Projeto Aplicado de Bando de Dados - Parque de Diversão

Ana Laura Lopes, Fernanda Farias Uberti e Marina Pianta

Porto Alegre, 2023

1. MODELAGEM CONCEITUAL

1.1. Apresentação do domínio de aplicação

O domínio do projeto é um parque de diversão em que possui oito entidades. São Funcionario, Caixa, Ticket, Cliente, Premio, Empresa, Estande, Brinquedo e Alimento.

- **Descrição das Entidades**

A entidade Funcionario possui como características código, nome, CPF, treinamento, celular e CEP, no qual o atributo **código é a chave primária** e o atributo **treinamento** recebe **1 para aqueles que possuem treinamento** e **0 para aqueles que não possuem treinamento.**

A entidade Caixa possui como características guiche, data_caixa, numTransacoes, valorInicialCaixa, valorFinalCaixa e lucro, no qual o atributo **guiche é a chave primária**.

A entidade Ticket possui como características código, valor e tipo, no qual o atributo **código é a chave primária**.

A entidade Cliente possui como características código, nome, dataNascimento e celular, no qual o atributo **código é a chave primária** e o atributo **dataNascimento** é solicitado para **ver se o cliente possui idade mínima para ir no brinquedo**, visto que alguns possuem.

A entidade Premio possui como características código, valor e nome, no qual o atributo **código é a chave primária** e o atributo **valor** é **a quantia de pontos** que se ganhou em brinquedos.

A entidade Empresa possui como características código, especialidade, CNPJ, email, nome e celular, no qual o atributo **código é a chave primária**.

A entidade Estande é uma entidade genérica que possui como características código, localParque e valorTicket, no qual o atributo **código é a chave primária**.

A entidade Brinquedo é uma entidade especialista em que herda todos os atributos da sua entidade mãe que são código, localParque e valorTicket. Além deles possui como características dataCompra, nome e dataManutencao. O atributo **valorTicket** herdado pela sua entidade mãe será o valor de cada brinquedo, podendo variar.

A entidade Alimento é uma entidade especialista em que herda todos os atributos da sua entidade mãe que são código, localParque e valorTicket. Além deles possui como características peso, quantidade e tipoAlimento. O atributo **tipoAlimento** será preenchido com o nome do produto como pipoca, bata frita e assim por diante. O atributo **valorTicket** herdado pela sua entidade mãe será o valor de cada alimento, podendo variar.

- **Cardinalidades e Relacionamentos**

Relacionamento das entidades Funcionario e Caixa: Nessa relação o relacionamento é denominado como "opera", **Funcionario** possui a cardinalidade (0,1), ou seja, ***um funcionário pode ou não operar um caixa*** e **Caixa** possui cardinalidade (1,1), ou seja, ***um caixa é operado por no mínimo um funcionário e no máximo um funcionário***. Como é um relacionamento binário 1:1 não é necessária nenhuma tabela auxiliar e também não possui nenhum atributo.

Relacionamento das entidades Caixa e Ticket: Nessa relação o relacionamento é denominado como "vende", **Caixa** possui a cardinalidade (0,n), ou seja, ***um caixa vende nenhum ou muitos tickets*** e

Ticket possui cardinalidade (0,1), ou seja, ***um ticket é vendido por nenhum ou um***. Como é um relacionamento binário 1:n não é necessária nenhuma tabela auxiliar e possui um atributo chamado **codigoVenda**.

Relacionamento das entidades Cliente e Ticket: Nessa relação o relacionamento é denominado como "compra", **Cliente** possui a cardinalidade (1,n), ou seja, ***um cliente compra um ou muitos tickets*** e **Ticket** possui cardinalidade (1,1), ou seja, ***um ticket é comprado por um e apenas um cliente***. Como é um relacionamento binário 1:1 não é necessária nenhuma tabela auxiliar e possui um atributo chamado **quantidadeTicket**.

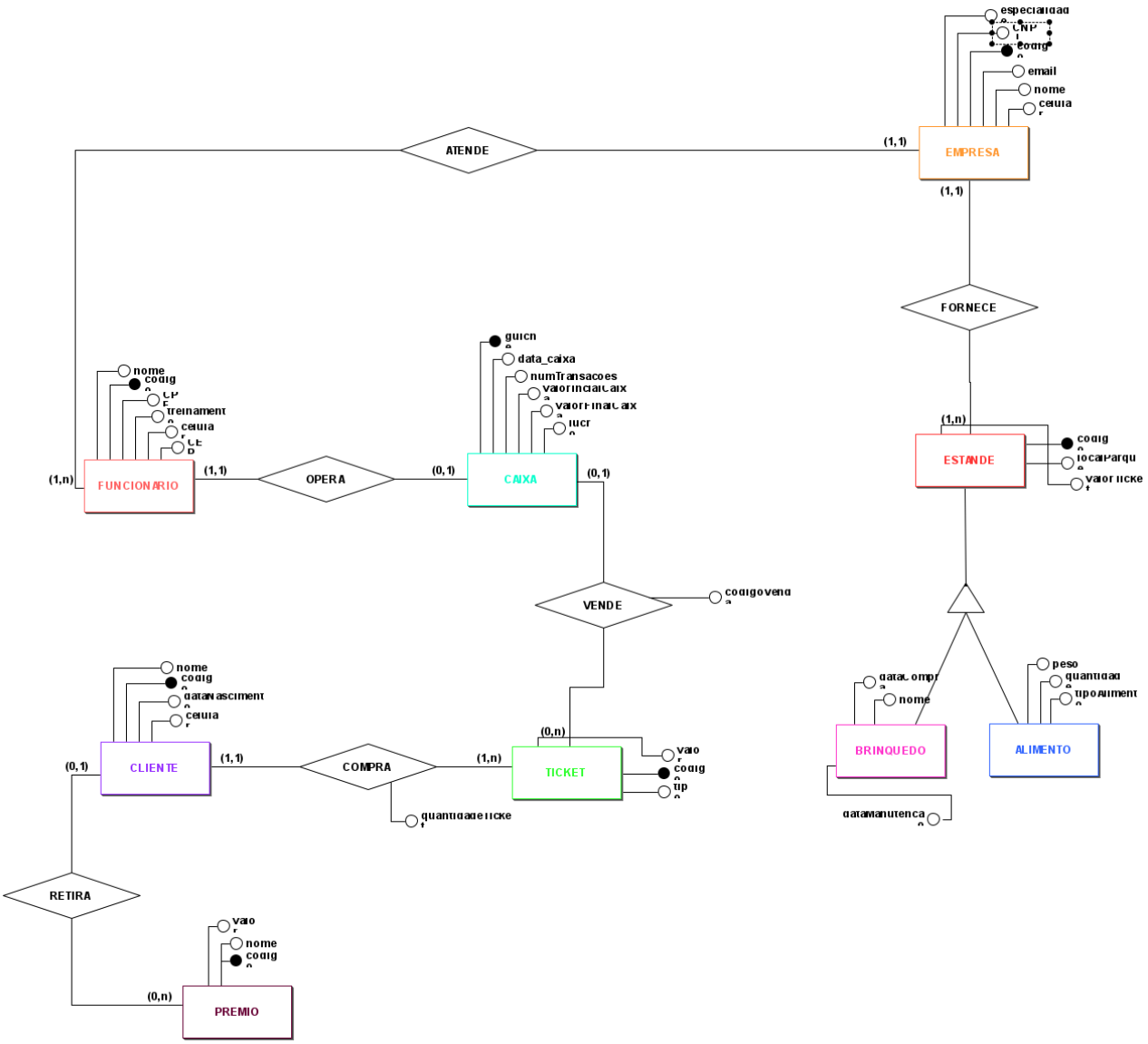
Relacionamento das entidades Cliente e Premio: Nessa relação o relacionamento é denominado como "retira", **Cliente** possui a cardinalidade (0,n), ou seja, ***um cliente retira nenhum ou vários prêmios*** e **Premio** possui cardinalidade (0,1), ou seja, ***um prêmio é retirado por nenhum ou por um cliente.*** Como é um relacionamento binário 1:n não é necessária nenhuma tabela auxiliar e também não possui nenhum atributo.

Relacionamento das entidades Funcionario e Empresa: Nessa relação o relacionamento é denominado como "atende", **Funcionario** possui a cardinalidade (1,1), ou seja, ***um funcionário atende uma e apenas uma empresa*** e **Empresa** possui cardinalidade (1,n), ou seja, ***uma empresa é atendida por um ou vários funcionários***. Como é um relacionamento binário 1:n não é necessária nenhuma tabela auxiliar e também não possui nenhum atributo.

Relacionamento das entidades Empresa e Estande: Nessa relação o relacionamento é denominado como "fornece", **Empresa** possui a cardinalidade (1,n), ou seja, ***uma empresa fornece para um ou vários estandes*** e **Estande** possui cardinalidade (1,1), ou seja ***um estande é fornecido por uma e apenas uma empresa***. Como é um relacionamento binário 1:n não é necessária nenhuma tabela auxiliar e também não possui nenhum atributo.

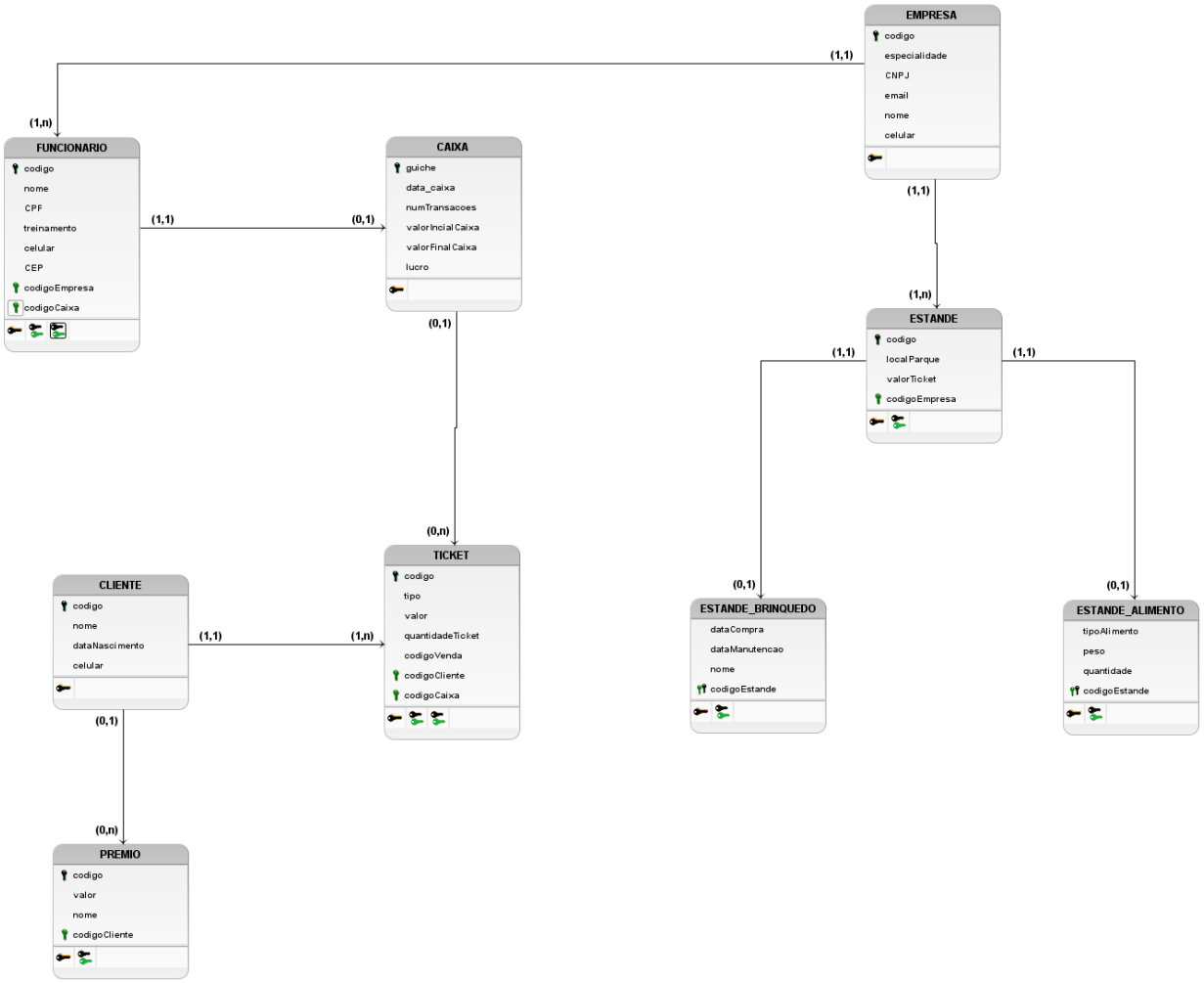
Relacionamento entre a entidade genérica Estande e suas especialistas Brinquedo e Alimento: Esse relacionamento é uma **especialização total**, no qual Estande é uma entidade genérica (entidade mãe) e Brinquedo e Alimento são entidades especialistas (entidade(s) filha(s)). Por conta desse relacionamento, Brinquedo e Alimento herdam todos os atributos de Estande, inclusive sua **chave primária**.

1.2. Diagrama Entidade-Relacionamento



2. MODELAGEM LÓGICA

2.1. Modelo lógico visual



2.2. Modelo lógico textual

CAIXA (guiche, data_Caixa, numTransacoes, valorInicialCaixa, valorFinalCaixa, lucro, codigoFun**)**
codigoFun **referencia FUNCIONÁRIO**

FUNCIONÁRIO (codigo, nome, CPF, treinamento, celular, CEP, codigoEmpresa)
codigoEmpresa **referencia EMPRESA**

TICKET (codigo, tipo, valor, quantidadeTicket, codigoVenda, codigoCliente, codigoCaixa)
codigoCaixa **referencia CAIXA**
codigoCliente **referencia CLIENTE**

CLIENTE (codigo, nome, dataNascimento, celular)

PREMIO (codigo, valor, nome, codigoCliente)
codigoCliente **referencia CLIENTE**

EMPRESA (codigo, CNPJ, especialidade, email, nome, celular)

ESTANDE (codigo, localParque, valorTicket, **codigoEmpresa**)
codigoEmpresa **referencia EMPRESA**

BRINQUEDO (codigoEstande, dataManutencao, dataCompra, nome)
codigoEstande **referencia ESTANDE**

ALIMENTO (codigoEstande, tipoAlimento, peso, quantidade)
codigoEstande **referencia ESTANDE**

2.3. Normalização

Como não foi feito a normalização na etapa de entrega do modelo lógico, optamos por não realizar nessa etapa pois alteraria alguns pontos e poderia afetar no desempenho do projeto. O projeto normalizado teria que ter a tabela em que guardasse as chaves primárias de EMPRESA, FUNCIONARIO E CAIXA e a tabela que guardasse as chaves primárias de TICKET, CLIENTE e CAIXA. Além disso, deveria ser feito no back-end o cálculo para poder gerar o lucro na entidade CAIXA. Com essas alterações, o projeto já está de acordo com a **terceira forma normal** e consequentemente na primeira e segunda forma normal.

3. MODELAGEM FÍSICA

3.1. Modelo físico em notação visual

Não fizemos o modelo físico em notação visual no Astah, por isso não pusemos imagem.

3.2. Dicionário de Dados

FUNCIONARIO			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo	integer	NOT NULL, PK	Código único para funcionário.
nome	varchar(50)	NOT NULL	Nome completo do funcionário.
CPF	integer	NOT NULL	Número de CPF do funcionário.
treinamento	number	NOT NULL	Armazena se o funcionário possui treinamento para conduzir os brinquedos do parque. Se é apto recebe 1, senão 0.
celular	integer	NOT NULL	Número de celular do funcionário.
CEP	Integer	NOT NULL	Número de CEP do funcionário.

codigoEmpresa	integer	NULL, FK	Código a empresa em que o funcionário trabalha. O campo restrições é NULL pois se esse funcionário trabalha no parque, ou seja, não trabalha para uma empresa, ele não possuirá o código de empresa. Se o funcionário trabalha para uma empresa, o campo será preenchido.

CAIXA			
Atributo	**Tipo**	**Restrições**	**Descrição**
guiche	integer	NOT NULL, FK	Código único do caixa
data_caixa	date	NOT NULL	Data do dia em que o caixa está operando. A representação da data é DD-MM-AAAA.
numTransacoes	integer	NOT NULL	Quantidades de transações que o caixa fez no dia.
valorInicialCaixa	integer	NOT NULL	Valor inicial de dinheiro que o caixa começou.
valorFinalCaixa	integer	NOT NULL	Valor final de dinheiro em que o caixa obteve no fim do dia com todas as transações.
lucro	integer	NOT NULL	Diferença entre o valorInicialCaixa e valorFinalCaixa que resulta no lucro que o caixa obteve no dia com todas as transações.
codigoFunc	integer	NOT NULL, FK	Código do funcionário que operou o caixa no dia.

TICKET			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo	integer	NOT NULL, FK	Código único do ticket
tipo	varchar(30)	NOT NULL	O ticket possui três tipo: **brinquedos**, **alimentos** e **prêmios**. O de brinquedo e alimento possuem um valor monetário, já o de prêmio não possui pois são os pontos conseguidos nos brinquedos.
valor	integer	NULL	Valor monetário de cada ticket. Possui o campo restrições NULL pois para ticket do tipo prêmio não possuem valor.
quantidadeTicket	integer	NOT NULL	Esse atributo veio do relacionamento com CLIENTE. Por isso, ele é a quantidade de ticket que o cliente comprou.

codigoVenda	integer	NOT NULL	Código único da venda do ticket. Possui o campo restrições NULL pois se for um ticket tipo prêmio ele não será vendido.
codigoCliente	integer	NOT NULL, FK	Código do cliente que comprou.
codigoCaixa	integer	NULL, FK	Código do caixa que foi realizado a venda. Possui o campo restrições NULL pois se for um ticket tipo prêmio ele não será comprado.

CLIENTE			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo	integer	NOT NULL, FK	Código único do cliente.
nome	varchar(30)	NOT NULL	Nome do cliente.
dataNascimento	date	NOT NULL	A data de nascimento é requerida pois será utilizada para calcular a idade do cliente para ver se é permitido ir brinquedos que possuem idade mínima. A representação da data é DD-MM-AAAA.
celular	integer	NOT NULL	Celular do cliente.

PREMIO			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo	integer	NOT NULL, FK	Código único do prêmio
valor	integer	NOT NULL	O valor em pontos de cada prêmio.
nome	varchar(30)	NOT NULL	Nome do brinquedo.
codigoCliente	integer	NOT NULL, FK	Esse atributo veio do relacionamento com CLIENTE. Por isso, ele é o código do cliente que está retirando o brinquedo

EMPRESA			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo	integer	NOT NULL, PK	Código único para a empresa.
nome	varchar(50)	NOT NULL	Nome da empresa.

CNPJ	integer	NOT NULL	Número de CNPJ da empresa.
especialidade	varchar(50)	NOT NULL	Especialidade da empresa como alimentos, brinquedos, sanitários e outros.
email	varchar(50)	NOT NULL	Email da empresa.
celular	Integer	NOT NULL	Número de celular da empresa.

ESTANDE			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo	integer	NOT NULL, FK	Código único do estande
localParque	varchar(50)	NOT NULL	Área do parque em que o estande está.
valorTicket	integer	NOT NULL	Valor do ticket do estande.
codigoEmpresa	integer	NOT NULL, FK	Código da empresa do estande.

BRINQUEDO			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo_estande	integer	NOT NULL, PK e FK	Código único do estande do brinquedo.
dataManutencao	date	NOT NULL	A data da última manutenção do brinquedo. A representação da data é DD-MM-AAAA.
dataCompra	date	NOT NULL	A data da compra do brinquedo. A representação da data é DD-MM-AAAA.
Nome	varchar(50)	NOT NULL	Nome do brinquedo.

ALIMENTO			
Atributo	**Tipo**	**Restrições**	**Descrição**
codigo_estande	integer	NOT NULL, PK e FK	Código único do estande do alimento.

tipoAlimento	varchar(50)	NOT NULL	Tipo do alimento vendido, seja comida ou bebida.
peso	integer	NOT NULL	Peso do alimento, seja em mililitros ou quilogramas.
quantidade	integer	NOT NULL	Quantidade de cada alimento no estoque.

3.3. Implementação em SQL

3.3.1. Criação de tabelas

```
CREATE TABLE FUNCIONARIO (
   codigo INTEGER NOT NULL,
   nome VARCHAR(50) NOT NULL,
   CPF INTEGER NOT NULL,
   treinamento NUMBER NOT NULL,
   celular INTEGER NOT NULL,
   CEP INTEGER NOT NULL,
   codigoEmpresa INTEGER  NULL,
   PRIMARY KEY (codigo),
   FOREIGN KEY (codigoEmpresa) REFERENCES EMPRESA(codigo)
) DEFAULT COLLATION BINARY_AI;
```

```
CREATE TABLE CAIXA (
   guiche INTEGER NOT NULL,
   data_caixa DATE NOT NULL,
   numTransacoes INTEGER NOT NULL,
   valorInicialCaixa INTEGER NOT NULL,
   valorFinalCaixa INTEGER NOT NULL,
   lucro INTEGER NOT NULL,
   codigoFunc INTEGER NOT NULL,
   PRIMARY KEY (guiche),
   FOREIGN KEY (codigoFunc) REFERENCES FUNCIONARIO(codigo)
) DEFAULT COLLATION BINARY_AI;
```

```
CREATE TABLE CLIENTE (
   codigo INTEGER NOT NULL,
   nome VARCHAR(50) NOT NULL,
   dataNascimento DATE NOT NULL,
   celular INTEGER NOT NULL,
   PRIMARY KEY (codigo)
) DEFAULT COLLATION BINARY_AI;
```

```
CREATE TABLE TICKET (
   codigo INTEGER NOT NULL,
   tipo VARCHAR(30) NOT NULL,
   valor INTEGER NULL,
   quantidadeTicket INTEGER NOT NULL,
   codigoVenda INTEGER NULL,
   codigoCliente INTEGER NOT NULL,
   codigoCaixa INTEGER NULL,
   PRIMARY KEY (codigo),
   FOREIGN KEY (codigoCliente) REFERENCES CLIENTE(codigo),
   FOREIGN KEY (codigoCaixa) REFERENCES CAIXA(guiche)
) DEFAULT COLLATION BINARY_AI;

CREATE TABLE PREMIO (
   codigo INTEGER NOT NULL,
   valor INTEGER NOT NULL,
   nome VARCHAR(50) NOT NULL,
   codigoCliente INTEGER NOT NULL,
   PRIMARY KEY (codigo),
   FOREIGN KEY (codigoCliente) REFERENCES CLIENTE(codigo)
) DEFAULT COLLATION BINARY_AI;


CREATE TABLE EMPRESA (
   codigo INTEGER NOT NULL,
   nome VARCHAR(50) NOT NULL,
   CNPJ INTEGER NOT NULL,
   especialidade VARCHAR(50) NOT NULL,
   email VARCHAR(50) NOT NULL,
   celular INTEGER NOT NULL,
   PRIMARY KEY (codigo)
) DEFAULT COLLATION BINARY_AI;


CREATE TABLE ESTANDE (
   codigo INTEGER NOT NULL,
   localParque VARCHAR(50) NOT NULL,
   valorTicket INTEGER NOT NULL,
   codigoEmpresa INTEGER NOT NULL,
   PRIMARY KEY (codigo),
   FOREIGN KEY (codigoEmpresa) REFERENCES EMPRESA(codigo)
) DEFAULT COLLATION BINARY_AI;
```

```
CREATE TABLE BRINQUEDO (
   codigo_estande INTEGER NOT NULL,
   dataManutencao DATE NOT NULL,
   dataCompra DATE NOT NULL,
   nome VARCHAR(50) NOT NULL,
   PRIMARY KEY (codigo_estande),
   FOREIGN KEY (codigo_estande) REFERENCES ESTANDE(codigo)
) DEFAULT COLLATION BINARY_AI;
```

```
CREATE TABLE ALIMENTO (
   codigo_estande INTEGER NOT NULL,
   tipoAlimento VARCHAR(50) NOT NULL,
        peso INTEGER NOT NULL,
   quantidade INTEGER NOT NULL,
   PRIMARY KEY (codigo_estande),
   FOREIGN KEY (codigo_estande) REFERENCES ESTANDE(codigo)
) DEFAULT COLLATION BINARY_AI;
```

3.3.2. Inserção de dados

```
INSERT INTO CAIXA (guiche, data_caixa, numTransacoes, valorInicialCaixa, valorFinalCaixa, lucro, codigoFunc) VALUES (2, TO_DATE('10-08-2023', 'DD-MM-YYYY'), 3000, 500, 7000, 6500, 1);
INSERT INTO CAIXA (guiche, data_caixa, numTransacoes, valorInicialCaixa, valorFinalCaixa, lucro, codigoFunc) VALUES (1, TO_DATE('10-08-2023', 'DD-MM-YYYY'), 3500, 500, 8000, 7500, 2);
INSERT INTO CAIXA (guiche, data_caixa, numTransacoes, valorInicialCaixa, valorFinalCaixa, lucro, codigoFunc) VALUES (3, TO_DATE('12-08-2023', 'DD-MM-YYYY'), 2500, 300, 2000, 1700, 3);
```

```
INSERT INTO FUNCIONARIO (codigo, nome, CPF, treinamento, celular, CEP, codigoEmpresa) VALUES (1, 'Marina Pianta', 87598725698, 1, 51997500578, 94764123, null);
INSERT INTO FUNCIONARIO (codigo, nome, CPF, treinamento, celular, CEP, codigoEmpresa) VALUES (2, 'Fernanda Farias', 49782516452, 1, 51998475213, 95890460, null);
INSERT INTO FUNCIONARIO (codigo, nome, CPF, treinamento, celular, CEP, codigoEmpresa) VALUES (3, 'Ana Laura Lopes', 18957964102, 1, 51984596782, 90456289, null);
INSERT INTO FUNCIONARIO (codigo, nome, CPF, treinamento, celular, CEP, codigoEmpresa) VALUES (4, 'Carol Sampaio', 84609743812, 0, 51999996425, 98459347, 1);
```

```
INSERT INTO CLIENTE (codigo, nome, dataNascimento, celular) VALUES (1, 'Ana Lucia Barros', TO_DATE('17-08-1949', 'DD-MM-YYYY'), 51980506879);
INSERT INTO CLIENTE (codigo, nome, dataNascimento, celular) VALUES (2, 'Lucas Silveira', TO_DATE('23-11-1999', 'DD-MM-YYYY'), 51997064231);
INSERT INTO CLIENTE (codigo, nome, dataNascimento, celular) VALUES (3, 'Caio Pereira', TO_DATE('08-02-2003', 'DD-MM-YYYY'), 51996457389);
```

```
INSERT INTO TICKET (codigo, tipo, valor, quantidadeTicket, codigoVenda, codigoCliente, codigoCaixa)
VALUES (1, 'Brinquedo', 5, 15, 5, 2, 3);
INSERT INTO TICKET (codigo, tipo, valor, quantidadeTicket, codigoVenda, codigoCliente, codigoCaixa)
VALUES (2, 'Alimento', 6, 20, 6, 1, 2);
INSERT INTO TICKET (codigo, tipo, valor, quantidadeTicket, codigoVenda, codigoCliente, codigoCaixa)
VALUES (4, 'Premio', null, 10, null, 1, null);


INSERT INTO PREMIO (codigo, valor, nome, codigoCliente) VALUES (01, 10, 'Bola Gigante Colorida', 1);
INSERT INTO PREMIO (codigo, valor, nome, codigoCliente) VALUES (02, 2, 'Chaveiro', 1);
INSERT INTO PREMIO (codigo, valor, nome, codigoCliente) VALUES (03, 5, 'Estralinho', 3);


INSERT INTO EMPRESA (codigo, nome, CNPJ, especialidade, email, celular) VALUES (1, 'TechFun S.A.',
98765432000121, 'Brinquedo de parque de diversões', 'info@techfun.com',  2198765432);
INSERT INTO EMPRESA (codigo, nome, CNPJ, especialidade, email, celular) VALUES (2, 'Sabores Deliciosos',
55555555000101, 'Alimentos', 'contato@saboresdeliciosos.com', 4729768231);
INSERT INTO EMPRESA (codigo, nome, CNPJ, especialidade, email, celular) VALUES (3, 'Diversão Premiada
Eventos Ltda', 45678901000134, 'Premios', 'contato@diversaopremiada.com', 2298765432);
INSERT INTO EMPRESA (codigo, nome, CNPJ, especialidade, email, celular) VALUES (4, 'Higiene Total Ltda',
56789012000189, 'Limpeza/Sanitários', 'contato@higienetotal.com', 44987654321);


INSERT INTO ESTANDE (codigo, localParque, valorTicket, codigoEmpresa) VALUES (6, 'Zona A', 10, 1);
INSERT INTO ESTANDE (codigo, localParque, valorTicket, codigoEmpresa) VALUES (7, 'Zona A', 5, 1);
INSERT INTO ESTANDE (codigo, localParque, valorTicket, codigoEmpresa) VALUES (8, 'Zona A', 10, 1);
INSERT INTO ESTANDE (codigo, localParque, valorTicket, codigoEmpresa) VALUES (9, 'Zona C', 15, 2);
INSERT INTO ESTANDE (codigo, localParque, valorTicket, codigoEmpresa) VALUES (10, 'Zona C', 5, 2);
INSERT INTO ESTANDE (codigo, localParque, valorTicket, codigoEmpresa) VALUES (11, 'Zona C', 7, 2);


INSERT INTO BRINQUEDO (codigo_estande, dataManutencao, dataCompra, nome) VALUES (6,
TO_DATE('29122023', 'DD-MM-YYYY'), TO_DATE('15092020', 'DD-MM-YYYY'), 'Gira-Gira');
INSERT INTO BRINQUEDO (codigo_estande, dataManutencao, dataCompra, nome) VALUES
(7,TO_DATE('29122023', 'DD-MM-YYYY'), TO_DATE('15092020', 'DD-MM-YYYY'), 'Centopeia');
INSERT INTO BRINQUEDO (codigo_estande, dataManutencao, dataCompra, nome) VALUES (8,
TO_DATE('20022024', 'DD-MM-YYYY'), TO_DATE('14052019', 'DD-MM-YYYY'), 'Montanha Russa');


INSERT INTO ALIMENTO (codigo_estande, tipoAlimento, peso, quantidade) VALUES (9, 'Batata Frita', 0.25,
500);
INSERT INTO ALIMENTO (codigo_estande, tipoAlimento, peso, quantidade) VALUES (10, 'Pipoca', 0.10, 500);
INSERT INTO ALIMENTO (codigo_estande, tipoAlimento, peso, quantidade) VALUES (11, 'Refrigerantes',
0.0005, 1000);
```

3.3.3. Seleção de dados:

- **Exiba todos os registros de uma tabela usando atributo textual para ordenação decrescente;**

Ordena a tabela cliente pelo nome em ordem decrescente.

```
SELECT * FROM CLIENTE c ORDER BY c.nome DESC;
```

- **Apresente uma função de agregação (count, sum, max, min, avg);**

Apresenta a quantidade de estandes que cada empresas possui.

```
SELECT e.nome AS nome_empresa, COUNT(es.codigo) AS qtd_estandes FROM EMPRESA e, ESTANDE es WHERE e.codigo = es.codigoEmpresa GROUP BY e.nome;
```

- **Apresenta dados de duas tabelas com relacionamento 1 para N ou N para N;**

```
SELECT c.nome AS nome_cliente, c.codigo AS codigo_cliente, c.dataNascimento AS dataNascimento_cliente, c.celular AS celular_cliente, p.nome AS nome_premio, p.valor AS valor_premio, p.codigo AS codigo_premio FROM CLIENTE c JOIN PREMIO p ON c.codigo = p.codigoCliente;
```

- **Apresente o uso de operadores de BETWEEN ou IN nos critérios;**

Apresenta as empresas que possuem código entre 1 e 3.

```
SELECT e.codigo, e.nome AS nome_empresa FROM EMPRESA e WHERE e.codigo BETWEEN 1 AND 3;
```

- **Use cláusulas de junção (INNER JOIN ou LEFT JOIN ou RIGHT JOIN);**

Apresenta qual prêmio cada cliente possui.

```
SELECT c.nome AS nome_cliente, p.nome AS nome_premio FROM CLIENTE c LEFT JOIN PREMIO p ON c.codigo = p.codigoCliente;
```

4. CONSIDERAÇÕES FINAIS

Como este trabalho foi a primeira vez que tivemos a experiencia de criar e manipular um banco de dados completo, seguindo todos os passos juntos, acreditamos que o resultado obtivo foi positivo e satisfatório. Para melhorar o projeto e deixá-lo mais completo, poderiam ser feitos alguns upgrades, porém iria requerer mais habilidades e experiencia. Ao longo do processo das entregas parciais, a cada nova etapa fomos encontrando os erros e percebendo o que poderíamos melhorar, assim fomos aprimorando nosso conhecimento e prática.

Como melhorias poderiam ser criadas mais algumas tabelas, caso houvesse um maior aprofundamento e especificar melhor como seria o funcionamento do parque como eventos que poderiam ser realizados (tabela Eventos), equipamentos eletrônicos utilizados (Equipamentos_eletronicos), manutenções feitas no parque (Manutencao), porém, tivemos um tempo limitado para a realização do projeto.