Universidade Católica de Brasília

Pró-Reitoria Acadêmica

Curso de Bacharelado em Engenharia de Software

Trabalho de Disciplina de Laboratório de Banco de Dados

Sistema para gestão de Software Houses

Autores(a): Pedro Henrique nunes, Pedro Paulo, Vinicius Girão, Winiston Alle e Yuri Natanael
Orientador: Prof. Jefferson Salomão Rodrigues

Brasília - DF
2025

Sistema de Gestão Comercial para Software Houses

1. Introdução

Este documento apresenta a modelagem de banco de dados para um Sistema de Gestão Comercial para Software Houses, contemplando: cadastro de clientes e projetos, gerenciamento de tarefas, controle de orçamentos e pagamentos, e os relacionamentos entre entidades.

São exibidos os diagramas conceitual, lógico e físico, bem como os scripts SQL que implementam o modelo.

2. Modelo Conceitual (DER)

O Diagrama Entidade-Relacionamento (DER) representa as principais entidades e seus relacionamentos.

[CLIENTE]—<possui>—[PROJETO]—<contem>—[TAREFA]

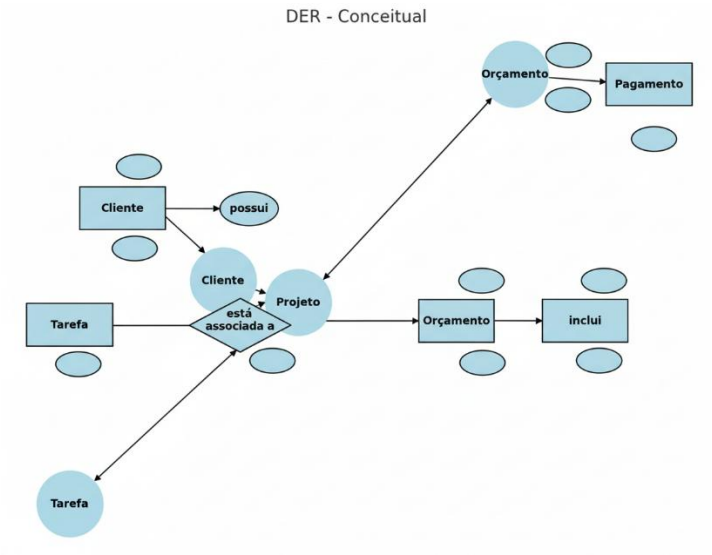
|

└—<gera>—[ORCAMENTO]—<possui>—[PAGAMENTO]

Legenda: O diagrama mostra as entidades **Cliente, Projeto, Tarefa, Orçamento e Pagamento**, com relacionamentos 1:N entre elas, conforme descrito no modelo conceitual.

Entidades:
- Cliente (id_cliente, nome, email, telefone, empresa)
- Projeto (id_projeto, nome, descrição, data_inicio, data_fim, status, valor_orçado, id_cliente)
- Tarefa (id_tarefa, titulo, descrição, status, prazo, estimativa_horas, id_projeto)
- Orçamento (id_orcamento, valor_total, data_criação, id_projeto)
- Pagamento (id_pagamento, valor, data_pagamento, forma_pagamento, parcela, status, id_orcamento)

Relacionamentos:
- Cliente 1:N Projeto
- Projeto 1:N Tarefa
- Projeto 1:N Orçamento
- Orçamento 1:N Pagamento



3. Modelo Lógico

O modelo lógico descreve as tabelas, atributos, tipos de dados e restrições.

CLIENTE(1)—<id_cliente>—(N)PROJETO

PROJETO(1)—<id_projeto>—(N)TAREFA

PROJETO(1)—<id_projeto>—(N)ORCAMENTO—<id_orcamento>—(N)PAGAMENTO

Legenda: O diagrama lógico apresenta as tabelas normalizadas, com atributos, chaves primárias e estrangeiras, evidenciando as relações entre as entidades do sistema.

Cliente:
- id_cliente (PK)
- nome (varchar)
- email (varchar)
- telefone (varchar)
- empresa (varchar)

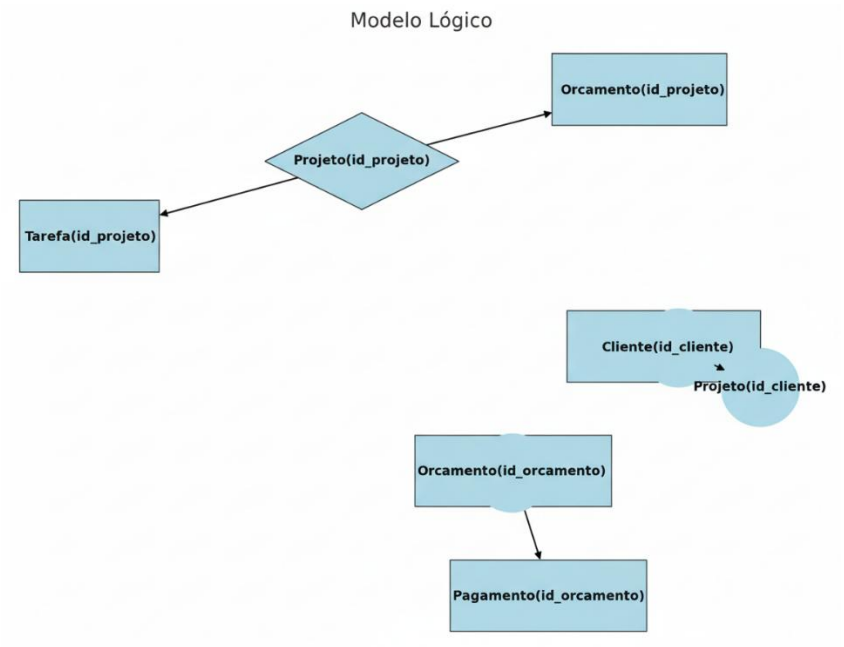
Projeto:
- id_projeto (PK)
- nome (varchar)
- descricao (text)
- data_inicio (date)
- data_fim (date)
- status (varchar)
- valor_orcado (decimal)
- id_cliente (FK → Cliente)

Tarefa:

- id_tarefa (PK)
- titulo (varchar)
- descricao (text)
- status (varchar)
- prazo (date)
- estimativa_horas (int)
- id_projeto (FK → Projeto)

Orcamento:
- id_orcamento (PK)
- valor_total (decimal)
- data_criacao (date)
- id_projeto (FK → Projeto)

Pagamento:
- id_pagamento (PK)
- valor (decimal)
- data_pagamento (date)
- forma_pagamento (varchar)
- parcela (int)
- status (varchar)
- id_orcamento (FK → Orcamento)

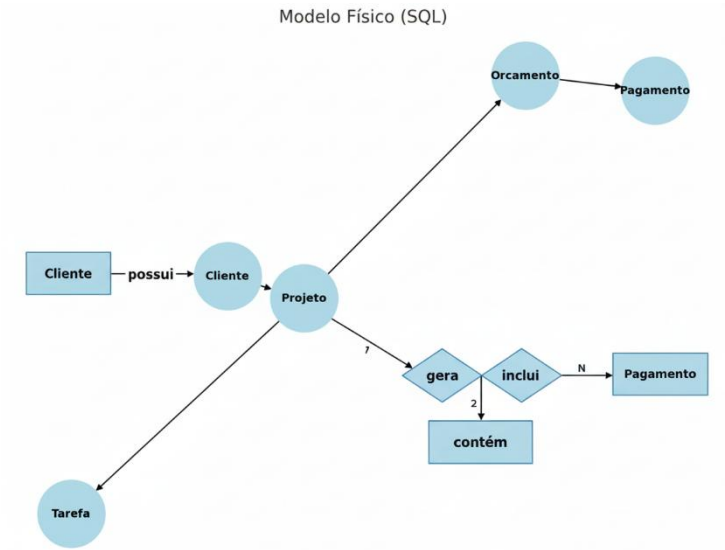


4. Modelo Físico (SQL)

```
CREATE DATABASE IF NOT EXISTS sistema_gestao;
USE sistema_gestao;
```

```
CREATE TABLE Cliente (
   id_cliente INT AUTO_INCREMENT PRIMARY KEY,
   nome VARCHAR(100) NOT NULL,
   email VARCHAR(100),
   telefone VARCHAR(20),
   empresa VARCHAR(100)
);

CREATE TABLE Projeto (
   id_projeto INT AUTO_INCREMENT PRIMARY KEY,
   nome VARCHAR(100) NOT NULL,
   descricao TEXT,
   data_inicio DATE,
   data_fim DATE,
   status VARCHAR(50),
   valor_orcado DECIMAL(10,2),
   id_cliente INT NOT NULL,
   CONSTRAINT fk_projeto_cliente FOREIGN KEY (id_cliente)
      REFERENCES Cliente(id_cliente)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Tarefa (
   id_tarefa INT AUTO_INCREMENT PRIMARY KEY,
   titulo VARCHAR(100) NOT NULL,
   descricao TEXT,
   status VARCHAR(50),
   prazo DATE,
   estimativa_horas INT,
   id_projeto INT NOT NULL,
   CONSTRAINT fk_tarefa_projeto FOREIGN KEY (id_projeto)
      REFERENCES Projeto(id_projeto)
      ON DELETE CASCADE
      ON UPDATE CASCADE
);

CREATE TABLE Orcamento (
   id_orcamento INT AUTO_INCREMENT PRIMARY KEY,
   valor_total DECIMAL(10,2) NOT NULL,
   data_criacao DATE,
   id_projeto INT NOT NULL,
   CONSTRAINT fk_orcamento_projeto FOREIGN KEY (id_projeto)
      REFERENCES Projeto(id_projeto)
```

```
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE Pagamento (
  id_pagamento INT AUTO_INCREMENT PRIMARY KEY,
  valor DECIMAL(10,2) NOT NULL,
  data_pagamento DATE,
  forma_pagamento VARCHAR(50),
  parcela INT,
  status VARCHAR(20) DEFAULT 'pendente',
  id_orcamento INT NOT NULL,
  CONSTRAINT fk_pagamento_orcamento FOREIGN KEY (id_orcamento)
    REFERENCES Orcamento(id_orcamento)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);
```



5. Manipulação de Dados (SQL)

5.1 Inserção de Dados

-- Clientes

INSERT INTO Cliente (nome, email, telefone, empresa) VALUES

('Acme Corp', 'contato@acme.com', '6199990001', 'Acme Corp'),

('Beta Solutions', 'financeiro@beta.com', '6199990002', 'Beta Solutions'),

('Gamma Ltda', 'vendas@gamma.com', '6199990003', 'Gamma Ltda')

```
-- Projetos
INSERT INTO Projeto (nome, descricao, data_inicio, data_fim, status, valor_orcado, id_cliente) VALUES
('Portal Acme', 'Portal corporativo', '2025-06-01', '2025-09-30', 'em andamento', 25000.00, 1),
('ERP Beta', 'ERP com módulo de compras', '2025-05-15', '2025-11-30', 'planejado', 80000.00, 2);

-- Tarefas
INSERT INTO Tarefa (titulo, descricao, status, prazo, estimativa_horas, id_projeto) VALUES
('Design Home', 'Layout da página inicial', 'concluída', '2025-06-10', 24, 1),
('Backend', 'API principal do sistema', 'em andamento', '2025-07-30', 160, 1);

-- Orçamentos
INSERT INTO Orcamento (valor_total, data_criacao, id_projeto) VALUES
(25000.00, '2025-05-20', 1),
(80000.00, '2025-04-30', 2);

-- Pagamentos
INSERT INTO Pagamento (valor, data_pagamento, forma_pagamento, parcela, status, id_orcamento) VALUES
(12500.00, '2025-06-01', 'transferencia', 1, 'pago', 1),
(12500.00, NULL, 'transferencia', 2, 'pendente', 1),
(40000.00, NULL, 'boleto', 1, 'pendente', 2);
```

5.2 Consultas de Exemplo (SELECT)

```
-- Projetos e seus clientes
SELECT p.nome AS Projeto, c.nome AS Cliente, p.status, p.valor_orcado
FROM Projeto p
JOIN Cliente c ON c.id_cliente = p.id_cliente;
```

```
-- Tarefas de um projeto específico
SELECT t.titulo, t.status, t.prazo
FROM Tarefa t
WHERE t.id_projeto = 1;

-- Pagamentos pendentes
SELECT pg.id_pagamento, pg.valor, pg.status
FROM Pagamento pg
WHERE pg.status = 'pendente';
```

6. Evidências de Funcionamento

6.1 Criação de Tabelas

Query OK, 0 rows affected (0.04 sec)

6.2 Inserção de Dados

Query OK, 3 rows affected (0.02 sec)

6.3 Consulta de Projetos e Clientes

Projeto	Cliente	Status	Valor Orçado
Portal Acme	Acme Corp	em andamento	25000.00
ERP Beta	Beta Solutions	planejado	80000.00

6.4 Consulta de Tarefas de Projeto

Título	Status	Prazo
Design Home	concluída	2025-06-10
Backend	em andamento	2025-07-30

6.5 Atualização de Status de Tarefa

Antes:

id_tarefa	titulo	status
2	Backend	em andamento

Depois:

id_tarefa	titulo	status
2	Backend	concluída

7. Conclusão

O banco de dados proposto atende às necessidades de um software houses, permitindo controlar: clientes, projetos e tarefas, orçamentos e pagamentos, status e prazos de execução. Esse modelo pode ser expandido futuramente para incluir funcionários, lançamentos de horas e contratos formais.

Apêndice C – Repositório Git

Nome do projeto: Sistema de Gestão Comercial para Software Houses
Descrição: Repositório contendo todos os arquivos-fonte, recursos e documentação do projeto Sistema de Gestão Comercial, incluindo códigos SQL, diagramas, relatórios e demais materiais utilizados no desenvolvimento do sistema.

Link do repositório: https://github.com/seuusuario/gestao-comercial-software-houses

Observações:

- **Todos os arquivos estão organizados em pastas por funcionalidade, incluindo cadastro de clientes, projetos, tarefas, orçamentos e pagamentos.**
- **Contém instruções para criação e teste do banco de dados no MySQL Workbench.**
- **Inclui diagramas de modelagem, relatórios e exemplos de scripts SQL para execução imediata.**