

ENAP – ESCOLA NACIONAL DE ADMINISTRAÇÃO PÚBLICA
MBA - CURSO EM CIÊNCIA DE DADOS E INTELIGÊNCIA ARTIFICIAL
APLICADAS
DISCIPLINA: INTRODUÇÃO A BANCO DE DADOS E LINGUAGEM SQL
PROFESSOR: MÁRCIO DE CARVALHO VICTORINO

ALUNO: JOÃO BATISTA DE SOUZA LEÃO

PROJETO DE BANCO DE DADOS

1. INTRODUÇÃO

O presente trabalho tem por objetivo desenvolver um modelo de base de dados que reflita a gestão de processos administrativos e judiciais no âmbito da Procuradoria Regional Federal da 1ª Região – PGF/AGU, com sede em Brasília (DF).

A gestão dos metadados processuais e dos conteúdos dos documentos produzidos é crucial para a realização da jurimetria (estatística e análise dados aplicadas ao Direito), assim como o desenvolvimento de ferramentas de automação capaz de capturar padrões processuais, em especial os de natureza repetitiva.

A escolha do tema para o desenvolvimento da base de dados se deu em virtude do possível aproveitamento para o projeto e o trabalho a ser apresentado ao final do curso.

2. REQUISITOS FUNCIONAIS

O desenvolvimento da base de dados buscou atender aos requisitos funcionais a seguir relatados.

Todos os processos internos, sejam administrativos ou judiciais, devem ter os seguintes campos a serem cadastrados: valor econômico, se há protocolo eletrônico, o Número Único do Processo (NUP), nos formatos só com dígitos e também no formato completo com os demais caracteres, espécie de processo, que

pode ser apenas judicial ou administrativo, data e hora de abertura e encerramento do processo, caso haja, título do processo, número alternativo, descrição do processo, pessoa responsável pelo cadastro (a procedência), setor atual e de início, se há acesso restrito, data e hora de quando foi criado e atualizado. Ainda um processo pode estar vinculado a um outro processo, quando se refiram a matéria em comum.

O processo pode ou não está relacionado a um processo judicial, com os dados a serem obtidos mediante integração com sistemas do Poder Judiciário. Os campos do processo judicial devem ser os seguintes: número do processo judicial, tanto no formato de apenas dígitos quanto no formato completo com outros caracteres, número alternativo caso haja, classe nacional (padronizada pelo CNJ), órgão julgador, data e hora do ajuizamento, da citação e do trânsito em julgado caso haja esses dados, se há intervenção do Ministério Público, se o processo é eletrônico, o valor da causa, a data e hora de quando foi criado e atualizado. Ainda no âmbito dos processos judiciais, as partes devem ser cadastradas como interessadas no processo, assim como a pessoa jurídica de direito público que esteja sendo representada judicialmente pela procuradoria.

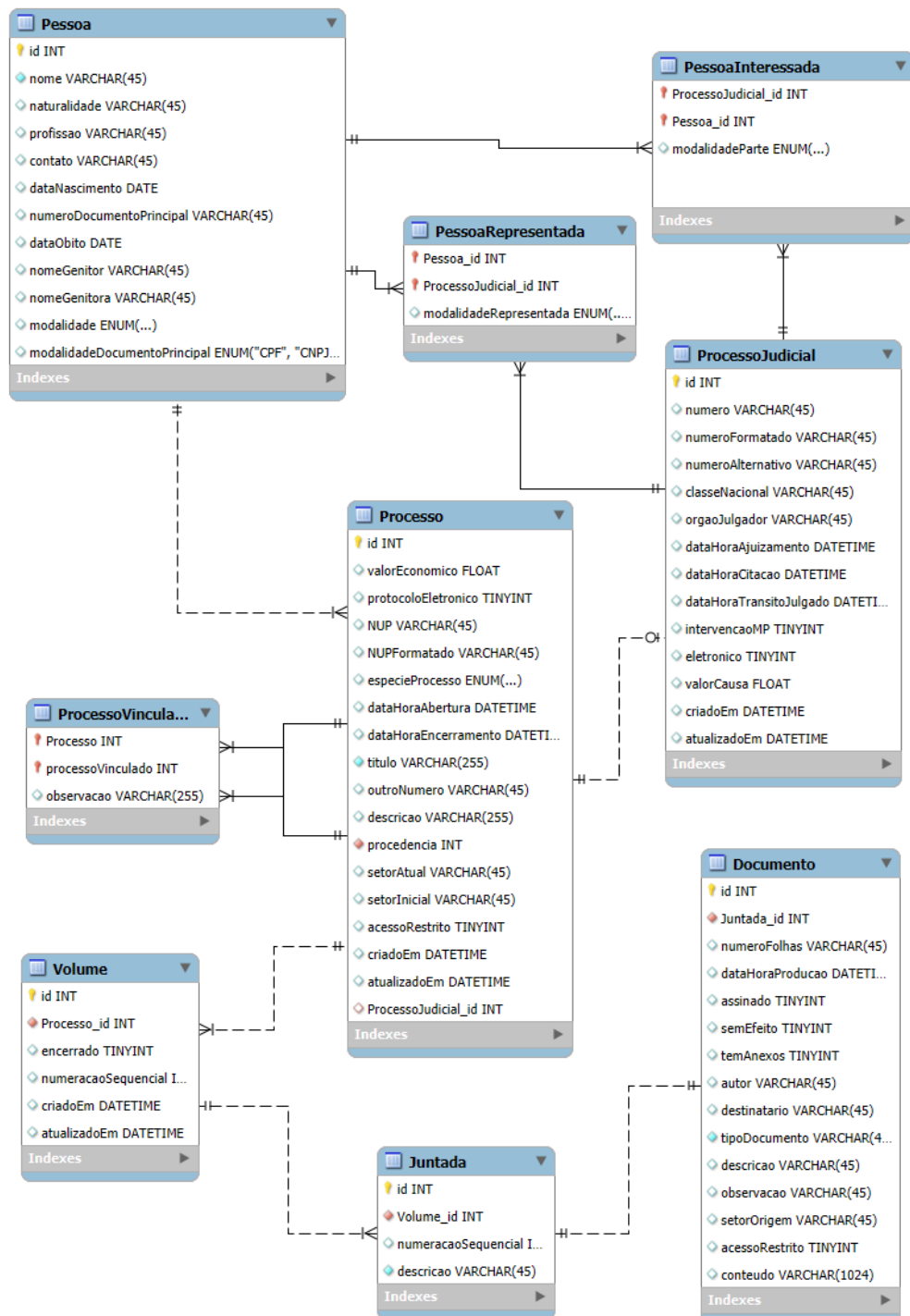
As pessoas que figuram como partes nos processos judiciais ou são responsáveis internamente pelo cadastramento dos processos deve ser cadastradas com os seguintes campos: nome, naturalidade, profissão, contato, data de nascimento, número e modalidade do documento principal (CPF ou CNPJ), data de óbito caso haja, nomes dos genitores, como também a modalidade, se é pessoa física ou jurídica.

O processo possui um ou vários volumes, que devem ser cadastrados com os seguintes campos: se foi ou não encerrado, a numeração sequencial do volume, a data e hora em que foi criado e atualizado. Os volumes são compostos por movimentações processuais denominadas de juntadas. Cada juntada deve possuir uma numeração sequencial e a descrição de forma opcional. Ainda, cada juntada é composta por um ou mais documentos. Cada documento deve ser registrado, de modo a constar os seguintes campos: número de folhas, data e hora da produção, se está ou não assinado, se está sem efeito, se tem anexos, quem é o autor e o destinatário, o tipo do documento (que deve ser considerado como obrigatório), a descrição e a observação que devem ser opcionais, o setor de

origem, se há acesso restrito ou não e o conteúdo, a ser também armazenado caso possível.

3. MODELO CONCEITUAL/LÓGICO

A seguir é exposto o diagrama EER (Diagrama de Entidade-Relação Melhorado, em tradução livre), exportado do programa MySQL Workbench.



4. BASE DE DADOS CRIADA E CARREGADA COM DADOS EM UM SGBD

O script de criação da base de dados (DDL) é apresentado a seguir.

```
-- -----  
-- Schema dbprf1  
-- -----  
CREATE SCHEMA IF NOT EXISTS `dbprf1` DEFAULT CHARACTER SET utf8 ;  
USE `dbprf1` ;  
  
-- -----  
-- Table `dbprf1`.`Pessoa`  
-- -----  
CREATE TABLE IF NOT EXISTS `dbprf1`.`Pessoa` (  
  `id` INT NOT NULL,  
  `nome` VARCHAR(45) NOT NULL,  
  `naturalidade` VARCHAR(45) NULL,  
  `profissao` VARCHAR(45) NULL,  
  `contato` VARCHAR(45) NULL,  
  `dataNascimento` DATE NULL,  
  `numeroDocumentoPrincipal` VARCHAR(45) NULL,  
  `dataObito` DATE NULL,  
  `nomeGenitor` VARCHAR(45) NULL,  
  `nomeGenitora` VARCHAR(45) NULL,  
  `modalidade` ENUM('FISICA', 'JURIDICA') NULL,  
  `modalidadeDocumentoPrincipal` ENUM("CPF", "CNPJ") NULL,  
  PRIMARY KEY (`id`))  
ENGINE = InnoDB;  
  
-- -----  
-- Table `dbprf1`.`ProcessoJudicial`  
-- -----  
CREATE TABLE IF NOT EXISTS `dbprf1`.`ProcessoJudicial` (  
  `id` INT NOT NULL,  
  `numero` VARCHAR(45) NULL,  
  `numeroFormatado` VARCHAR(45) NULL,  
  `numeroAlternativo` VARCHAR(45) NULL,  
  `classeNacional` VARCHAR(45) NULL,  
  `orgaoJulgador` VARCHAR(45) NULL,  
  `dataHoraAjuizamento` DATETIME NULL,  
  `dataHoraCitacao` DATETIME NULL,  
  `dataHoraTransitoJulgado` DATETIME NULL,  
  `intervencaoMP` TINYINT NULL,
```

```

`eletronico` TINYINT NULL,
`valorCausa` FLOAT NULL,
`criadoEm` DATETIME NULL,
`atualizadoEm` DATETIME NULL,
PRIMARY KEY (`id`))
ENGINE = InnoDB;

```

```

-----
-- Table `dbprf1`.`Processo`
-----

```

```

CREATE TABLE IF NOT EXISTS `dbprf1`.`Processo` (
  `id` INT NOT NULL,
  `valorEconomico` FLOAT NULL,
  `protocoloEletronico` TINYINT NULL,
  `NUP` VARCHAR(45) NULL,
  `NUPFormatado` VARCHAR(45) NULL,
  `especieProcesso` ENUM('JUDICIAL', 'ADMINISTRATIVO') NULL,
  `dataHoraAbertura` DATETIME NULL,
  `dataHoraEncerramento` DATETIME NULL,
  `titulo` VARCHAR(255) NOT NULL,
  `outroNumero` VARCHAR(45) NULL,
  `descricao` VARCHAR(255) NULL,
  `procedencia` INT NOT NULL,
  `setorAtual` VARCHAR(45) NULL,
  `setorInicial` VARCHAR(45) NULL,
  `acessoRestrito` TINYINT NULL,
  `criadoEm` DATETIME NULL,
  `atualizadoEm` DATETIME NULL,
  `ProcessoJudicial_id` INT NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_Processo_ProcessoJudicial1_idx` (`ProcessoJudicial_id` ASC) VISIBLE,
  INDEX `fk_procedencia_Pessoa_idx` (`procedencia` ASC) VISIBLE,
  CONSTRAINT `fk_Processo_ProcessoJudicial1`
    FOREIGN KEY (`ProcessoJudicial_id`)
    REFERENCES `dbprf1`.`ProcessoJudicial` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_procedencia_Pessoa`
    FOREIGN KEY (`procedencia`)
    REFERENCES `dbprf1`.`Pessoa` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

-- Table `dbprf1`.`Volume`

```
CREATE TABLE IF NOT EXISTS `dbprf1`.`Volume` (  
  `id` INT NOT NULL,  
  `Processo_id` INT NOT NULL,  
  `encerrado` TINYINT NULL,  
  `numeracaoSequencial` INT NULL,  
  `criadoEm` DATETIME NULL,  
  `atualizadoEm` DATETIME NULL,  
  PRIMARY KEY (`id`),  
  INDEX `fk_Volume_Processo1_idx` (`Processo_id` ASC) VISIBLE,  
  CONSTRAINT `fk_Volume_Processo1`  
    FOREIGN KEY (`Processo_id`)  
    REFERENCES `dbprf1`.`Processo` (`id`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

-- Table `dbprf1`.`Juntada`

```
CREATE TABLE IF NOT EXISTS `dbprf1`.`Juntada` (  
  `id` INT NOT NULL,  
  `Volume_id` INT NOT NULL,  
  `numeracaoSequencial` INT NULL,  
  `descricao` VARCHAR(45) NOT NULL,  
  PRIMARY KEY (`id`),  
  INDEX `fk_Juntada_Volume1_idx` (`Volume_id` ASC) VISIBLE,  
  CONSTRAINT `fk_Juntada_Volume1`  
    FOREIGN KEY (`Volume_id`)  
    REFERENCES `dbprf1`.`Volume` (`id`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

-- Table `dbprf1`.`PessoaInteressada`

```
CREATE TABLE IF NOT EXISTS `dbprf1`.`PessoaInteressada` (  
  `id` INT NOT NULL,  
  `Pessoa_id` INT NOT NULL,  
  `descricao` VARCHAR(45) NOT NULL,  
  PRIMARY KEY (`id`),  
  INDEX `fk_PessoaInteressada_Pessoa1_idx` (`Pessoa_id` ASC) VISIBLE,  
  CONSTRAINT `fk_PessoaInteressada_Pessoa1`  
    FOREIGN KEY (`Pessoa_id`)  
    REFERENCES `dbprf1`.`Pessoa` (`id`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

```

`ProcessoJudicial_id` INT NOT NULL,
`Pessoa_id` INT NOT NULL,
`modalidadeParte` ENUM('POLO ATIVO', 'POLO PASSIVO', 'TERCEIRO
INTERESSADO') NULL,
PRIMARY KEY (`ProcessoJudicial_id`, `Pessoa_id`),
INDEX `fk_ProcessoJudicial_has_Pessoa_Pessoa1_idx` (`Pessoa_id` ASC)
VISIBLE,
CONSTRAINT `fk_ProcessoJudicial_has_Pessoa_ProcessoJudicial1`
FOREIGN KEY (`ProcessoJudicial_id`)
REFERENCES `dbprf1`.`ProcessoJudicial` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_ProcessoJudicial_has_Pessoa_Pessoa1`
FOREIGN KEY (`Pessoa_id`)
REFERENCES `dbprf1`.`Pessoa` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

-- -----
-- Table `dbprf1`.`PessoaRepresentada`
-- -----

```

```

CREATE TABLE IF NOT EXISTS `dbprf1`.`PessoaRepresentada` (
`Pessoa_id` INT NOT NULL,
`ProcessoJudicial_id` INT NOT NULL,
`modalidadeRepresentada` ENUM('FUNDACAO PUBLICA', 'AUTARQUIA') NULL,
PRIMARY KEY (`Pessoa_id`, `ProcessoJudicial_id`),
INDEX `fk_Pessoa_has_ProcessoJudicial_ProcessoJudicial1_idx`
(`ProcessoJudicial_id` ASC) VISIBLE,
INDEX `fk_Pessoa_has_ProcessoJudicial_Pessoa1_idx` (`Pessoa_id` ASC)
VISIBLE,
CONSTRAINT `fk_Pessoa_has_ProcessoJudicial_Pessoa1`
FOREIGN KEY (`Pessoa_id`)
REFERENCES `dbprf1`.`Pessoa` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Pessoa_has_ProcessoJudicial_ProcessoJudicial1`
FOREIGN KEY (`ProcessoJudicial_id`)
REFERENCES `dbprf1`.`ProcessoJudicial` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

-- -----
-- Table `dbprf1`.`Documento`
-- -----

```
CREATE TABLE IF NOT EXISTS `dbprf1`.`Documento` (  
  `id` INT NOT NULL,  
  `Juntada_id` INT NOT NULL,  
  `numeroFolhas` VARCHAR(45) NULL,  
  `dataHoraProducao` DATETIME NULL,  
  `assinado` TINYINT NULL,  
  `semEfeito` TINYINT NULL DEFAULT 0,  
  `temAnexos` TINYINT NULL,  
  `autor` VARCHAR(45) NULL,  
  `destinatario` VARCHAR(45) NULL,  
  `tipoDocumento` VARCHAR(45) NOT NULL,  
  `descricao` VARCHAR(45) NULL,  
  `observacao` VARCHAR(45) NULL,  
  `setorOrigem` VARCHAR(45) NULL,  
  `acessoRestrito` TINYINT NULL,  
  `conteudo` VARCHAR(1024) NULL,  
  PRIMARY KEY (`id`),  
  INDEX `fk_Documento_Juntada1_idx` (`Juntada_id` ASC) VISIBLE,  
  CONSTRAINT `fk_Documento_Juntada1`  
    FOREIGN KEY (`Juntada_id`)  
    REFERENCES `dbprf1`.`Juntada` (`id`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

-- -----
-- Table `dbprf1`.`ProcessoVinculado`
-- -----

```
CREATE TABLE IF NOT EXISTS `dbprf1`.`ProcessoVinculado` (  
  `Processo` INT NOT NULL,  
  `processoVinculado` INT NOT NULL,  
  `observacao` VARCHAR(255) NULL,  
  PRIMARY KEY (`Processo`, `processoVinculado`),  
  INDEX `fk_Processo_has_Processo_Processo2_idx` (`processoVinculado` ASC)  
  VISIBLE,  
  INDEX `fk_Processo_has_Processo_Processo1_idx` (`Processo` ASC) VISIBLE,  
  CONSTRAINT `fk_Processo_has_Processo_Processo1`  
    FOREIGN KEY (`Processo`)  
    REFERENCES `dbprf1`.`Processo` (`id`)
```



```

ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Processo_has_Processo_Processo2`
FOREIGN KEY (`processoVinculado`)
REFERENCES `dbprf1`.`Processo` (`id`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

A seguir, apresenta-se o print screen das tabelas carregadas com os dados.

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.documento;`

id	Juntada_id	numeroFolhas	dataHoraProducao	assinado	semEfeito	temAnexos	autor	destinatario	tipoDocumento	descricao	observacao	setorOrigem	acessoRestrito	conteudo
1	1	10	2023-01-10 09:00:00	1	0	0	João da Silva	Maria Souza	Petição	Petição inicial	Observações	Sector A	0	Conteúdo do documento
2	2	5	2023-02-15 11:00:00	1	0	0	Maria Souza	João da Silva	Contestação	Contestação	Observações	Sector C	1	Conteúdo do documento

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.juntada;`

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.pessoa;`

id	Volume_id	numeracaoSequencial	descricao
1	1	1	Juntada inicial
2	1	2	Juntada de contestação

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.pessoa;`

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.pessoa;`

id	nome	naturalidade	profissao	contato	dataNascimento	numeroDocumentoPrincipal	dataObito	nomeGenitor	nomeSentora	modalidade	modalidadeDocumentoPrincipal
1	João da Silva	Brasileiro	Advogado	joao.silva@email.com	1980-05-15	12345678900		Pai de João	Mãe de João	FISICA	CPF
2	Maria Souza	Brasileira	Engenheira	maria.souza@email.com	1990-10-20	98765432100		Pai de Maria	Mãe de Maria	FISICA	CPF

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.pessoaintressada;`

Result Grid

	ProcessoJudicial_id	Pessoa_id	modalidadeParte
1	1	1	POLO ATIVO
1	1	2	POLO PASSIVO

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.pessoarepresentada;`

Result Grid

	Pessoa_id	ProcessoJudicial_id	modalidadeRepresentada
1	1	1	AUTARQUIA

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.processo;`

Result Grid

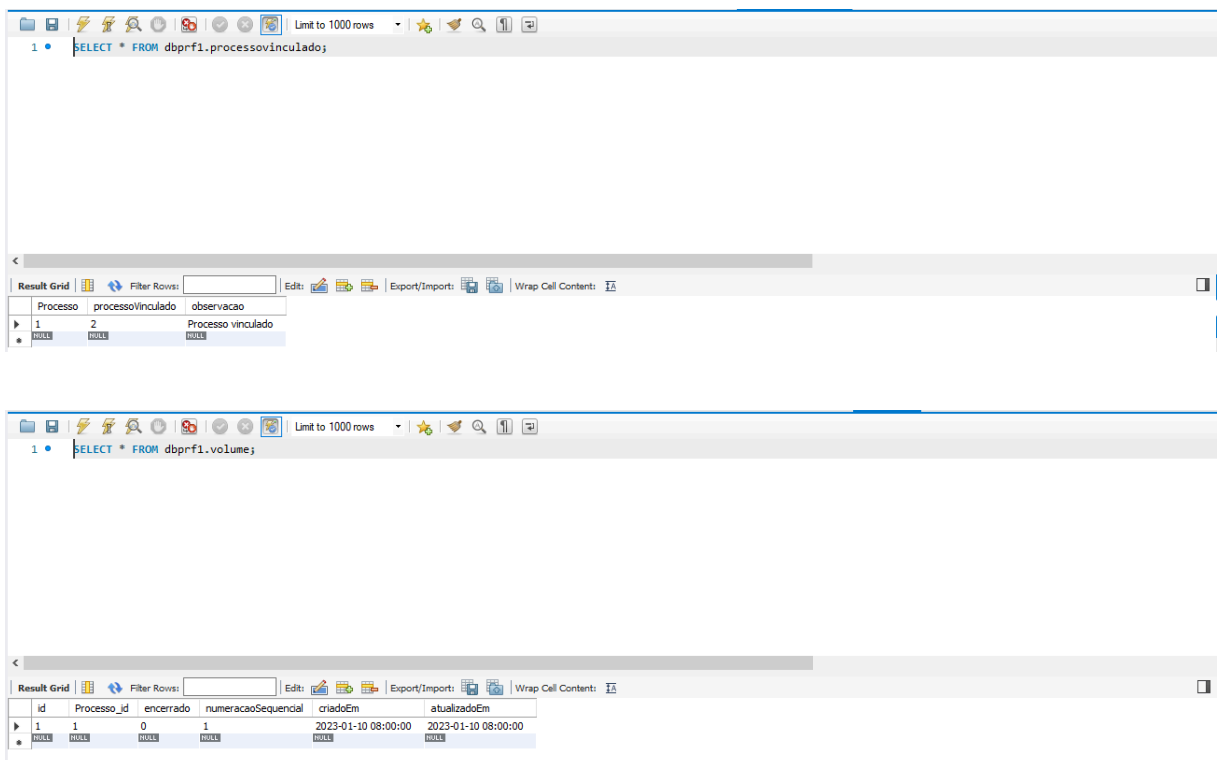
	id	valorEconomico	protocoloEletronico	NUP	NUPFormatado	especieProcesso	dataHoraAbertura	dataHoraEncerramento	titulo	outroNumero	descricao	procedencia	setorAtual	setorInicial
1	1000	1	123456789	1234567-89	ADMINISTRATIVO	2023-01-10 08:00:00			Processo Administrativo		Descrição do processo	1	Setor A	Setor B
2	5000	1	987654321	9876543-21	JUDICIAL	2023-02-15 10:00:00			Processo Judicial		Descrição do processo judicial	2	Setor C	Setor D

Limit to 1000 rows

1 • `SELECT * FROM dbprf1.processojudicial;`

Result Grid

	id	numero	numeroFormatado	numeroAlternativo	classeNacional	orgaoJulgador	dataHoraAjuizamento	dataHoraCitacao	dataHoraTransitoJulgado	intervencaoMP	eletronico	valorCausa	criadoEm	atualizadoEm
1	123456789	1234567-89			Ação Ordinária	Vare Cível	2023-02-15 10:00:00	2023-03-01 14:00:00	2023-03-20 17:00:00	1	1	5000	2023-02-15 10:00:00	2023-03-20 17:00:00



5. CONSULTAS SQL

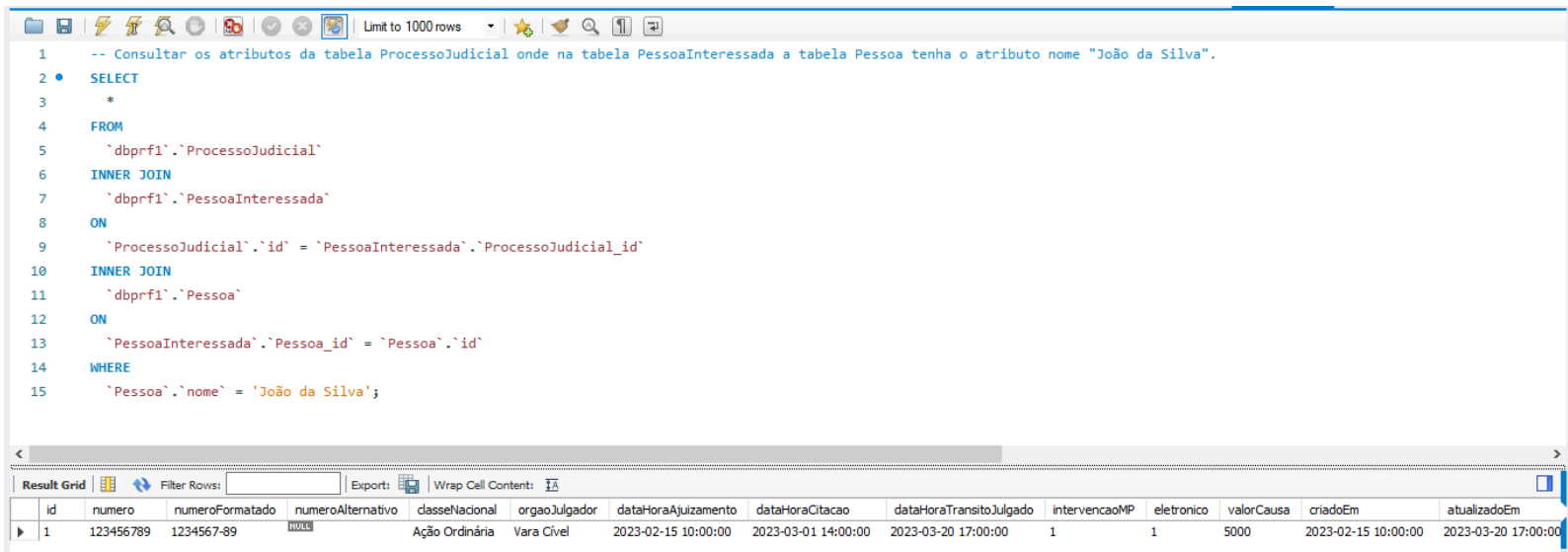
Por fim, são apresentadas cinco consultas tanto em linguagem natural quanto em linguagem SQL, assim como o print screen do resultado da execução da consulta.

5.1. Consulta 1

```
1 -- Consultar todos os atributos da tabela Processo, onde o atributo especieProcesso é igual a Judicial.
2 • SELECT
3     *
4 FROM
5     `dbprf1`.`Processo`
6 WHERE
7     especieProcesso = 'Judicial';
```

[illegible]

5.2. Consulta 2

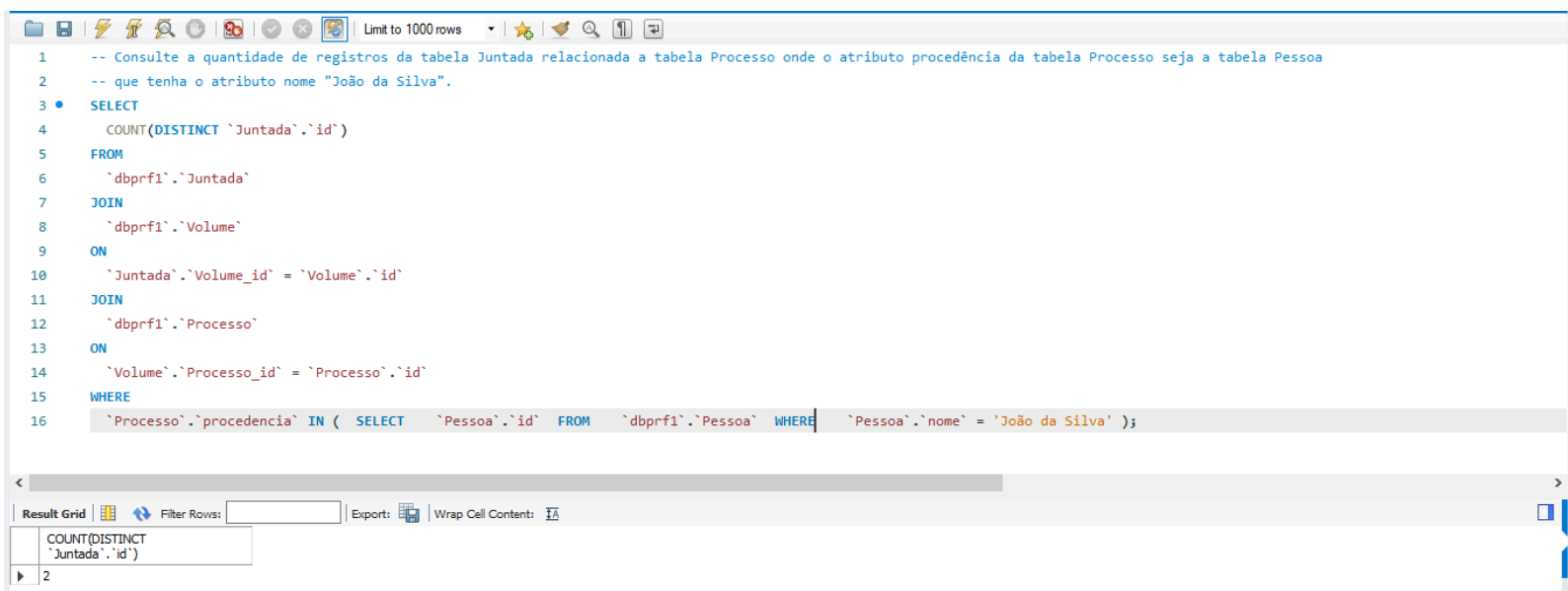


The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, search, and execution. The query editor contains a SQL query that joins three tables: `ProcessoJudicial`, `PessoaInteressada`, and `Pessoa` from a database named `dbprf1`. The query filters for records where the name is 'João da Silva'. Below the query editor, the 'Result Grid' tab is active, displaying a table with 14 columns and one data row.

```
1  -- Consultar os atributos da tabela ProcessoJudicial onde na tabela PessoaInteressada a tabela Pessoa tenha o atributo nome "João da Silva".
2  SELECT
3      *
4  FROM
5      `dbprf1`.`ProcessoJudicial`
6  INNER JOIN
7      `dbprf1`.`PessoaInteressada`
8  ON
9      `ProcessoJudicial`.`id` = `PessoaInteressada`.`ProcessoJudicial_id`
10 INNER JOIN
11     `dbprf1`.`Pessoa`
12 ON
13     `PessoaInteressada`.`Pessoa_id` = `Pessoa`.`id`
14 WHERE
15     `Pessoa`.`nome` = 'João da Silva';
```

id	numero	numeroFormatado	numeroAlternativo	classeNacional	orgaoJulgador	dataHoraAjuizamento	dataHoraCitacao	dataHoraTransitoJulgado	intervencaoMP	eletronico	valorCausa	criadoEm	atualizadoEm
1	123456789	1234567-89	NULL	Ação Ordinária	Vara Cível	2023-02-15 10:00:00	2023-03-01 14:00:00	2023-03-20 17:00:00	1	1	5000	2023-02-15 10:00:00	2023-03-20 17:00:00

5.3. Consulta 3



The screenshot shows a SQL IDE interface. The query editor contains a SQL query that counts the number of distinct records in the `Juntada` table, joined with `Volume` and `Processo` tables. The query filters for records where the name is 'João da Silva'. Below the query editor, the 'Result Grid' tab is active, displaying a table with one column and one data row.

```
1  -- Consulte a quantidade de registros da tabela Juntada relacionada a tabela Processo onde o atributo procedência da tabela Processo seja a tabela Pessoa
2  -- que tenha o atributo nome "João da Silva".
3  SELECT
4      COUNT(DISTINCT `Juntada`.`id`)
5  FROM
6      `dbprf1`.`Juntada`
7  JOIN
8      `dbprf1`.`Volume`
9  ON
10     `Juntada`.`Volume_id` = `Volume`.`id`
11 JOIN
12     `dbprf1`.`Processo`
13 ON
14     `Volume`.`Processo_id` = `Processo`.`id`
15 WHERE
16     `Processo`.`procedencia` IN ( SELECT `Pessoa`.`id` FROM `dbprf1`.`Pessoa` WHERE `Pessoa`.`nome` = 'João da Silva' );
```

COUNT(DISTINCT `Juntada`.`id`)
2

5.4. Consulta 4

</

5.5. Consulta 5