

# Interagindo com Arquivos de Texto

Flávio Codeço Coelho  
*FGV-EMAp*  
(Dated: September 10, 2019)

## CONTENTS

I. Abrindo Arquivos de Texto	1
I.1. Exercício	6
I.2. Exercício:	6
I.3. Abrindo um grande número de documentos texto	7
I.4. Outros recursos do DHBB	8
II. Extraíndo Informação Estruturada	9
III. Exportando para Bancos de Dados	11
IV. Exercícios	12

## I. ABRINDO ARQUIVOS DE TEXTO

Neste curso de mineração de textos usaremos como material principal de trabalho, os verbetes do Dicionário Histórico e Biográfico do Brasil – DHBB. Estes verbetes são disponíveis para Download público.

Neste capítulo vamos aprender a interagir com os verbetes no disco e extrair informações simples a partir dos mesmos.

Vamos começar importando alguma bibliotecas que nos serão úteis nesta tarefa:

```
In [1]: import os
        import glob

In [5]: print("alô turma!")

alô turma!
```

Assumindo que os dados do DHBB já foram baixados para um diretório local, podemos começar inspecionando o diretório e listando o seu conteúdo.

```
In [3]: caminho = "F:/dhbb-master/text/*.text"
        caminho = "../../dhbb/text/*.text"
        arquivos = glob.glob(caminho)
        len(arquivos)
```

[3]: 7687

Temos 7687 verbetes neste diretório. Vamos agora ver como abrir um destes verbetes e inspecionar o seu conteúdo:

```
In [4]: arquivos[0]
```

[4]: '../../dhbb/text/1.text'

Para abrir um arquivo utilizamos um bloco `with`.

```
In [6]: with open(arquivos[0], 'r', encoding='utf8') as arquivo_aberto:
        verbete = arquivo_aberto.read()
        print(verbete)
```

```
---
```

```
title: COELHO, Machado
natureza: biográfico
sexo: m
cargos:
- dep. fed. DF 1927-1929
- dep. fed. DF 1930
- const. 1946
- dep. fed. SP 1946-1951
```

```
---
```

ñJosé Machado Coelho de Castroz nasceu em Lorena (SP).

Estudou no Ginásio Diocesano de São Paulo e bacharelou-se em 1910 pela Faculdade de Ciências Jurídicas e Sociais. Dedicando-se à advocacia, foi promotor público em Cunha (SP) e depois delegado de polícia no Rio de Janeiro, então Distrito Federal.

Iniciou sua vida política como deputado federal pelo Distrito Federal, exercendo o mandato de 1927 a 1929. Reeleito para a legislatura iniciada em maio de 1930, ocupava sua cadeira na Câmara quando, em 3 de outubro, foi deflagrado o movimento revolucionário liderado por Getúlio Vargas. Ligado ao governo federal, encontrava-se ao lado do presidente Washington Luís, no palácio Guanabara, no momento de sua deposição no dia 24 de outubro. Junto com outros companheiros também solidários ao regime deposto e que se haviam asilado em embaixadas e legações, foi enviado em novembro para o estrangeiro. Em outubro de 1932, estava presente no porto de Alcântara, em Lisboa, para receber os revolucionários constitucionalistas exilados pelo governo de Getúlio Vargas após a derrota da revolução irrompida em julho desse ano em São Paulo.

Com a redemocratização do país em 1945, candidatou-se pelo estado de São Paulo, na legenda do Partido Social Democrático (PSD), às eleições para a Assembléia Nacional Constituinte (ANC) realizadas em dezembro desse ano. Obteve uma suplência e, em julho de 1946, foi convocado para participar dos trabalhos constituintes. Com a promulgação da nova Carta (18/9/1946) e a transformação da Constituinte em Congresso ordinário, integrou a Comissão Permanente de Obras Públicas da Câmara Federal, tendo votado em janeiro de 1948 a favor da cassação dos mandatos dos parlamentares comunistas. Deixou a Câmara em janeiro de 1951.

Foi ainda presidente da Companhia de Cimento Vale do Paraíba.

Faleceu no Rio de Janeiro no dia 17 de maio de 1975.

Uma outra maneira de abrir um arquivo, seria como se segue, mas teríamos que usar uma linha de código a mais, para fechar o arquivo, que podemos economizar, lendo o arquivo dentro de um bloco `with` como fizemos anteriormente.

```
In [19]: arquivo_aberto = open(arquivos[0], 'r', encoding='utf8')
```

```
verbete = arquivo_aberto.read()
arquivo_aberto.close()
```

A variável **verbete** que criamos na célula anterior é uma variável do tipo **string**, que é o tipo usado pelo Python para representar um bloco de texto. Podemos manipular o texto dentro de uma **string** de diversas maneiras:

```
In [20]: type(verbete)
```

```
[20]: str
```

```
In [21]: print(verbete.split('---')[1])
```

```
title: COELHO, Machado
natureza: biográfico
sexo: m
cargos:
- dep. fed. DF 1927-1929
- dep. fed. DF 1930
- const. 1946
- dep. fed. SP 1946-1951
```

Tipos de dados em Python, também conhecidos como objetos, possuem métodos. O método **split** do tipo **string** segmenta uma string nas posições em que ocorram uma sequência específica de caracteres, retornando um outro tipo de dado, denominado **lista**.

```
In [22]: type(verbete.split('---'))
```

```
[22]: list
```

Listas são sequências de objetos de quaisquer tipos que também apresentam seu conjunto de métodos. Para descobrir os métodos de qualquer objeto, basta colocar um ponto após o nome da variável e pressionar a tecla <tab>. Listas são delimitadas por colchetes: `[]` (lista vazia). Abaixo vamos dividir o **verbete** em uma lista de **strings**.

```
In [23]: l = verbete.split('---')
1
```

```
[23]: ['',
       '\ntitle: COELHO, Machado\nnatureza: biográfico\nsexo: m\ncargos:\n - dep. fed.\nDF 1927-1929 \n - dep. fed. DF 1930\n - const. 1946\n - dep. fed. SP\n1946-1951\n',
       '\n\nJosé Machado Coelho de Castro nasceu em Lorena (SP).\n\nEstudou no\nGinásio Diocesano de São Paulo e bacharelou-se em 1910 pela\nFaculdade de\nCiências Jurídicas e Sociais. Dedicando-se à advocacia, foi\npromotor público em\nCunha (SP) e depois delegado de polícia no Rio de\nJaneiro, então Distrito\nFederal.\n\nIniciou sua vida política como deputado federal pelo Distrito\nFederal,\nexercendo o mandato de 1927 a 1929. Reeleito para a legislatura\niniciada\nem maio de 1930, ocupava sua cadeira na Câmara quando, em 3 de\noutubro,\nfoi deflagrado o movimento revolucionário liderado por Getúlio\nVargas.\nLigado ao governo federal, encontrava-se ao lado do\npresidente\nWashington Luís, no palácio Guanabara, no momento de sua deposição\nno\ndia 24 de outubro. Junto com outros companheiros também solidários
```

ao regime deposto e que se haviam asilado em embaixadas e legações, foi enviado em novembro para o estrangeiro. Em outubro de 1932, estava presente no porto de Alcântara, em Lisboa, para receber os revolucionários constitucionalistas exilados pelo governo de Getúlio Vargas após a derrota da revolução irrompida em julho desse ano em São Paulo. Com a redemocratização do país em 1945, candidatou-se pelo estado de São Paulo, na legenda do Partido Social Democrático (PSD), às eleições para a Assembleia Nacional Constituinte (ANC) realizadas em dezembro desse ano. Obteve uma suplência e, em julho de 1946, foi convocado para participar dos trabalhos constituintes. Com a promulgação da nova Carta (18/9/1946) e a transformação da Constituinte em Congresso ordinário, integrou a Comissão Permanente de Obras Públicas da Câmara Federal, tendo votado em janeiro de 1948 a favor da cassação dos mandatos dos parlamentares comunistas. Deixou a Câmara em janeiro de 1951. Foi ainda presidente da Companhia de Cimento Vale do Paraíba. Faleceu no Rio de Janeiro no dia 17 de maio de 1975.

Note que nas strings acima existem várias ocorrências da sequência de caracteres '\n'. Esta sequência identifica quebra de linhas. Podemos então utilizá-la para dividir o cabeçalho do verbete em uma lista de linhas:

```
In [24]: cabeçalho = verbete.split('---')[1]
        cabeçalho.splitlines()
```

```
[24]: ['',
        'title: COELHO, Machado',
        'natureza: biográfico',
        'sexo: m',
        'cargos:',
        ' - dep. fed. DF 1927-1929 ',
        ' - dep. fed. DF 1930',
        ' - const. 1946',
        ' - dep. fed. SP 1946-1951']
```

Elementos de uma lista podem ser acessados por sua posição na sequência, por exemplo para acessar a 3ª string da lista:

```
In [25]: print(l[2])
```

José Machado Coelho de Castro nasceu em Lorena (SP).

Estudou no Ginásio Diocesano de São Paulo e bacharelou-se em 1910 pela Faculdade de Ciências Jurídicas e Sociais. Dedicando-se à advocacia, foi promotor público em Cunha (SP) e depois delegado de polícia no Rio de Janeiro, então Distrito Federal.

Iniciou sua vida política como deputado federal pelo Distrito Federal, exercendo o mandato de 1927 a 1929. Reeleito para a legislatura iniciada em maio de 1930, ocupava sua cadeira na Câmara quando, em 3 de outubro, foi deflagrado o movimento revolucionário liderado por Getúlio Vargas. Ligado ao governo federal, encontrava-se ao lado do presidente Washington Luís, no palácio Guanabara, no momento de sua deposição no dia 24 de outubro. Junto com outros companheiros também solidários ao regime deposto e que se haviam asilado em embaixadas e legações, foi enviado em novembro para o estrangeiro. Em outubro de 1932, estava presente no porto de Alcântara, em Lisboa, para receber os

revolucionários constitucionistas exilados pelo governo de Getúlio Vargas após a derrota da revolução irrompida em julho desse ano em São Paulo.

Com a redemocratização do país em 1945, candidatou-se pelo estado de São Paulo, na legenda do Partido Social Democrático (PSD), às eleições para a Assembléia Nacional Constituinte (ANC) realizadas em dezembro desse ano. Obteve uma suplência e, em julho de 1946, foi convocado para participar dos trabalhos constituintes. Com a promulgação da nova Carta (18/9/1946) e a transformação da Constituinte em Congresso ordinário, integrou a Comissão Permanente de Obras Públicas da Câmara Federal, tendo votado em janeiro de 1948 a favor da cassação dos mandatos dos parlamentares comunistas. Deixou a Câmara em janeiro de 1951.

Foi ainda presidente da Companhia de Cimento Vale do Paraíba.

Faleceu no Rio de Janeiro no dia 17 de maio de 1975.

Muitas vezes, as atrings podem vir acompanhadas de um ou mais espaços no início ou no fim. Para removê-los podemos usar o método `strip` como exemplificado abaixo. Caso queiramos remover apenas os espaços no início ou no fim, podemos usar `lstrip` ou `rstrip`, respectivamente.

```
In [15]: " gjsldfkgj ".strip()
```

```
[15]: 'gjsldfkgj'
```

Um outro tipo de estrutura de dados fundamental no Python, é chamado um dicionário, e é denotado por um conjunto de pares de (chave: valor). Abaixo vamos construir um dicionário com os campos de um verbete.

```
In [16]: campos = {l.split(':')[0].strip() : l.split(':')[1].strip() for l in
                cabeçalho.split('\n') if l and ':' in l}
                campos
```

```
[16]: {'title': 'COELHO, Machado',
       'natureza': 'biográfico',
       'sexo': 'm',
       'cargos': ''}
```

No exemplo acima usamos um laço `for` para percorrer repetidamente o campos do cabeçalho e inseri-los um-a-um no dicionário, em apenas uma linha de código. esta maneira de preencher o dicionário é chamada de “*dict comprehension*”. Para entendermos melhor como funciona um laço `for`, e exatamente a sequência de operações realizada acima, vamos escrever “por extenso” o código acima.

```
In [17]: campos = {}
         for linha in cabeçalho.split('\n'):
             if linha and ':' in linha:
                 chave, valor = linha.split(':')
                 campos[chave.strip()] = valor.strip()

         campos
```

```
[17]: {'title': 'COELHO, Machado',
      'natureza': 'biográfico',
      'sexo': 'm',
      'cargos': ''}
```

### I.1. Exercício

Construa para 5 verbetes, um dicionário com o seguinte conteúdo: {"nome-do-cargo": "período"} para todos os cargos de cada verbetado.

```
In [37]: resposta = {}
        for verbeta in arquivos[10:15]:
            with open(verbeta, 'r', encoding='utf8') as verb:
                cabecalho = verb.read().split('---')[1]
                cargos = cabecalho.split('cargos:')[1]
                lista_de_cargos = [cargo.strip('- ') for cargo in cargos.splitlines() if
                cargo.strip('- ') != ""]
                for cargo in lista_de_cargos:
                    if len(cargo.split()) > 1:
                        per = cargo.split()[-1]
                        nome = ' '.join(cargo.split()[:-1])
                    else:
                        nome = cargo.split()[0]
                        per = "NA"
                    resposta[nome] = per
        #     print(cargos)
        print(resposta)

{'const.': '1987-1988', 'dep. fed. DF': '1987-1991'}
{'const.': '1987-1988', 'dep. fed. DF': '1987-1991', 'militar': 'NA', 'gov. RR':
'1967-1974', 'dep. fed. RR': '1975-1983', 'sen. RR': '1991'}
{'const.': '1987-1988', 'dep. fed. DF': '1987-1991', 'militar': 'NA', 'gov. RR':
'1967-1974', 'dep. fed. RR': '1975-1983', 'sen. RR': '1991', 'gov. MT': '1991-1995'}
{'const.': '1946', 'dep. fed. DF': '1987-1991', 'militar': 'NA', 'gov. RR':
'1967-1974', 'dep. fed. RR': '1975-1983', 'sen. RR': '1991', 'gov. MT': '1991-1995',
'rev.': '1930', 'interv. BA': '1942-1945', 'sen. BA': '1946-1955', 'autor.': 'NA',
'Amélia': 'Coutinho'}
{'const.': '1946', 'dep. fed. DF': '1987-1991', 'militar': 'NA', 'gov. RR':
'1967-1974', 'dep. fed. RR': '1975-1983', 'sen. RR': '1991', 'gov. MT': '1991-1995',
'rev.': '1930', 'interv. BA': '1942-1945', 'sen. BA': '1946-1955', 'autor.': 'NA',
'Amélia': 'Coutinho', 'sen. MT': '2007'}
```

Na célula acima contruímos uma variável de tipo *Dicionário*, que é basicamente um conjunto de pares, delimitado por {}. Estes pares são chamados pares chave: valor, como dissemos anteriormente

### I.2. Exercício:

Modifique o código acima para criar outro dicionário com a seguinte estrutura:

```
{
    "nome": ["nome do verbetado", "nome do verbetado", ...],
    "cargo": [cargo 1, cargo2, ...],
    "início": [1987, 1987, ...],
```

```
"fim": [1988, 1991, ...]
}
```

```
In [ ]:
```

```
In [ ]:
```

### I.3. Abrindo um grande número de documentos texto

Como vimos acima existem 7687 verbetes à nossa disposição no disco, mas não podemos abrir todos ao mesmo tempo pois, em primeiro lugar podem não caber na memória, em segundo lugar raramente precisaremos inspecioná-los todos ao mesmo tempo. O mais comum é analisá-los em sequência. Vamos inspecionar os primeiros 10:

```
In [13]: for a in arquivos[:10]:
           with open(a, 'r') as f:
               verbete = f.readlines()
           print('Verbete: ', a.split('.text')[0].split('/')[1])
           print(verbete[1])
```

```
Verbete: 1956
title: FERRAZ, Gabriel Lopes
```

```
Verbete: 10978
title: DESCONSI, Orlando
```

```
Verbete: 2687
title: LACERDA, Jorge
```

```
Verbete: 3429
title: MELO, Geraldo Medeiros de
```

```
Verbete: 2839
title: LIMA, Alceu Amoroso
```

```
Verbete: 2088
title: FONTES, Tomás
```

```
Verbete: 11055
title: FROSSARD, Denise
```

```
Verbete: 11650
title: MURAD, Jamil
```

```
Verbete: 12159
title: SÁ, Liliam
```

```
Verbete: 4940
title: SEIXAS, Luís Siqueira
```

```
In [14]: arquivos[1]
```

```
[14]: '../dhbb/text/10978.text'
```

Acima utilizamos uma estrutura de repetição, denominada “laço for” para abrir sequencialmente os arquivos. É importante notar que a cada volta do laço, o arquivo texto é atribuído à mesma variável, o que significa que nunca há mais do que apenas um verbete na memória. Desta forma poderíamos potencialmente analisar todos os milhares de verbetes ocupando apenas uma quantidade pequena e constante de memória. Outro detalhe do código acima é que, para facilitar a extração do título do verbete, Fizemos a leitura do arquivo com o método `readlines` que retorna o verbete já dividido em uma lista de linhas ao invés de uma `string`.

#### I.4. Outros recursos do DHBB

O arquivo do DHBB disponível no Github oferece outros recursos textuais para nos auxiliar em nossa pesquisa, como por exemplos dicionários com identificadores de “Entidades” presentes nos verbetes, como pessoas, organizações, eventos, etc.

```
In [26]: with open("../dhbb/dic/pessoa-individuo.txt", 'r') as f:
          pessoas = f.readlines()
          pessoas[:10]
```

```
[26]: ['Aarão Rebelo\n',
       'Aarão Steinbrüch\n',
       'Abalcazar Garcia\n',
       'Abdias Do Nascimento\n',
       'Abdon Goncalves Nanhay\n',
       'Abdon Gonçalves\n',
       'Abdon Sena\n',
       'Abdon de Mello\n',
       'Abdur R. Khan\n',
       'Abel Avila dos Santos\n']
```

```
In [28]: with open("../dhbb/dic/pessoa-papel.txt", 'r') as f:
          profissão = f.readlines()
          profissão[:10]
```

```
[28]: ['Advogado\n',
       'Advogado Geral da União\n',
       'Agente de investimento\n',
       'Agente de segurança judiciária\n',
       'Alfaiate\n',
       'Analista administrativo\n',
       'Analista de comércio exterior\n',
       'Antiquário\n',
       'Arcebispo\n',
       'Armador\n']
```

```
In [29]: with open("../dhbb/dic/evento.txt", 'r') as f:
          evento = f.readlines()
          evento[:10]
```

```
[29]: ['A Rusga\n',
       'ATENTADO DA TONELEIROS\n',
       'ATENTADO DO RIOCENTRO\n',
       'Aclamação de Amador Bueno\n',
       'Balaiada\n',
```



```
'Batalha da Maria Antônia\n',
'Batalha da Venda Grande\n',
'Batalha das Toninhas\n',
'Batalha de Santa Luzia\n',
'COMÍCIO DAS REFORMAS\n']
```

```
In [30]: with open("../dhbb/dic/organizacao.txt", 'r') as f:
          organização = f.readlines()
          organização[:10]
```

```
[30]: ['Abrigo Lar dos Velhos Vicentini\n',
       'Academia Alagoana de Letras\n',
       'Academia Brasileira de Ciências\n',
       'Academia Brasileira de Ciências Econômicas e Administrativas\n',
       'Academia Brasileira de Ciências Sociais e Políticas\n',
       'Academia Brasileira de Direito Empresarial\n',
       'Academia Brasileira de Letras\n',
       'Academia Brasileira de Música\n',
       'Academia Brasiliense de Letras\n',
       'Academia Cultural de Curitiba\n']
```

```
In [32]: with open("../dhbb/dic/formulacao-politica.txt", 'r') as f:
          politica = f.readlines()
          politica[:10]
```

```
[32]: ['anteprojeto Constitucional\n',
       'anteprojeto da Carta Magna\n',
       'anteprojeto da Comissão Provisória\n',
       'anteprojeto da Comissão Provisória de Estudos Constitucionais\n',
       'anteprojeto da Comissão de Sistematização\n',
       'anteprojeto da Consolidação das Leis do Trabalho\n',
       'anteprojeto da Constituição\n',
       'anteprojeto da Lei Orgânica da Magistratura\n',
       'anteprojeto da Lei de Acidentes no Trabalho\n',
       'anteprojeto da Lei de Direitos Autorais\n']
```

## II. EXTRAINDO INFORMAÇÃO ESTRUTURADA

Agora que sabemos como abrir arquivos de texto e ler o seu conteúdo, podemos experimentar a extração de informações específicas dos verbetes e organizá-la em uma tabela. Para isso vamos lançar mão de uma biblioteca chamada **Pandas** para organizar em uma estrutura tabular, chamada **DataFrame** os dados que vamos extrair.

```
In [15]: import pandas as pd
          pd.set_option("display.latex.repr", True)
```

Nós vimos acima que os verbetes contém uma seção inicial delimitada pelos caracteres --- vamos utilizar esta característica do texto para guiar nossa extração de informação. Como você pode perceber, já começamos a reutilizar código que escrevemos anteriormente. Para facilitar o reuso e reduzir a necessidade de escrever múltiplas vezes o mesmo código vamos aprender a organizá-lo melhor. Vamos começar definindo uma função.

```
In [16]: def tabula_verbete(n=None):
          """
```

```

Carrega todos os verbetes disponíveis, ou os primeiros n.
n: número de verbetes a tabular
"""
if n is None:
    n = len(arquivos)
linhas = []
for a in arquivos[:n]:
    with open(a, 'r') as f:
        verbete = f.read()
        cabeçalho = verbete.split('---')[1]
        campos = {l.split(':')[0].strip(): l.split(':')[1].strip() for l in
cabeçalho.split('\n')[:4] if l}
        campos['arquivo'] = os.path.split(a)[1]
#         campos['cargos'] = cabeçalho.split('cargos:')[1]
#         campos['corpo'] = verbete.split('---')[2]
        linhas.append(campos)
tabela = pd.DataFrame(data = linhas, columns=['arquivo', 'title', 'natureza',
'sexo'])
return tabela

```

A função acima inclui a maior parte do código que escrevemos anteriormente, só que encapsulado em uma função que nos permite executar a extração e tabulação do cabeçalho para o número de verbetes que desejarmos. Podemos ver abaixo que na verdade é muito rápido processar todos os verbetes.

```
In [17]: help(tabula_verbete)
```

Help on function tabula\_verbete in module \_\_main\_\_:

```

tabula_verbete(n=None)
    Carrega todos os verbetes disponíveis, ou os primeiros n.
    n: número de verbetes a tabular

```

```
In [18]: tab = tabula_verbete()
         tab.head()
```

```
[18]:
```

	arquivo	title	natureza	sexo
0	1956.text	FERRAZ, Gabriel Lopes	biográfico	m
1	10978.text	DESCONSI, Orlando	biográfico	m
2	2687.text	LACERDA, Jorge	biográfico	m
3	3429.text	MELO, Geraldo Medeiros de	biográfico	m
4	2839.text	LIMA, Alceu Amoroso	biográfico	m

Podemos visualizar uma descrição básica da tabela resultante

```
In [19]: tab.describe()
```

```
[19]:
```

	arquivo	title	natureza	sexo
count	7687	7687	7687	6722
unique	7687	7596	2	2
top	1638.text	ALBUQUERQUE, Carlos	biográfico	m
freq	1	3	6724	6517

Por exemplo fica fácil ver que no DHBB predominam biografias de personagens do sexo masculino.

```
In [20]: print(tab.sexo.value_counts())
```

```
m      6517
f       205
Name: sexo, dtype: int64
```

Percebemos também que a natureza predominante dos verbetes é biográfica e que só existem duas naturezas, mas qual a outra?

```
In [21]: print(tab.natureza.value_counts())
```

```
biográfico      6724
temático        963
Name: natureza, dtype: int64
```

### III. EXPORTANDO PARA BANCOS DE DADOS

Depois de realizarmos a nossa análise e tabular os resultados, podemos exportar a tabela em vários formatos. Em primeiro lugar, caso queiramos abrir nosso trabalho em uma planilha, devemos salvar no formato CSV, ou “comma-separated-values”. Este formato pode ser aberto imediatamente em uma planilha.

```
In [22]: tab.to_csv("minha_tabela.csv", sep=',')
```

Acima usamos o caractere “|” como separador para evitar confusões com as vírgulas existentes no texto. ## Exportando para um banco de dados relacional Para exportar para um banco relacional, precisamos de uma biblioteca adicional, o [SQLAlchemy](#). Esta biblioteca nos permite interagir com a maioria dos bancos relacionais. Aqui vamos usar o banco [SQLite](#).

```
In [32]: from sqlalchemy import create_engine
```

```
In [34]: engine = create_engine('sqlite:///minha_tabela.sqlite', echo=False)
         tab.to_sql('resultados', con=engine, if_exists='append')
```

Uma vez inserido no banco relacional, podemos fazer consultas aos dados usando a linguagem SQL. Abaixo obtemos o resultado da consulta em uma lista.

```
In [39]: engine.execute("select * from resultados where natureza='temático']").
         ↪fetchall()[:10]
```

```
[39]: [(354, '10989.text', 'Destacamento de Operações e Informações Centro de
Operações e Defesa Interna (DOI-CODI)', 'temático', None),
(1027, '11595.text', 'Agência Brasileira de Inteligência (Abin)', 'temático',
None),
(1028, '11596.text', 'Associação Brasileira de Emissoras de Rádio e Televisão
(ABERT)', 'temático', None),
(1029, '11597.text', 'Associação Nacional de Jornais (ANJ)', 'temático', None),
(1030, '11598.text', 'Associação Nacional dos Membros do Ministério Público
(CONAMP)', 'temático', None),
(1031, '11599.text', 'CAROS AMIGOS', 'temático', None),
(1034, '11600.text', 'CARTA CAPITAL', 'temático', None),
(1035, '11601.text', 'Central dos Trabalhadores e das Trabalhadoras do Brasil
(CTB)', 'temático', None),
(1036, '11602.text', 'Central Geral dos Trabalhadores do Brasil (CGTB)',
'temático', None),
(1037, '11603.text', 'Conselho de Comunicação Social (CCS)', 'temático', None)]
```

Se quisermos os resultados na forma de um Dataframe, podemos usar o **Pandas**.

```
In [40]: pd.read_sql_query("select * from resultados where natureza='temático'",
                           con=engine).head()
```

[40]:

	index	arquivo	title		natureza	sexo
0	354	10989.text	Destacamento de Operações e Informações Cent...		temático	None
1	1027	11595.text	Agência Brasileira de Inteligência (Abin)		temático	None
2	1028	11596.text	Associação Brasileira de Emissoras de Rádio e ...		temático	None
3	1029	11597.text	Associação Nacional de Jornais (ANJ)		temático	None
4	1030	11598.text	Associação Nacional dos Membros do Ministério ...		temático	None

#### IV. EXERCÍCIOS

1. Construa uma função para buscar apenas verbetes de personagens que tenham ocupado o cargo de deputado federal. Tabule os resultados incluindo o número de mandatos.
2. Construa uma função para buscar o primeiro verbete temático e apresente o seu conteúdo.
3. Encontre todos os verbetes que contenham “Academia Brasileira de Letras”. Que porcentagem destes correspondem a membros da dita academia?
4. Construa uma linha do tempo que represente a cobertura histórica do DHBB.

```
In [ ]:
```