🗄️ Diagrama da Base Relacional
Sistema de Estacionamento

Base de Dados: H2 Database (In-Memory) | JPA/Hibernate ORM

📋 TICKET

🔑 **CODIGO**	VARCHAR(255)	***PRIMARY KEY***
📅 **ENTRADA**	TIMESTAMP	*NOT NULL*
🚗 **PLACA**	VARCHAR(255)	*NOT NULL*
✅ **PAGO**	BOOLEAN	*DEFAULT FALSE*
📅 **SAIDA**	TIMESTAMP	*NULL*
💰 **VALOR**	DECIMAL(19,2)	*NULL*

💳 PAGAMENTO

🔑 **ID**	BIGINT	***PRIMARY KEY AUTO_INCREMENT***
📅 **DATA_PAGAMENTO**	TIMESTAMP	*NOT NULL*
🎫 **TICKET_CODIGO**	VARCHAR(255)	*NOT NULL*
💰 **VALOR**	DECIMAL(19,2)	*NOT NULL*

Relacionamento: TICKET ←→ PAGAMENTO

TICKET
CODIGO

1 ⟶ **0..***

PAGAMENTO
TICKET_CODIGO

Tipo: One-to-Many (1:N)
Descrição: Um ticket pode ter zero ou vários pagamentos
Implementação: Relacionamento lógico via campo TICKET_CODIGO

📊 Exemplo de Dados

Tabela TICKET:

CODIGO	ENTRADA	PLACA	PAGO	SAIDA	VALOR
TKT001	2024-01-15 08:30:00	ABC-1234	true	2024-01-15 10:45:00	15.00
TKT002	2024-01-15 09:15:00	DEF-5678	false	NULL	NULL
TKT003	2024-01-15 10:20:00	GHI-9012	true	2024-01-15 12:30:00	10.00

Tabela PAGAMENTO:

ID	TICKET_CODIGO	VALOR	DATA_PAGAMENTO
1	TKT001	15.00	2024-01-15 10:40:00
2	TKT003	10.00	2024-01-15 12:25:00

🔐 Chaves e Constraints

Chaves Primárias:
- TICKET.CODIGO (String)
- PAGAMENTO.ID (Long, Auto-increment)

Campos Obrigatórios:
- TICKET.ENTRADA (NOT NULL)
- TICKET.PLACA (NOT NULL)
- PAGAMENTO.TICKET_CODIGO (NOT NULL)
- PAGAMENTO.VALOR (NOT NULL)
- PAGAMENTO.DATA_PAGAMENTO (NOT NULL)

Valores Padrão:
- TICKET.PAGO = FALSE (inicialmente não pago)

📈 Índices Recomendados

Performance de Consultas:
• idx_ticket_placa ON TICKET(PLACA)
• idx_ticket_saida ON TICKET(SAIDA)
• idx_ticket_entrada ON TICKET(ENTRADA)

Relatórios por Período:
• idx_pagamento_data ON PAGAMENTO(DATA_PAGAMENTO)

Relacionamento:
• idx_pagamento_ticket_codigo ON PAGAMENTO(TICKET_CODIGO)

🔍 Consultas Típicas do Sistema

1. Buscar Ticket por Código

```
SELECT * FROM TICKET WHERE CODIGO = 'TKT001';
```

2. Relatório de Receita por Período

```
SELECT COUNT(*) as TOTAL_TICKETS_PAGOS, SUM(VALOR) as RECEITA_TOTAL, AVG(VALOR)
as VALOR_MEDIO FROM PAGAMENTO WHERE DATA_PAGAMENTO BETWEEN '2024-01-01' AND
'2024-01-31';
```

3. Tickets em Aberto (não pagos)

```
SELECT CODIGO, PLACA, ENTRADA, TIMESTAMPDIFF(MINUTE, ENTRADA, NOW()) as
MINUTOS_ESTACIONADO FROM TICKET WHERE PAGO = FALSE ORDER BY ENTRADA ASC;
```

4. Histórico de Pagamentos por Ticket

```
SELECT t.CODIGO, t.PLACA, p.DATA_PAGAMENTO, p.VALOR FROM TICKET t JOIN PAGAMENTO
p ON t.CODIGO = p.TICKET_CODIGO WHERE t.CODIGO = 'TKT001' ORDER BY
p.DATA_PAGAMENTO DESC;
```

5. Tickets por Placa do Veículo

```
SELECT CODIGO, ENTRADA, SAIDA, PAGO, VALOR FROM TICKET WHERE PLACA = 'ABC-1234'
ORDER BY ENTRADA DESC;
```

⚙️ Características Técnicas

🛡️ **Integridade de Dados:**
• Precisão monetária com DECIMAL(19,2)
• Timestamps para auditoria temporal
• Relacionamento lógico garantido pela aplicação

📈 **Escalabilidade:**
• Índices otimizados para consultas frequentes
• Estrutura preparada para grandes volumes
• Possibilidade de particionamento futuro

🔄 **Flexibilidade:**
• Suporte a múltiplos pagamentos por ticket
• Campos nullable para dados opcionais
• Facilita migração para outros SGBDs

📊 **Relatórios:**
• Consultas otimizadas por período
• Agregações para análise de receita
• Histórico completo de transações