-10-01', 'M', 'Pedreiro',
'Brasileira', 'Rua das Laranjeiras', '300', 'Apart.', 'Cto.',
'Canoinhas', 'SC'),

-- Clientes 4-15: Demais registros com variações de dados
(4, 'Adriana', '12321221222', '98777', '1989-09-10', 'F', 'Jornalista',
'Brasileira', 'Rua das Limas', '240', 'Casa', 'Sao Pedro', 'Porto
Vitoria', 'PR'),
(5, 'Amanda', '99982838828', '28382', '1991-03-04', 'F', 'Jorn.',
'Italiana', 'Av. Central', '100', NULL, 'Sao Pedro', 'General Carneiro',
'SP'),
(6, 'Angelo', '99982828181', '1323', '2000-01-01', 'M', 'Professor',
'Brasileiro', 'Av. Beira Mar', '300', NULL, 'Ctr.', 'Sao Paulo', 'SP'),
(7, 'Anderson', NULL, NULL, NULL, 'M', 'Prof.', 'Italiano', 'Av.
Brasil', '100', 'Apartamento', 'Santa Rosa', 'Rio de Janeiro', 'SP'),
(8, 'Camila', '9998282828', NULL, '2001-10-10', 'F', 'Professora',
'Norte americana', 'Rua Central', '4333', NULL, 'Centro', 'Uberlandia',
'MG'),
(9, 'Cristiano', NULL, NULL, NULL, 'M', 'Estudante', 'Alema', 'Rua do
Centro', '877', 'Casa', 'Centro', 'Porto Alegre', 'RS'),
(10, 'Fabricio', '82828282828', '32323', NULL, 'M', 'Estudante',
'Brasileiro', NULL, NULL, NULL, NULL, 'PU', 'SC'),
(11, 'Fernanda', NULL, NULL, NULL, 'F', NULL, 'Brasileira', NULL, NULL,
NULL, NULL, 'Porto Uniao', 'SC'),
(12, 'Gilmar', '88881818181', '888', '2000-02-10', 'M', 'Estud.', NULL,
'Rua das Laranjeiras', '200', NULL, 'C.Nova', 'Canoinhas', 'SC'),
(13, 'Diego', '1010191919', '111939', NULL, 'M', 'Professor', 'Alemao',
'Rua Central', '455', 'Casa', 'Cidade N.', 'Sao Paulo', 'SP'),
(14, 'Jeferson', NULL, NULL, '1983-07-01', 'M', NULL, 'Brasileiro',
NULL, NULL, NULL, NULL, 'Uniao da Vitoria', 'PR'),
(15, 'Jessica', NULL, NULL, NULL, 'F', 'Estudante', NULL, NULL, NULL,
NULL, NULL, 'Uniao da Vitoria', 'PR');

-- Verificação de dados cadastrados
-- select * from clientes

/*************************************************************
*   CONSULTAS BÁSICAS - TABELA CLIENTE
```

```
*   Descrição: Exemplos de consultas para extração de dados
*              com diferentes critérios de filtro
*********************************************************/

-- O nome, o gênero e a profissão de todos os clientes, ordenado pelo
nome em ordem decrescente
-- select nome, genero, profissao from cliente order by nome desc;

-- Os clientes que tenham a letra "R" no nome
-- select nome from cliente where nome ilike '%r%';

-- Os clientes que o nome inicia com a letra "C"
-- select nome from cliente where nome like 'C%';

-- Os clientes que o nome termina com a letra "A"
-- select nome from cliente where nome like '%a';

-- Os clientes que moram no bairro "Centro"
-- select nome, bairro from cliente where bairro = 'Centro' or bairro =
'Cto.' or bairro = 'Ctr.';

-- Os clientes que moram em complementos que iniciam com a letra "A"
-- select nome, complemento from cliente where complemento like 'A%';

-- Somente os clientes do sexo feminino
-- select nome, genero from cliente where genero like 'F';

-- Os clientes que não informaram o CPF
-- select nome, cpf from cliente where cpf is null;

-- Os clientes de nacionalidade "Brasileira"
-- select nome, nacionalidade from cliente where nacionalidade like
'Brasil%';

-- Os clientes que nasceram entre 01/01/2000 e 01/01/2002
-- select nome, data_nascimento from cliente where data_nascimento
between '01-01-2000' and '01-01-2002';

-- O nome do cliente e o logradouro, número, complemento, bairro,
município e UF concatenado de todos os clientes
-- select ' ' || nome || ' - ' || logradouro || ' - ' || numero || ' - '
|| complemento || ' - ' || bairro || ' - ' || municipio || ' - ' || uf
as "Dados do Cliente" from cliente;


/*****************************************************
```

```
*   ATUALIZAÇÕES E EXCLUSÕES - TABELA CLIENTE
*   Descrição: Exemplos de operações DML para manutenção
*              dos dados dos clientes
*****************************************************/

-- Atualização de cadastro do cliente Adriana para Adriano
-- select * from cliente;
update cliente set nome = 'Adriano', genero = 'M', numero = '241' where
idcliente = 4;

-- Inserção e exclusão de cliente
insert into cliente (idcliente, nome) values (16, 'Joao');
delete from cliente where idcliente = 16;

-- Inserção de novos clientes com diferentes níveis de completude
insert into cliente (idcliente, nome, cpf, rg, data_nascimento, genero,
profissao, municipio, uf)
values (16, 'Maicon', '12349596421', '1234', '1965-10-10', 'F',
'Empresario', 'Florianopolis', 'PR');
insert into cliente (idcliente, nome, genero, profissao, nacionalidade,
logradouro, numero, complemento, bairro, municipio, uf)
values (17, 'Getulio', 'F', 'Estudante', 'Brasileira', 'Rua Central',
'343', 'Apartamento', 'Centro', 'Curitiba', 'SC');
insert into cliente (idcliente, nome, genero, profissao, nacionalidade,
numero, complemento)
values (18, 'Sandra', 'M', 'Professor', 'Italiana', '12', 'Bloco A');

---- Atualização corretiva de dados

-- CPF para 45390569432, gênero para M, nacionalidade para Brasileira,
UF para SC
update cliente set cpf = '45390569432', genero = 'M', nacionalidade =
'Brasileira', uf = 'SC' where idcliente = 16;

-- Altera os dados do cliente Getúlio
-- Data de nascimento para 01/04/1978, gênero para M
update cliente set genero = 'M', data_nascimento = '1978-04-01' where
idcliente = 17;

-- Altera os dados da cliente Sandra
-- Genero para F, profissão para Professora, número para 123
update cliente set genero = 'F', profissao = 'Professora', numero =
'123' where idcliente = 18;

-- Apaga o cliente Maicon e Sandra
delete from cliente where idcliente = 16;
```

```sql
delete from cliente where idcliente = 18;


/*************************************************************
*   CRIAÇÃO DA TABELA PROFISSÃO
*   Descrição: Tabela auxiliar para padronização das
*              profissões dos clientes
*   Observações:
*   - Chave primária em idprofissao
*   - Nome da profissão deve ser único
*************************************************************/

create table profissao (
      -- Identificador único da profissão PK
      idprofissao integer not null,
      nome varchar(30) not null,

      -- Definição de constraints
      constraint pk_prf_idprofissao primary key (idprofissao),
      constraint un_prf_nome unique (nome)
);

-- Inserção inicial de profissões existentes na tabela Cliente
insert into profissao (idprofissao, nome) values
(1, 'Estudante'),
(2, 'Engenheiro'),
(3, 'Pedreiro'),
(4, 'Jornalista'),
(5, 'Professor');

-- Verificação dos dados de profissão
-- select * from profissao;
-- select profissao from cliente;


/*************************************************************
*   CRIAÇÃO DA TABELA NACIONALIDADE
*   Descrição: Tabela para cadastro padronizado das
*              nacionalidades dos clientes
*   Observações:
*   - Chave primária em idnacionalidade
*   - Nome da nacionalidade deve ser único
*   - Tabela usada para normalização de dados
*************************************************************/

create table nacionalidade (
```

```
      idnacionalidade integer not null,
      nome varchar(30) not null,

      constraint pk_ncn_idnacionalidade primary key (idnacionalidade),
      constraint un_ncn_nome unique (nome)
);

-- Inserção das nacionalidades existentes na tabela cliente
insert into nacionalidade (idnacionalidade, nome) values
(1, 'Brasileira'),
(2, 'Italiana'),
(3, 'Norte-americana'),
(4, 'Alema');

-- Verificação dos dados de nacionalidade
-- select * from nacionalidade;


/*********************************************************
*   CRIAÇÃO DA TABELA COMPLEMENTO
*   Descrição: Tipos de complementos de endereço
*   Observações:
*   - Normaliza os valores de complemento
*   - Evita variações de digitação
*********************************************************/

create table complemento (
      idcomplemento integer not null,
      nome varchar(30) not null,

      constraint pk_cpl_idcomplemento primary key (idcomplemento),
      constraint un_cpl_nome unique (nome)
);

-- Inserção inicial de complementos existentes na tabela cliente
insert into complemento (idcomplemento, nome) values
(1, 'Casa'),
(2, 'Apartamento');

-- Verificação dos dados de complemento
-- select * from complemento


/*********************************************************
*   CRIAÇÃO DA TABELA BAIRRO
*   Descrição: Cadastro de bairros para endereços
```

```
*   Observações:
*   - Padroniza os nomes de bairros
*   - Permite relacionamento com clientes
*************************************************************/

create table bairro (
      idbairro integer not null,
      nome varchar(30) not null,

      constraint pk_brr_idbairro primary key (idbairro),
      constraint un_brr_nome unique (nome)
);

-- Inserção inicial dos bairros existentes na tabela cliente
insert into bairro (idbairro, nome) values
(1, 'Cidade Nova'),
(2, 'Centro'),
(3, 'Sao Pedro'),
(4, 'Santa Rosa');

-- Verificação dos dados de bairro
--select * from bairro;


/*************************************************************
*   NORMALIZAÇÃO DA TABELA CLIENTE
*   Descrição: Substituição de colunas textuais por FK
*              para relacionamento com tabelas auxiliares
*************************************************************/

-- 1. ATUALIZAÇÃO DE PROFISSÃO
-- Remove a coluna antiga e adiciona a FK para profissão
alter table cliente drop profissao;
alter table cliente add idprofissao integer; --foreign key
alter table cliente add constraint fk_cln_idprofissao foreign key
(idprofissao) references profissao (idprofissao);

-- Atualiza os IDs de profissão conforme mapeamento:
-- Estudante(1), Engenheiro(2), Pedreiro(3), Jornalista(4), Professor(5)
update cliente set idprofissao = 1 where idcliente in (1, 9, 10, 12, 15,
17);
update cliente set idprofissao = 2 where idcliente = 2;
update cliente set idprofissao = 3 where idcliente = 3;
update cliente set idprofissao = 4 where idcliente in (4, 5);
update cliente set idprofissao = 5 where idcliente in (6, 7, 8, 13);
```

```sql
-- 2. ATUALIZAÇÃO DE NACIONALIDADE
-- Substitui a coluna texto por FK para nacionalidade
alter table cliente drop nacionalidade;
alter table cliente add idnacionalidade integer;
alter table cliente add constraint fk_cln_idnacionalidade foreign key
(idnacionalidade) references nacionalidade (idnacionalidade);

-- Atualiza os IDs de nacionalidade:
-- Brasileira(1), Italiana(2), Norte-americana(3), Alemã(4)
update cliente set idnacionalidade = 1 where idcliente in (1, 2, 3, 4,
6, 10, 11, 14);
update cliente set idnacionalidade = 2 where idcliente in (5, 7);
update cliente set idnacionalidade = 3 where idcliente = 8;
update cliente set idnacionalidade = 4 where idcliente in (9, 13);

-- 3. ATUALIZAÇÃO DE COMPLEMENTO
-- Substitui a coluna texto por FK para complemento
alter table cliente drop complemento;
alter table cliente add idcomplemento integer;
alter table cliente add constraint fk_cln_idcomplemento foreign key
(idcomplemento) references complemento (idcomplemento);

-- Atualiza os IDs de complemento:
-- Casa(1), Apartamento(2)
update cliente set idcomplemento = 1 where idcliente in (1, 4, 9, 13);
update cliente set idcomplemento = 2 where idcliente in (3, 2, 7);


-- 4. ATUALIZAÇÃO DE BAIRRO
-- Substitui a coluna texto por FK para bairro
alter table cliente drop bairro;
alter table cliente add idbairro integer;
alter table cliente add constraint fk_cln_idbairro foreign key
(idbairro) references bairro (idbairro);

-- Atualiza os IDs de bairro:
-- Cidade Nova(1), Centro(2), Sao Pedro(3), Santa Rosa(4)
update cliente set idbairro = 1 where idcliente in (1, 12, 13);
update cliente set idbairro = 2 where idcliente in (2, 3, 6, 8, 9);
update cliente set idbairro = 3 where idcliente in (4, 5);
update cliente set idbairro = 4 where idcliente = 7;

-- Verificação final da estrutura
-- select * from cliente;
```

```sql
/*********************************************************
*   CRIAÇÃO DA TABELA UF
*   Descrição: Cadastro de Unidades Federativas (Estados)
*   Observações:
*   - Armazena nome completo e sigla
*   - Restrições de unicidade em nome e sigla
*********************************************************/

create table uf (
      iduf integer not null,
      nome varchar(30) not null,
      sigla char(2) not null,

      constraint pk_ufd_idunidade_federacao primary key (iduf),
      constraint un_ufd_nome unique (nome),
      constraint un_ufd_sigla unique (sigla)
);

-- Inserção inicial dos estados existentes na tabela cliente
insert into uf (iduf, nome, sigla) values
(1, 'Santa Catarina', 'SC'),
(2, 'Parana', 'PR'),
(3, 'Sao Paulo', 'SP'),
(4, 'Minas Gerais', 'MG'),
(5, 'Rio Grande do Sul', 'RS'),
(6, 'Rio de Janeiro', 'RJ');

-- Verificação dos dados
-- select * from uf;


/*********************************************************
*   CRIAÇÃO DA TABELA MUNICÍPIO
*   Descrição: Cadastro de municípios com relacionamento
*              com a tabela UF
*   Observações:
*   - Chave estrangeira para UF
*   - Nome do município deve ser único
*********************************************************/

create table municipio (
      idmunicipio integer not null,
      nome varchar(30) not null,
      iduf integer not null,

      constraint pk_mnc_idmunicipio primary key (idmunicipio),
```

```
      constraint un_mnc_nome unique (nome),
      constraint fk_mnc_iduf foreign key (iduf) references uf (iduf)
);

-- Inserção inicial de municípios existentes na tabela cliente
insert into municipio (idmunicipio, nome, iduf) values
(1, 'Porto Uniao', 1),
(2, 'Canoinhas', 1),
(3, 'Porto Vitoria', 2),
(4, 'General Carneiro', 2),
(5, 'Sao Paulo', 3),
(6, 'Rio de Janeiro', 6),
(7, 'Uberlandia', 4),
(8, 'Porto Alegre', 5),
(9, 'Uniao da Vitoria', 2);

-- Verificação dos dados
-- select * from municipio;


/*************************************************************
*   ATUALIZAÇÃO DA TABELA CLIENTE - RELACIONAMENTO COM MUNICÍPIO
*   Descrição: Substituição das colunas municipio e uf
*              por relacionamento com tabela municipio
*   Observações:
*   - Remove colunas textuais antigas
*   - Adiciona FK para município (que já contém relação com UF)
*************************************************************/

-- 1. REMOÇÃO DAS COLUNAS ANTIGAS
alter table cliente drop municipio;
alter table cliente drop uf;

-- 2. ADIÇÃO DA NOVA COLUNA E CHAVE ESTRANGEIRA
alter table cliente add idmunicipio integer;
alter table cliente add constraint fk_cliente_idmunicipio foreign key
(idmunicipio) references municipio (idmunicipio);

-- 3. ATUALIZAÇÃO DOS MUNICÍPIOS DOS CLIENTES
update cliente set idmunicipio = 1 where idcliente in (1, 2, 10, 11);
update cliente set idmunicipio = 2 where idcliente in (3, 13);
update cliente set idmunicipio = 3 where idcliente in (4);
update cliente set idmunicipio = 4 where idcliente in (5);
update cliente set idmunicipio = 5 where idcliente in (6, 13);
update cliente set idmunicipio = 6 where idcliente in (7);
update cliente set idmunicipio = 7 where idcliente in (8);
```

```sql
update cliente set idmunicipio = 8 where idcliente in (9);
update cliente set idmunicipio = 9 where idcliente in (14, 15);


/*********************************************************
*   CRIAÇÃO DA TABELA FORNECEDOR
*   Descrição: Cadastro de empresas fornecedoras de produtos
*   Observações:
*   - Nome do fornecedor deve ser único
*   - Tabela básica para relacionamento com produtos
*********************************************************/

create table fornecedor (
      idfornecedor integer not null,
      nome varchar(50) not null,

      constraint pk_frn_idfornecedor primary key (idfornecedor),
      constraint un_frn_nome unique (nome)
);

-- Inserção inicial de fornecedores
insert into fornecedor (idfornecedor, nome) values
(1, 'Cap. Computadores'),
(2, 'AA. Computadores'),
(3, 'BB. Maquinas');

-- Verificação dos dados
-- select * from fornecedor;


/*********************************************************
*   CRIAÇÃO DA TABELA VENDEDOR
*   Descrição: Cadastro de vendedores/representantes
*   Observações:
*   - Nome do vendedor deve ser único
*   - Tabela independente para flexibilidade
*********************************************************/

create table vendedor (
      idvendedor integer not null,
      nome varchar(50) not null,

      constraint pk_vnd_idvendedor primary key (idvendedor),
      constraint un_vnd_nome unique (nome)
);
```

```sql
-- Inserção inicial de vendedores
insert into vendedor (idvendedor, nome) values
(1, 'Andre'),
(2, 'Alisson'),
(3, 'Jose'),
(4, 'Ailton'),
(5, 'Maria'),
(6, 'Suelem'),
(7, 'Aline'),
(8, 'Silvana');

-- Verificação dos dados
-- select * from vendedor;


/**********************************************************
*   CRIAÇÃO DA TABELA TRANSPORTADORA
*   Descrição: Cadastro de empresas de transporte/logística
*   Observações:
*   - Relacionamento com município para endereço
*   - Nome da transportadora deve ser único
**********************************************************/

create table transportadora (
      idtransportadora integer not null,
      idmunicipio integer,
      nome varchar(50) not null,
      logradouro varchar(50),
      numero varchar(10),

      constraint pk_trn_idtransportadora primary key (idtransportadora),
      constraint fk_trn_idmunicipio foreign key (idmunicipio) references
municipio (idmunicipio),
      constraint un_trn_nome unique (nome)
);

-- Inserção inicial de transportadoras
insert into transportadora (idtransportadora, idmunicipio, nome,
logradouro, numero) values
(1, 9, 'BS. Transportes', 'Rua das Limas', '01'),
(2, 5, 'Uniao Transportes', null, null);

-- Verificação dos dados
-- select * from transportadora
```

```sql
/*********************************************************
*   CRIAÇÃO DA TABELA PRODUTO
*   Descrição: Cadastro de itens comercializados
*   Observações:
*   - Relacionamento obrigatório com fornecedor
*   - Valor deve ser sempre positivo
*********************************************************/

create table produto (
      idproduto integer not null,
      idfornecedor integer not null,
      nome varchar(50) not null,
      valor float not null,

      constraint pk_prd_idproduto primary key (idproduto),
      constraint fk_prd_idfornecedor foreign key (idfornecedor)
references fornecedor (idfornecedor)
);

-- Inserção inicial de produtos
insert into produto (idproduto, idfornecedor, nome, valor) values
(1, 1, 'Microcomputador', 800),
(2, 1, 'Monitor', 500),
(3, 2, 'Placa mae', 200),
(4, 2, 'HD', 150),
(5, 2, 'Placa de video', 200),
(6, 3, 'Memoria RAM', 100),
(7, 1, 'Gabinete', 35);

-- Verificação dos dados
-- select * from produto;


/*********************************************************
*   CRIAÇÃO DA TABELA PEDIDO
*   Descrição: Registro de vendas/pedidos dos clientes
*   Observações:
*   - Relacionamentos obrigatórios com cliente e vendedor
*   - Transportadora pode ser nula (retirada no local)
*   - Data e valor são obrigatórios
*********************************************************/

create table pedido (
      idpedido integer not null,
      idcliente integer not null,
      idtransportadora integer,
```

```
      idvendedor integer not null,
      data_pedido date not null,
      valor float not null,

      constraint pk_pdd_idpedido primary key (idpedido),
      constraint fk_pdd_idcliente foreign key (idcliente) references
cliente (idcliente),
      constraint fk_pdd_idtransportadora foreign key (idtransportadora)
references transportadora (idtransportadora),
      constraint fk_pdd_idvendedor foreign key (idvendedor) references
vendedor (idvendedor)
);

-- Inserção inicial de pedidos
insert into pedido (idpedido, data_pedido, valor, idcliente,
idtransportadora, idvendedor) values
(1, '2008-04-01', 1300, 1, 1, 1),
(2, '2008-04-01', 500, 1, 1, 1),
(3, '2008-04-02', 300, 11, 2, 5),
(4, '2008-04-05', 1000, 8, 1, 7),
(5, '2008-04-06', 200, 9, 2, 6),
(6, '2008-04-06', 1985, 10, 1, 6),
(7, '2008-04-06', 800, 3, 1, 7),
(8, '2008-04-06', 175, 3, null, 7),
(9, '2008-04-07', 1300, 12, null, 8),
(10, '2008-04-10', 200, 6, 1, 8),
(11, '2008-04-15', 300, 15, 2, 1),
(12, '2008-04-20', 300, 15, 2, 5),
(13, '2008-04-20', 350, 9, 1, 7),
(14, '2008-04-23', 300, 2, 1, 5),
(15, '2008-04-25', 200, 11, null, 5);

-- Verificação dos pedidos
-- select * from pedido;


/*********************************************************
*   CRIAÇÃO DA TABELA PEDIDO_PRODUTO
*   Descrição: Itens que compõem cada pedido (tabela associativa)
*   Observações:
*   - Chave primária composta (idpedido + idproduto)
*   - Quantidade e valor unitário são obrigatórios
*   - Permite análise detalhada por produto
*********************************************************/

create table pedido_produto (
```

```
	idpedido integer not null,
	idproduto integer not null,
	quantidade integer not null,
	valor_unitario float not null,

	constraint pk_pdp_idpedidoproduto primary key (idpedido,
idproduto),
	constraint fk_pdp_idpedido foreign key (idpedido) references
pedido (idpedido),
	constraint fk_pdp_idproduto foreign key (idproduto) references
produto (idproduto)
);

--select * from produto
--select * from pedido

-- Inserção inicial dos itens dos pedidos
insert into pedido_produto (idpedido, idproduto, quantidade,
valor_unitario) values
(1, 1, 1, 800),
(1, 2, 1, 500),
(2, 2, 1, 500),
(3, 4, 2, 150),
(4, 1, 1, 800),
(4, 3, 1, 200),
(5, 3, 1, 200),
(6, 1, 2, 800),
(6, 7, 1, 35),
(6, 5, 1, 200),
(7, 1, 1, 800),
(8, 7, 5, 35),
(9, 1, 1, 800),
(9, 2, 1, 500),
(10, 5, 1, 200),
(11, 5, 1, 200),
(11, 6, 1, 100),
(12, 2, 1, 500),
(13, 3, 1, 200),
(13, 4, 1, 150),
(14, 6, 3, 100),
(15, 3, 1, 200);

-- Verificação dos itens
-- select * from pedido_produto;
```

```sql
/*********************************************************
*   CONSULTAS COM JOIN (RELACIONAMENTO ENTRE TABELAS)
*********************************************************/

-- Consulta completa de dados de clientes com informações relacionadas
-- Retorna: nome, profissão, nacionalidade, endereço completo e dados
pessoais
select
      cln.nome as cliente,
      prf.nome as profissao,
      ncn.nome as nacionalidade,
      cmp.nome as complemento,
      mnc.nome as municipio,
      uf.nome as unidade_federacao,
      brr.nome as bairro,
      cln.rg,
      cln.cpf,
      cln.data_nascimento,
      cln.genero,
      cln.logradouro,
      cln.numero
from
      cliente cln
left outer join
      profissao prf on cln.idprofissao = prf.idprofissao
left outer join
      nacionalidade ncn on cln.idnacionalidade = ncn.idnacionalidade
left outer join
      complemento cmp on cln.idcomplemento = cmp.idcomplemento
left outer join
      municipio mnc on cln.idmunicipio = mnc.idmunicipio
left outer join
      uf on mnc.iduf = uf.iduf
left outer join
      bairro brr on cln.idbairro = brr.idbairro


/*********************************************************
*   CONSULTAS COM GROUP BY (AGRUPAMENTO DE DADOS)
*********************************************************/

-- Lista clientes e datas de pedido, incluindo quem não fez pedidos
-- Ordenado alfabeticamente pelo nome do cliente
select
      cln.nome,
      pdd.data_pedido
```

```
from
      cliente cln
left outer join
      pedido pdd on pdd.idcliente = cln.idcliente
order by
      cln.nome


/*********************************************************
*   FUNÇÕES AGREGADAS (CÁLCULOS SOBRE CONJUNTOS)
*********************************************************/

-- 1. Média de vendas por vendedor (acima de R$300)
select  idvendedor, avg(valor) from pedido group by idvendedor having
avg(valor) > 300

-- 2. Total de vendas por vendedor
select idvendedor, sum(valor) from pedido group by idvendedor

-- 3. Total de municípios cadastrados
select count (idmunicipio) from municipio

-- 4. Municípios por UF
select count (idmunicipio) from municipio where iduf = 1 or iduf = 2

-- 5. Média de preços do fornecedor Cap.Computadores
select avg(valor) from produto where idfornecedor = 1


/*********************************************************
*   VIEW CLIENTE_DADOS
*   Descrição: Visão consolidada de todos os dados do cliente
*   Observações:
*   - Formata gênero para texto completo
*   - Inclui todas as informações relacionadas
*********************************************************/

-- Cria uma view consolidada com todos os dados do cliente
-- Inclui formatação de gênero (M/F para Masculino/Feminino)
create view cliente_dados as
select
      cln.nome as cliente,
      prf.nome as profissao,
      ncn.nome as nacionalidade,
      cmp.nome as complemento,
      mnc.nome as municipio,
```

```
      uf.nome as unidade_federacao,
      brr.nome as bairro,
      cln.rg,
      cln.cpf,
      cln.data_nascimento,
      case cln.genero
            when 'M' then 'Masculino'
            when 'F' then 'Feminino'
      end as genero,
      cln.logradouro,
      cln.numero
from
      cliente cln
left outer join
      profissao prf on cln.idprofissao = prf.idprofissao
left outer join
      nacionalidade ncn on cln.idnacionalidade = ncn.idnacionalidade
left outer join
      complemento cmp on cln.idcomplemento = cmp.idcomplemento
left outer join
      municipio mnc on cln.idmunicipio = mnc.idmunicipio
left outer join
      uf on mnc.iduf = uf.iduf
left outer join
      bairro brr on cln.idbairro = brr.idbairro

-- Consulta da view
--select * from cliente_dados;
```