Aluno: Gabriel Lucas Dias de Sant' Anna
Professor: Fábio Cardozo
Curso: Pós graduação em Administração de Banco de Dados
Matéria: Projeto de Banco de Dados

Prova de Banco de Dados

Sumário

- Questão 1 - Modelagem Conceitual e Lógica
- Questão 2 - Modelo Entidade Relacionamento
- Questão 3 - Formas Normais
- Questão 4 - Banco de Dados Orientado a Objetos

QUESTÃO 1 - MODELAGEM CONCEITUAL E LÓGICA

a) Identificação e Listagem de Entidades e Atributos

Com base no mini mundo apresentado do Sistema de Gestão da Agência de Turismo "Mundo Aventura" e considerando a flexibilidade do PostgreSQL para abordar conceitos de BDOO, as seguintes entidades foram identificadas, juntamente com seus principais atributos, chaves primárias (PK), chaves estrangeiras (FK) e tipos de dados sugeridos. Notamos a aplicação de **tipos compostos** para atributos que representam estruturas complexas, alinhando-se à discussão da Questão 4.

b) Identificação e Descrição dos Relacionamentos

c) Modelo Lógico Relacional Resultante

SGDB usado: PostgreSQL

```
 -- Tipo composto para endereço
CREATE TYPE endereco AS (
    rua VARCHAR(255),
    numero VARCHAR(10),
    bairro VARCHAR(100),
    cidade VARCHAR(100),
    estado CHAR(2),
    cep VARCHAR(10)
);

-- Tipo composto para contato
```

```
CREATE TYPE contato AS (
    email VARCHAR(255),
    telefone_principal VARCHAR(20),
    telefone_secundario VARCHAR(20)
);

-- Tabela: CATEGORIAS_FIDELIDADE
CREATE TABLE CATEGORIAS_FIDELIDADE (
    id SERIAL PRIMARY KEY,
    nome_categoria VARCHAR(20) NOT NULL UNIQUE,
    percentual_desconto DECIMAL(5,2) NOT NULL
);

-- ...demais tabelas omitidas para concisão...
```

QUESTÃO 2 - MODELO ENTIDADE RELACIONAMENTO (Atualizada e Revisada)

a) Diagrama ER completo (Legenda)

- PK sinalizado com símbolo de uma chave dourada
- FG's sinalizadas com símbolo de 2 chaves cor prata
- Cardinalidade demarcada no contato de cada linha de conexão entre as entidades.
- Não possui atributos multivalorados e nem entidades fracas

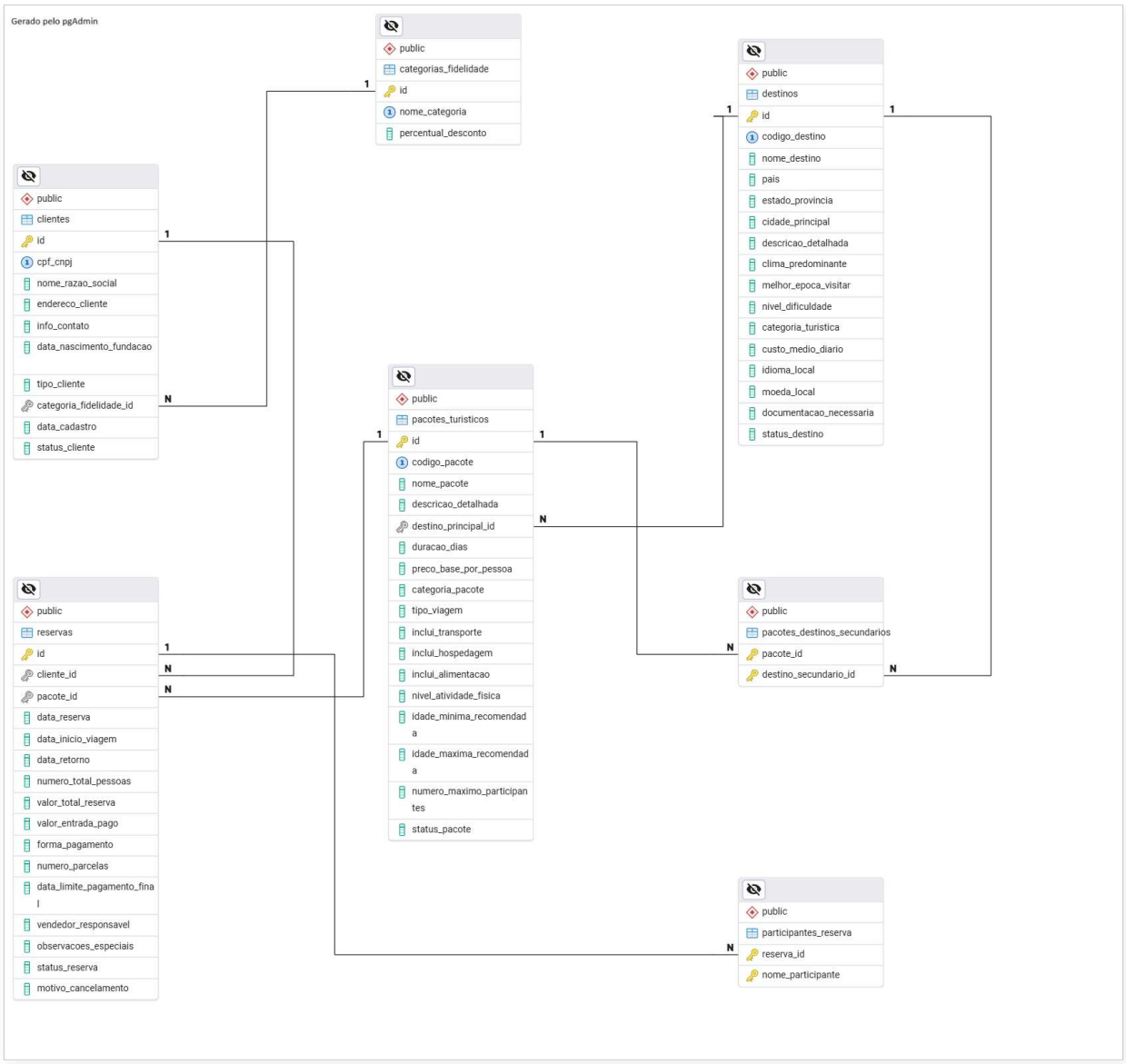

Figura 1: DER detalhado gerado pelo pgAdmin

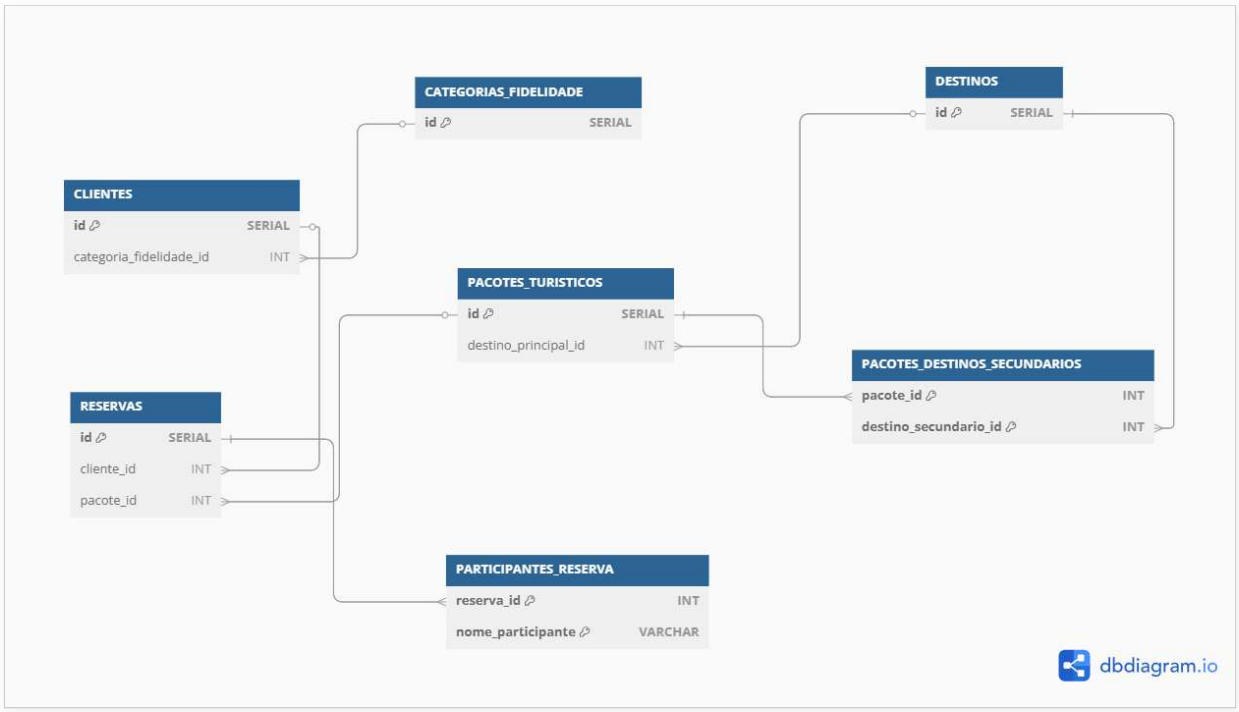

Figura 2: DER simplificado gerado pelo dbdiagram.io

---

b) Entidades Associativas

No modelo, identificamos duas entidades associativas essenciais para a normalização e correta representação dos relacionamentos N:M e atributos multivalorados:

- **PACOTES_DESTINOS_SECUNDARIOS**: Resolve o relacionamento N:M entre PACOTES_TURISTICOS e DESTINOS para destinos secundários.
- **PARTICIPANTES_RESERVA**: Resolve o atributo multivalorado lista_nomes_participantes da entidade RESERVAS.

---

c) Restrições de Integridade

Restrições de Domínio

- **CLIENTES.tipo_cliente**: Deve ser 'individual', 'corporativo' ou 'grupo'.
- **CLIENTES.categoria_fidelidade_id**: Deve referenciar um id válido em CATEGORIAS_FIDELIDADE.
- **CLIENTES.status_cliente**: Deve ser 'ativo' ou 'inativo'.
- **DESTINOS.nivel_dificuldade**: Deve ser 'facil', 'moderado' ou 'dificil'.
- **DESTINOS.categoria_turistica**: Deve ser 'praia', 'montanha', 'cidade historica', 'aventura', 'cultural' ou 'gastronomico'.
- **DESTINOS.status_destino**: Deve ser 'ativo', 'inativo' ou 'sazonal'.
- **PACOTES_TURISTICOS.categoria_pacote**: Deve ser 'economico', 'standard', 'premium' ou 'luxo'.
- **PACOTES_TURISTICOS.tipo_viagem**: Deve ser 'individual', 'casal', 'familia' ou 'grupo'.
- **PACOTES_TURISTICOS.inclui_alimentacao**: Deve ser 'cafe', 'meia pensao' ou 'pensao completa'.
- **PACOTES_TURISTICOS.nivel_atividade_fisica**: Deve ser 'baixo', 'medio' ou 'alto'.
- **PACOTES_TURISTICOS.status_pacote**: Deve ser 'ativo', 'inativo' ou 'sazonal'.
- **RESERVAS.forma_pagamento**: Deve ser 'cartao', 'transferencia', 'dinheiro' ou 'parcelado'.
- **RESERVAS.status_reserva**: Deve ser 'pendente', 'confirmada', 'paga', 'cancelada', 'em andamento' ou 'finalizada'.

Restrições de Integridade Referencial

- **RESERVAS.cliente_id** referencia **CLIENTES.id**
- **PACOTES_TURISTICOS.destino_principal_id** referencia **DESTINOS.id**
- **RESERVAS.pacote_id** referencia **PACOTES_TURISTICOS.id**
- **PACOTES_DESTINOS_SECUNDARIOS.pacote_id** referencia **PACOTES_TURISTICOS.id**
- **PACOTES_DESTINOS_SECUNDARIOS.destino_secundario_id** referencia **DESTINOS.id**
- **PARTICIPANTES_RESERVA.reserva_id** referencia **RESERVAS.id**

Restrições de Negócio Específicas do Sistema

- Descontos por Categoria de Fidelidade: aplicar descontos (15% para Platinum, 10% para Ouro, 5% para Prata) no valor_total_reserva com base em CLIENTES.categoria_fidelidade.
- Desconto para Grupos: Se RESERVAS.numero_total_pessoas > 10, aplicar desconto adicional de 10%.
- Regras de Cancelamento: lógica para calcular motivo_cancelamento e valor da multa com base na data_reserva e data_inicio_viagem.
- Destino Principal Obrigatório: PACOTES_TURISTICOS.destino_principal_id não pode ser nulo.
- Pacotes Sazonais: lógica para verificar disponibilidade de pacotes sazonais em períodos específicos do ano.
- Valor de Entrada Mínimo: RESERVAS.valor_entrada_pago deve ser no mínimo 30% de RESERVAS.valor_total_reserva.
- Prazo de Pagamento para Clientes Corporativos: Se CLIENTES.tipo_cliente = 'corporativo', RESERVAS.data_limite_pagamento_final deve ser estendido para 45 dias após a data_reserva.

QUESTÃO 3 - FORMAS NORMAIS

a) Primeira Forma Normal (1FN)

Discussão sobre atomicidade dos atributos, ausência de grupos repetitivos e unicidade das linhas. Todas as tabelas propostas estão em 1FN.

b) Segunda Forma Normal (2FN)

Discussão sobre dependências funcionais parciais. Todas as tabelas propostas estão em 2FN.

c) Terceira Forma Normal (3FN)

Discussão sobre dependências transitivas. Todas as tabelas propostas estão em 3FN. Benefícios da normalização destacados.

QUESTÃO 4 - BANCO DE DADOS ORIENTADO A OBJETOS

a) Modelagem de uma Entidade como "Classe" Orientada a Objetos no PostgreSQL

```
 -- Tipo composto para endereço
CREATE TYPE endereco AS (
    rua VARCHAR(255),
    numero VARCHAR(10),
    bairro VARCHAR(100),
    cidade VARCHAR(100),
    estado CHAR(2),
```

```
    cep VARCHAR(10)
);

-- Tipo composto para contato
CREATE TYPE contato AS (
    email VARCHAR(255),
    telefone_principal VARCHAR(20),
    telefone_secundario VARCHAR(20)
);

-- Tabela "pessoa" como superclasse
CREATE TABLE pessoa (
    id SERIAL PRIMARY KEY,
    nome VARCHAR(255) NOT NULL,
    data_nascimento DATE,
    info_contato contato,
    endereco_residencial endereco
);

-- Tabela "cliente" herdando de "pessoa"
CREATE TABLE cliente (
    cpf_cnpj VARCHAR(14) UNIQUE NOT NULL,
    data_cadastro DATE DEFAULT CURRENT_DATE,
    status_cliente VARCHAR(20) CHECK (status_cliente IN ('ativo', 'inativo')) D
) INHERITS (pessoa);

-- Função para calcular idade do cliente
CREATE OR REPLACE FUNCTION calcular_idade_cliente(p_id INTEGER)
RETURNS INTEGER AS $$
DECLARE
    data_nasc DATE;
BEGIN
    SELECT data_nascimento INTO data_nasc FROM pessoa WHERE id = p_id;
    RETURN EXTRACT(YEAR FROM AGE(NOW(), data_nasc));
END;
$$ LANGUAGE plpgsql;
```

b) Implementação de um Método Específico

```
 -- Função que verifica se o cliente pode realizar uma nova reserva
CREATE OR REPLACE FUNCTION pode_realizar_reserva(p_id_cliente INTEGER)
RETURNS BOOLEAN AS $$
DECLARE
    ativo BOOLEAN;
    qtd_reservas INTEGER;
BEGIN
    SELECT status_cliente = 'ativo' INTO ativo FROM cliente WHERE id = p_id_cli
    SELECT COUNT(*) INTO qtd_reservas FROM reservas WHERE cliente_id = p_id_cli
    RETURN ativo AND qtd_reservas < 5;
END;
$$ LANGUAGE plpgsql;
```

c) Comparação: BDOO (PostgreSQL Objeto-Relacional) vs Banco Relacional Tradicional

- **Vantagens:** Reutilização de estruturas, herança, encapsulamento de regras de negócio, atributos complexos.
- **Desvantagens:** Consultas mais complexas, suporte parcial a recursos OO, menor compatibilidade com ferramentas de mercado, possível impacto de performance.

O PostgreSQL permite simular muitos conceitos de BDOO, trazendo flexibilidade e expressividade ao modelo de dados. Porém, para sistemas com grande volume de dados e necessidade de relatórios complexos, o modelo relacional tradicional ainda pode ser mais eficiente e amplamente suportado.