



UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
CIN0137 BANCO DE DADOS

ANITA MONTEIRO DE SIQUEIRA PEREIRA BEZERRA
ELLIAN DOS SANTOS RODRIGUES
GABRIELA TAINA COSTA BRITO
MICHELLY EMANUELA DA SILVA
SANDRÍRAMES ALBINO FAUSTO
THALES VINICIUS GOMES FRAGA
YUUMI KUNII

SISTEMA DE GERENCIAMENTO DE CINEMA

Recife-PE
2025

ANITA MONTEIRO DE SIQUEIRA PEREIRA BEZERRA
ELLIAN DOS SANTOS RODRIGUES
GABRIELA TAINA COSTA BRITO
MICHELLY EMANUELA DA SILVA
SANDRÍRAMES ALBINO FAUSTO
THALES VINICIUS GOMES FRAGA
YUUMI KUNII

SISTEMA DE GERENCIAMENTO DE CINEMA

Projeto para a disciplina de Banco de Dados do curso de
Bacharelado em Sistemas de Informação do Centro de
Informática da Universidade Federal de Pernambuco.

Prof. Dr. Flavio Arthur Oliveira Santos .

Sumário

	Páginas
1 Minimundo	4
2 Entidade Relacionamento Estendido (EER)	5
2.1 Especialização da Entidade Pessoa	5
2.2 Especialização da Entidade Funcionário	5
2.3 Especialização, Relacionamentos e Cardinalidades	6
3 Modelo Relacional	7
3.1 Geração das Tabelas a partir do Modelo EER	7
4 Script SQL	7
4.1 Criação das Tabelas	7
4.2 Povoamento do Banco	11
4.3 Consultas	15

1 Minimundo

Um cinema é composto por diversas salas, cada uma contendo um número de identificação, capacidade e um tipo de tela. As salas são utilizadas para a realização das sessões, que são dependentes de um filme, uma sala e um horário, além de possuírem um idioma e uma identificação. Os filmes exibidos em uma sessão contêm uma identificação, diretor, gênero, título, duração e classificação.

A entidade Pessoa representa as pessoas que interagem com o cinema. Cada pessoa cadastrada possui CPF, nome, celular e endereço, podendo ser um cliente ou um funcionário. Os clientes compram produtos e também assistem às sessões. Os funcionários recebem um salário e se dividem em vendedores, que recebem comissão, e zeladores, que possuem turno de trabalho.

Os vendedores realizam vendas de produtos, que são a união de comida e ingressos. As comidas possuem identificação, nome e preço, enquanto os ingressos possuem identificação da sessão, poltrona, nome, preço e tipo. Os zeladores, por sua vez, são responsáveis pela limpeza das salas.

Algumas justificativas:

A entidade Sessão é considerada fraca, pois só existe em função de um filme exibido em uma sala em determinado horário.

Para unificar o processo de venda, definimos a entidade Produto, que é a união de Ingresso e Comida. Dessa forma, tanto ingressos (vendidos na bilheteria) quanto comidas (como pipoca e refrigerantes, vendidas na lanchonete) são tratados de forma integrada.

2 Entidade Relacionamento Estendido (EER)

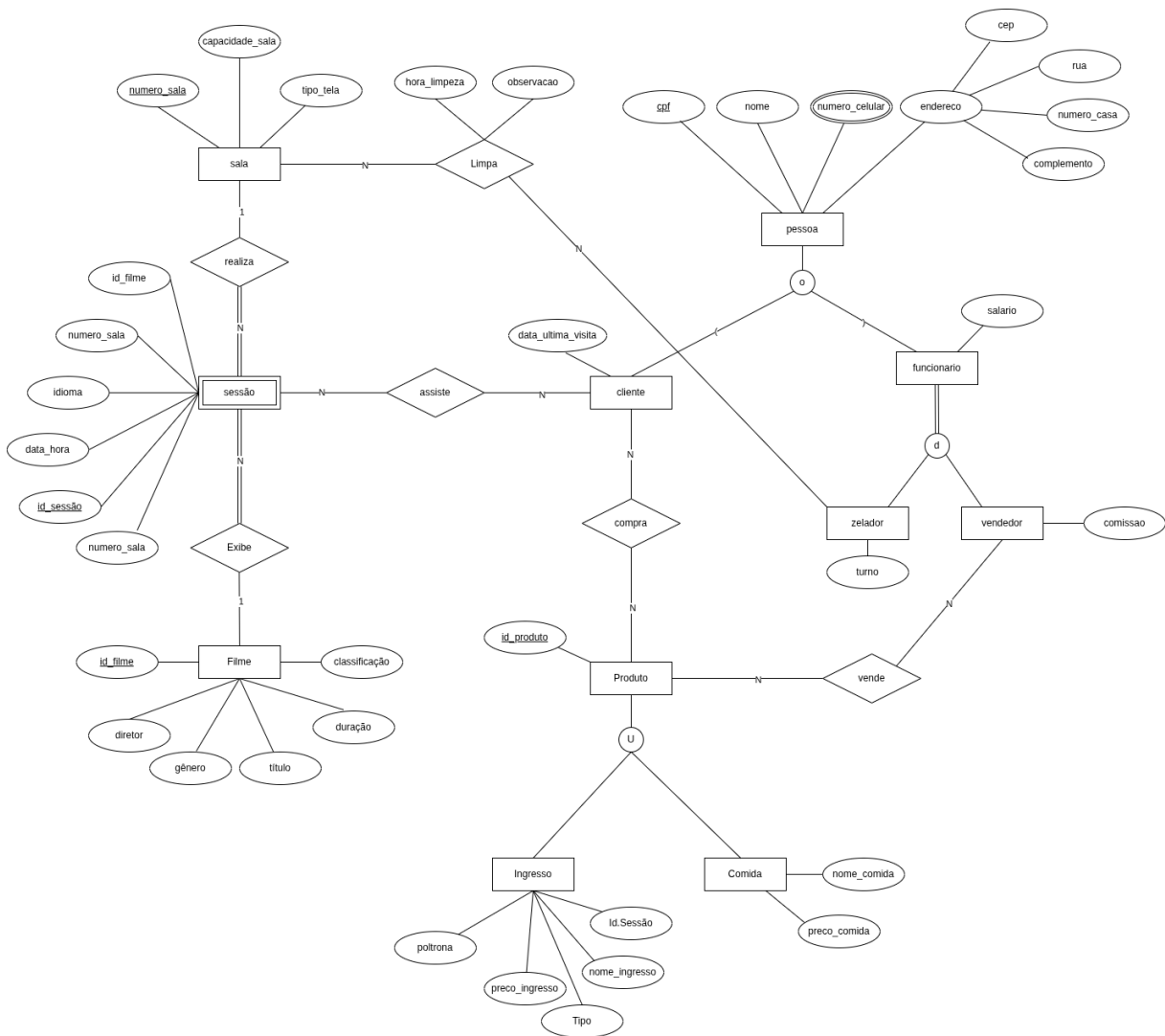


Figura 1. Diagrama EER

2.1 Especialização da Entidade Pessoa

A entidade **Pessoa** é modelada como uma superclasse, contendo atributos simples, compostos e multivalorados. A especialização de Pessoa origina as subclasses **Cliente** e **Funcionário**. Como uma pessoa pode ser simultaneamente cliente e funcionário, a especialização é do tipo **sobreposta** (*overlap*), sendo representada pelo símbolo **O** no diagrama.

2.2 Especialização da Entidade Funcionário

A entidade **Funcionário** é especializada em **Zelador** e **Vendedor**. Nesse caso, cada funcionário precisa exercer apenas uma das funções, mas não ambas simultaneamente. Assim, essa especialização é **disjunta** (*disjoint*), indicada pelo símbolo **D** e pelo uso de **duas linhas** no conector de especialização.

2.3 Especialização, Relacionamentos e Cardinalidades

Superclasse	Subclasses	Tipo	Descrição
Pessoa	Cliente, Funcionário	Overlap (O)	Uma pessoa pode ser cliente, funcionário ou ambos simultaneamente.
Funcionário	Zelador, Vendedor	Disjoint (D)	Um funcionário deve exercer apenas uma função.

Tabela 1. Especializações do Modelo EER

Relacionamento	Entidades Envolvidas	Cardinalidade	Descrição
Vende	Vendedor – Produto	N:N	Um vendedor pode vender vários produtos, e um produto pode ser vendido por vários vendedores.
Compra	Cliente – Produto	N:N	Um cliente pode comprar vários produtos, e um produto pode ser comprado por vários clientes.
Assiste	Cliente – Sessão	N:N	Um cliente pode assistir várias sessões, e uma sessão pode ser assistida por vários clientes.
Limpa	Zelador – Sala	N:N	Um zelador pode limpar várias salas, e uma sala pode ser limpa por vários zeladores.
Realiza	Sala – Sessão	1:N	Uma sessão ocorre em apenas uma sala, mas uma sala pode realizar várias sessões.
Exibe	Filme – Sessão	1:N	Uma sessão exibe apenas um filme, enquanto um filme pode ser exibido em várias sessões.

Tabela 2. Relacionamentos do Modelo EER e suas Cardinalidades

3 Modelo Relacional

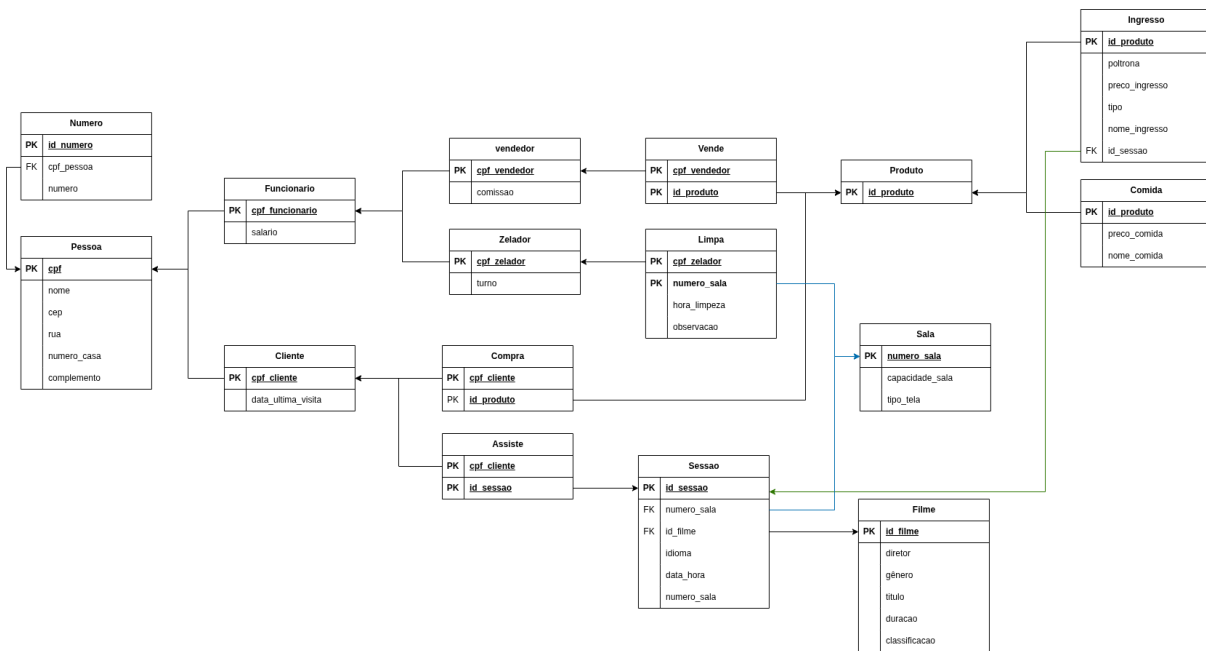


Figura 2. Modelo Relacional

3.1 Geração das Tabelas a partir do Modelo EER

Como apresentado na Figura, todas as tabelas do modelo relacional foram derivadas diretamente a partir do diagrama EER. As chaves primárias são identificadas em negrito acompanhadas da sigla PK, enquanto as chaves estrangeiras estão marcadas com FK, indicando suas referências para outras tabelas.

Além disso, as setas presentes no diagrama relacional ilustram o processo de propagação de chaves. Esse comportamento pode ser observado nas entidades Funcionário e Cliente, que herdam o atributo CPF da entidade Pessoa, já que resultam de uma especialização do tipo overlap. Dessa forma, ambas mantêm a mesma chave primária que identifica a pessoa correspondente. Outro aspecto derivado do EER é o tratamento do atributo multivalorado Número associado à entidade Pessoa. Por se tratar de um atributo capaz de assumir múltiplos valores, ele não pode ser armazenado diretamente como uma única coluna. Assim, criou-se uma tabela específica para esse atributo.

4 Script SQL

4.1 Criação das Tabelas

-- Tabela Pessoa

```
CREATE TABLE Pessoa (  
    cpf CHAR(11) PRIMARY KEY,  
    nome VARCHAR(100) NOT NULL,  
    cep CHAR(8) ,  
    rua VARCHAR(150) ,  
    numero_casa INTEGER,  
    complemento VARCHAR(50)  
);
```

-- Tabela Numero (Telefones)

```
CREATE TABLE Numero (  
    id_numero SERIAL PRIMARY KEY,  
    cpf_pessoa CHAR(11) NOT NULL,  
    numero VARCHAR(15) NOT NULL,  
    FOREIGN KEY (cpf_pessoa) REFERENCES Pessoa(cpf)  
);
```

-- Tabela Sala

```
CREATE TABLE Sala (  
    numero_sala INTEGER PRIMARY KEY,  
    capacidade_sala INTEGER NOT NULL CHECK (capacidade_sala >  
        0) ,  
    tipo_sala VARCHAR(50)  
);
```

-- Tabela Filme

```
CREATE TABLE Filme (  
    id_filme SERIAL PRIMARY KEY,  
    diretor VARCHAR(100) ,  
    genero VARCHAR(50) ,  
    titulo VARCHAR(150) NOT NULL,  
    duracao INTEGER CHECK (duracao > 0) ,  
    classificacao VARCHAR(20)  
);
```

-- Tabela Produto (Super-tipo para Ingresso e Comida)

```
CREATE TABLE Produto (  
    id_produto SERIAL PRIMARY KEY
```



```

);

-- Tabela Funcionario (Herda de Pessoa)
CREATE TABLE Funcionario (
    cpf_funcionario CHAR(11) PRIMARY KEY,
    salario NUMERIC(10, 2) NOT NULL CHECK (salario >= 0),
    FOREIGN KEY (cpf_funcionario) REFERENCES Pessoa(cpf)
);

-- Tabela Cliente (Herda de Pessoa)
CREATE TABLE Cliente (
    cpf_cliente CHAR(11) PRIMARY KEY,
    data_ultima_visita DATE,
    FOREIGN KEY (cpf_cliente) REFERENCES Pessoa(cpf)
);

-- Tabela Vendedor (Herda de Funcionario)
CREATE TABLE Vendedor (
    cpf_vendedor CHAR(11) PRIMARY KEY,
    comissao NUMERIC(5, 2) CHECK (comissao >= 0 AND comissao <=
        100),
    FOREIGN KEY (cpf_vendedor) REFERENCES Funcionario(
        cpf_funcionario)
);

-- Tabela Zelador (Herda de Funcionario)
CREATE TABLE Zelador (
    cpf_zelador CHAR(11) PRIMARY KEY,
    turno VARCHAR(20) NOT NULL,
    FOREIGN KEY (cpf_zelador) REFERENCES Funcionario(
        cpf_funcionario)
);

-- Tabela Sessao
CREATE TABLE Sessao (
    id_sessao SERIAL PRIMARY KEY,
    id_filme INTEGER NOT NULL,
    idioma VARCHAR(50),
    data_hora TIMESTAMP NOT NULL,

```

```

        numero_sala INTEGER NOT NULL,
        FOREIGN KEY (id_filme) REFERENCES Filme(id_filme),
        FOREIGN KEY (numero_sala) REFERENCES Sala(numero_sala)
    );

-- Tabela Ingresso (Herda de Produto)
CREATE TABLE Ingresso (
    id_produto INTEGER PRIMARY KEY,
    poltrona VARCHAR(10),
    preco_ingresso NUMERIC(10, 2) NOT NULL CHECK (
        preco_ingresso >= 0),
    tipo VARCHAR(50),
    nome_ingresso VARCHAR(100),
    id_sessao INTEGER NOT NULL,
    FOREIGN KEY (id_produto) REFERENCES Produto(id_produto),
    FOREIGN KEY (id_sessao) REFERENCES Sessao(id_sessao)
);

-- Tabela Comida (Herda de Produto)
CREATE TABLE Comida (
    id_produto INTEGER PRIMARY KEY,
    preco_comida NUMERIC(10, 2) NOT NULL CHECK (preco_comida >=
        0),
    nome_comida VARCHAR(100),
    FOREIGN KEY (id_produto) REFERENCES Produto(id_produto)
);

-- Tabela Vende (Vendedor – Produto)
CREATE TABLE Vende (
    cpf_vendedor CHAR(11) NOT NULL,
    id_produto INTEGER NOT NULL,
    PRIMARY KEY (cpf_vendedor, id_produto),
    FOREIGN KEY (cpf_vendedor) REFERENCES Vendedor(cpf_vendedor
        ),
    FOREIGN KEY (id_produto) REFERENCES Produto(id_produto)
);

-- Tabela Compra (Cliente – Produto)

```

```

CREATE TABLE Compra (
    cpf_cliente CHAR(11) NOT NULL,
    id_produto INTEGER NOT NULL,
    PRIMARY KEY (cpf_cliente , id_produto),
    FOREIGN KEY (cpf_cliente) REFERENCES Cliente(cpf_cliente),
    FOREIGN KEY (id_produto) REFERENCES Produto(id_produto)
);

-- Tabela Assiste (Cliente - Sessao)
CREATE TABLE Assiste (
    cpf_cliente CHAR(11) NOT NULL,
    id_sessao INTEGER NOT NULL,
    PRIMARY KEY (cpf_cliente , id_sessao),
    FOREIGN KEY (cpf_cliente) REFERENCES Cliente(cpf_cliente),
    FOREIGN KEY (id_sessao) REFERENCES Sessao(id_sessao)
);

-- Tabela Limpa (Zelador - Sala)
CREATE TABLE Limpa (
    cpf_zelador CHAR(11) NOT NULL,
    numero_sala INTEGER NOT NULL,
    hora_limpeza TIME NOT NULL,
    observacao TEXT,
    PRIMARY KEY (cpf_zelador , numero_sala),
    FOREIGN KEY (cpf_zelador) REFERENCES Zelador(cpf_zelador),
    FOREIGN KEY (numero_sala) REFERENCES Sala(numero_sala)
);

```

4.2 Povoamento do Banco

```

-- 1. Inserindo Pessoas
INSERT INTO Pessoa (cpf , nome , cep , rua , numero_casa ,
    complemento) VALUES
('11111111111', 'Ana Silva', '50000001', 'Rua das Flores', 101,
    'Apt 1'),
('22222222222', 'Bruno Souza', '50000002', 'Av. Brasil', 200,
    NULL),
('33333333333', 'Carla Dias', '50000003', 'Rua do Sol', 35, '

```

```

    Casa B'),
('444444444444', 'Daniel Lima', '50000004', 'Av. Boa Viagem',
    1500, 'Cob 01'),
('555555555555', 'Eduardo Melo', '51000001', 'Rua A', 10, NULL),
    -- Vendedor
('666666666666', 'Fernanda Costa', '51000002', 'Rua B', 20, NULL
    ), -- Vendedor
('777777777777', 'Gabriel Rocha', '51000003', 'Rua C', 30, NULL)
    , -- Vendedor
('888888888888', 'Helena Alves', '51000004', 'Rua D', 40, NULL),
    -- Vendedor
('999999999999', 'Igor Santos', '52000001', 'Rua X', 5, NULL),
    -- Zelador
('101010101010', 'Julia Moraes', '52000002', 'Rua Y', 6, NULL),
    -- Zelador
('121212121212', 'Kleber Pires', '52000003', 'Rua Z', 7, NULL),
    -- Zelador
('131313131313', 'Laura Nunes', '52000004', 'Rua W', 8, NULL);
    -- Zelador

```

-- 2. Inserindo Telefones

```

INSERT INTO Numero (cpf_pessoa, numero) VALUES
('111111111111', '81999991111'),
('222222222222', '81999992222'),
('333333333333', '81999993333'),
('444444444444', '81999994444');

```

-- 3. Inserindo Salas

```

INSERT INTO Sala (numero_sala, capacidade_sala, tipo_sala)
VALUES
(1, 100, '2D Padr o'),
(2, 150, '3D IMAX'),
(3, 80, 'VIP'),
(4, 120, '4DX');

```

-- 4. Inserindo Filmes

```

INSERT INTO Filme (diretor, genero, titulo, duracao,
    classificacao) VALUES
('Christopher Nolan', 'Fic o', 'Interestelar', 169, '10 anos

```

```

    '),
('Greta Gerwig', 'Com dia', 'Barbie', 114, '12 anos'),
('Francis Ford Coppola', 'Drama', 'O Poderoso Chef o', 175,
'14 anos'),
('Hayao Miyazaki', 'Anima o', 'A Viagem de Chihiro', 125, '
Livre');

```

-- 5. Inserindo Produtos

```

INSERT INTO Produto DEFAULT VALUES; -- ID 1
INSERT INTO Produto DEFAULT VALUES; -- ID 2
INSERT INTO Produto DEFAULT VALUES; -- ID 3
INSERT INTO Produto DEFAULT VALUES; -- ID 4
INSERT INTO Produto DEFAULT VALUES; -- ID 5
INSERT INTO Produto DEFAULT VALUES; -- ID 6
INSERT INTO Produto DEFAULT VALUES; -- ID 7
INSERT INTO Produto DEFAULT VALUES; -- ID 8

```

-- 6. Inserindo Clientes

```

INSERT INTO Cliente (cpf_cliente, data_ultima_visita) VALUES
('111111111111', '2023-10-01'),
('222222222222', '2023-10-05'),
('333333333333', '2023-10-10'),
('444444444444', '2023-10-15');

```

-- 7. Inserindo Funcionarios

```

INSERT INTO Funcionario (cpf_funcionario, salario) VALUES
('555555555555', 2500.00),
('666666666666', 2500.00),
('777777777777', 2500.00),
('888888888888', 2600.00),
('999999999999', 1800.00),
('10101010101', 1800.00),
('12121212121', 1900.00),
('13131313131', 1800.00);

```

-- 8. Inserindo Comidas

```

INSERT INTO Comida (id_produto, preco_comida, nome_comida)
VALUES
(1, 20.00, 'Pipoca Grande'),

```

```

(2, 15.00, 'Refrigerante 500ml'),
(3, 12.00, 'Nachos'),
(4, 8.00, 'Chocolate');

-- 9. Inserindo Sessoes
INSERT INTO Sessao (id_filme, idioma, data_hora, numero_sala)
VALUES
(1, 'Legendado', '2023-11-01 14:00:00', 1),
(2, 'Dublado', '2023-11-01 16:00:00', 2),
(3, 'Legendado', '2023-11-01 19:00:00', 3),
(4, 'Dublado', '2023-11-01 20:00:00', 4);

-- 10. Inserindo Vendedores
INSERT INTO Vendedor (cpf_vendedor, comissao) VALUES
('555555555555', 5.00),
('666666666666', 5.00),
('777777777777', 4.50),
('888888888888', 6.00);

-- 11. Inserindo Zeladores
INSERT INTO Zelador (cpf_zelador, turno) VALUES
('999999999999', 'Manh '),
('101010101010', 'Tarde '),
('121212121212', 'Noite '),
('131313131313', 'Madrugada ');

-- 12. Inserindo Ingressos
INSERT INTO Ingresso (id_produto, poltrona, preco_ingresso,
    tipo, nome_ingresso, id_sessao) VALUES
(5, 'A10', 30.00, 'Inteira', 'Interestelar 14h', 1),
(6, 'B05', 15.00, 'Meia', 'Barbie 16h', 2),
(7, 'C12', 50.00, 'VIP', 'O Poderoso Chef o 19h', 3),
(8, 'D08', 40.00, 'Inteira', 'Chihiro 20h', 4);

-- 13. Inserindo Vende (Vendedor vende Produto)
INSERT INTO Vende (cpf_vendedor, id_produto) VALUES
('555555555555', 1), -- Eduardo vendeu Pipoca
('666666666666', 5), -- Fernanda vendeu Ingresso A10
('777777777777', 2), -- Gabriel vendeu Refri

```

```

('888888888888', 7); -- Helena vendeu Ingresso VIP

-- 14. Inserindo Compra (Cliente compra Produto)
INSERT INTO Compra (cpf_cliente, id_produto) VALUES
('111111111111', 1), -- Ana comprou Pipoca
('222222222222', 5), -- Bruno comprou Ingresso A10
('333333333333', 2), -- Carla comprou Refri
('444444444444', 7); -- Daniel comprou Ingresso VIP

-- 15. Inserindo Assiste (Cliente assiste Sessao)
INSERT INTO Assiste (cpf_cliente, id_sessao) VALUES
('111111111111', 1), -- Ana assiste Sessao 1
('222222222222', 2), -- Bruno assiste Sessao 2
('333333333333', 3), -- Carla assiste Sessao 3
('444444444444', 4); -- Daniel assiste Sessao 4

-- 16. Inserindo Limpa
INSERT INTO Limpa (cpf_zelador, numero_sala, hora_limpeza,
observacao) VALUES
('999999999999', 1, '13:00:00', 'Limpeza pr -sess o'),
('101010101010', 2, '15:30:00', 'Recolhimento de lixo'),
('121212121212', 3, '18:30:00', 'Limpeza completa'),
('131313131313', 4, '22:00:00', 'Manuten o de poltrona');

```

4.3 Consultas

```

-- Clientes que Assistiram Filmes
SELECT P.nome
FROM Pessoa P
WHERE P.cpf IN (
    SELECT A.cpf_cliente
    FROM Assiste A
    WHERE A.id_sessao IN (
        SELECT S.id_sessao
        FROM Sessao S
        JOIN Filme F ON S.id_filme = F.id_filme
    )
);

```

```
-- ou
select p.nome, f.titulo as titulo_filme
from pessoa as p
join assiste as a on p.cpf = a.cpf_cliente
join sessao as s on a.id_sessao = s.id_sessao
join filme as f on s.id_filme = f.id_filme;
```

```
-- Relat rio de Limpezas Noturnas por Sala
```

```
SELECT
    L.numero_sala ,
    COUNT(*) AS qtd_limpezas
FROM
    Limpa L
WHERE
    L.hora_limpeza >= '18:00:00'
GROUP BY
    L.numero_sala
HAVING
    COUNT(*) >= 1;
```

```
-- Relat rio de Desempenho dos Vendedores
```

```
SELECT
    P.nome ,
    COUNT(V.id_produto) AS qtd_vendas ,
    CASE
        WHEN COUNT(V.id_produto) >= 1 THEN 'Vendeu Algo'
        ELSE 'Nenhuma Venda'
    END AS status_desempenho
FROM
    Pessoa P
    JOIN Funcionario F ON P.cpf = F.cpf_funcionario
    JOIN Vendedor Ven ON F.cpf_funcionario = Ven.cpf_vendedor
    LEFT JOIN Vende V ON Ven.cpf_vendedor = V.cpf_vendedor
GROUP BY
    P.nome , P.cpf
```



```

ORDER BY
    qtd_vendas DESC;

-- Produtos N o Vendidos
SELECT
    Pr.id_produto ,
    COALESCE(C.nome_comida , I.nome_ingresso) AS nome_produto ,
    CASE
        WHEN C.id_produto IS NOT NULL THEN 'Comida'
        ELSE 'Ingresso'
    END AS tipo_produto
FROM
    Produto Pr
    LEFT JOIN Vende V ON Pr.id_produto = V.id_produto
    LEFT JOIN Comida C ON Pr.id_produto = C.id_produto
    LEFT JOIN Ingresso I ON Pr.id_produto = I.id_produto
WHERE
    V.cpf_vendedor IS NULL;

-- Faturamento por Sess o
WITH FaturamentoSessao AS (
    SELECT
        S.id_sessao ,
        F.titulo ,
        SUM(I.preco_ingresso) AS total_arrecadado
    FROM
        Sessao S
        JOIN Filme F ON S.id_filme = F.id_filme
        JOIN Ingresso I ON S.id_sessao = I.id_sessao
    GROUP BY
        S.id_sessao , F.titulo
)
SELECT
    titulo ,
    total_arrecadado
FROM

```

```
FaturamentoSessao
WHERE
    total_arrecadado > 0;
-- ou
select f.titulo , sum(i.preco_ingresso) as total_arrecadado
from filme as f
join sessao as s on s.id_filme = f.id_filme
join ingresso as i on i.id_sessao = s.id_sessao
group by f.titulo
order by total_arrecadado desc;
```