FACULDADE SENAI “GASPAR RICARDO JÚNIOR”

CFP 402

PROJETO FINAL – BANCO DE DADOS

ANDRÉ LUCAS FERREIRA

LEONARDO RODRIGUES VIEIRA

NICKOLAS RAPHAEL MACHADO

ODIRLEI LIMA

SOROCABA

2025

FACULDADE SENAI “GASPAR RICARDO JÚNIOR”

CFP 402

PROJETO FINAL – BANCO DE DADOS

ANDRÉ LUCAS FERREIRA

LEONARDO RODRIGUES VIEIRA

NICKOLAS RAPHAEL MACHADO

ODIRLEI LIMA

Nº2, N°26, N°33, N°34, Tecnólogo em Análise e Desenvolvimento de Sistemas, 1º semestre.

Trabalho solicitado no componente curricular de banco de dados sob orientação do Prof.º André Souza até a data 20/06/2025.

Sorocaba, 16 de maio de 2025.

SUMÁRIO

1 INTRODUÇÃO

A gestão e interpretação eficiente de dados é um elemento essencial para o funcionamento de qualquer sistema informatizado hodierno, especialmente em setores que envolvem grande fluxo de informações, como por exemplo um sistema de um hotel. Neste contexto, este trabalho tem como foco o desenvolvimento e a estruturação de um banco de dados relacional voltado à operação de um sistema de gerenciamento de hotelaria, utilizando a linguagem SQL por meio de comandos DDL (Data Definition Language), DML (Data Manipulation Language), DQL (Data Query Language), DCL (Data Control Language) e DTL (Data Transaction Langugage).

O objetivo principal deste projeto é implementar a base estrutural de um sistema de banco de dados que suporte as principais operações administrativas de um hotel, como o cadastro e gerenciamento de clientes (pessoas físicas e jurídicas), controle de reservas, registro de pagamentos, serviços oferecidos, contratos firmados, bem como os dados de funcionários, hospedagens e estrutura física das unidades (endereços, quartos e contatos). O banco de dados foi projetado com foco na integridade referencial, normalização das informações e flexibilidade para que o banco seja manutenível e expansível.

A estrutura desenvolvida contempla 18 tabelas inter-relacionadas, com destaque para entidades centrais como “cliente”, “hotel”, “reserva” e “quarto”. A modelagem também abrange aspectos específicos do domínio, como a distinção entre clientes “pessoa física” e “pessoa jurídica”, o relacionamento entre reservas e serviços contratados, e a associação entre hotéis e seus respectivos atendentes, contatos e endereços. Os relacionamentos foram implementados com o uso de chaves estrangeiras e regras de integridade referencial, garantindo a consistência dos dados, evitando redundâncias em operações de inserção, atualização e exclusão.

Este trabalho busca não apenas apresentar a modelagem e implementação do banco de dados, mas também demonstrar sua aplicabilidade prática em cenários reais, evidenciando a importância de um projeto de banco de dados bem estruturado para a eficiência operacional e tomada de decisão em ambientes corporativos.

2 MODELAGEM CONCEITUAL

A modelagem conceitual é uma etapa fundamental no desenvolvimento de sistemas de banco de dados, pois permite representar de forma abstrata os elementos essenciais para resolução do problema, bem como os relacionamentos entre eles e facilitar a construção na linguagem SQL posteriormente. Para o sistema de gerenciamento de hotelaria proposto, foi elaborado um modelo conceitual que contempla todas as entidades bases e interações necessárias ao funcionamento do sistema, com o objetivo de garantir integridade, normalização e clareza na estruturação dos dados.

2.1 Entidades e Atributos

O modelo é composto por diversas entidades que representam os principais componentes do sistema:

- **Cliente**: representa o cliente do hotel, podendo ser pessoa física ou jurídica. Possui os atributos “id_cliente” (chave primária) e “status”;

- **Pessoa Física**: subentidade de “Cliente” (especialização), inclui os atributos “nome”, “CPF” e “data de nascimento”;

- **Pessoa Jurídica**: também subentidade de “Cliente”, composta pelos atributos “CNPJ”, “razão social” e “nome fantasia”;

- **Hotel**: entidade que representa cada unidade hoteleira. Possui os atributos “id_hotel” (chave primária) e “nome fantasia”;

- **Atendente**: funcionário responsável por processos de atendimento e reserva. Contém os atributos “id_atendente”(chave primária), “id_hotel” (chave estrangeira), “CPF”, “data de nascimento” e “nome”;

- **Reserva**: armazena informações sobre a hospedagem de um cliente, incluindo “id_reserva” (chave primária), “id_cliente” (chave estrangeira), “id_quarto” (chave estrangeira), “id_atendente” (chave estrangeira), “data de entrada”, “data de saída” e “status”;

- **Quarto**: representa os quartos disponíveis no hotel, com os atributos “id_quarto” (chave primária), “id_hotel” (chave estrangeira), “preço diário”, “nome”, “tipo”, “capacidade” e “status”;

- **Serviços**: define os serviços adicionais disponíveis para contratação, como café da manhã, almoço ou jantar. Atributos incluem “id_serviço” (chave primária), “tipo” e “preço”;

- **Pagamento**: vinculado à reserva, armazena os dados de pagamento com os atributos “id_pagamento” (chave primária), “id_reserva” (chave estrangeira), “forma de pagamento”, “valor”, “situação” e “data”;

- **Contrato**: contém os registros contratuais das reservas. Inclui os atributos “id_contrato” (chave primária), “id_reserva” (chave estrangeira), “número do contrato”, “data de início”, “data de fim” e “situação”;

- **HospedeReserva**: entidade associativa que representa hóspedes vinculados a uma reserva, podendo incluir acompanhantes. Contém “id_cliente” (chave primária/estrangeira), “id_reserva” (chave primária/estrangeira), “tipo de hóspede” e “data de check-in";

- **Telefone**: armazena os contatos telefônicos de clientes e hotéis. Possui “id_telefone” (chave primária), “id_cliente” (chave estrangeira), “id_hotel” (chave estrangeira), “DDD”, “número” e “tipo”;

- **Email**: semelhante à entidade Telefone, armazena “id_email” (chave primária, “id_cliente” (chave estrangeira), “id_hotel” (chave estrangeira) e “email”;

- **Endereço**: armazena dados de localização associados a clientes e hotéis. Inclui “id_endereço” (chave primária), “id_cliente” (chave estrangeira), “id_hotel” (chave estrangeira), “logradouro”, “número”, “CEP”, “cidade”, “estado”, “bairro”, “país” e “complemento”;

- **Contrata**: entidade associativa que representa a contratação de um ou mais serviços em uma determinada reserva. Seus atributos são compostos pelas chaves estrangeiras “id_serviço” (chave primária/estrangeira) e “id_reserva” (chave primária/estrangeira).

2.2 Relacionamentos

Os relacionamentos entre as entidades são expressos por meio de ligações com cardinalidades e restrições bem definidas, garantindo coerência nos vínculos entre os dados. Entre os principais relacionamentos, destacam-se:

- O **cliente** pode ser uma **pessoa física** ou **jurídica (especialização)**, em uma relação 1:1;

- Um **cliente** pode possuir vários **endereços**, **telefones** e **e-mails** (relacionamentos 1:N);

- Um **hotel** pode possuir vários **quartos**, **atendentes**, **endereços**, **telefones** e **e-mails** (relacionamentos 1:N);

- Um **atendente** está relacionado a apenas um **hotel**, mas um hotel pode ter vários atendentes (N:1);

- A **reserva** é feita por um **cliente**, realizada por um **atendente**, e vinculada a um **quarto** (relacionamentos N:1);

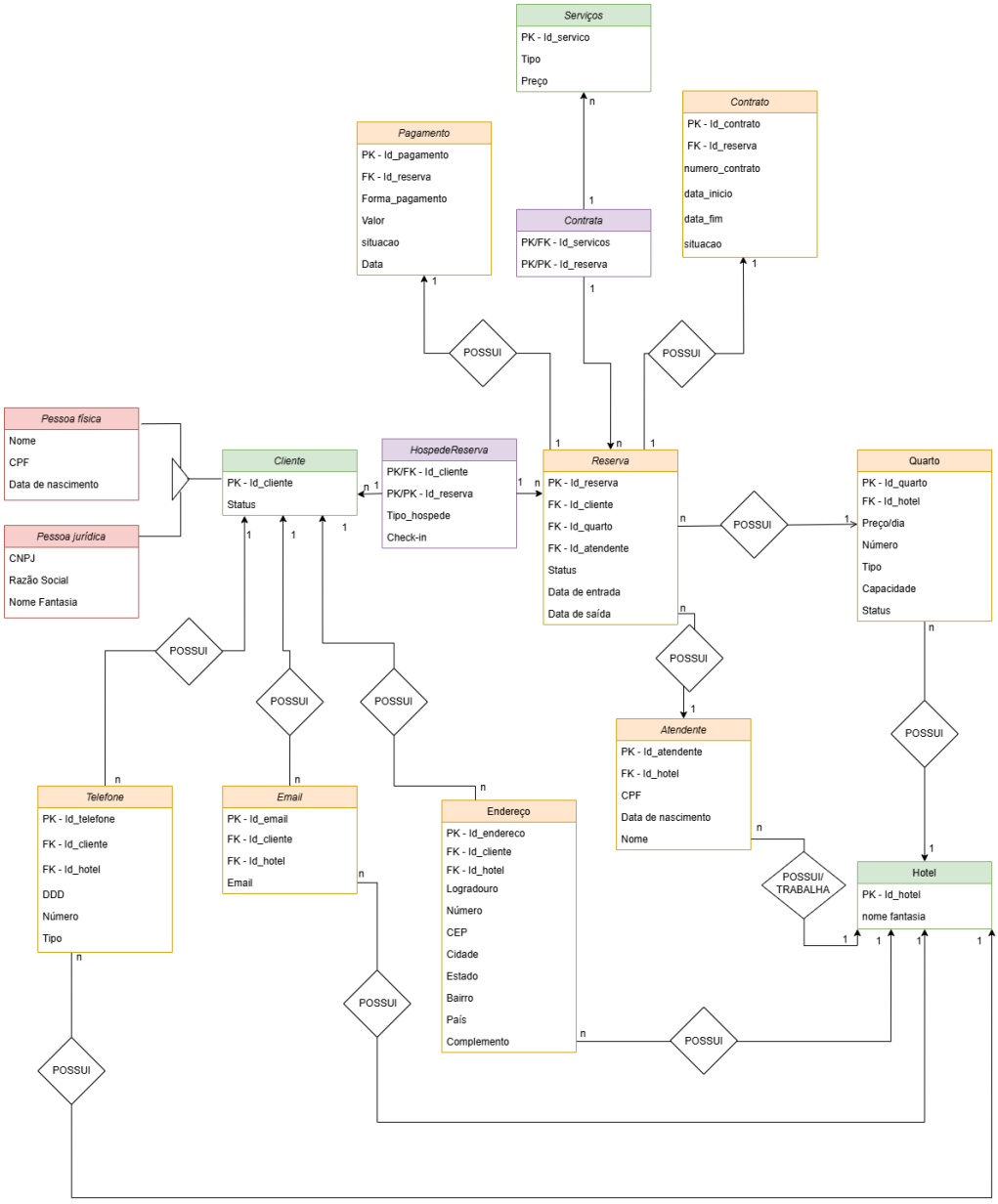
- Uma **reserva** pode possuir um ou mais **hóspedes** (clientes), representados na entidade associativa **HospedeReserva** (N:N);

- Cada **reserva** pode ter **um contrato** e **um pagamento** associado (1:1);

- A relação entre **serviços** e **reservas** ocorre através da entidade associativa **Contrata**, possibilitando que uma reserva inclua múltiplos serviços e vice-versa (N:N).

Este modelo conceitual fornece uma base sólida e bem estruturada para a futura implementação física do banco de dados, assegurando escalabilidade, integridade e organização das informações no contexto de um sistema hoteleiro.

Figura 1 – Diagrama de Entidade e Relacionamento



Fonte: Elaborado pelos autores.

3 MODELAGEM LÓGICA

O processo de modelagem de dados segue etapas fundamentais que visam transformar um modelo conceitual, representado por meio de um Diagrama Entidade-Relacionamento (DER), em um modelo lógico e, posteriormente, físico, de modo a facilitar a implementação em um Sistema Gerenciador de Banco de Dados (SGBD). A seguir, detalha-se as principais transformações realizadas a partir do DER para a estrutura de tabelas do banco de dados do sistema de reservas hoteleiras.

1. Entidades e Tabelas

Cada entidade presente no DER foi convertida diretamente em uma tabela no modelo relacional. Por exemplo, as entidades Cliente, Hotel, Quarto, Reserva, Atendente, Serviço e Contrato tornaram-se tabelas com o mesmo nome no banco de dados. Cada tabela recebeu uma chave primária, com a utilização do tipo SERIAL, que gera automaticamente os ID’s únicos.

2. Especialização/Generalização

A entidade Cliente foi especializada em duas subentidades: Pessoa Física e Pessoa Jurídica. Essa especialização foi implementada por meio da técnica de chave primária compartilhada. Assim, as tabelas “pessoa_fisica” e “pessoa_juridica” possuem atributos específicos, mas utilizam o campo “id” — um atributo comum entre elas — como chave primária, que também funciona como chave estrangeira referenciando a tabela cliente, a qual armazena os atributos compartilhados pelas especializações. Esse modelo garante a integridade referencial e preserva a relação de herança entre as tabelas.

3. Relacionamentos

Os relacionamentos entre entidades do DER foram representados por meio da transformação em chaves estrangeiras ou em tabelas associativas, dependendo da cardinalidade envolvida:

- Relacionamentos **1:N** foram implementados com o uso de **chaves estrangeiras**. Por exemplo:
    - A tabela “reserva” contém “cliente_id”, “quarto_id” e “atendente_id”, todos como chaves estrangeiras;

    - As tabelas “atendente” e “quarto” possuem a chave estrangeira “hotel_id”.

- Relacionamentos **N:N** foram mapeados para **tabelas associativas** — já representadas no DER — com chaves compostas. Exemplos:

    - A relação entre as tabelas “reserva” e “servico” foi representada pela tabela “contrata”, que contém “reserva_id” e “servico_id” como chaves estrangeiras e juntas formam a chave primária composta;

    - A relação entre as tabelas “reserva” e “cliente” para representar hóspedes adicionais foi mapeada pela tabela “hospede_reserva”.

4. Atributos compostos e multivalorados

Atributos como “endereço”, “telefone” e “e-mail”, que podem ocorrer múltiplas vezes para um mesmo cliente ou hotel, foram transformados em entidades independentes no modelo relacional, em conformidade com os princípios da **Primeira Forma Normal (1FN)**:

- A tabela “endereco” armazena os dados de localização com os campos de detalhamento (logradouro, número, cidade etc.), e contém as chaves estrangeiras “cliente_id” e “hotel_id” para indicar a associação com os respectivos registros;

- Da mesma forma, “telefone” e “email” foram transformados em tabelas próprias, com referências as tabelas “cliente” e “hotel”, permitindo múltiplos registros desses dados para cada entidade.

5. Restrições de Integridade

Foram utilizados diversos tipos de restrições com o objetivo de garantir a integridade dos dados, seguindo os princípios da **Terceira Forma Normal (3NF)** e da **Segunda Forma Normal (2NF)** como por exemplo:

- **Chaves primárias** (PRIMARY KEY) foram aplicadas a todas as tabelas para garantir a unicidade dos registros;

- **Chaves estrangeiras** (FOREIGN KEY) asseguram a integridade referencial entre os dados relacionados;

- **Restrições de domínio** (CHECK) foram utilizadas em colunas com valores restritos, como os tipos de serviço, tipo de quarto, tipo de telefone e tipo de endereço, que possuem valores padrões específicos;

- O campo “cpf” na tabela “pessoa_fisica” e o campo “cnpj” em “pessoa_juridica” possuem restrição de unicidade (UNIQUE) para evitar duplicidade de documentos;

- A tabela “pagamento” e “contrato” têm um relacionamento 1:1 com a tabela “reserva”, o que foi garantido pela cláusula “UNIQUE” sobre o campo “reserva_id”, garantindo com que cada reserva possua apenas um contrato e apenas um pagamento.

4 DESCRIÇÃO DAS TABELAS E RELACIONAMENTOS

A estrutura do banco de dados foi projetada para um sistema de reservas hoteleiras, contemplando as necessidades operacionais, cadastrais e transacionais de clientes, hotéis e reservas. A seguir, apresenta-se a descrição das tabelas, seus atributos e os relacionamentos entre elas.

1. Tabela cliente

A tabela cliente representa os usuários do sistema que realizam reservas. Possui os seguintes atributos:

- id: identificador único do cliente (chave primária, SERIAL);
- ativo: campo booleano que indica se o cliente está ativo no sistema.

Esta entidade é generalizada em duas subentidades: “pessoa_fisica” e “pessoa_juridica”.

2. Tabela pessoa_fisica

Tabela especializada de cliente, representa indivíduos. Seus atributos são:

- id: chave primária e chave estrangeira para a tabela cliente;
- nome: nome completo;
- cpf: número de CPF, com restrição de unicidade;
- nascimento: data de nascimento.

3. Tabela pessoa_juridica

Outra especialização de cliente, representa empresas. Contém:

- id: chave primária e chave estrangeira para cliente;
- nome_fantasia: nome comercial;
- razao_social: razão social da empresa;
- cnpj: número de CNPJ, também com restrição de unicidade.

4. Tabela hotel

Representa os hotéis disponíveis no sistema. Atributos:

- id: identificador único do hotel;
- nome_fantasia: nome comercial do hotel.

5. Tabela atendente

Armazena os funcionários responsáveis pelo atendimento de reservas. Atributos:

- id: chave primária;
- hotel_id: chave estrangeira referenciando a tabela “hotel”;
- nome: nome do atendente;
- cpf: número de CPF, único;

- nascimento: data de nascimento.

  Relação **N:1** com hotel.

6. Tabela quarto

Contém os dados dos quartos dos hotéis. Atributos:

- id: identificador do quarto;
- hotel_id: chave estrangeira para hotel;
- preco: valor da diária;
- nome: identificação do quarto;
- tipo: categoria (Suite ou Padrao);
- capacidade: número máximo de hóspedes;
- liberado: indica se está disponível para reserva.

Relação **N:1** com hotel.

7. Tabela reserva

Registra as reservas realizadas pelos clientes. Atributos:

- id: chave primária;
- cliente_id: chave estrangeira para a tabela “cliente”;
- quarto_id: chave estrangeira para a tabela “quarto”;

- atendente_id: chave estrangeira para a tabela “atendente”;
- concluido: booleano indicando se a reserva foi finalizada;
- data_entrada: data prevista de entrada;
- data_saida: data prevista de saída.

Relaciona-se com as tabelas “cliente”, “quarto” e “atendente” em relações **N:1**.

8. Tabela servico

Define os serviços adicionais oferecidos aos clientes. Atributos:

- id: chave primária;
- tipo: tipo de serviço (Nenhum, Café da manhã, Almoço, Jantar);
- preco: valor do serviço.

9. Tabela contrata

Tabela associativa entre as tabelas “reserva” e “servico”, representando um relacionamento **N:N**. Atributos:

- servico_id: chave estrangeira para “servico”;
- reserva_id: chave estrangeira para “reserva”.

Chave primária composta por ambos os campos.

10. Tabela hospede_reserva

Tabela associativa que relaciona múltiplos “clientes” a uma “reserva” (como hóspedes acompanhantes). Atributos:

- cliente_id: chave estrangeira para cliente;
- reserva_id: chave estrangeira para reserva.

Relacionamento **N:N**, com chave primária composta.

11. Tabela contrato

Registra os contratos firmados após as reservas. Atributos:

- id: chave primária;
- reserva_id: chave estrangeira para a tabela “reserva”, com restrição UNIQUE (1:1);
- numero_contrato: número identificador do contrato (único);
- data_inicio: data de início da vigência;
- data_fim: data de encerramento (opcional);
- situacao: estado atual do contrato (Ativo, Encerrado etc.).

12. Tabela pagamento

Registra os pagamentos das reservas. Atributos:

- id: chave primária;

- reserva_id: chave estrangeira para a tabela “reserva”, com UNIQUE;
- data_pagamento: data de efetivação;
- valor: valor pago;
- forma_pagamento: método de pagamento (ex: cartão, boleto);
- situacao: status do pagamento (Pago, Pendente etc.).

13. Tabela endereco

Armazena os endereços dos clientes e hotéis. Atributos:

- id: chave primária;
- cliente_id e hotel_id: chaves estrangeiras opcionais (um ou outro deve ser preenchido);
- logradouro, bairro, numero, complemento, cep, cidade, estado, pais: dados do endereço;
- tipo: se é comercial ou residencial.

14. Tabela telefone

Registra números de telefone de clientes e hotéis. Atributos:

- id: chave primária;
- cliente_id, hotel_id: chaves estrangeiras opcionais;
- ddd: código de área;

- numero: número do telefone;
- tipo: tipo de telefone (Fixo, Movel, Recado).

15. Tabela email

Armazena e-mails de contato. Atributos:

- id: chave primária;
- cliente_id, hotel_id: chaves estrangeiras opcionais;
- email: endereço eletrônico.

A modelagem segue os princípios de normalização até a Terceira Forma Normal (3FN), evitando redundância e promovendo integridade dos dados. O uso de especialização ─ como no caso de clientes para pessoa física e pessoa jurídica ─ tabelas associativas para relacionamentos N:N, e controle de integridade com chaves estrangeiras e restrições CHECK e UNIQUE confere solidez ao modelo e aderência às boas práticas em bancos de dados relacionais.

5 MANIPULAÇÃO DE DADOS

CÓDIGO 1 - DML com DTL

```
BEGIN;

SAVEPOINT ponto1;
/* 1.0 -- INSERIR TABELA
-------------------------------------------------------------------------------------------------------------------*/
INSERT INTO cliente (ativo) values
(TRUE), (TRUE), (FALSE), (TRUE), (FALSE), (TRUE);

/* 1.1 -- INSERINDO EM PESSOA FISICA */
INSERT INTO pessoa_fisica (id, nome, cpf, nascimento)
VALUES (1, 'Leonardo Rodrigues', '124.456.789-00', '2003-01-24'),
(2, 'Lucas Ianovski', '125.456.789-00', '1998-08-01'),
(3, 'Nicolas Cadique', '126.456.789-00', '2002-07-28');

/* 1.2 -- INSERINDO EM PESSOA JURIDICA */
INSERT INTO pessoa_juridica (id, nome, razao_social, cnpj)
VALUES (4, 'Kaualcáida', 'Kauan Cáida.ltda', '91.739.439/0001-96'),
(5, 'Matias', 'São Matias Roscas.ltda', '22.444.433/0001-79'),
(6, 'André', 'Talarica Quimicas.ltda', '05.070.470/0001-45');

/* 2 -- INSERINDO EM QUARTOS */
INSERT INTO hotel (id, nome_fantasia)
VALUES(1, 'Hotel Splannate');

INSERT INTO quarto (hotel_id, preco, nome, tipo, capacidade, liberado)
VALUES (1, 2.150, 'Suite meia noite', 'Suite', 40, FALSE),
(1, 1.500, 'Premium night', 'Padrao', 20, FALSE),
(1, 2.000, 'Suite manhã', 'Suite', 30, TRUE);

/* 3 -- INSERÇÃO COM DO ... BEGIN*/
DO $$
DECLARE
id INT;
BEGIN
-- Cliente 1
INSERT INTO cliente (ativo) VALUES (true) RETURNING id INTO id;
INSERT INTO pessoa_fisica (id, nome, cpf, nascimento)
VALUES (id, 'João Machado', '123.456.789-00', '1970-04-23');
INSERT INTO endereco (cliente_id, logradouro, numero, cidade, estado, cep, tipo)
VALUES (id, 'Rua das Flores', '100', 'São Paulo', 'SP', '01000-000', 'Residencial');
INSERT INTO telefone (cliente_id, ddd, numero, tipo)
VALUES (id, '11', '912345678', 'Movel');
INSERT INTO email (cliente_id, email)
```

```
VALUES (id, 'joao@email.com');

-- Cliente 2
INSERT INTO cliente (ativo) VALUES (true) RETURNING id INTO id;
INSERT INTO pessoa_fisica (id, nome, cpf, nascimento)
VALUES (id, 'Maria Souza', '987.654.321-00', '1985-08-12');
INSERT INTO endereco (cliente_id, logradouro, numero, cidade, estado, cep, tipo)
VALUES (id, 'Rua dos Sonhos', '200', 'Rio de Janeiro', 'RJ', '22000-000', 'Residencial');
INSERT INTO telefone (cliente_id, ddd, numero, tipo)
VALUES (id, '21', '998877665', 'Fixo');
INSERT INTO email (cliente_id, email)
VALUES (id, 'maria@email.com');

-- Cliente 3
INSERT INTO cliente (ativo) VALUES (true) RETURNING id INTO id;
INSERT INTO pessoa_fisica (id, nome, cpf, nascimento)
VALUES (id, 'Carlos Lima', '111.222.333-44', '1992-01-30');
INSERT INTO endereco (cliente_id, logradouro, numero, cidade, estado, cep, tipo)
VALUES (id, 'Av. Central', '300', 'Belo Horizonte', 'MG', '31000-000', 'Residencial');
INSERT INTO telefone (cliente_id, ddd, numero, tipo)
VALUES (id, '31', '934567890', 'Movel');
INSERT INTO email (cliente_id, email)
VALUES (id, 'carlos@email.com');
END $$;

/* 4 UPDATE
-------------------------------------------------------------------------------------------------------------*/

UPDATE pessoa_fisica
SET nome = 'João Pedro da Silva'
WHERE cpf = '123.456.789-00';

UPDATE cliente
SET ativo = FALSE
WHERE id = (
SELECT id FROM pessoa_juridica
WHERE cnpj = '05.070.470/0001-45'
);

/* 5 DELETE
-------------------------------------------------------------------------------------------------------------*/

DELETE FROM cliente
WHERE id = (
SELECT id FROM pessoa_fisica WHERE cpf = '111.222.333-44'
);
```

```
-- CASO OS SCRIPTS SEJAM EXECUTADOS COM SUCESSO:
COMMIT;

-- CASO OCORRA UM RESULTADO NÃO ESPERADO:
ROLLBACK TO SAVEPOINT ponto1;
```

Fonte: Elaborado pelos autores.

O presente código SQL demonstra uma aplicação prática da linguagem de manipulação de dados (DML) associada a mecanismos de controle transacional (DTL – Data Transaction Language) em um sistema de gestão hoteleira modelado em um banco de dados relacional.

A execução tem início com o comando “BEGIN”, seguido pela criação de um ponto de recuperação (SAVEPOINT ponto1), que permite o retorno parcial da transação em caso de falhas, visando preservar a integridade dos dados presentes na tabela.

A primeira etapa compreende a inserção de registros na tabela “cliente”, sendo que cada entrada é posteriormente especializada em “pessoa_fisica” ou “pessoa_juridica”, representando, respectivamente, clientes do tipo pessoa natural ou jurídica. Os dados são inseridos com associação direta ao campo “id”, que funciona como chave primária compartilhada entre as entidades generalizada (cliente) e especializadas (pessoa natural e jurídica), respeitando o princípio da herança em modelagem de dados.

A seguir, é inserida a estrutura de hospedagem, com um hotel e seus respectivos quartos, identificados por atributos como “nome”, “tipo”, “capacidade”, “preço” e “estado de liberação”. Tais inserções visam compor a base de dados necessária ao funcionamento do sistema.

Na sequência, por meio de um bloco “DO ... BEGIN”, são realizadas inserções encadeadas, com controle de variáveis (DECLARE) e uso do “RETURNING INTO”, que captura o “id” do cliente recém-inserido para realizar as inserções correspondentes em tabelas associadas: “pessoa_fisica”, “endereco”, “telefone” e “email”. Essa abordagem confere maior dinamismo e integridade referencial à inserção em múltiplas tabelas, tendo em vista que a partir do controle de variáveis fornecido por esta abordagem, a chance de erros é diminuta.

De forma complementar, podem ser executadas instruções de atualização (UPDATE), a fim de modificar atributos específicos de registros já existentes – como a alteração do nome de um cliente físico e a desativação de um cliente jurídico a partir de seu CNPJ. Posteriormente, aplica-se uma instrução de remoção (DELETE), que exclui um cliente físico com base em seu CPF.

Por fim, o controle da transação é concluído com “COMMIT”, persistindo as alterações em caso de sucesso, ou com “ROLLBACK TO SAVEPOINT ponto1”, que permite desfazer apenas parte da transação até o ponto estabelecido, oferecendo flexibilidade no tratamento de falhas e mantendo a consistência da base de dados.

Este script ilustra, de maneira robusta, o uso iniciante e intermediário de comandos DML e DTL para simular operações reais de cadastro, manutenção e exclusão de dados em um sistema de banco de dados relacional, com ênfase na atomicidade, consistência, isolamento e durabilidade (ACID) das transações.

CÓDIGO 2 – Exemplos DQL

```sql
SELECT pf.nome, pf.cpf
FROM pessoa_fisica pf
JOIN cliente c ON pf.id = c.id
JOIN reserva r ON c.id = r.cliente_id
WHERE r.data_entrada = '2025-06-10';


SELECT e.*, pf.nome, pf.cpf
FROM endereco e
JOIN cliente c ON e.cliente_id = c.id
JOIN pessoa_fisica pf ON c.id = pf.id
WHERE pf.cpf = '123.456.789-00';
```

Fonte: Elaborado pelos autores.

O código apresentado utiliza a Linguagem de Consulta de Dados (DQL), evidenciado especificamente pelo comando “SELECT”, com o intuito de extrair informações a partir de uma ou mais tabelas de um banco de dados relacional.

A primeira consulta visa retornar o nome e o CPF de clientes do tipo pessoa física, cuja participação em reservas está associada à data de entrada específica de '2025-06-10'. Para isto, emprega-se a junção (JOIN) entre as tabelas “pessoa_fisica”, “cliente” e “reserva”, utilizando como critério de ligação as chaves primárias compartilhadas entre as tabelas (id), respeitando o princípio da integridade referencial. Tal operação demonstra a capacidade de responder perguntas como: “quais clientes realizaram check-in em determinada data?”

A segunda instrução tem como objetivo recuperar os dados completos de endereço (e.*), juntamente com o nome e CPF da pessoa física vinculada, tendo como filtro o número do CPF. A consulta efetua junções entre as tabelas “endereco”, “cliente” e “pessoa_fisica”, novamente tendo como base as correspondências entre chaves primárias para reconstruir o relacionamento entre os registros. Esta consulta é representativa da capacidade do sistema de fornecer informações cadastrais, o que pode ser utilizado para fins administrativos ou de atendimento personalizado.

Ambas as consultas são fundamentais para a verificação de dados de suma importância, sendo exemplos clássicos de como uma base de dados bem estruturada permite extrações precisas, rápidas e confiáveis, fomentando a tomada de decisão baseada em dados e a realização de processos internos com eficiência.

6 CONTROLE DE ACESSO

CÓDIGO 3 – Controle de acesso com CTL

```
/* Criação de usuários */
CREATE USER adminitrador_01 WITH PASSWORD 'admin2025';
CREATE USER atendente_01 WITH PASSWORD 'atend0125';

/* Concede todas as permissões para o administrador*/
GRANT ALL PRIVILEGES ON SCHEMA public TO administrador_01;
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO administrador_01;
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO
administrador_01;

/* Permissões futuras */
ALTER DEFAULT PRIVILEGES IN SCHEMA public
GRANT ALL PRIVILEGES ON TABLES TO administrador_01;

ALTER DEFAULT PRIVILEGES IN SCHEMA public
GRANT ALL PRIVILEGES ON SEQUENCES TO administrador_01;

/* Concede permissão de apenas acessar ao schema */
GRANT USAGE ON SCHEMA public TO atendente_01;

/* Concede permissão de leitura em todas as tabelas e sequências */
GRANT SELECT ON ALL TABLES IN SCHEMA public TO atendente_01;
GRANT SELECT ON ALL SEQUENCES IN SCHEMA public TO atendente_01;

/* Reservas */
GRANT SELECT, INSERT, UPDATE ON reserva TO atendente_01;

/* Clientes */
GRANT SELECT, INSERT, UPDATE ON cliente TO atendente_01;
GRANT SELECT, INSERT, UPDATE ON pessoa_fisica TO atendente_01;
GRANT SELECT, INSERT, UPDATE ON pessoa_juridica TO atendente_01;

/* Pagamentos */
GRANT SELECT, INSERT, UPDATE ON pagamento TO atendente_01;

/* Contratos (gerados com a reserva) */
GRANT SELECT, INSERT, UPDATE ON contrato TO atendente_01;

/* Dados de contato dos clientes */
GRANT SELECT, INSERT, UPDATE ON endereco TO atendente_01;
GRANT SELECT, INSERT, UPDATE ON telefone TO atendente_01;
GRANT SELECT, INSERT, UPDATE ON email TO atendente_01;
```

```
/* Hospedes */
GRANT SELECT, INSERT, DELETE ON hospede_reserva TO atendente_01;

/* Serviços */
GRANT SELECT, INSERT, DELETE ON contrata TO atendente_01;

/* Permissão de apenas leitura para tabelas futurasz */
ALTER DEFAULT PRIVILEGES IN SCHEMA public
GRANT SELECT ON TABLES TO atendente_01;

ALTER DEFAULT PRIVILEGES IN SCHEMA public
GRANT SELECT ON SEQUENCES TO atendente_01;
```

Fonte: Elaborado pelos autores.

A administração de acessos a um banco de dados é um aspecto crucial para a segurança, organização e integridade do sistema de informação. O código CTL (Command Transaction Language) apresentado tem como objetivo definir e atribuir permissões de acesso a diferentes perfis de usuários, garantindo que cada grupo tenha acesso apenas às informações e operações necessárias à sua função no sistema.

O perfil de administrador foi criado para ter acesso total ao banco de dados. Isso inclui a permissão para realizar todas as operações sobre tabelas, sequências e objetos do banco, como inserir, atualizar, excluir e consultar dados, além de criar e modificar a estrutura do banco, como adicionar novas tabelas, alterar esquemas ou gerenciar outros usuários. Esse nível de permissão é fundamental para garantir que o administrador possa manter, expandir e garantir o funcionamento seguro e eficiente do sistema.

Por outro lado, o perfil de atendente foi configurado com um nível de acesso mais restrito, condizente com as atividades operacionais que realiza no sistema. Esse usuário possui acesso de leitura geral a todo o banco de dados, o que permite consultar qualquer informação necessária para o atendimento ao cliente ou para o gerenciamento das operações diárias. Contudo, ele só tem permissão para inserir e atualizar dados nas tabelas que fazem parte diretamente de suas atividades rotineiras, como reservas, cadastros de clientes, contratos, pagamentos e dados de contato. Além disso, também tem autorização para remover registros relacionados à hospedagem e à contratação de serviços, que são dinâmicos e suscetíveis a alterações no processo de atendimento.

Outro ponto relevante do código é que ele estabelece uma configuração padrão para futuras tabelas que venham a ser criadas no “schema” do banco de dados. Isso garante que, automaticamente, o administrador mantenha todos os privilégios completos, enquanto o atendente continuará com permissões de leitura nas novas estruturas. Esse recurso é extremamente importante para manter a coerência da política de segurança e acesso, evitando falhas administrativas caso ocorram expansões no banco.

Portanto, a aplicação desse código é essencial não apenas para a segurança dos dados, mas também para a organização dos processos, evitando que usuários operacionais tenham acesso a funções críticas do banco, preservando assim a integridade e a estabilidade do sistema.

7 CONCLUSÃO

O desenvolvimento do banco de dados relacional para um sistema de gestão hoteleira foi conduzido de forma progressiva e sistemática, iniciando-se com a modelagem conceitual e lógica da base de dados, seguida da implementação prática por meio de scripts SQL. O processo contemplou a criação das tabelas e relacionamentos com o uso de DDL (Data Definition Language), assegurando a normalização e a integridade referencial por meio de chaves primárias e estrangeiras.

Posteriormente, procedeu-se à inserção e manipulação dos dados utilizando DML (Data Manipulation Language) e a asseguração de sua integridade a partir do DTL (Data Transaction Language), com comandos “INSERT”, “UPDATE” e “DELETE”, organizados em blocos transacionais com “BEGIN”, “SAVEPOINT”, “ROLLBACK” e “COMMIT”, demonstrando práticas de controle de transações e consistência em operações críticas. Por fim, com a aplicação de DQL (Data Query Language), realizaram-se consultas relacionais com junções múltiplas (JOIN), voltadas à recuperação de dados específicos para fins analíticos e operacionais.

Ao longo do projeto, diversos aprendizados relevantes foram assimilados, sendo a principal delas a importância de um planejamento posterior e de uma modelagem de dados robusta e bem estruturada, capaz de sustentar as operações de manipulação e consulta sem comprometer a integridade das informações. A prática com transações demonstrou o valor dos mecanismos de controle de consistência, oferecendo maior segurança em operações críticas.

Além disso, foi possível compreender a necessidade de clareza semântica nas estruturas de dados e nas consultas, destacando como uma base relacional bem normalizada pode gerar resultados precisos e confiáveis mesmo diante de múltiplas dependências entre as entidades.

Ainda que o projeto atenda aos requisitos propostos, há espaço para muita evolução e refinamento. Dentre estas melhorias, podemos citar a normalização até a Quinta Forma Normal (5NF) como um exemplo, que visa eliminar ainda mais as redundâncias existentes em bancos de dados SQL, mesmo que menos comumente utilizadas. Além disso, poderíamos levar em consideração também a automatização de processos utilizando "triggers", que responderiam certos eventos com ações específicas para aqueles eventos,

como atualizar automaticamente o status de um quarto quando for desocupado.

Em síntese, este projeto constituiu-se como uma sólida aplicação dos fundamentos de banco de dados relacionais, integrando teoria e prática com foco na aplicabilidade real em sistemas administrativos, e abrindo caminho para aprimoramentos técnicos futuros.

8 REFERÊNCIAS

- HEUSER, Carlos A. *Projeto de Banco de Dados*. Bookman.
- SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. *Sistemas de Banco de Dados*. McGraw-Hill.
- CHEN, Peter. *The Entity-Relationship Model: Toward a Unified View of Data*. ACM Transactions on Database Systems, 1976.
- DATE, C. J. *Introdução a Sistemas de Banco de Dados*. Campus.
- PostgreSQL Official Docs – SQL Commands: https://www.postgresql.org/docs/current/sql-commands.html
- PostgreSQL Data Types: https://www.postgresql.org/docs/current/datatype.html
- Databricks Transações ACID: https://www.databricks.com/br/glossary/acid-transactions#:~:text=ACID%20é%20uma%20sigla%20para,ser%20chamada%20de%20transação%20ACID
- Hashtag treinamentos: https://www.hashtagtreinamentos.com/trigger-em-sql