Aula 4 — Criação do Banco e manipulação de dados

Objetivos da Aula

- Revisar o modelo lógico do projeto ABP.
- Revisar e aplicar 1ª, 2ª e 3ª Forma Normal no *clima_alerta*.
- Produzir versão **normalizada** do modelo lógico.
- Criar o banco de dados *clima_alerta*.
- Criar as tabelas normalizadas (versão inicial do schema).
- Garantir chaves primárias, estrangeiras e restrições básicas.

---

1) O que é Normalização?

É um processo para **organizar tabelas e colunas** de modo a:

- Evitar **redundância** de dados.
- Garantir **integridade**.
- Facilitar manutenção e consultas.

---

2) As Formas Normais — passo a passo com exemplos

1FN (Primeira Forma Normal)

- Regra: **sem atributos multivalorados ou compostos**.
- Cada coluna armazena apenas **um valor atômico**.

Exemplo errado (não normalizado):

USUARIO(id_usuario, nome, email, telefones)

- O atributo telefones pode ter vários números.

Correção (1FN):

USUARIO(id_usuario, nome, email)

TELEFONE(id_telefone, numero, id_usuario FK)

2FN (Segunda Forma Normal)

- Regra: estar na 1FN e **nenhum atributo não-chave depende de parte da chave primária**.
- Afeta tabelas com **chave composta**.

Exemplo errado (não normalizado):

RELATO(id_evento, id_usuario, data_hora, nome_usuario)

- nome_usuario depende só de id_usuario, não da chave composta inteira.

Correção (2FN):

RELATO(id_relato PK, id_evento FK, id_usuario FK, data_hora)

USUARIO(id_usuario PK, nome)

3FN (Terceira Forma Normal)

- Regra: estar na 2FN e **sem dependência transitiva** (coluna que depende de outra coluna não-chave).

Exemplo errado:

LOCALIZACAO(id_localizacao, cidade, estado, nome_estado)

- nome_estado depende de estado, não da PK.

Correção (3FN):

LOCALIZACAO(id_localizacao, cidade, estado)

ESTADO(sigla_estado PK, nome_estado)

---

3) Aplicando no projeto clima_alerta

Passo 1 — Revisar tabelas (modelo lógico da Aula 4):

- tipo_evento
- localizacao
- usuario
- evento
- relato
- alerta

Passo 2 — Procurar problemas

- Usuário pode ter múltiplos **telefones** → criar tabela TELEFONE (1FN).
- Nome do usuário não deve estar em RELATO, apenas FK (2FN).
- Se quisermos guardar nome_estado, separar em tabela ESTADO (3FN).

Passo 3 — Modelo normalizado (resumido)

TIPO_EVENTO(id_tipo_evento PK, nome, descricao)

LOCALIZACAO(id_localizacao PK, latitude, longitude, cidade, sigla_estado FK)

ESTADO(sigla_estado PK, nome_estado)

USUARIO(id_usuario PK, nome, email UNIQUE, senha_hash)

TELEFONE(id_telefone PK, numero, id_usuario FK)

EVENTO(id_evento PK, titulo, descricao, data_hora, status,
id_tipo_evento FK, id_localizacao FK)

RELATO(id_relato PK, texto, data_hora,
id_evento FK, id_usuario FK)

ALERTA(id_alerta PK, mensagem, data_hora, nivel,
id_evento FK)

---

4) Criando o Banco de Dados

Passo 1 — Acessando o PostgreSQL

- Se for no pgAdmin: botão direito em **Databases → Create Database**.

Passo 2 — Criar o banco *clima_alerta*

```
CREATE DATABASE clima_alerta;
```

Passo 3 — Conectar no banco

- pelo pgAdmin → conectar no novo banco.

---

5) Criando Tabelas (usando modelo normalizado)

Agora vamos **traduzir o modelo lógico normalizado da Aula** em SQL.
Exemplo com as tabelas principais:

```
-- Tabela de Tipos de Evento
CREATE TABLE tipo_evento (
  id_tipo_evento SERIAL PRIMARY KEY,
  nome VARCHAR(100) NOT NULL,
  descricao TEXT
);
```

```
-- Tabela de Estados
CREATE TABLE estado (
    sigla_estado CHAR(2) PRIMARY KEY,
    nome_estado VARCHAR(100) NOT NULL
);

-- Tabela de Localização
CREATE TABLE localizacao (
    id_localizacao SERIAL PRIMARY KEY,
    latitude NUMERIC(9,6) NOT NULL,
    longitude NUMERIC(9,6) NOT NULL,
    cidade VARCHAR(100) NOT NULL,
    sigla_estado CHAR(2) NOT NULL REFERENCES estado(sigla_estado)
);

-- Tabela de Usuários
CREATE TABLE usuario (
    id_usuario SERIAL PRIMARY KEY,
    nome VARCHAR(150) NOT NULL,
    email VARCHAR(150) UNIQUE NOT NULL,
    senha_hash VARCHAR(255) NOT NULL
);

-- Tabela de Telefones do Usuário
CREATE TABLE telefone (
    id_telefone SERIAL PRIMARY KEY,
    numero VARCHAR(20) NOT NULL,
    id_usuario INT NOT NULL REFERENCES usuario(id_usuario)
);
```

```
-- Tabela de Eventos
CREATE TABLE evento (
    id_evento SERIAL PRIMARY KEY,
    titulo VARCHAR(150) NOT NULL,
    descricao TEXT,
    data_hora TIMESTAMP NOT NULL,
    status VARCHAR(30) CHECK (status IN ('Ativo','Em Monitoramento','Resolvido')),
    id_tipo_evento INT NOT NULL REFERENCES tipo_evento(id_tipo_evento),
    id_localizacao INT NOT NULL REFERENCES localizacao(id_localizacao)
);

-- Tabela de Relatos
CREATE TABLE relato (
    id_relato SERIAL PRIMARY KEY,
    texto TEXT NOT NULL,
    data_hora TIMESTAMP NOT NULL,
    id_evento INT NOT NULL REFERENCES evento(id_evento),
    id_usuario INT NOT NULL REFERENCES usuario(id_usuario)
);

-- Tabela de Alertas
CREATE TABLE alerta (
    id_alerta SERIAL PRIMARY KEY,
    mensagem TEXT NOT NULL,
    data_hora TIMESTAMP NOT NULL,
    nivel VARCHAR(20) CHECK (nivel IN ('Baixo','Médio','Alto','Crítico')),
    id_evento INT NOT NULL REFERENCES evento(id_evento)
);
```