Relatório de Estágio Interno Supervisionado Obrigatório

PONDOC

CEFET/RJ – CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA CELSO SUCKOW DA FONSECA

Discente: Isabelle Ferreira de Araujo

Curso Técnico em Informática

Turma: 4BINFO (2021)

Professor Supervisor: Eduardo Bezerra da Silva

Professor(a) Orientador(a): Carolina de Lima Aguilar

Rio de Janeiro, RJ

Julho de 2022

APROVAÇÃO

RELATÓRIO DE ESTÁGIO SUPERVISIONADO OBRIGATÓRIO: PONDOC

ISABELLE FERREIRA DE ARAUJO
<isabelletecn@gmail.com>

Resumo: A partir de produções bibliográficas dos docentes do Programa de Pós-graduação em Ciência da Computação (PPCIC) do CEFET-RJ, o projeto gerou um relatório em Excel para análise de pontuação destes docentes, dando importância a avaliação Qualis das conferências e das revistas de publicação.

Palavras-chave: Python, Excel, Análise de dados, Análise sintática

Rio de Janeiro, 28 de julho de 2022

_____	_____
Eduardo Bezerra da Silva	Carolina de Lima Aguilar
Professor Supervisor – CEFET-RJ	Professor(a) Orientador(a) – CEFET-RJ

_____

Isabelle Ferreira de Araujo

Estagiário(a) – CEFET- RJ

Rio de Janeiro, RJ

Julho de 2022

Sumário

Introdução

O presente relatório tem como objetivo apresentar e descrever o desenvolvimento das atividades desempenhadas no estágio interno supervisionado obrigatório no CEFET/RJ. Isabelle Ferreira de Araujo exerceu suas atividades no estágio via online, na área de desenvolvimento de software em Python, atendendo ao professor Eduardo Bezerra da Silva.

O discente ficou ciente do estágio através do e-mail enviado pelo COINFO que divulgava oportunidades de estágios internos supervisionados por professores do COINFO. O objetivo do estágio era, a partir de produções bibliográficas dos docentes do Programa de Pós-graduação em Ciência da Computação (PPCIC) do CEFET-RJ, gerar um relatório em formato de planilha do Excel para análise de pontuação destes docentes, dando importância a avaliação Qualis das conferências e das revistas de publicação.

O estágio foi realizado à distância, de forma online. O professor Eduardo Bezerra junto a aluna realizava reuniões semanais para traçar objetivos, avaliar atividades realizadas, analisar e gerar soluções para possíveis problemas dessas atividades.

Estágio na Empresa

O estágio foi realizado de 28 de dezembro de 2021 a 16 de junho de 2022. As atividades da discente Isabelle Araujo tinham como primeiro objetivo extrair dados da url: https://eic.cefet-rj.br/lattes, e passá-los para uma função, fornecida por Eduardo Bezerra, que separou e classificou cada informação. O segundo era fazer um script que criasse um banco de dados com os nomes completos e possíveis formas de referências bibliográficas dos discentes e docentes, e informações Qualis. O terceiro era criar uma função para verificar se algum e qual(is) docente(s) do PPCIC e discentes participaram de conferências, publicações em revistas ou publicações aceitas, além de normalizar as citações para nomes completos. A última era transcrever essas informações em um arquivo existente com a extensão xlsx.

Mesmo iniciando o estágio sem nenhum conhecimento prévio em Python, a adaptação foi fácil, pois foi dado um tempo antes das atividades para que a aluna pudesse aprender o básico da linguagem. Em caso de dúvidas era possível se comunicar com o professor supervisor por mensagens e pedir ajuda, além de poder se comunicar com outros alunos em estágios diferentes.

No geral, a discente Isabelle F. de Araujo teve uma ótima adaptação e realizou suas atividades com a ajuda e orientação do professor supervisor Eduardo Bezerra dentro do prazo previsto. Com as atividades aprendeu a compreender os erros, buscar possíveis soluções nos meios disponíveis e com outras pessoas, e a corrigi-los.

Atividades Realizadas

Após o período de estudos da linguagem Python, foram iniciadas as atividades. A primeira atividade foi analisar a estrutura HTML das URLs para que fosse realizada a construção da função htmlInfos, que filtraria as informações das produções/trabalhos dos professores pesquisadores e discentes. Após isso foi necessária a criação de funções para filtrar, organizar e classificar os tipos de dados que htmlInfos retornou.

A segunda atividade foi a construção do banco de dados com informações dos discentes, professores pesquisadores do PPCIC e informações sobre revistas e conferências. Com essas informações foi possível normalizar as referências bibliográficas dos pesquisadores e discentes para seus nomes completos, além de remover referencias de outros participantes que não fossem do PPCIC ou discentes do CEFET/RJ.

A terceira foi pontuar quais pesquisadores participaram de quais produções ou trabalhos, e transferir estes dados, as informações classificadas e algumas fórmulas específicas uma a uma para uma planilha EXCEL preexistente, default.xlsx ou com outro nome que o usuário desejar.

PONDOC

Linguagem: Python (3.10.5)

Bibliotecas:

- bs4 (0.0.1)
- fuzzywuzzy (0.18.0)
- openpyxl (3.0.9)
- pandas (1.4.2)
- psycopg2 (2.9.3)
- pyparsing (3.0.6)
- requests (2.27.1)

Instalação

1. Python
   Realize o download do Python pelo site oficial seguindo as instruções nele contidas.
   Obs.: Pandas é instalado junto ao Python.
2. pondoc
   i. Clone o repositório ou faça o download do arquivo zip.

```
git clone https://github.com/araujobtc/pondoc.git
```

No mesmo diretório onde clonou o repositório, instale os pré-requisitos abaixo.

3. bs4

```
pip install bs4
```

4. fuzzywuzzy

```
pip install fuzzywuzzy
```

5. openpyxl

```
pip install openpyxl
```

6. psycopg2

```
pip install psycopg2
```

7. pyparsing

```
pip install pyparsing
```

8. requests

```
pip install requests
```

database.py

Arquivo com funções construídas com base na biblioteca psycopg2, para conectar, criar, deletar ou atualizar dados em um banco de dados existente.

conect_db

A função tem o propósito de gerar uma conexão com um banco de dados. Para que funcione é necessária a alteração das informações sobre o banco de dados e o usuário. Informações: host, dbname (banco de dados), user, password, port. Retorna uma variável com a conexão com o banco de dados pelo psycopg2.

create_drop_db

Recebe como parâmetro um script SQL. Se conecta ao banco de dados com a conect_db e executa o script SQL. Pode ser utilizado para criar ou dropar uma tabela do banco de dados. Não retorna nada.

Exemplo parâmetro:

```
create_drop_db(‘‘‘CREATE TABLE teste ( t   VARCHAR(2),
                                       PRIMARY KEY (t)’’’))
```

insert_delete_db

Recebe como parâmetro um script SQL. Se conecta ao banco de dados com a conect_db e tenta executar o script SQL Pode ser utilizado para inserir ou deletar dados de alguma tabela. Ao tentar executar o script, se obter sucesso não retorna nada, porém se ocorrer um erro, o apresenta no terminal junto a detalhes.

Exemplo parâmetro e erro: Tentativa de duplicar chave primária.

```
insert_delete_db(‘INSERT INTO teste (t) VALUES (“2b”)’)
```

```
Error: duplicate key value violates unique comstraint
"teste_pkey"
DETAIL:  Key (t)=(2b) already exists.
```

consult_db

Esta função tem o objetivo de apresentar os dados existentes no banco de dados, utilizando conect_db para se conectar, executa um script SQL e realiza uma pesquisa. A partir dessa pesquisa a função gera uma lista com diversas tuplas contendo os resultados. é possível utilizar a biblioteca pandas para visualizar melhor as informações.

Exemplos parâmetro e resultados:

```
import pandas as pd
print(‘Sem pandas: ’, consult_db(‘SELECT * FROM teste’))
print(‘Com pandas:\n’, pd.DataFrame(consult_db(‘SELECT * FROM
teste’)))
```

```
Sem pandas: [(1a), (2b)]
Com pandas:
      0
0     2b
```

scriptsdb.py

O arquivo scriptsdb.py deve ser rodado antes de todo o programa e tem a finalidade de encaminhar os scripts SQL para as funções do database.py no intuito da construção das tabelas e inserção de dados nas mesmas, utilizando os arquivos PPCICresearchers.json, Discentes.csv e Qualis.csv, que contêm, respectivamente: nomes completos e possíveis referências bibliográficas de professores pesquisadores; nomes completos e possíveis referências bibliográficas de discentes; ISSNs, nomes e nota qualis de revistas.

Tabelas:

researchers	**students**	**qualis**
colunas: nome (VARCHAR=255) referência (VARCHAR=50)	**colunas:** nome (VARCHAR=255) referência (VARCHAR=50)	**colunas:** issn (VARCHAR=9) nome (VARCHAR=255) qualis (VARCHA=2)
primary key: nome, referência	**primary key**: nome, referência	**primary key**: issn, nome

reader.py: htmlInfos

A função tem o objetivo de extrair conteúdo de uma URL passada como parâmetro, utilizando a biblioteca requests, e filtrá-lo, com bs4, deixando passar e atribuindo a uma lista apenas strings com as informações sobre os trabalhos/artigos dos pesquisadores e/ou dos discentes. Ao final a função retorna a lista gerada.

Exemplo parâmetro e resultado:

```
print(htmlInfos('https://eic.cefet-rj.br/lattes/ppcic-2022/PB4-0.html'))
```

```
['ASSIS JR., L. B. ; SILVA, F. L. E. ; HENRIQUES, F. R. ; GUERRA, R. P. O. ; CARVALHO, C. S. ; BRANDAO, D. N.. Construção e Validação de um Sistema IoT de Baixo Custo para Detecção de Vazamento de Água em Residências (to appear). Em: XIII Computer on the Beach, v. 1, 2022. B4 (Computer on The Beach).', ..., 'SOUZA, J. A. ; PINHO, A. B. C. ; FERREIRA, C. B. M. ; MONTEIRO, A. F. A. ; HENRIQUES, F. R.. An IoT-based System for Landslides Warnings Using a Real Time Slope Monitoring Model. Em: XIII Computer on the Beach, 2022. B4 (Computer on The Beach).']
```

analyzer.py

Arquivo com funções construídas com base na biblioteca PyParsing, para analisar, filtrar e classificar as informações obtidas com a função htmlInfos.

makeDecoratingParseAction

makeDecoratingParseAction junto à setParseAction tem a finalidade de 'decorar' e classificar tokens retornados de pesquisas de strings com estruturas e conteúdo específicos, a função adiciona um marcador aos tokens para indicar qual correspondência foi feita a partir da pesquisa. Recebe como parâmetro o título do marcador.

Exemplo parâmetro e resultado:

```
A = Word(nums).setParseAction(makeDecoratingParseAction("A"))
expr = OneOrMore(A)
print(expr.parseString("123 456 7"))
```

```
[('A', '123'), ('A', '456'), ('A', '7')]
```

parseRef

A função recebe como parâmetro um item por vez do resultado gerado por htmlInfos que utiliza URLs semelhantes à https://eic.cefet-rj.br/lattes/ppcic-2022/PB0-0, https://eic.cefet-rj.br/lattes/ppcic-2022/PB7-0, e https://eic.cefet-rj.br/lattes/ppcic-2022/PB4-0. As URLs sofrem alterações de acordo com o ano/período que o usuário escolher.

parseRef tem o objetivo de procurar, filtrar e classificar as informações de seu parâmetro (string). Após realizar buscas na string e encontrar um resultado que se encaixa nos parâmetros da pesquisa, o resultado é classificado com makeDecoratingParseAction e setParseAction.

É retornado por ela os autores, o título do trabalho/produção, o nome da revista ou conferência, o ano em que foram realizados ou publicados, o ISSN se houver, e a avaliação qualis.

Obs.: Em teste realizados, foi descoberta que a existência de "." no trecho correspondente ao título do trabalho/produção ou nome da revista/ conferência da string, ocasiona em um erro, impedindo que a função encontre o final correto e obtenha o trecho desejado completo.

Exemplo 1 parseRef – parâmetro e resultado

```
print(parseRef('ARAUJO, ISABELLE ; BEZERRA, EDUARDO. Título.
Revista. v. 1, p. 1, issn: 0000-0000, 2022. A1.'.upper()))
```

```
ParseResults([('author', ParseResults(['ARAUJO', 'ISABELLE'],
{'author_name_before_comma': 'ARAUJO',
'author_name_after_comma': ['ISABELLE']})), ('author',
ParseResults([' BEZERRA', 'EDUARDO'],
{'author_name_before_comma': ' BEZERRA',
'author_name_after_comma': ['EDUARDO']})), ('title',
ParseResults(['TÍTULO'], {'Title': ['TÍTULO']})), '.',
('conference_journal', ParseResults(['REVISTA'],
{'conferjournal_name': ['REVISTA']})), ('Remaining',
ParseResults(['V.', '1,', 'P.', '1,', 'ISSN:', '0000-0000,'],
{})), ('year', ParseResults(['2022'], {})), ('qualis',
ParseResults(['A1'], {}))], {'AuthorLst': [('author',
ParseResults(['ARAUJO', 'ISABELLE'],
{'author_name_before_comma': 'ARAUJO',
'author_name_after_comma': ['ISABELLE']})), ('author',
ParseResults([' BEZERRA', 'EDUARDO'],
{'author_name_before_comma': ' BEZERRA',
'author_name_after_comma': ['EDUARDO']}))], 'Title':
'title'})
```

Exemplo 2 parseRef – parâmetro e resultado:

```
print(parseRef('ARAUJO, ISABELLE F. ; BEZERRA, EDUARDO.
Título. Em: Conferência, v. 1, 2022. C'.upper()))
```

```
ParseResults([('author', ParseResults(['ARAUJO', 'ISABELLE',
'F.'], {'author_name_before_comma': 'ARAUJO',
'author_name_after_comma': ['ISABELLE', 'F.']})), ('author',
ParseResults([' BEZERRA', 'EDUARDO'],
{'author_name_before_comma': ' BEZERRA',
'author_name_after_comma': ['EDUARDO']})), ('title',
ParseResults(['TÍTULO'], {'Title': ['TÍTULO']})), '.',
('conference_journal', ParseResults(['CONFERÊNCIA'],
{'conferjournal_name': ['CONFERÊNCIA']})), ('Remaining',
ParseResults(['V.', '1,'], {})), ('year',
ParseResults(['2022'], {})), ('qualis', ParseResults(['C'],
{}))], {'AuthorLst': [('author', ParseResults(['ARAUJO',
'ISABELLE', 'F.'], {'author_name_before_comma': 'ARAUJO',
'author_name_after_comma': ['ISABELLE', 'F.']})), ('author',
ParseResults([' BEZERRA', 'EDUARDO'],
{'author_name_before_comma': ' BEZERRA',
'author_name_after_comma': ['EDUARDO']}))], 'Title':
'title'})
```

infosCitation

infosCitation trata as informações retornadas por parseRef e parseConferenceRef, separando os dados de acordo com suas classificações, removendo suas marcações e os adicionando em variáveis específicas para cada um. A função retorna, em geral, uma lista contendo as referências dos autores, o título, o nome da revista/conferência, o ano, e a avaliação qualis. Se as informações vierem de parseRef pode haver ou não uma variável contendo ISSN.
Exemplo parâmetro e resultado:

```
print(infosCitation(ParseResults([('author',
ParseResults(['ARAUJO', 'ISABELLE'],
{'author_name_before_comma': 'ARAUJO',
'author_name_after_comma': ['ISABELLE']})), ('author',
ParseResults([' BEZERRA', 'EDUARDO'],
{'author_name_before_comma': ' BEZERRA',
'author_name_after_comma': ['EDUARDO']})), ('title',
ParseResults(['TÍTULO'], {'Title': ['TÍTULO']})), '.',
('conference_journal', ParseResults(['REVISTA'],
{'conferjournal_name': ['REVISTA']})), ('Remaining',
ParseResults(['V.', '1,', 'P.', '1,', 'ISSN:', '0000-0000,'],
{})), ('year', ParseResults(['2022'], {})), ('qualis',
ParseResults(['A1'], {}))], {'AuthorLst': [('author',
ParseResults(['ARAUJO', 'ISABELLE'],
{'author_name_before_comma': 'ARAUJO',
'author_name_after_comma': ['ISABELLE']})), ('author',
ParseResults([' BEZERRA', 'EDUARDO'],
{'author_name_before_comma': ' BEZERRA',
'author_name_after_comma': ['EDUARDO']}))], 'Title':
'title'})))
```

```
([['ARAUJO', 'ISABELLE'], [' BEZERRA', 'EDUARDO']], 'TÍTULO',
' REVISTA', ‘0000-0000’, '2022', 'A1')
```

parsePublication

parsePublication recebe como parâmetro um item do resultado htmlInfos. Elas têm apenas a finalidade de passar este item para parseRef e em seguida passar os resultados obtidos para infosCitation. parsePublication retorna o resultado gerado por infosCitation.

normalizer.py: normalizer

normalizer recebe como parâmetro a lista com os autores gerada por infosCitation. A função compara, uma a uma, as referências obtidas do site com as referências de professores pesquisadores e de discentes do banco de dados. Caso não ocorra combinação, aquela referência do site é simplesmente ignorada e é adicionada apenas um vazio (“ “) a duas listas, umas para os alunos(as) e outra para os pesquisadores. Em caso de combinação, é adicionado a uma lista ou outra o nome completo do professor ou discente. É retornado a lista com os nomes completos dos participantes que forem discentes do CEFET-RJ ou professor do PPCIC e estejam no banco de dados.

Exemplo parâmetro e resultado:

```
print(normalizer([['ARAUJO', 'ISABELLE', 'F.'], [' BEZERRA',
'EDUARDO']]))
```

```
(['EDUARDO BEZERRA DA SILVA'], '')
```

handler.py

qualisInfos

A função tem o objetiva de verificar a real avaliação qualis de uma conferência ou revista. Recebe duas listas com todos os nomes das revistas/conferências e possivelmente os ISSNs obtidos do site, com elas é feita a comparação do nome com a relação qualis do banco de dados. É criada duas listas nesta função, qualis e color, umas para a nota qualis e uma para receber 0, caso a combinação seja perfeita, e 1, caso a combinação não seja perfeita ou não haja.

Para saber quão perfeita é a combinação, utilizamos fuzzywuzzy para medir, em porcentagem, e em casos de combinação perfeita (100%) é adicionado a qualis a avaliação qualis e a color um 0, e se não for 100%, mas estiver acima de 95%, ao invés de 0 é adicionado 1 a color. Se a combinação é menor que 95% é realizada a comparação do ISSN com o banco de dados, neste caso é necessária a combinação de 100%, uma vez que seja perfeita, ocorre o mesmo processo, é adicionado a qualis a avaliação qualis e a color um 1. Por último caso não ocorra nenhuma combinação, é adicionado a qualis a avaliação: ‘NI’ e a color: 1.

Exemplo parâmetro e resultado:

```
print(qualisInfos(['REVISTA', 'CONFERÊNCIA'], ['0000-0000',
'']))
```

```
(['NI', 'NI'], [1, 1])
```

refInfos

Como parâmetro, recebe no formato de lista todos os itens de htmlInfos, com exceção dos autores que vai como ‘ ’, após serem passados por parseJournalPublication, parseConferencePublication, a função separa novamente os itens, os colocando em uma lista específica para cada, separando também as informações de revistas das informações das conferências. Em seguida passa a lista com os nomes das revistas/conferências e os ISSNs para qualisInfos, o resultado de qualisInfos é retornado por refInfos junto as listas criadas nela.

Exemplo parâmetro e resultado:

```
print(refInfos([['','TÍTULO', 'REVISTA', '0000-0000', '2022', 'NÃO IDENTIFICADO (REVISTA)'], ['','TÍTULO 2', 'REVISTA 2', '1111-2222', '2022', 'B5']], [['', 'TÍTULO', 'CONFERÊNCIA', '2022', 'C'], ['', 'TÍTULO 2', 'CONFERÊNCIA 2', '2022', 'A1']]))
```

```
(['TÍTULO', 'TÍTULO 2'], ['REVISTA', 'REVISTA 2'], ['0000-0000', '1111-2222'], ['2022', '2022'], ['NI', 'NI'], [1, 1], ['TÍTULO', 'TÍTULO 2'], ['CONFERÊNCIA', 'CONFERÊNCIA 2'], ['2022', '2022'], ['C', 'A1'])
```

researchersCorrelation

O parâmetro desta função é a lista com os nomes completos dos pesquisadores autores (resultado do normalizer). Primeiro é criado um dicionário com todos os nomes completos dos professores, em seguida, com lista, é feita uma correlação de quais professores participaram de quais trabalhos. A correlação é feita inserindo ‘ ‘ ou 1 nas listas, do dicionário, relacionadas a estes pesquisadores.

Exemplo parâmetro e resultado:

```
print(researchersCorrelation(['EDUARDO BEZERRA DA SILVA']))
```

```
{'DIEGO BARRETO HADDAD': [''], 'DIEGO NUNES BRANDÃO': [''], 'DOUGLAS DE OLIVEIRA CARDOSO': [''], 'EDUARDO BEZERRA DA SILVA': [1], 'EDUARDO SOARES OGASAWARA': [''], 'FELIPE DA ROCHA HENRIQUES': [''], 'GLAUCO FIOROTT AMORIM': [''], 'GUSTAVO PAIVA GUEDES E SILVA': [''], 'JOAO ROBERTO DE TOLEDO QUADROS': [''], 'JOEL ANDRÉ FERREIRA DOS SANTOS': [''], 'JORGE DE ABREU SOARES': [''], 'KELE TEIXEIRA BELLOZE': [''], 'LAURA SILVA DE ASSIS': [''], 'PEDRO HENRIQUE GONZÁLEZ SILVA': [''], 'RAFAELLI DE CARVALHO COUTINHO': ['']}
```

transcriber.py

conferData, periodData, lconferData, lperiodData

Funções que passam uma a uma os itens das listas retornadas por refInfos, além de fórmulas para planilhas EXCEL, para suas respectivas folhas num arquivo com extensão .xlsx escolhido pelo usuário. conferData recebe os títulos, o nome das conferências, o ano, a lista com os discentes que participaram e um caminho para o arquivo e insere esses dados na folha ‘Conferencias’. periodData é semelhante ao conferData, sendo alterado apenas o nome das conferências e pelos nomes das revistas, e a relação de 0 e 1 (color),

gerado em qualisInfos, e insere estes dados em 'Periodicos'. Em lconferData é passado a lista com os nomes das conferências e a lista com as notas qualis do site como parâmetro e a função insere essas informações em 'LConferencias'. O lperiodData difere bastante recebendo listas com os nomes das revistas, ISSNs, color, e avaliação qualis e inserindo esses dados em 'LPeriodicos'.

Obs.: não é passado para conferData e lconferData a relação de 0 e 1, gerado em qualisInfos por falta de informações sobre as conferências para que houvesse comparação. E não é passado para lconferData a nota qualis real pelo mesmo motivo.

insertData

Esta função tem o objetivo de inserir os dados que foram obtidos durante o projeto na planilha EXCEL, podendo ser default.xlsx ou outro arquivo escolhido pelo usuário com extensão xlsx. insertData utiliza openpyxl abrir o arquivo e permitir a edição dele, em seguida ela chama e passa as informações para as funções: conferData, periodData, lconferData, lperiodData, que inserem os dados no arquivo. Também insere a correlação de professores e trabalhos gerada por researchersCorrelation. Após a função inserir os dados no arquivo, no caso de ter sido usado o arquivo default.xlsx, ele é salvo em um novo arquivo com o nome contendo o período, exemplo: 'producao2020-2022.xlsx'. Se o usuário inserir o nome de um arquivo diferente os dados serão inseridos e salvos nele.

main.py: main

A main é a função principal deste projeto, ela utiliza argparse para que o usuário informe o arquivo que deseja editar, caso não informe é entendido que deseja editar o arquivo default, e de qual período ele deseja as informações sobre produções/trabalhos que serão inseridas no arquivo.

Uma vez que o usuário informe o período e o arquivo, a função percorre todos os anos do período escolhido e passa as URLs, com alterações no ano, para htmlInfos e o resultado é passado para parseJournalPublication, parseConferencePublication que filtra, separa e classifica os dados.

Após a classificação dos dados, a lista de autores é normalizada com o normalizer e é passada para insertData junto a lista com os resultados gerados por parseJournalPublication, parseConferencePublication.

Conclusão

Enfim, o estágio interno foi uma ótima forma de pôr em prática o que foi estudado no CEFET-RJ e de adquirir novos conhecimentos. Com o decorrer do estágio, houve a necessidade de realizar pesquisas, e aprender a entender e corrigir os problemas que surgiam, entretanto, isso se deu por termos tido a demanda de usar recursos que estão fora da grade de ensino.

Referências Bibliográficas

BEAUTIFUL Soup Documentation. crummy, c2020. Disponível em: crummy. Último acesso em: 02/03/2022.

BEZERRA, Eduardo. ESU-Isabelle. colab google, 2022. Disponível em: colab google. Último acesso em: 07/04/2022.

CLARK, Charlie; GAZONI, Eric. openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files. readthedocs, 2022. Disponível em: readthedocs. Último acesso em: 15/06/2022.

DISTINGUISH matches in pyparsing. stackoverflow, 2015. Disponível em: stackoverflow. Último acesso em: 12/01/2022.

FUZZYWUZZY. pypi, c2020. Disponível em: pypi. Último acesso em: 10/06/2022

GETTING a pure list out of 'pyparsing.ParseResults'. stackoverflow, 2012. Disponível em: stackoverflow. Último acesso em: 12/01/2022.

MCGUIRE, Paul. Module pyparsing. pythonhosted, 2017. Disponível em: pythonhosted. Último acesso em: 07/04/2022.

MCNAMARA, John. Working with Python Pandas and XlsxWriter. readthedocs, c2022. Disponível em: readthedocs. Último acesso em: 30/03/2022.

MCNAMARA, John. Creating Excel files with Python and XlsxWriter. readthedocs, c2022. Disponível em: readthedocs. Último acesso em: 28/03/2022.

PANDAS Tutorial. w3schools, c2022. Disponível em: w3schools. Último acesso em: 28/03/2022.

PYPARSING documentation. readthedocs, 2022. Disponível em: readthedocs. Último acesso em: 15/05/2022.

PYPARSING: bibliographic citations. stackoverflow, 2017. Disponível em: stackoverflow. Último acesso em: 12/01/2022.

PYPARSING: What does Combine() do?. stackoverflow, 2010. Disponível em: stackoverflow. Último acesso em: 07/04/2022.

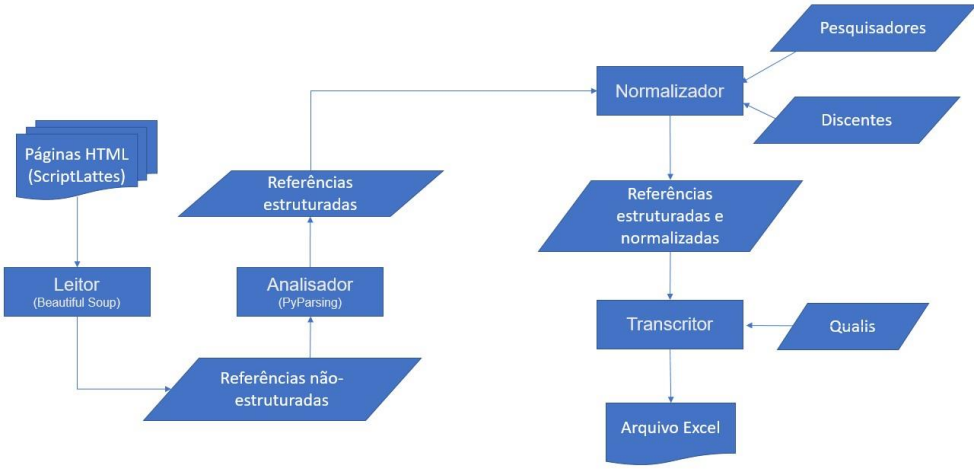
PYTHON - pyparsing unicode characters. stackoverflow, 2010. Disponível em: stackoverflow. Último acesso em: 07/04/2022.

REQUESTS: HTTP for Humans. readthedocs, 2022. Disponível em: readthedocs. Último acesso em: 02/03/2022.

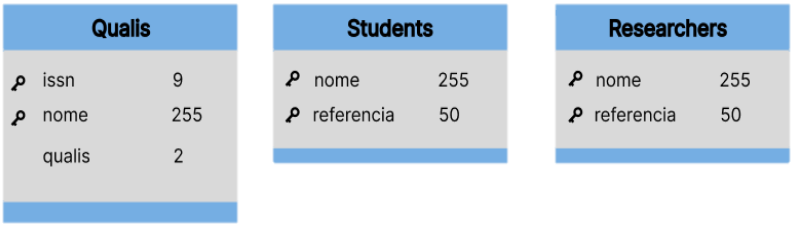
STRICKLER, John. Text Parsing Tools. accelebrate, 2015. Disponível em: accelebrate. Último acesso em: 12/01/2022.

Anexos

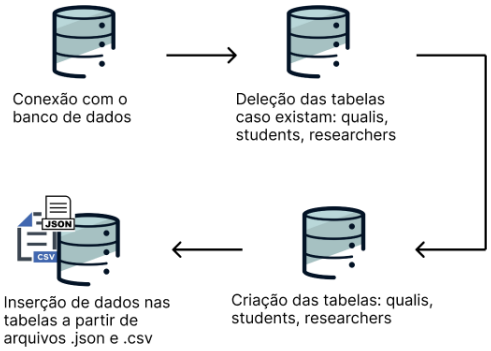
Anexo 01 – Fluxograma pondoc



Anexo 02 – Diagrama Banco de dados



Anexo 03 – Sequência de chamadas dos scripts SQL



Anexo 04 – Transferência de dados