# Relatório Técnico

**Guilherme de Lima de Menezes**[1], **Eduardo dos Santos Paim**[1], **Diogo Montanha Giordano**[1]

[1]Instituto de Informática – Universidade Federal do Pampa (UNIPAMPA)
Alegrete – RS – Brazil

{guilhermemenezes.aluno,eduardopaim.aluno,diogogiordano.aluno}@unipampa.edu.br

***Abstract.*** *The report outlines the development of a relational database project focusing on 2022 electoral processes from the TSE (Superior Electoral Court). The process includes normalizing tables up to the third normal form, creating logical and relational models, and implementing the physical database. Tools such as LucidChart, Data Grip, Microsoft Excel, Pycharm, and Postgres, along with languages like SQL and Python, were employed. The project enhances practical understanding of data modeling and its impact on system efficiency.*

***Resumo.*** *O relatório descreve o desenvolvimento de um projeto de banco de dados relacionais focado em processos eleitorais do TSE em 2022. O trabalho inclui a normalização das tabelas até a terceira forma normal, criação dos modelos lógico e relacional, e a implementação do banco de dados físico. Ferramentas como LucidChart, Data Grip, Microsoft Excel, Pycharm e Postgres, além de linguagens como SQL e Python, foram utilizadas. O projeto contribui para a compreensão prática da modelagem de dados e seu impacto na eficiência de sistemas.*

## 1. Introdução

O seguinte relatório tem como objetivo relatar os procedimentos adotados e durante o desenvolvimento do trabalho de banco de dados. O trabalho consiste em apresentar técnicas e métodos de modelagem de banco de dados relacionais. Foram aplicadas técnicas a partir de um banco de dados existente e abertos que armazena os processos que foram aberto no Tribunal Superior Eleitoral no ano de 2022

Nos capítulos seguintes teremos as seções: Contextualização2 do problema, Normalização3 que consta os processo de normalização das tabelas, Projeto lógico e relacional4 que aborda os modelos e diagramas criados pelo grupo , Projeto físico5 que consta a criação do e manipulação do banco de dados e visõe. Por fim o no capítulo Conclusões 6 abordaremos a considerações finais do trabalho.

## 2. Contextualização

Em todo o ano eleitoral o Tribunal Superior Eleitoral é bastante exigido com o aumento de processos eleitorais criados por diferentes atores e com diferentes interesses, com isso é importante que se tenha um banco de dados com esses processos contribuindo com um papel fundamental na garantia da integridade, transparência eficiência do sistema eleitoral em um país. Portanto aplicar os conceitos de engenharia reversa e poder manipular o banco de dados com consultas através de software torna a tarefa de análise menos cansativa.

No desenvolvimento do trabalho foi utilizados as seguinte ferramentas de apoio : LucidChart, Data Grip, Microsoft Excel, Pycharm, Postgres. Além disso, desenvolvemos utilizando a linguagem de programação SQL (Structured Query Language) e Python.

## 3. Normalização

A normalização em bancos de dados é um processo que visa organizar os dados de uma maneira eficiente, reduzindo a redundância e a dependência de dados

Com a análise dos dados contidos na base de dados escolhidas, foi verificado que os atributos contidos nas quatro tabelas (Processo, recursos, partes e decisões) não possuíam tabelas aninhadas, portanto, as tabelas já se encontrava na 1º forma normal por não ter tabelas aninhas e posteriormente fomos transformando até a 3º forma normal.

Na 2º forma normal foram retirada as dependências parciais e separadas as respectivas tabelas com suas chaves Mesmo após a 2º forma normal ainda foram encontrados problemas com dependência transitiva, então separamos algumas pequenas tabelas maiores como por exemplo relator e decisões em outras tabelas menores com suas chaves primárias, e sendo referenciadas como chave estrangeira 'FK' quando necessário.

### 3.1. Primeira Forma Normal

Na figura 1 verifica-se as tabelas Processos Eleitorais, Partes, Recursos e Decisões na primeira forma normal sem as chaves primárias definidas.

ProcessosEleitorais (DT_GERACAO, HH_GERACAO, ANO_ELEICAO, idProcesso, DT_AUTUACAO, DT_BAIXA, SG_UF_TRIBUNAL_ORIGEM, NR_INSTANCIA_ORIGEM, SG_UF_TRIBUNAL, NR_INSTANCIA, DT_DISTRIBUICAO, CD_TIPO_DISTRIBUICAO, DS_TIPO_DISTRIBUICAO, CD_RELATOR, NM_RELATOR, CD_TIPO_CARGO_RELATOR, DS_TIPO_CARGO_RELATOR, CD_CLASSE, SG_CLASSE, DS_CLASSE, CD_ASSUNTO_PRINCIPAL, DS_ASSUNTO_PRINCIPAL, ST_CONCLUSO, ST_EM_PAUTA, ST_SOBRESTADO, ST_PEDIDO_VISTA, ST_CARGA_VISTA_MPE,ST_RECURSAL, ST_REMESSA_SUPERIOR, QT_DECISOES)

Partes (DT_GERACAO, HH_GERACAO, ANO_ELEICAO, SG_UF_ORGAO, SG_TRIBUNAL_ORIGEM, NR_INSTANCIA, idProcesso, DS_POLO, TP_PARTE, ST_PARTE_PRINCIPAL, NM_PARTE, NM_SOCIAL_PARTE, ST_CANDIDATO, SQ_CANDIDATO)

Recursos (DT_GERACAO, HH_GERACAO, ANO_ELEICAO, DS_IDENTIFICACAO_RECURSO, NR_RECURSO,DT_AUTUACAO, DT_BAIXA, idProcesso_ORIGEM, SG_UF_TRIBUNAL_ORIGEM, NR_INSTANCIA_ORIGEM, SG_UF_TRIBUNAL, NR_INSTANCIA, DT_DISTRIBUICAO, CD_TIPO_DISTRIBUICAO, DS_TIPO_DISTRIBUICAO, CD_RELATOR, NM_RELATOR, CD_TIPO_CARGO_RELATOR, DS_TIPO_CARGO_RELATOR, CD_CLASSE, SG_CLASSE, DS_CLASSE, CD_ASSUNTO_PRINCIPAL, DS_ASSUNTO_PRINCIPAL, DS_TIPO_RECURSO, DS_NATUREZA_RECURSO, ST_CONCLUSO, QT_DECISOES, DT_ULTIMA_DECISAO, DS_ULTIMA_DECISAO)

Decisões (DT_GERACAO, HH_GERACAO, ANO_ELEICAO, NR_PROCESSO, SG_UF_TRIBUNAL_ORIGEM, SQ_DECISAO, DT_DECISAO, NM_AUTOR_DECISAO, DS_TIPO_DECISAO)

**Figure 1. 1º Forma Normal**

### 3.2. Segunda Forma Normal

Na figura 2 podemos visualizar as tabelas na segunda forma normal com as chaves primárias definidas e também com a criação das tabelas relator, assunto, classe.

Processo (idPROCESSO, SQ_DECISAO, idRecurso, CD_ASSUNTO_PRINCIPAL, CD_CLASSE,
CD_RELATOR, DT_GERACAO, HH_GERACAO, ANO_ELEICAO,DT_AUTUACAO,DT_BAIXA,SG_UF_TRIBUNAL_ORIGEM,NR_INSTANCIA_ORIGEM, SG_UF_TRIBUNAL,NR_INSTANCIA, DT_DISTRIBUICAO)
SQ_DECISAO referencia Decisao
idRecurso referencia Recurso
CD_ASSUNTO_PRINCIPAL referencia Assunto
CD_CLASSE referencia Classe
CD_RELATOR referencia Relator

Relator (CD_TIPO_CARGO_RELATOR, DS_TIPO_CARGO_RELATOR, NM_RELATOR).

Assunto (idPROCESSO, CD_ASSUNTO_PRINCIPAL, DS_ASSUNTO_PRINCIPAL)

Recursos (idRECURSO, idTipoRecurso, DS_TIPO_RECURSO, idNatureza, DS_NATUREZA_RECURSO, DS_IDENTIFICACAO_RECURSO, DT_AUTUACAO, DT_BAIXA, SG_UF_TRIBUNAL, NR_INSTANCIA, DT_GERACAO, HH_GERACAO, ANO_ELEICAO, SG_UF_TRIBUNAL_ORIGEM, NR_INSTANCIA_ORIGEM, ST_CONCLUSO)

Classe (CD_CLASSE, CD_DS_CLASSE, DS_CLASSE, SG_CLASSE )

Decisão (SQ_DECISAO, idDecisao, DS_TIPO_DECISAO , idAutorDecisao, NM_AUTOR_DECISAO, DT_GERACAO, HH_GERACAO, ANO_ELEICAO, SG_UF_TRIBUNAL_ORIGE, DT_DECISAO)

Partes (idParte, id_TipoParte, TP_PARTE, idPolo, DS_POLO, SQ_CANDIDATO, ST_PARTE_PRINCIPAL, NM_SOCIAL_PARTE, ST_CANDIDATO, NM_PARTE)

**Figure 2. 2º Forma Normal**

### 3.3. Terceira Forma Normal

Na figura 3 verifica-se normalização na terceira forma normal com as chaves primárias definidas e também com a tipo relator, tipo recursos, natureza classe descrição, tipo decisão, autor decisão, tipo parte e polo. Nesta tabela também podemos ver todas as chaves estrangeiras sendo referenciadas.

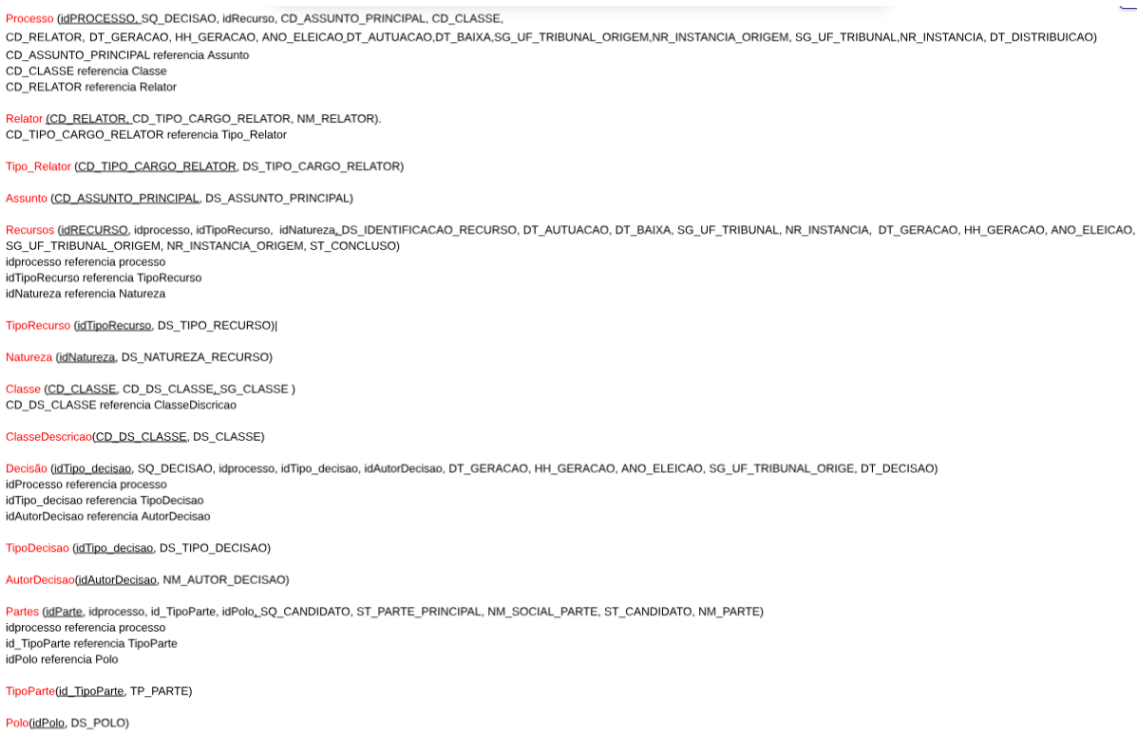


**Figure 3. 3º Forma Normal**

Não foi necessário colocar na quarta forma normal pois na terceira forma normal já satisfazia o problema de dependências.

## 4. Projeto Lógico e Relacional

Para melhor visualização dos modelos ER e Lógico pode-se acessar o LucidChart

O modelo entidade relacionamento é uma estrutura de dados de negócio criada em forma de um gráfico, com várias caixas representando atividades, funções e entidades. Esse modelo é usado para a criação de um banco de dados relacionais com elementos chave ideais para vincular essas tabelas.

Nesta figura 4 podemos visualizar o modelo entidade relacionamento que foi criado pelo grupo, ele sofreu alterações durante todo o processo mas aqui vamos apresentar o modelo final somente, o modelo anterior pode ser encontrado na pagina da WIKIdo grpo Já na figuraz5 podemos visualizar o modelo Lógico criado pelo grupo e que foi usado para modelar o bando de dados.

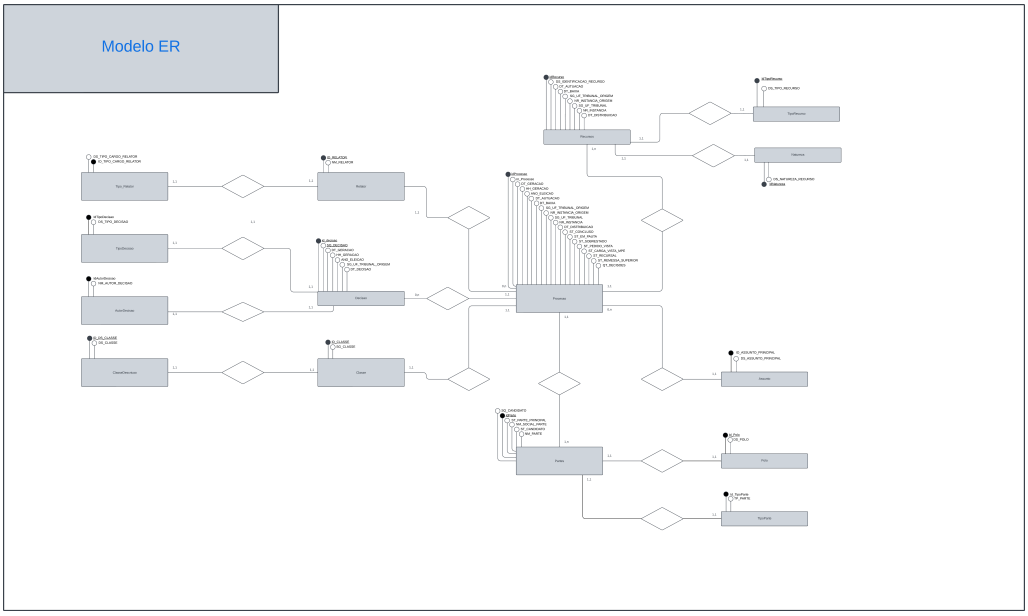


**Figure 4. Modelo Entidade Relacionamento**

## 5. Projeto Físico

O projeto físico foi montado com base nos modelos lógico e relacional e no projeto conceitual modelados anteriormente, segue na figura 6 um diagrama do modelo elaborado.

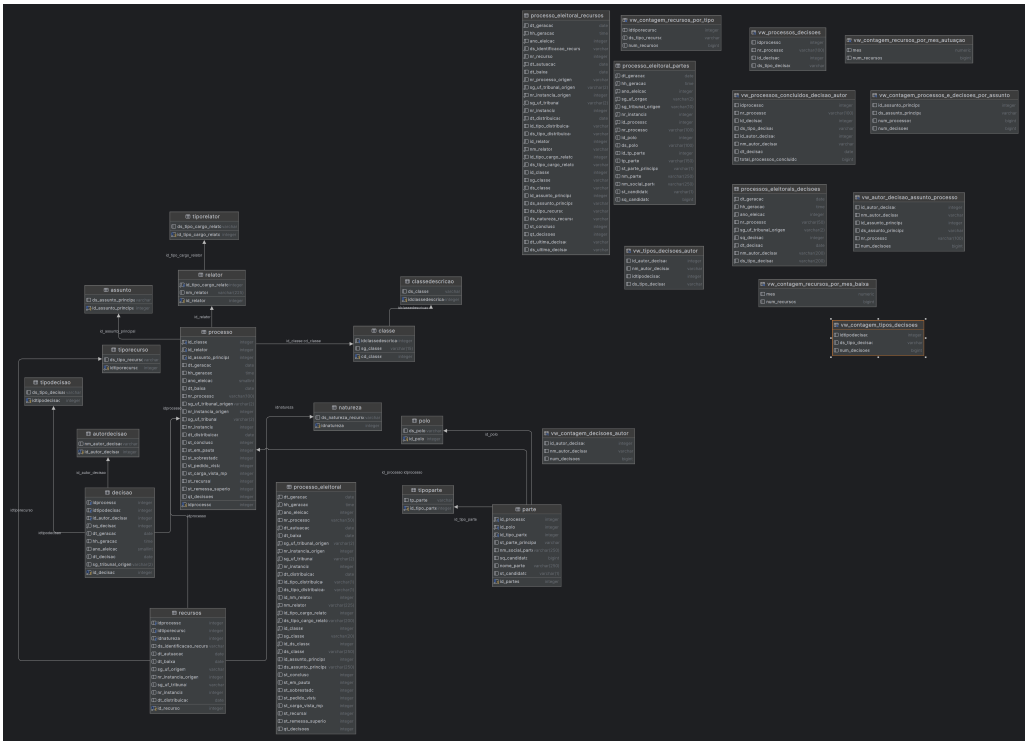

**Figure 6. Diagrama Modelo Físico.**

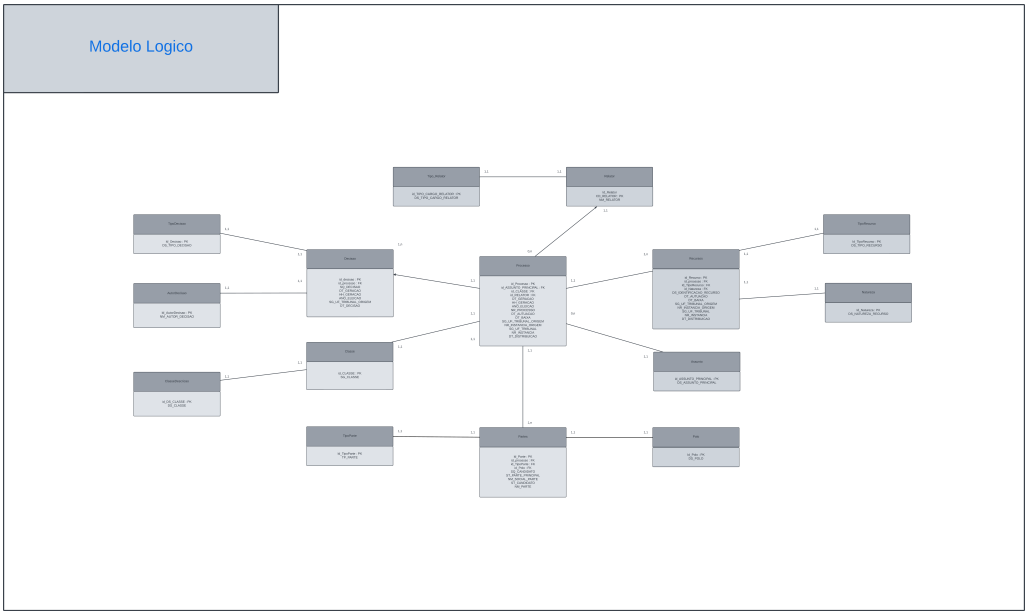


**Figure 5. Modelo Lógico**

## 5.1. DDL Linguagem de Definição de Dados

DDLs significa ”Data Definition Language” e uma linguagem usada para descrever e definir a estrutura de dados em um banco de dados.As DDLs são usadas para criar, alterar ou excluir tabelas, índices, restrições e outros objetos de banco de dados.

### 5.1.1. DDLs

```
create table natureza
(
    idnatureza          integer not null
        constraint natureza_pk
            primary key,
    ds_natureza_recurso varchar
);

alter table natureza
    owner to postgres;

create table tiporecurso
(
    idtiporecurso   serial
        constraint tiporecurso_pk
            primary key,
    ds_tipo_recurso varchar
);

alter table tiporecurso
    owner to postgres;
```

```
create table assunto
(
    id_assunto_principal integer not null
        constraint assunto_pk
            primary key,
    ds_assunto_principal varchar
);

alter table assunto
    owner to postgres;

create table tipoparte
(
    id_tipo_parte serial
        constraint tipoparte_pk
            primary key,
    tp_parte      varchar
);

alter table tipoparte
    owner to postgres;

create table polo
(
    id_polo serial
        constraint polo_pk
            primary key,
    ds_polo varchar
);

alter table polo
    owner to postgres;

create table tiporelator
(
    id_tipo_cargo_relator serial
        constraint tipocargorelator_pk
            primary key,
    ds_tipo_cargo_relator varchar
);

alter table tiporelator
    owner to postgres;

create table relator
(
    id_relator            serial
        constraint relator_pk
```

```
            primary key,
    id_tipo_cargo_relator integer not null
        constraint relator_fk
            references tiporelator,
    nm_relator            varchar(225)
);

alter table relator
    owner to postgres;

create table autordecisao
(
    id_autor_decisao serial
        constraint autordecisao_pk
            primary key,
    nm_autor_decisao varchar
);

alter table autordecisao
    owner to postgres;

create table tipodecisao
(
    idtipodecisao   serial
        constraint tipodecisao_pk
            primary key,
    ds_tipo_decisao varchar
);

alter table tipodecisao
    owner to postgres;

create table classedescricao
(
    idclassedescricao integer not null
        constraint classedescricao_pk
            primary key,
    ds_classe         varchar
);

alter table classedescricao
    owner to postgres;

create table classe
(
    cd_classe         integer not null
        constraint classe_pk
            primary key,
    idclassedescricao integer not null
```

```
        constraint classedescricao_fk
            references classedescricao,
    sg_classe         varchar(15)
);

alter table classe
    owner to postgres;

create table processo
(
    idprocesso            serial
        constraint processo_pk
            primary key,
    id_classe             integer not null
        constraint classe_fk
            references classe,
    id_relator            integer not null
        constraint relator_fk
            references relator,
    id_assunto_principal  integer
        constraint assunto_fk
            references assunto,
    dt_geracao            date,
    hh_geracao            time,
    ano_eleicao           smallint,
    dt_baixa              date,
    nr_processo           varchar(100),
    sg_uf_tribunal_origem varchar(2),
    nr_instancia_origem   integer,
    sg_uf_tribunal        varchar(2),
    nr_instancia          integer,
    dt_distribuicao       date,
    st_concluso           integer,
    st_em_pauta           integer,
    st_sobrestado         integer,
    st_pedido_vista       integer,
    st_carga_vista_mpe    integer,
    st_recursal           integer,
    st_remessa_superior   integer,
    qt_decisoes           integer
);

alter table processo
    owner to postgres;

create table decisao
(
    id_decisao         serial
        constraint decisao_pk
```

```
            primary key,
    idprocesso          integer
        constraint processo_fk
            references processo,
    idtipodecisao       integer
        constraint tipodecisao_fk
            references tipodecisao,
    id_autor_decisao    integer
        constraint autordecisao_fk
            references autordecisao,
    sq_decisao          integer not null,
    dt_geracao          date,
    hh_geracao          time,
    ano_eleicao         smallint,
    dt_decisao          date,
    sg_tribunal_origem varchar(2)
);

alter table decisao
    owner to postgres;

create table parte
(
    id_partes           serial
        constraint partes_pk
            primary key,
    id_processo         integer not null
        constraint processo_fk
            references processo,
    id_polo             integer not null
        constraint polo_fk
            references polo,
    id_tipo_parte       integer not null
        constraint tipoparte_fk
            references tipoparte,
    st_parte_principal varchar,
    nm_social_parte    varchar(250),
    sq_candidato        bigint,
    nome_parte          varchar(250),
    st_candidato        varchar(1)
);

alter table parte
    owner to postgres;

create table processo_eleitoral_recursos
(
    dt_geracao                  date        not null,
    hh_geracao                  time        not null,
```

```
    ano_eleicao               integer    not null,
    ds_identificacao_recurso  varchar    not null,
    nr_recurso                integer    not null,
    dt_autuacao               date       not null,
    dt_baixa                  date,
    nr_processo_origem        varchar    not null,
    sg_uf_tribunal_origem     varchar(2) not null,
    nr_instancia_origem       integer    not null,
    sg_uf_tribunal            varchar(2) not null,
    nr_instancia              integer    not null,
    dt_distribuicao           date       not null,
    id_tipo_distribuicao      varchar,
    ds_tipo_distribuicao      varchar,
    id_relator                integer    not null,
    nm_relator                varchar    not null,
    id_tipo_cargo_relator     integer    not null,
    ds_tipo_cargo_relator     varchar    not null,
    id_classe                 integer,
    sg_classe                 varchar,
    ds_classe                 varchar    not null,
    id_assunto_principal      integer,
    ds_assunto_principal      varchar,
    ds_tipo_recurso           varchar,
    ds_natureza_recurso       varchar,
    st_concluso               integer,
    qt_decisoes               integer,
    dt_ultima_decisao         varchar,
    ds_ultima_decisao         varchar
);

alter table processo_eleitoral_recursos
    owner to postgres;

create table processos_eleitorais_decisoes
(
    dt_geracao            date         not null,
    hh_geracao            time         not null,
    ano_eleicao           integer      not null,
    nr_processo           varchar(50),
    sg_uf_tribunal_origem varchar(2)   not null,
    sq_decisao            integer      not null,
    dt_decisao            date         not null,
    nm_autor_decisao      varchar(200),
    ds_tipo_decisao       varchar(200) not null
);

alter table processos_eleitorais_decisoes
    owner to postgres;
```

```
create table processo_eleitoral_partes
(
    dt_geracao          date         not null,
    hh_geracao          time         not null,
    ano_eleicao         integer      not null,
    sg_uf_orgao         varchar(2)   not null,
    sg_tribunal_origem  varchar(10)  not null,
    nr_instancia        integer      not null,
    id_processo         integer      not null,
    nr_processo         varchar(100) not null,
    id_polo             integer,
    ds_polo             varchar(100),
    id_tp_parte         integer,
    tp_parte            varchar(150),
    st_parte_principal  varchar(1),
    nm_parte            varchar(250),
    nm_social_parte     varchar(250),
    st_candidato        varchar(1),
    sq_candidato        bigint
);

alter table processo_eleitoral_partes
    owner to postgres;

create table processo_eleitoral
(
    dt_geracao            date         not null,
    hh_geracao            time         not null,
    ano_eleicao           integer      not null,
    nr_processo           varchar(50),
    dt_autuacao           date         not null,
    dt_baixa              date,
    sg_uf_tribunal_origem varchar(2)   not null,
    nr_instancia_origem   integer      not null,
    sg_uf_tribunal        varchar(2)   not null,
    nr_instancia          integer      not null,
    dt_distribuicao       date         not null,
    id_tipo_distribuicao  varchar(1),
    ds_tipo_distribuicao  varchar(1),
    id_nm_relator         integer,
    nm_relator            varchar(225) not null,
    id_tipo_cargo_relator integer      not null,
    ds_tipo_cargo_relator varchar(200) not null,
    id_classe             integer      not null,
    sg_classe             varchar(20)  not null,
    id_ds_classe          integer      not null,
    ds_classe             varchar(250) not null,
    id_assunto_principal  integer,
    ds_assunto_principal  varchar(250),
```

```
    st_concluso           integer,
    st_em_pauta           integer,
    st_sobrestado         integer,
    st_pedido_vista       integer,
    st_carga_vista_mpe    integer,
    st_recursal           integer,
    st_remessa_superior   integer,
    qt_decisoes           integer
);

alter table processo_eleitoral
    owner to postgres;

create table recursos
(
    id_recurso               integer not null
        constraint recursos_pk
            primary key,
    idprocesso               integer
        constraint processo_fk
            references processo,
    idtiporecurso            integer
        constraint tiporecurso_fk
            references tiporecurso,
    idnatureza               integer
        constraint natureza_fk
            references natureza,
    ds_identificacao_recurso varchar,
    dt_autuacao              date,
    dt_baixa                 date,
    sg_uf_origem             varchar,
    nr_instancia_origem      integer,
    sg_uf_tribunal           varchar,
    nr_instancia             integer,
    dt_distribuicao          date
);

alter table recursos
    owner to postgres;
```

## 5.2. DML

### 5.2.1. Manipulando os dados

Abaixo podemos ver os código utilizado para manipular os dados, em um primeiro momento fez-se criação das tabela e manipulação da tabelas originais **??** no qual importamos direto o arquivo csv disponível no site do TSE para poder fazer a manipulação no banco de dados modelado pelo grupo **??**

```
create table if not exists processo_eleitoral (

        DT_GERACAO date not null,
        HH_GERACAO time not null,
        ANO_ELEICAO integer not null,
        NR_PROCESSO varchar(50),
        DT_AUTUACAO date not null,
        DT_BAIXA date,
        SG_UF_TRIBUNAL_ORIGEM varchar(2) not null,
        NR_INSTANCIA_ORIGEM integer not null,
        SG_UF_TRIBUNAL varchar(2) not null,
        NR_INSTANCIA integer not null,
        DT_DISTRIBUICAO date not null,
        id_TIPO_DISTRIBUICAO varchar(1),
        DS_TIPO_DISTRIBUICAO varchar(1),
        id_RELATOR integer not null,
        NM_RELATOR varchar(225) not null,
        id_TIPO_CARGO_RELATOR integer not null,
        DS_TIPO_CARGO_RELATOR varchar(50) not null,
        id_CLASSE integer not null,
        SG_CLASSE varchar (20) not null,
        DS_CLASSE varchar (250) not null,
        id_ASSUNTO_PRINCIPAL integer,
        DS_ASSUNTO_PRINCIPAL varchar (250),
        ST_CONCLUSO     integer,
        ST_EM_PAUTA     integer,
        ST_SOBRESTADO integer,
        ST_PEDIDO_VISTA integer,
        ST_CARGA_VISTA_MPE      integer,
        ST_RECURSAL     integer ,
        ST_REMESSA_SUPERIOR     integer ,
        QT_DECISOES integer
)

create table if not exists processo_eleitoral (

        DT_GERACAO date not null,
        HH_GERACAO time not null,
        ANO_ELEICAO integer not null,
        NR_PROCESSO varchar(50),
        DT_AUTUACAO date not null,
        DT_BAIXA date,
        SG_UF_TRIBUNAL_ORIGEM varchar(2) not null,
        NR_INSTANCIA_ORIGEM integer not null,
        SG_UF_TRIBUNAL varchar(2) not null,
        NR_INSTANCIA integer not null,
        DT_DISTRIBUICAO date not null,
        id_TIPO_DISTRIBUICAO varchar(1),
```

```
        DS_TIPO_DISTRIBUICAO varchar(1),
        ID_NM_RELATOR integer,
        NM_RELATOR varchar(225) not null,
        ID_TIPO_CARGO_RELATOR integer not null,
        DS_TIPO_CARGO_RELATOR varchar(200) not null,
        ID_CLASSE integer not null,
        SG_CLASSE varchar (20) not null,
        ID_DS_CLASSE integer not null,
        DS_CLASSE varchar (250) not null,
        ID_ASSUNTO_PRINCIPAL integer,
        DS_ASSUNTO_PRINCIPAL varchar (250),
        ST_CONCLUSO     integer,
        ST_EM_PAUTA     integer,
        ST_SOBRESTADO integer,
        ST_PEDIDO_VISTA integer,
        ST_CARGA_VISTA_MPE      integer,
        ST_RECURSAL     integer ,
        ST_REMESSA_SUPERIOR     integer ,
        QT_DECISOES integer
)



create table if not exists processo_eleitoral_partes(

        DT_GERACAO date not null,
        HH_GERACAO time not null,
        ANO_ELEICAO integer not null,
        SG_UF_ORGAO     varchar(2) not null,
        SG_TRIBUNAL_ORIGEM      varchar(10) not null,
        NR_INSTANCIA    integer not null,
        id_processo integer not null,
        NR_PROCESSO     varchar(100) not null,
        id_polo integer,
        DS_POLO varchar(100),
        id_tp_parte integer,
        TP_PARTE        varchar(150),
        ST_PARTE_PRINCIPAL varchar(1),
        NM_PARTE varchar(250),
        NM_SOCIAL_PARTE varchar(250),
        ST_CANDIDATO varchar(1),
        SQ_CANDIDATO bigint
)
-- 13

create table if not exists processos_eleitorais_decisoes(

        DT_GERACAO date not null,
        HH_GERACAO time not null,
```

```
        ANO_ELEICAO integer not null,
        NR_PROCESSO varchar(50),

        SG_UF_TRIBUNAL_ORIGEM varchar(2) not null,
    sq_decisao  integer not null,
        DT_DECISAO      date not null,
        NM_AUTOR_DECISAO varchar(200),
        DS_TIPO_DECISAO varchar(200) not null

);


create table if not exists processo_eleitoral_recursos (

        DT_GERACAO date not null,
        HH_GERACAO time not null,
        ANO_ELEICAO integer not null,
        DS_IDENTIFICACAO_RECURSO varchar not null,
        NR_RECURSO integer not null,
        DT_AUTUACAO     date not null,
        DT_BAIXA        varchar,
        NR_PROCESSO_ORIGEM      varchar not null,
        SG_UF_TRIBUNAL_ORIGEM   varchar(2) not null,
        NR_INSTANCIA_ORIGEM     integer not null,
        SG_UF_TRIBUNAL  varchar(2) not null,
        NR_INSTANCIA    integer not null,
        DT_DISTRIBUICAO date not null,
        id_TIPO_DISTRIBUICAO    varchar,
        DS_TIPO_DISTRIBUICAO    varchar,
        id_RELATOR      integer not null,
        NM_RELATOR      varchar not null,
        id_TIPO_CARGO_RELATOR   integer not null,
        DS_TIPO_CARGO_RELATOR   varchar not null,
        id_CLASSE       integer,
        SG_CLASSE       varchar,
        DS_CLASSE       varchar not null,
        id_ASSUNTO_PRINCIPAL    integer,
        DS_ASSUNTO_PRINCIPAL    varchar,
        DS_TIPO_RECURSO varchar,
        DS_NATUREZA_RECURSO     varchar,
        ST_CONCLUSO     integer,
        QT_DECISOES     integer,
        DT_ULTIMA_DECISAO       varchar,
        DS_ULTIMA_DECISAO varchar

)
```

## 5.3. Visões

### 5.3.1. Tipos de decisões autor

Afigura7 utiliza uma sub-consulta para numerar as linhas por autor de decisão e tipo de decisão, selecionando apenas a primeira linha para cada autor. Retorna informações sobre o autor de decisão, tipo de decisão e o número de decisão associado.

```sql
CREATE OR REPLACE VIEW vw_tipos_decisoes_autor AS
SELECT
    id_autor_decisao,
    nm_autor_decisao,
    idtipodecisao,
    ds_tipo_decisao
FROM (
    SELECT
        a.id_autor_decisao,
        a.nm_autor_decisao,
        d.idtipodecisao,
        td.ds_tipo_decisao,
        ROW_NUMBER() OVER (PARTITION BY a.id_autor_decisao ORDER BY td.ds_tipo_decisao) AS row_num
    FROM
        autordecisao a
    JOIN
        decisao d ON a.id_autor_decisao = d.id_autor_decisao
    JOIN
        tipodecisao td ON d.idtipodecisao = td.idtipodecisao
) AS numbered_rows
WHERE
    row_num = 1;
```

**Figure 7. Visão 1**

### 5.3.2. Contagem de Decisões por autor

A figura8 Conta o número total de decisões para cada autor de decisão. Retorna informações sobre o autor de decisão e o número total de decisões associadas.

```sql
CREATE OR REPLACE VIEW vw_contagem_decisoes_autor AS
SELECT
    a.id_autor_decisao,
    a.nm_autor_decisao,
    COUNT(d.id_decisao) AS num_decisoes
FROM
    autordecisao a
JOIN
    decisao d ON a.id_autor_decisao = d.id_autor_decisao
GROUP BY
    a.id_autor_decisao, a.nm_autor_decisao;
```

**Figure 8. Visão 2**

### 5.3.3. Informações processos e decisões

A figura9 Combina informações dos processos com suas decisões e tipos de decisões. Retorna o ID do processo, número do processo, ID da decisão e tipo de decisão associados.

```sql
CREATE OR REPLACE VIEW vw_processos_decisoes AS
SELECT
    p.idprocesso,
    p.nr_processo,
    d.id_decisao,
    td.ds_tipo_decisao
FROM
    processo p
    JOIN decisao d ON p.idprocesso = d.idprocesso
    JOIN tipodecisao td ON d.idtipodecisao = td.idtipodecisao;
```

**Figure 9. Visão 3**

### 5.3.4. Total de decisões por tipo

Na figura 10 podemos ver a contagem do número total de decisões para cada tipo de decisão. Retorna informações sobre o tipo de decisão e o número total de decisões associadas.

```sql
CREATE OR REPLACE VIEW vw_contagem_tipos_decisoes AS
SELECT
    td.idtipodecisao,
    td.ds_tipo_decisao,
    COUNT(d.id_decisao) AS num_decisoes
FROM
    tipodecisao td
    LEFT JOIN decisao d ON td.idtipodecisao = d.idtipodecisao
GROUP BY
    td.idtipodecisao, td.ds_tipo_decisao;
```

**Figure 10. Visão 4**

### 5.3.5. Total de Recursos por tipo

A figura11 verifica-se o número total de recursos para cada tipo de recurso. Retorna informações sobre o tipo de recurso e o número total de recursos associados.

```sql
CREATE OR REPLACE VIEW vw_contagem_recursos_por_tipo AS
SELECT
    tr.idtiporecurso,
    tr.ds_tipo_recurso,
    COUNT(DISTINCT r.id_recurso) AS num_recursos
FROM
    tiporecurso tr
    LEFT JOIN recursos r ON tr.idtiporecurso = r.idtiporecurso
GROUP BY
    tr.idtiporecurso, tr.ds_tipo_recurso;
```

**Figure 11. Visão 5**

### 5.3.6. Recursos por mês de autuação

A figura12 podemos contar o número total de recursos autuados por mês. Retorna informações sobre o mês e o número total de recursos autuados associados.

```sql
CREATE OR REPLACE VIEW vw_contagem_recursos_por_mes_autuaçao AS
SELECT
    EXTRACT(MONTH FROM pe.dt_autuacao) AS mes,
    COUNT(*) AS num_recursos
FROM
    processo_eleitoral_recursos pe
GROUP BY
    mes
ORDER BY
    mes;
```

**Figure 12. Visão 6**

### 5.3.7. Recurso por mês de baixa

A figura13 podemos consultar o número total de recursos baixados por mês. Retorna informações sobre o mês e o número total de recursos baixados associados.

A figura 14 consulta-se o número total de processos e decisões para cada assunto. Retorna informações sobre o assunto, o número total de processos e o número total de decisões associadas.

### 5.3.8. Autor decisões e processos

A figura 15 combina informações sobre autor de decisão, assunto, processo e conta o número total de decisões para cada combinação. Retorna informações sobre o autor de decisão, assunto, número do processo e o número total de decisões associadas.

```sql
CREATE VIEW vw_contagem_recursos_por_mes_baixa AS
SELECT
    EXTRACT(MONTH FROM pe.dt_baixa) AS mes,
    COUNT(*) AS num_recursos
FROM
    processo_eleitoral_recursos pe
GROUP BY
    mes
ORDER BY
    mes;
```

**Figure 13. Visão 7**

```sql
CREATE or replace VIEW vw_contagem_processos_e_decisoes_por_assunto AS
SELECT
    a.id_assunto_principal,
    a.ds_assunto_principal,
    COUNT(DISTINCT p.idprocesso) AS num_processos,
    COUNT(DISTINCT d.id_decisao) AS num_decisoes
FROM
    assunto a
    LEFT JOIN processo p ON a.id_assunto_principal = p.id_assunto_principal
    LEFT JOIN decisao d ON p.idprocesso = d.idprocesso
GROUP BY
    a.id_assunto_principal, a.ds_assunto_principal
HAVING
    COUNT(DISTINCT p.idprocesso) > 0 OR COUNT(DISTINCT d.id_decisao) > 0;
```

**Figure 14. Visão 8**

### 5.3.9. Processos concluídos

A figura16 seleciona informações sobre processos concluídos com detalhes sobre a decisão e autor da decisão. Retorna o ID do processo, número do processo, ID da decisão, tipo de decisão, ID do autor de decisão, nome do autor de decisão e a data da decisão.

## 6. Conclusões

O principal foco do projeto foi compreender e aplicar de maneira prática o processo de modelagem de dados. Nesse contexto, foram abordados os passos executados, destacando os pontos de atenção em cada modificação realizada ao longo do processo. Os resultados obtidos foram demonstrados, evidenciando as versões desenvolvidas com base nos conhecimentos adquiridos durante as aulas do componente de Banco de Dados.

Adicionalmente, enfrentamos o desafio de modelar um banco de dados de grande porte com complexidades médias. Esse desafio incentivou o grupo a constantemente buscar soluções e melhorias ao longo do processo. Dessa forma, durante essa jornada, conseguimos desenvolver e aplicar conhecimentos e habilidades que contribuem significati-

```sql
CREATE or replace VIEW vw_autor_decisao_assunto_processo AS
SELECT
    ad.id_autor_decisao,
    ad.nm_autor_decisao,
    a.id_assunto_principal,
    a.ds_assunto_principal,
    p.nr_processo,
    COUNT(d.id_decisao) AS num_decisoes
FROM
    autordecisao ad
    JOIN decisao d ON ad.id_autor_decisao = d.id_autor_decisao
    JOIN processo p ON d.idprocesso = p.idprocesso
    LEFT JOIN assunto a ON p.id_assunto_principal = a.id_assunto_principal
GROUP BY
    ad.id_autor_decisao, ad.nm_autor_decisao, a.id_assunto_principal, a.ds_assunto_principal, p.nr_processo;
```

**Figure 15. Visão 9**

```sql
CREATE or replace VIEW vw_processos_concluidos_decisao_autor AS
SELECT
    p.idprocesso,
    p.nr_processo,
    d.id_decisao,
    td.ds_tipo_decisao,
    ad.id_autor_decisao,
    ad.nm_autor_decisao,
    d.dt_decisao
FROM
    processo p
    JOIN decisao d ON p.idprocesso = d.idprocesso
    JOIN tipodecisao td ON d.idtipodecisao = td.idtipodecisao
    JOIN autordecisao ad ON d.id_autor_decisao = ad.id_autor_decisao
WHERE
    p.st_concluso = 1;
```

**Figure 16. Visão 10**

vamente para nosso crescimento profissional e educacional. O projeto também incluiu a otimização da base de dados por meio do desenvolvimento de novos diagramas, modelos e do projeto físico, agregando valor à estrutura do banco de dados trabalhado.

## References