****

**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)

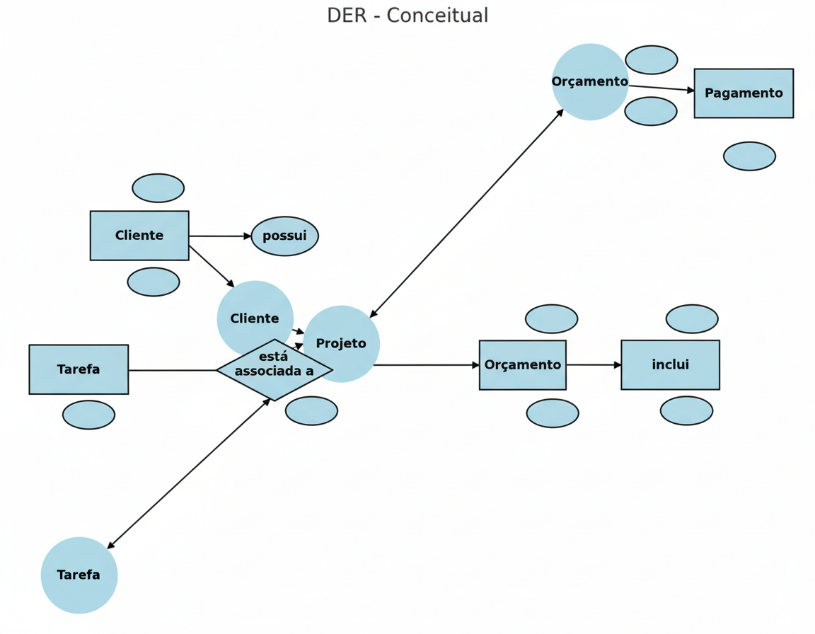
O Diagrama Entidade-Relacionamento (DER) representa as principais entidades e seus relacionamentos.

[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  
  


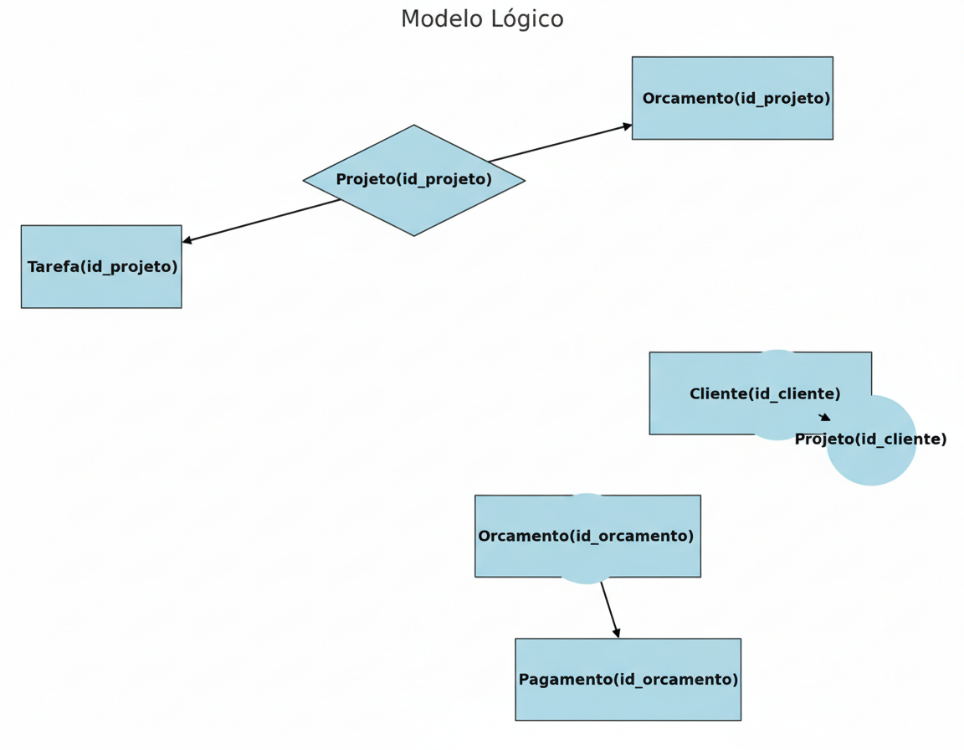
# 3. Modelo Lógico

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  
);

Diagrama

O conteúdo gerado por IA pode estar incorreto.

# 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**](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.**