Universidade de Brasília - UnB
Dep. Ciência da Computação
CIC0097 - BANCOS DE DADOS - Turma 01

Projeto de Banco de Dados para Projetos de Pesquisa vinculados à UnB

Ricardo de Carvalho Nabuco	Giovanni Daldegan	Rafael Oliveira Bonach
231021360	232002520	221008365

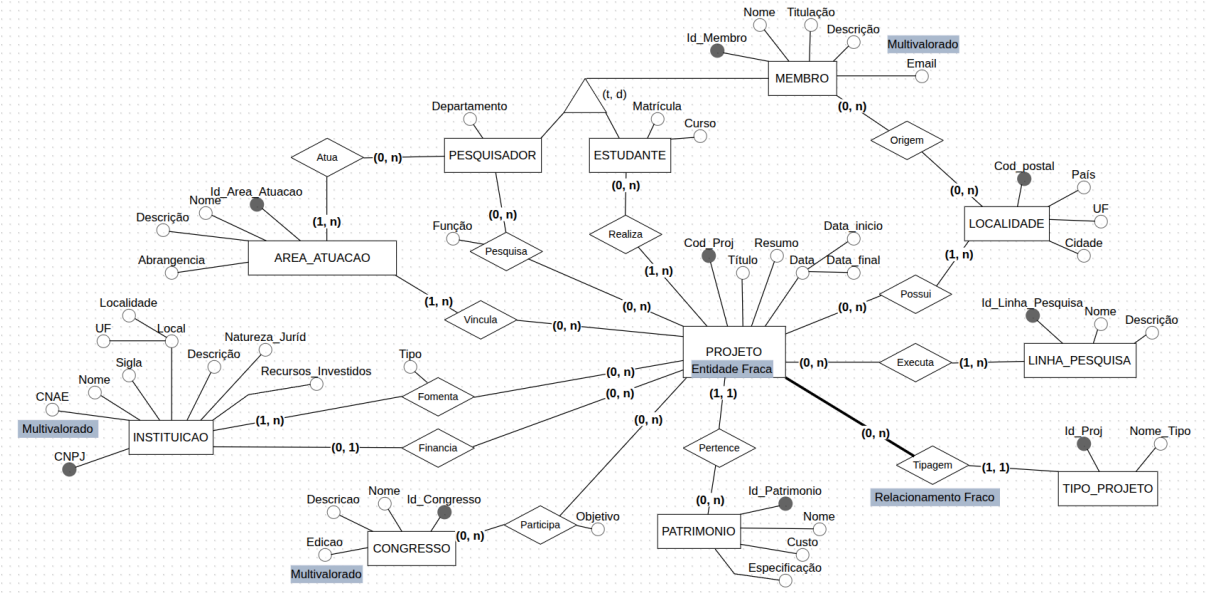
Introdução

Este projeto consiste na idealização de um banco de dados que centraliza e trata os Projetos de Pesquisa da Universidade de Brasília (UnB), informando dados importantes de cada projeto, como os professores orientadores, alunos participantes, área de atuação, instituições fomentadoras e financiadoras, participações em congressos, entre outras informações essenciais para a gestão da área de pesquisa da UnB. Esse gerenciamento é simulado no sistema desenvolvido para o projeto, disponível no link para seu repositório no GitHub ao final do documento.

Para definir os requisitos e modelagem do banco de dados, foram utilizados como referência o Painel analítico dos Grupos de Pesquisa da UnB e o Diretório de Grupos de Pesquisa no Brasil da Plataforma Lattes do CNPq.

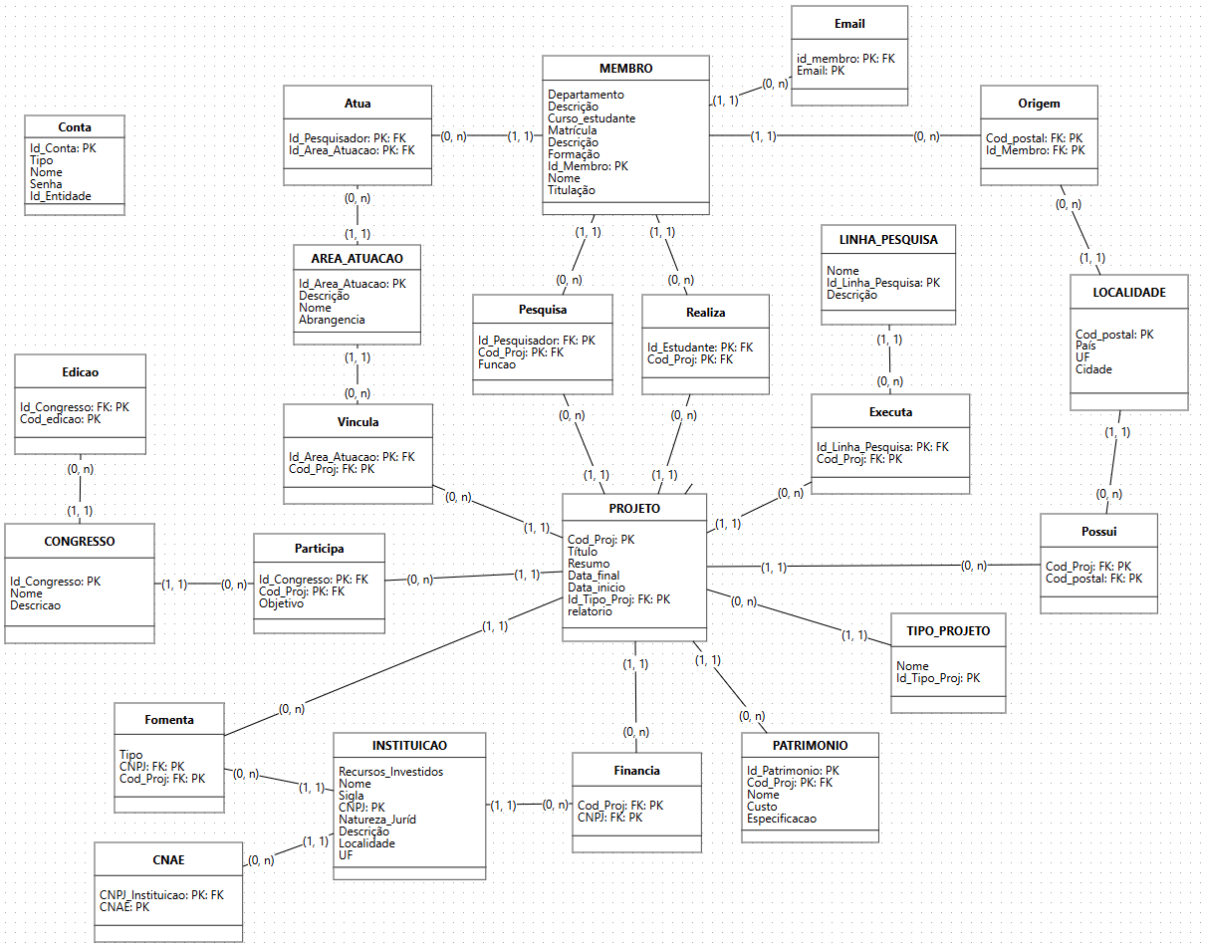
Modelo Entidade Relacionamento

Considerando os requisitos definidos, usando a ferramenta BR Modelo Web, o seguinte modelo entidade relacionamento (MER) foi desenvolvido, estabelecendo as relações lógicas entre as entidades pertencentes a ele.



Modelo Relacional

Com o modelo entidade relacionamento em mãos, utilizou-se a função do BR Modelo Web de converter um MER em modelo relacional (MR) e foi corrigido e refinado o resultado da conversão. É importante pontuar que toda instituição precisa de pelo menos um registro de CNAE, ou seja, é obrigatório que todo registro da tabela INSTITUIÇÃO seja referenciado por pelo menos um registro da tabela CNAE, significando uma cardinalidade CNAE (1,N) : (1,1) INSTITUIÇÃO na prática. No geral, foi necessário fazer poucas correções no modelo convertido e determinar os tipos e *constraints* de cada atributo.



Código SQL

O código SQL a seguir foi utilizado para criar todas as tabelas do MR do banco de dados. Para visualizar os códigos SQL de criação das *procedures* utilizadas para o CRUD do sistema e das *views* necessárias para a visualização eficiente dos dados armazenados, além dos demais códigos do sistema em Python, acesse o link do repositório no GitHub ao final do documento ou clicando no link a seguir: https://github.com/RafaBonach/BD-Pesquisas-UnB.

```
CREATE TABLE MEMBRO (
  Id_Membro         SERIAL        PRIMARY KEY,
  Nome              VARCHAR(45)   NOT NULL,
  Titulação         VARCHAR(15)   NOT NULL,
```

```
  Descrição        text,

  /* Atributo de Pesquisador da UnB */
  Departamento     VARCHAR(30),

  /* Atributos de Estudante da UnB */
  Matrícula        INT          UNIQUE,
  Curso_estudante  VARCHAR(30)
);

/* Projeto tem um tipo */
CREATE TABLE TIPO_PROJETO (
  Id_Tipo_Proj  INT          GENERATED ALWAYS AS IDENTITY,
  Nome_Tipo     VARCHAR(40)  NOT NULL,

  PRIMARY KEY  (Id_Tipo_Proj)
);

CREATE TABLE PROJETO (
  Cod_Proj        INT          GENERATED ALWAYS AS IDENTITY,
  Id_Tipo_Proj    INT,
  Título          VARCHAR(200)  NOT NULL,
  Data_final      DATE          NOT NULL,
  Data_inicio     DATE          NOT NULL,
  Resumo          text,
  relatorio       bytea,

  PRIMARY KEY  (Cod_Proj, Id_Tipo_Proj),
  UNIQUE       (Cod_Proj),
  FOREIGN KEY  (Id_Tipo_Proj)    REFERENCES TIPO_PROJETO(Id_Tipo_Proj) ON
DELETE CASCADE
);

CREATE TABLE INSTITUICAO (
  CNPJ                bigint       PRIMARY KEY,
  Nome                VARCHAR(50)  NOT NULL,
  Sigla               VARCHAR(10),
  Natureza_Juríd      VARCHAR(60),
  UF                  CHAR(2)      NOT NULL,
  Localidade          VARCHAR(50)  NOT NULL,
  Recursos_Investidos  bigint,
  Descrição           text
);

CREATE TABLE CONGRESSO (
  Id_Congresso  INT          GENERATED ALWAYS AS IDENTITY,
```

```
  Nome          VARCHAR(45)  NOT NULL,
  Descricao     text,

  PRIMARY KEY  (Id_Congresso)
);

CREATE TABLE LINHA_PESQUISA (
  Id_Linha_Pesquisa  INT          GENERATED ALWAYS AS IDENTITY,
  Nome               VARCHAR(120)  NOT NULL,
  Descrição          text,

  PRIMARY KEY  (Id_Linha_Pesquisa)
);

CREATE TABLE AREA_ATUACAO (
  Id_Area_Atuacao  INT          GENERATED ALWAYS AS IDENTITY,
  Abrangencia      VARCHAR(40)  NOT NULL,
  Nome             VARCHAR(45)  NOT NULL,
  Descrição        text,

  PRIMARY KEY  (Id_Area_Atuacao)
);

CREATE TABLE LOCALIDADE (
  Cod_postal  INT          PRIMARY KEY,
  País        VARCHAR(45)  NOT NULL,
  UF          CHAR(2)      NOT NULL,
  Cidade      VARCHAR(45)  NOT NULL
);

CREATE TABLE PATRIMONIO (
  Cod_Proj       INT,
  Id_Patrimonio  INT          GENERATED ALWAYS AS IDENTITY,
  Nome           VARCHAR(30)  NOT NULL,
  Custo          INT          NOT NULL,
  Especificacao  text,

  PRIMARY KEY  (Id_Patrimonio, Cod_Proj),
  FOREIGN KEY  (Cod_Proj)  REFERENCES  PROJETO(Cod_Proj) ON DELETE CASCADE
);


/* Email de membro (multivalorado) */
CREATE TABLE Email (
  Email      VARCHAR(30),
  id_membro  INT,
```

```
  PRIMARY KEY (Email, id_membro),
  FOREIGN KEY (id_membro) REFERENCES MEMBRO(id_Membro) ON DELETE CASCADE
);

/* Membro tem origem em Localidade */
CREATE TABLE Origem (
  Cod_postal  INT,
  Id_Membro   INT,

  PRIMARY KEY  (Id_Membro, Cod_postal),
  FOREIGN KEY  (Id_Membro)   REFERENCES MEMBRO(Id_Membro)
    ON DELETE CASCADE,
  FOREIGN KEY  (Cod_postal)  REFERENCES LOCALIDADE(Cod_postal)
    ON DELETE CASCADE
);

/* Pesquisador atua em Área de atuação */
CREATE TABLE Atua (
  Id_Pesquisador   INT,
  Id_Area_Atuacao  INT,

  PRIMARY KEY  (Id_Pesquisador, Id_Area_Atuacao),
  FOREIGN KEY  (Id_Pesquisador)   REFERENCES  MEMBRO(Id_Membro)
    ON DELETE CASCADE,
  FOREIGN KEY(Id_Area_Atuacao)    REFERENCES  AREA_ATUACAO(Id_Area_Atuacao)
    ON DELETE CASCADE
);

/* Pesquisador pesquisa Projeto */
CREATE TABLE Pesquisa (
  Id_Pesquisador  INT,
  Cod_Proj        INT,
  Funcao          VARCHAR(12)  NOT NULL,

  PRIMARY KEY  (Id_Pesquisador, Cod_Proj),
  FOREIGN KEY  (Id_Pesquisador)  REFERENCES MEMBRO(Id_Membro)
    ON DELETE CASCADE,
  FOREIGN KEY  (Cod_Proj)        REFERENCES PROJETO(Cod_Proj)
    ON DELETE CASCADE
);

/* Estudante realiza Projeto */
CREATE TABLE Realiza (
  Id_Estudante  INT,
  Cod_Proj      INT,
```

```
  PRIMARY KEY  (Id_Estudante, Cod_Proj),
  FOREIGN KEY  (Id_Estudante)  REFERENCES MEMBRO(Id_Membro)
    ON DELETE CASCADE,
  FOREIGN KEY  (Cod_Proj)      REFERENCES PROJETO(Cod_Proj)
    ON DELETE CASCADE
);


/* Projeto possui Localidade */
CREATE TABLE Possui (
  Cod_Proj    INT,
  Cod_postal  INT,

  PRIMARY KEY  (Cod_Proj, Cod_postal),
  FOREIGN KEY  (Cod_Proj)    REFERENCES PROJETO(Cod_Proj)
    ON DELETE CASCADE,
  FOREIGN KEY  (Cod_postal)  REFERENCES LOCALIDADE(Cod_postal)
    ON DELETE CASCADE
);

/* Projeto vincula Área de atuação */
CREATE TABLE Vincula (
  Cod_Proj         INT,
  Id_Area_Atuacao  INT,

  PRIMARY KEY  (Cod_Proj, Id_Area_Atuacao),
  FOREIGN KEY  (Cod_Proj)         REFERENCES PROJETO(Cod_Proj)
    ON DELETE CASCADE,
  FOREIGN KEY  (Id_Area_Atuacao)  REFERENCES AREA_ATUACAO(Id_Area_Atuacao)
    ON DELETE CASCADE
);

/* Projeto executa uma Linha de pesquisa */
CREATE TABLE Executa (
  Cod_Proj           INT,
  Id_Linha_Pesquisa  INT,

  PRIMARY KEY  (Cod_Proj, Id_Linha_Pesquisa),
  FOREIGN KEY  (Cod_Proj)           REFERENCES PROJETO(Cod_Proj)
    ON DELETE CASCADE,
  FOREIGN KEY(Id_Linha_Pesquisa) REFERENCES LINHA_PESQUISA(Id_Linha_Pesquisa)
    ON DELETE CASCADE
);

/* Projeto participa de um Congresso */
```

```sql
CREATE TABLE Participa (
  Id_Proj       INT,
  Id_Congresso  INT,
  Objetivo      text,

  PRIMARY KEY (Id_Proj, Id_Congresso),
  FOREIGN KEY (Id_Proj)       REFERENCES PROJETO(Cod_Proj)
    ON DELETE CASCADE,
  FOREIGN KEY (Id_Congresso)  REFERENCES CONGRESSO(Id_Congresso)
    ON DELETE CASCADE
);


/* Instituição tem múltiplos CNAE */
CREATE TABLE CNAE (
  CNPJ_Instituicao  bigint,
  CNAE              VARCHAR(20),

  PRIMARY KEY  (CNPJ_Instituicao, CNAE),
  FOREIGN KEY  (CNPJ_Instituicao)  REFERENCES  INSTITUICAO(CNPJ)
    ON DELETE CASCADE
);

/* Instituição financia Projeto */
CREATE TABLE Financia (
  Cod_Proj  INT,
  CNPJ      bigint,

  PRIMARY KEY  (Cod_Proj, CNPJ),
  FOREIGN KEY  (Cod_Proj)  REFERENCES PROJETO(Cod_Proj) ON DELETE CASCADE,
  FOREIGN KEY  (CNPJ)      REFERENCES INSTITUICAO(CNPJ) ON DELETE CASCADE
);

/* Instituição fomenta Projeto */
CREATE TABLE Fomenta (
  CNPJ      bigint,
  Cod_Proj  INT,
  Tipo      VARCHAR(20)  NOT NULL,

  PRIMARY KEY  (CNPJ, Cod_Proj),
  FOREIGN KEY  (CNPJ)      REFERENCES INSTITUICAO(CNPJ) ON DELETE CASCADE,
  FOREIGN KEY  (Cod_Proj)  REFERENCES PROJETO(Cod_Proj) ON DELETE CASCADE
);


/* Congresso tem edição */
```

```sql
CREATE TABLE Edicao (
  Id_Congresso  INT,
  Cod_edicao    INT,

  PRIMARY KEY  (Id_Congresso, Cod_edicao),
  FOREIGN KEY  (Id_Congresso)  REFERENCES CONGRESSO(Id_Congresso)
    ON DELETE CASCADE
);


/* Contas de usuário do sistema */
/* Tipo: 0, 1, 2, 3 -> instituiçao, pesquisador, estudante, colaborador
externo */
CREATE TABLE Conta (
  Id_Conta     SERIAL       PRIMARY KEY,
  Tipo         INT          NOT NULL,
  Nome         VARCHAR(15)  NOT NULL,
  Senha        CHAR(6)      NOT NULL,
  Id_Entidade  INT,

  UNIQUE (Nome, Senha, Id_Entidade)
);
```

Consultas em Álgebra relacional

1. Liste todos os dados da professora “Maristela”, seus emails e o nome dos projetos nos quais ela é coordenadora.

$$Maristela \leftarrow \sigma_{Nome='Maristela'}(MEMBRO)$$
$$Maristela\_email \leftarrow EMAIL \bowtie Maristela$$
$$Maristela\_proj \leftarrow \pi_{(Id\_Membro,\ Título)}(PROJETO \bowtie (Pesquisa \bowtie_{Funcao='Coordenador'} Maristela))$$

$$Maristela\_email\_proj \leftarrow Maristela\_email \bowtie Maristela\_proj$$

2. Liste o nome de pesquisadores, os projetos que eles fazem parte e os congressos que seus projetos participaram.

$$TB\_Proj\_Congr_0 \leftarrow \pi_{(Cod\_proj,\ Titulo,\ Nome)}(PROJETO \bowtie_{id\_proj=cod\_proj} (Participa \bowtie CONGRESSO))$$
$$TB\_Proj\_Congr \leftarrow \rho_{(Cod\_proj,\ Proj\_nome,\ Congr\_Nome)}(TB\_Proj\_Congr_0)$$
$$TB\_Membro\_info \leftarrow (MEMBRO \bowtie (Realiza \bowtie (Pesquisa \bowtie TB\_Proj\_Congr)))$$

$$\pi_{(Nome,\ Proj\_nome,\ Congr\_nome)}(MEMBRO \bowtie_{MEMBRO.id\_membro=TB\_Membro\_info.id\_membro} TB\_Membro\_info)$$

3. Liste o nome das instituições que fomentam projetos do tipo “Educação”.

$Proj\_educacao \leftarrow \pi_{Cod\_Proj}(\sigma_{Nome='Educação'}(TIPO\_PROJETO \bowtie PROJETO))$
$\pi_{Nome}(INSTITUICAO \bowtie (FOMENTA \bowtie Proj\_educacao))$

4. Selecione o nome de todos os projetos que fazem parte da linha de pesquisa “Psicologia Educacional”.

$\pi_{título}(\sigma_{projeto.título = 'Psicologia\ Educacional'}(PROJETO \bowtie (EXECUTE \bowtie LINHA\ PESQUISA)))$

5. Liste todos os colaboradores externos do país de origem “Espanha”.

$\pi_{Nome}(\sigma_{Pais = 'Espanha'}(LOCALIDADE \bowtie (ORIGEM \bowtie MEMBRO)))$

Avaliação das formas normais

Para a análise de formas normais, vamos considerar a relação TB_Inicial como a junção das tabelas PROJETO, INSTITUICAO, CNAE, Fomenta, e Financia (levando em conta que Fomenta e Financia representam relacionamentos entre as entidades Projeto e Instituição, ambas com chaves primárias {Cod_Proj, CNPJ}).

TB_Inicial (Cod_Proj, Título, Resumo, Data_final, Data_inicial, Id_Tipo_Proj, CNPJ, Nome, Sigla, Natureza_Juríd, Descrição, UF, Localidade, Recursos_investidos, CNAE, Tipo)

Tomemos {Cod_Proj, CNPJ} como chave candidata. Analisando quanto à Primeira Forma Normal (1FN), a relação está normalizada, já que cada coluna contém um valor indivisível.

Porém, quanto à Segunda Forma Normal (2FN), a relação está desnormalizada, pois a chave candidata {Cod_Proj, CNPJ} **não determina funcionalmente** os atributos complementares a ela, já que:

- Cod_Proj ➜ {Título, Resumo, Data_final, Data_inicial, Tipo}
- CNPJ ➜ {Nome, Sigla, Natureza_Juríd, Descrição, UF, Localidade, Recursos_investidos, CNAE}

Para normalizar a relação de acordo com a 2FN, é necessário dividir esses dados em 3 novas relações:

- TB_Projeto (Cod_Proj, Título, Resumo, Data_final, Data_inicial), com chave {Cod_Proj};
- TB_Instituicao_CNAE (CPNJ, Nome, Sigla, Natureza_Juríd, Descrição, UF, Localidade, Recursos_investidos, CNAE) com chave {CPNJ};
- TB_REL_Projeto_Instituicao (Cod_Proj, CPNJ, Tipo), onde {Cod_Proj, CPNJ} ➜ Tipo.

Agora, além de estarem normalizadas quanto à 2FN, essas relações também estão normalizadas de acordo com a Terceira Forma Normal (3FN), pois nenhum atributo complementar às chaves primárias determina funcionalmente outro atributo complementar, ou seja, não há **dependência transitiva**.

Porém, para otimizar o armazenamento desses dados, já que TB_REL_Projeto_Instituicao terá linhas com coluna “Tipo” nulo, podemos separá-la em duas relações, com significados distintos (como indicam os nomes):

- TB_Fomenta (Cod_Proj, CPNJ, Tipo), onde {Cod_Proj, CPNJ} ➜ Tipo;
- TB_Financia (Cod_Proj, CPNJ).

Similarmente, podemos separar TB_Instituicao_CNAE em duas relações:

- TB_Instituicao (CPNJ, Nome, Sigla, Natureza_Juríd, Descrição, UF, Localidade, Recursos_investidos), com chave CNPJ;
- TB_CNAE (CPNJ, CNAE), com chave {CNPJ, CNAE}.

Como resultado, temos as relações TB_Projeto, TB_Instituicao, TB_CNAE, TB_Fomenta, TB_Financia, equivalentes às tabelas originais PROJETO, INSTITUICAO, CNAE, Fomenta e Financia, respectivamente, demonstrando que essas tabelas estão normalizadas em relação às formas normais 1FN, 2FN e 3FN.

Repositório GitHub

Link para o repositório do sistema desenvolvido e os códigos SQL usados para gerar o banco de dados (incluindo *procedures* e *views*): https://github.com/RafaBonach/BD-Pesquisas-UnB

Referências

UNIVERSIDADE DE BRASÍLIA. **Decanato de Pesquisa e Inovação**. *Painel analítico dos Grupos de Pesquisa da UnB*. [S. l.], maio de 2020. Disponível em: http://pesquisa.unb.br/grupos-de-pesquisa/painel-analitico-grupos-de-pesquisa?menu=373. Acesso em: 7 maio 2025.

CNPQ. **Diretório dos Grupos de Pesquisa no Brasil**. *Consulta parametrizada*. [*S. l.*], 202-. Disponível em: http://dgp.cnpq.br/dgp/faces/consulta/consulta_parametrizada.jsf. Acesso em: 7 maio 2025.

POSTGRESQL: Documentation: 17. *PostgreSQL 17.5 Documentation*. [S. l.], 8 maio 2025. Disponível em: https://www.postgresql.org/docs/17/index.html. Acesso em: 3 jul. 2025.