

**PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO
CIENTÍFICA E TECNOLÓGICA**

UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE

DARCY RIBEIRO

Centro CCT

Labotatório LCMAT

Relatório do período: Fevereiro 2022 - Janeiro 2023

Relatório Anual PIBIC-10

Bolsista: Daniel Brito dos Santos

Matricula: 00119110393

Orientadora: Prof. Dra. Annabell Del Real Tamariz

Curso: Bacharelado em Ciência da Computação

Título do Projeto: Project-driven *Data Science*: Aprendendo e Mapeando

Título do Plano de Trabalho: Aplicação Exploratória da *Data Science* em um Projeto
Real de Dados

Fonte financiadora: PIBIC/UENF

1 Etapas propostas no plano de trabalho

O plano de trabalho desse último ano (2022) consistiu na definição e execução de um projeto real de dados relevante, intercalado com o estudo de três dos principais fundamentos da ciência de dados: banco de dados, modelos preditivos e visualização de dados.

- A - Definição do Projeto de Dados
- B - Execução do Projeto de Dados
- C - Banco de Dados
- D - Modelos Preditivos
- E - Visualização de Dados
- F - Elaboração do Relatório Final

Concluimos com sucesso os primeiros três itens com a definição do projeto intitulado “AcaDem” relacionado a informações acadêmicas dos estudantes do curso de bacharelado em Ciência da Computação da própria UENF, a sua execução incluindo: definição da arquitetura, desenvolvimento computacional de software, modelagem e construção de banco de dados.

Entretanto, a demanda de tempo necessário para tal desenvolvimento estendeu-se mais do que o esperado. De modo que não foi possível o aprofundamento desejado nas áreas de modelos preditivos (D) e visualização de dados (E). Além disso, devido a concentração do escopo, e aos dados disponíveis para o desenvolvimento do software, apenas visualizações simples e parciais puderam ser constituídas.

Desse modo, também não foi possível aplicar um modelo preditivo no “AcaDem”, porque os dados nele trabalhados são sensíveis, portanto, o desenvolvimento e análise se deu a partir de um subconjunto de extratos fornecidos voluntariamente por colegas do curso em questão. Para realizar uma melhor análise dos dados, seria necessário um conjunto muito maior de dados para que algum modelo preditivo pudesse ser aplicado. Não obstante, o estudo dos modelos preditivos se mostrou extremamente promissor para um desenvolvimento futuro que será apresentado no Capítulo 7.

Assim, os itens D e E foram parcialmente cumpridos. Nas seções seguintes serão detalhados todos os itens trabalhados neste relatório.

2 Introdução

A definição de oferta e organização de disciplinas é um dos grandes desafios dos coordenadores de cursos na nossa universidade. Acreditamos que parte fundamental desse problema é a inexistência de dados que facilitem esse processo decisório.

Nesse sentido, o projeto de dados selecionado nesse ano, intitulado “AcaDem” consistiu em construir um software que dê ao coordenador do curso rápido acesso a demanda por cada matéria da grade que seus alunos precisam cursar. Para isso, inicialmente é necessário apenas os dados que todos os coordenadores têm, sejam eles: o conjunto de extratos acadêmicos de seus alunos e a grade curricular de seu curso. O software também constrói um banco de dados local simples que estrutura e organiza todos os dados extraídos dos extratos acadêmicos. Chamamos esse projeto de “AcaDem” em referência aos termos Demanda e Acadêmica.

Dessa forma, concentraremos o escopo deste projeto de pesquisa na funcionalidade de “agregar as demandas dos alunos”, uma vez que para realizá-la será necessário toda a modelagem e implementação de um software para 1- extrair os dados dos extratos escolares e 2- gerar um banco de dados com as informações extraídas. Também teremos o enfoque concentrado no curso de Ciência da Computação, uma vez que é o curso do bolsista e portanto tem maior facilidade de obter extratos voluntariamente cedidos pelos colegas, e *feedback* com professores e a própria coordenação do curso. Tais limitações de escopo são importantes para possibilitar o desenvolvimento do programa, e acreditamos que uma outra versão necessitaria de poucas modificações para ser generalizada aos outros cursos no caso de futuros desenvolvimentos e uso por parte destes outros coordenadores.

Assim, este projeto prático será a consolidação dos fundamentos teóricos construídos no ano anterior e ainda suscitará aprofundamentos práticos em tópicos importantes da ciência de dados. Além disso, o banco de dados gerado permite o desenvolvimento futuro de diversas outras funcionalidades. Exemplo disso seria o desenvolvimento de *dashboards* interativos que além do número de alunos que demanda cada disciplina, poderia apresentar outras informações como o número de alunos que demandam disciplinas de horários conflitantes, ou o histograma do ano de matrícula dos alunos interessados em determinada matéria. Neles o coordenador poderia ter um conjunto de informações que garantiriam que suas decisões pudessem ser tomadas da melhor maneira, informadas por dados concretos, claramente visualizados. As possibilidades crescem exponencialmente, por essa razão acreditamos que o primeiro passo fundamental na construção de uma cultura baseada em dados é criar a infraestrutura para extração desses dados a partir do extrato escolar de cada aluno e construir um banco de dados já tratados e prontos para futuros desenvolvimentos.

3 Objetivos

O objetivo do plano de trabalho desse ano de 2022 de pesquisa foi estruturado da seguinte maneira:

1. Identificar um problema real e relevante no qual pudéssemos definir e estruturar um Projeto de Dados para aborda-lo. De modo que possamos aplicar os conceitos estudados no primeiro ano da bolsa de IC, bem como direcionar os aprofundamentos deste segundo ano.
2. Executar o projeto definido de acordo com as diretrizes estudadas no ano anterior.
3. Estudar e aplicar no projeto os principais conceitos e ferramentas da área de banco de dados no contexto da obtenção de dados, especialmente na estruturação e consulta de bancos de dados relacional com a linguagem SQL.

Na prática, ele se traduziu em consolidar e aprofundar o conhecimento acumulado no ano anterior enquanto desenvolvemos um projeto relevante que esperamos auxiliar os coordenadores de curso de graduação e os alunos de nossa universidade.

4 Metodologia

Selecionamos o problema por meio da observação das necessidades ao nosso redor, conversa com as partes interessadas e ponderação quanto a viabilidade e impacto positivo de uma solução de dados para o problema. Desse modo percebemos que o AcaDem é factível, e pode ter impactos positivos tanto para os coordenadores quanto para os alunos.

Após a seleção do projeto, “AcaDem”, ele é executado de acordo com as diretrizes construídas no ano anterior da pesquisa. Ou seja, utilizamos o *Data Science Work Model* (DSWM) (CRISAN; FIORE-GARTLAND; TORY, 2021) para orientar a prática do projeto. Na seção 4.1 detalharemos esse modelo de trabalho e como aplicamos cada uma de suas etapas. Já na seção 4.2 abordaremos a modelagem e implementação do software do projeto, falaremos sobre nossas escolhas técnicas, tipos de dados e detalharemos o funcionamento das principais funções do programa.

4.1 Modelo de Trabalho em Ciência de Dados (DSWM)

O modelo de trabalho *Data Science Work Model* desenvolvido por Crisan, Fiore-Gartland e Tory (2021) sintetiza a literatura científica, pesquisas de mercado e entrevista com diversos profissionais desse campo. Ele estrutura a prática de ciência de dados nos seguintes quatro (4) macro processos constituídos de 14 subprocessos no total, como se mostra na sequência:

- **Preparação:** Definição de necessidades, Obtenção de dados, Criação e Perfilamento de dados, e Manipulação de Dados
- **Análise de dados:** Experimentação, Exploração, Modelagem, Verificação, e Interpretação.
- **Instalação:** Monitoramento e Refinamento
- **Comunicação:** Disseminação e Documentação

Apresentamos na Figura 1 uma visualização desse modelo (DSWM).

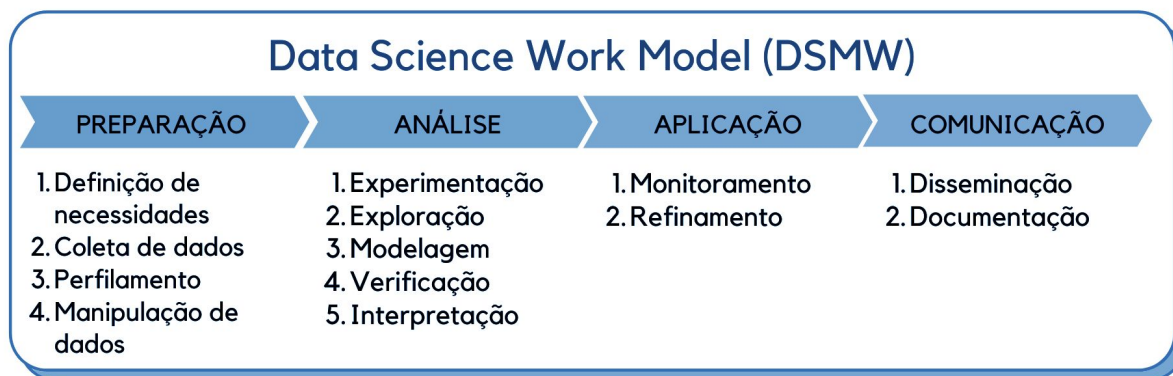


Figura 1 – Modelo de trabalho DSWM

No desenvolvimento do nosso projeto, a aplicação dessa estrutura se deu da seguinte forma:

4.1.1 Preparação

O processo de preparação em um projeto de dados diz respeito as suas primeiras atividades, desde a definição das necessidades até a transformação dos dados em um formato adequado aos objetivos do projeto. Em nosso projeto, esse macro processo foi nossa principal atividade, uma vez que o escopo era obter os dados de demanda, ou seja, tais dados deveriam ser extraídos e posteriormente processados. À seguir especificamos as sub atividades desse processo.

4.1.1.1 Definição de necessidades

O primeiro passo em qualquer projeto de dados é definir o que se espera dele. Segundo o modelo, a definição do problema de dados é o processo de traduzir objetivos analíticos, geralmente definidos pelas partes interessadas, em um conjunto de requerimentos viáveis.

Dentre tais requerimentos estão:

- Dados necessários
- Planos para analisá-los
- Definir quais serão as entregas, como por exemplo: relatórios, modelos e infraestrutura computacional.

No nosso caso, compreendemos que a necessidade das partes interessadas é obter o número de estudantes que demandam cada disciplina oferecida pelo curso. Essa necessidade implica nas seguintes necessidades do projeto: entregar um programa que gera tais informações. Obter dados dos quais podemos extrair tais informações. Processar os dados extraídos para gerar tais informações.

A principal fonte de dados que os coordenadores têm disponível para ser processada são os extratos acadêmicos dos estudantes. Esse extrato (um por estudante por período) é gerado automaticamente pelo Sistema Acadêmico da UENF. No extrato acadêmico aparecem todas as informações de cada estudante, como nome, matrícula, disciplinas cursadas e suas respectivas notas, período concluído, dentre outras. Também sabemos que todos os coordenadores têm as grades curriculares de seus respectivos cursos onde estão registradas as disciplinas a serem cursadas, assim como os pré requisitos de cada uma delas. A partir desses dados seria possível determinar para cada aluno quais disciplinas ainda não concluídas estão presentes na grade de seu curso e dentre elas, quais ele cumpre os pré-requisitos. Ou seja, pode-se obter uma lista com as disciplinas demandadas por cada aluno por cada período aberto. Finalmente, podemos combinar tais listas para gerar uma tabela final com o número de alunos que demanda cada disciplina.

Portanto, as necessidades do projeto são: extratos acadêmicos dos estudantes e a respectiva grade curricular de seu curso. Entregar uma tabela final com o código de cada disciplina e sua respectiva demanda. Implementar um software que efetue ambos. Logo, as entregas são o próprio software, uma tabela com a demanda por cada disciplina, e nesse processo podemos construir um banco de dados local com os dados extraídos.

4.1.1.2 Obtenção de dados

Essa etapa envolve coletar, preparar e armazenar os dados que serão usados em um projeto de ciência de dados. Pode envolver a extração de dados de diferentes fontes, como bancos de dados, arquivos, APIs ou web scraping. Os dados podem ser coletados de fontes internas, como sistemas de gerenciamento de banco de dados da empresa, ou de fontes externas, como arquivos abertos ou bancos de dados públicos. Por fim, os dados (coletados e/ou extraídos) são armazenados em um lugar acessível para uso posterior da aplicação desenvolvida, como um banco de dados ou um arquivo de dados. Isso permite que os dados sejam acessados facilmente para análise e modelagem posterior.

Como em nosso projeto as fontes de dados são os extratos acadêmicos dos alunos e a grade curricular do curso, essa etapa envolve processá-las para extrair os dados de interesse.

Vale esclarecer, que por ser um projeto de pesquisa que tenta mexer com informações sensíveis dos alunos e para não violar a Lei Geral de Proteção de Dados Pessoais (LGPD), Lei nº 13.709/2018, não foi solicitado nenhum acesso ao sistema acadêmico para obter o extrato dos alunos e a metodologia para contornar este problema foi de pedir aos colegas do curso que disponibilizaram seus extratos, voluntariamente e o compromisso de não divulgar nenhuma informação sensível.

A obtenção dos dados foi a etapa mais duradoura e trabalhosa, pois envolveu toda a modelagem e implementação do software necessário para processar todos os extratos,

construir o banco de dados e determinar as demandas de cada aluno por cada matéria, conforme detalhamos na seção 4.2.

4.1.1.3 Criação e Perfilamento de dados

A criação de dados diz respeito a geração de dados sintéticos para aumentar determinado conjunto de dados, geralmente tendo em vista alguma análise numérica ou simulação.

O perfilamento de dados, por outro lado, é o processo de realizar uma análise estatística dos dados de modo a obter informações gerais sobre suas características e estrutura. Esse processo é útil para entender o conjunto de dados e identificar possíveis problemas ou tendências. Ele pode ajudar a identificar valores ausentes, valores extremos, *outliers* ou valores duplicados nos dados. Além disso, o perfilamento de dados pode ajudar a entender a distribuição dos dados, bem como a relação entre diferentes variáveis.

Nenhuma das duas etapas se aplicam ao nosso projeto visto que não buscamos uma análise numérica, nem a modelagem de algum fenômeno. Reforçando a observação de Crisan, Fiore-Gartland e Tory (2021) que cada projeto aplica o DSMW de forma dinâmica, de modo que muitas vezes diversos dos seus 14 subprocessos não são necessários.

4.1.1.4 Manipulação de dados

A Manipulação de dados é a etapa final do pré-processamento. Nela os dados são transformados, combinados, reestruturados, linhas faltantes e duplicatas são tratadas, unidades convertidas, colunas transformadas, dentre outras atividades.

Em nosso projeto, como nós mesmos extraímos os dados, foi possível já o fazermos no formato que julgamos mais adequado ao nosso objetivo, conforme detalharemos na seção 4.2. A única manipulação se deu para lidar com exceções no processo. Um exemplo de exceção que foi tratado em nosso projeto é quando um aluno do curso de Computação cursa a disciplina “Introdução a Probabilidade e Estatística” oferecida por outro laboratório com código PROD12049 ao invés de ter cursado a disciplina presente na grade do curso com código MAT01201. Nesse caso, permitimos a utilização do arquivo disciplinas_equivalentes.csv, que é uma tabela com duas colunas onde cada linha define um par de disciplinas equivalentes no formato: (disciplina_externa, disciplina_grade). Assim, esse arquivo é carregado no código e nosso algoritmo considera a sigla presente na grade para efetuar os cálculos de disciplinas cursadas e por consequente disciplinas demandadas.

Assim, a principal manipulação de dados foi a própria leitura e processamento dos pdfs. O programa extrai os textos dos pdf, em seguida extrai os dados do texto extraído, para depois estruturar e transformar esses dados de modo que podemos calcular quais disciplinas cada aluno demanda, e finalmente calcular quantos alunos demandam cada disciplina. Logo em seguida manipulamos novamente os dados para construir e apresentar a tabela de resultados bem como construir e preencher o banco de dados local com todos os dados extraídos. Cada uma dessas operações é detalhada na seção 4.2

4.1.2 Analysis

Esta etapa diz respeito à exploração, verificação e experimentação dos dados adquiridos e transformados na etapa anterior. No caso específico do nosso projeto “AcaDem” esse conjunto de atividade englobaria a exploração dos dados coletados de modo a encontrar padrões, relações e características interessantes. Tal conhecimento seria fruto de visualizações e modelagens realizadas nos dados. Isto é, criação de diversos gráficos e modelos preditivos.

Entretanto, a sensibilidade dos dados impede que tais atividades sejam realizadas. Seria interessante em um desenvolvimento futuro implementar um módulo de anonimização dos dados que permitisse até disponibilizá-los em público, o que certamente traria conhecimento, e descobertas relevantes aos interesses da universidade. Infelizmente tal atividade foge do escopo dessa pesquisa, pois lidar com dados sensíveis e a sua dessensibilização é uma área de estudo própria (DOMINGO-FERRER; SÁNCHEZ; SORIA-COMAS, 2016) que envolve expertise e responsabilidade ímpar.

Assim, essa etapa foi substituída por outras duas atividades:

4.1.2.1 Visualização de dados

Para a visualização de dados, estamos estruturando um material didático de Introdução a Ciência de Dados ¹ (BRITO, 2022b) firmemente baseado no livro Wickham e Grolemund (2016) que enfatiza a visualização como ferramenta fundamental para todo um projeto de dados. Nesse material criamos diversas visualizações, além de utilizar os conceitos e ferramentas de comunicação indicadas por ele como a ferramenta Quarto ² para criar apresentações e documentos interativos. Além do enfoque na visualização, também abordamos virtualmente todas as outras atividades em um projeto de dados como a coleta, definição de perguntas e transformação de dados. A principal área não abordada nesse material é a parte de modelagem de dados e aprendizado de máquina, pois segundo o autor é uma área própria de conhecimento que deve ser estudada em um escopo dedicado.

4.1.2.2 Modelagem de dados

Conforme dito na subseção 4.1.2.1, Wickham e Grolemund (2016) considera que a modelagem de dados é parte fundamental no ferramental de um cientista de dados, porém o seu estudo sugere dedicação e materiais à parte da metodologia tradicional da ciência de dados. Isto porque, modelos preditivos são um assunto da disciplina de Aprendizado de Máquina, uma área do conhecimento própria que se ocupa das técnicas e algoritmos que permitem que máquinas aprendam com dados, sem serem explicitamente programadas. (DONOHO, 2017a)

¹ <https://github.com/dbs-97/introds>

² <https://quarto.org/>

Nesse sentido, compreendemos que o próximo desenvolvimento dessa pesquisa deve ser dedicado a esta área do conhecimento. De modo que o bolsista iniciou seus estudos nesta área de aprendizado de máquina. Nesse processo o bolsista entrou em contato e se interessou muito pela linha temática de Inferência Ativa (FRISTON; SAMOTHRAKIS; MONTAGUE, 2012), nela pretende dedicar os próximos passos de sua pesquisa. Conforme detalhamos na Capítulo 7.

4.1.3 Deployment

A instalação (*deployment*) de uma solução de dados diz respeito ao processo de tornar o modelo construído acessível ao usuário final em seu ambiente “real”, também conhecido como ambiente de “produção”. No caso do “AcaDem” significa disponibilizar o software de modo que coordenadores possam instalar e utilizar o software para analisar a demanda dos alunos de seus cursos.

Nesse sentido, utilizamos o modelo de distribuição padrão de pacotes Python (PYPA, 2023), permitindo que o usuário utilize a biblioteca pip (PIP..., 2022) para instalar o AcaDem em sua máquina e executá-lo a partir das instruções disponíveis em seu repositório *github*³ (BRITO, 2022a), conforme apresentamos na seção 5.1.

4.1.4 Communication

A comunicação é uma das atividades mais importantes de projeto de dados. Ela envolve um conjunto de processos que é essencial para a circulação de artefatos e conhecimento. Podemos dividir essas atividades em dois sub-processos principais:

4.1.4.1 Dissemination

Disseminação envolve criar e circular o aprendizado dos processos de ciência de dados, geralmente na forma de relatórios, apresentações e *dashboards*. Eles costumam resumir o trabalho e apresentar os achados.

4.1.4.2 Documentation

Documentação é o processo de registrar, descrever e sintetizar o trabalho realizado e seus artefatos. Normalmente inclui dados, código, fluxo de trabalhos e modelos. Um artefato muito comum nessa área são os Jupyter notebooks (KLUYVER; RAGAN-KELLEY; PÉREZ, 2016), equivalentes digitais a cadernos de laboratório, compostos por células que podem conter anotações, dados e códigos.

³ https://github.com/dbs-97/demanda_academica

4.1.4.3 AcaDem

Em nosso projeto a comunicação se dá por diversos canais. Este documento e as apresentações no CONFLICT são exemplos de comunicação do projeto. Além da participação acadêmica, temos os repositórios públicos com a documentação de toda essa pesquisa ⁴ (BRITO, 2021), a documentação e código fonte do AcaDem (BRITO, 2022a), e ainda o material didático que temos desenvolvido de Introdução a Ciência de Dados (BRITO, 2022b).

4.2 Implementação

Conforme detalhamos na seção 4.1, o projeto AcaDem envolve primordialmente o processamento dos dados presentes nos extratos acadêmicos dos alunos do curso de Ciência da Computação (CC) da UENF. Esses extratos são arquivos no formato pdf (ISO Central Secretary, 2020) gerados pelo sistema acadêmico da universidade, acessíveis aos coordenadores dos cursos. A partir desses extratos e da grade curricular queremos extrair o número de alunos que demanda cada disciplina, bem como construir um banco de dados local com todos os dados extraídos.

Gostaria de observar que as disciplinas inseridas na grade curricular do curso de CC como “Análise”, “Projeto e Desenvolvimento de Software” foram primordiais para a estruturação e execução desse projeto. Nesta seção em particular, utilizaremos muitos dos conceitos aprendidos nelas para representar e estruturar o nosso programa/software.

Dessa forma, apresentaremos diagramas, estruturas de pastas, exemplos de arquivos e bibliotecas utilizadas. Nesta seção apresentamos a modelagem e implementação do software necessário para tal processamento. Começando pelo levantamento de requisitos, como se apresenta na próxima seção.

4.2.1 Requisitos

A definição de engenharia de requisitos segundo Kotonya e Sommerville (1998) é o processo de determinar, analisar, especificar e validar os requisitos necessários para um sistema ou produto funcionar como previsto. Isso inclui identificar e documentar os requisitos funcionais e não funcionais, bem como as restrições e objetivos do sistema ou produto. Essa etapa é crítica no processo de desenvolvimento de um sistema pois ajuda a garantir que o produto final atenda às necessidades das partes interessadas e seja adequado ao seu propósito previsto.

Em nosso caso, por se tratar de um projeto de iniciação científica cujo o enfoque é na ciência de dados, especialmente coleta e extração dos dados, essa etapa da engenharia de software foi executada de modo menos formal, conforme as necessidades do projeto.

⁴ https://github.com/dbs-97/ds_fundamentals_research

Nesse caso, os principais requisitos funcionais do AcaDem envolvem suas entradas e saídas. Ele deve ser capaz de ler os extratos dos alunos no formato pdf e a grade de seu respectivo curso. Já como saída o sistema deve construir um banco de dados local contendo todos os dados extraídos bem como um arquivo tabular com os resultados de demanda, ou seja, uma tabela apresentando cada disciplina e o seu respectivo número de alunos demandantes.

Considerando ainda o caráter pioneiro e experimental do projeto, especialmente frente a complexidade da manipulação de dados, permitimos que os requisitos mantivemos o escopo dos requisitos restritos apenas aos essenciais ao seu funcionamento e adequação ao objetivo. De modo que todas as outras características puderam ser emergentes das nossas restrições.

4.2.2 Decisões de arquitetura

Considerando os requisitos e o contexto do sistema, selecionamos a linguagem Python (VAN ROSSUM GUIDO AND DRAKE, 2009) para sua implementação. Tanto por ser considerada atualmente a *lingua franca* da ciência de dados (Stack Overflow, 2020) e portanto ser a linguagem utilizada nessa pesquisa até então, quanto também por suas características. Dentre elas podemos ressaltar a legibilidade e amplo suporte de sua comunidade no que diz respeito a disponibilidade e qualidade de bibliotecas para os mais diversos fins e vasto material de referência em artigos e fóruns além das documentações oficiais da linguagem e cada biblioteca.

Além da linguagem, definimos utilizar o formato csv (SHAFRANOVICH, 2005) para arquivo de resultados e o arquivo com a grade curricular do curso em função de sua simplicidade e legibilidade. Logo, para efetuar tal manipulação utilizamos a biblioteca “csv” nativa de Python. Já para a leitura dos extratos em pdf utilizamos a biblioteca pyxpdf.

Além dessas, Python possui outras bibliotecas populares para tratamento de dados, como NumPy e Pandas, que poderiam ser úteis para o projeto. Entretanto, restringimos o uso de bibliotecas ao mínimo, tendo em vista facilitar a sua adoção pelos coordenadores, pois dessa forma temos uma instalação mais simples, com menor demanda de armazenamento e memória principal durante a execução. Desse modo, com exceção da biblioteca de leitura de pdfs (pyxpdf) nós implementamos todas as operações de forma manual, utilizando as ferramentas, bibliotecas e estruturas de dados nativas da linguagem Python.

Finalmente, decidimos utilizar o banco de dados SQLite por diversos motivos. O principal motivo é a grande adequação que ele tem com o nosso programa. SQLite é um banco de dados desenvolvido para ser rápido e leve em recursos, sua onipresença a nível global é prova do sucesso em desempenhar tais tarefas. Além de ser código aberto, gratuito, e também dispor de material instrutivo facilmente acessível online. Sua principal limitação está em seu escopo reduzido para dados de menor volume, menor complexidade

estrutural e que não necessitam de alta disponibilidade. Esse é exatamente nosso caso: dados estruturalmente simples por serem tabulares, não hierárquicos e apenas textuais. E bem abaixo da capacidade máxima de 140 terabytes que o SQLite oferece. (HIPPI, 2000) (SQLITE.ORG, 2021)

4.2.3 Estrutura do sistema

Nesta sessão apresentamos a maneira como o sistema foi estruturado incluindo diagramas, apresentação de seus módulos, sua estrutura de pastas, comportamento e suas principais funções.

4.2.3.1 Diagrama de fluxo de dados

Esse diagrama é uma ferramenta muito útil para representar os principais processos de um programa enfatizando suas respectivas entradas e saídas. Dessa forma, a Figura 2 apresenta o diagrama de contexto do sistema, isto é, representamos todo o programa como um único processo com os dados que entram e saem a nível global.

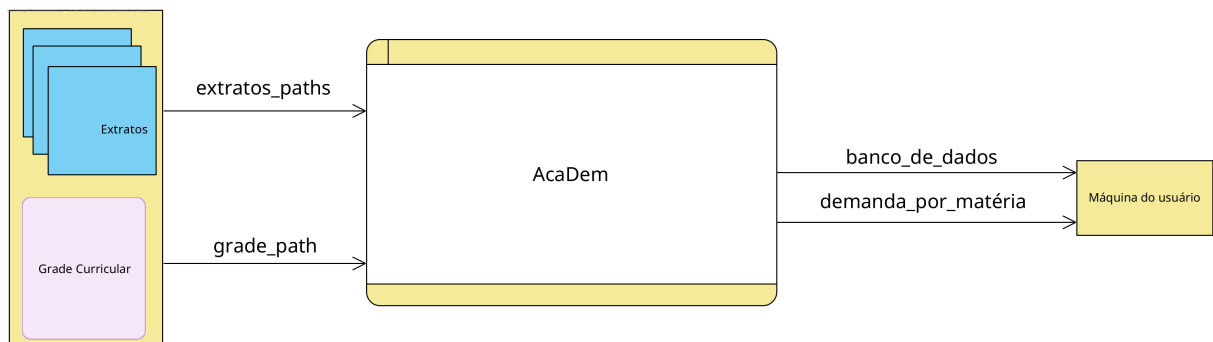


Figura 2 – Diagrama de Contexto

Já a Figura 3 detalha o processo principal do programa em seus subprocessos.

acrescenta uma camada de detalhes apresentando os módulos que compõem o AcaDem e os dados que fluem entre si.

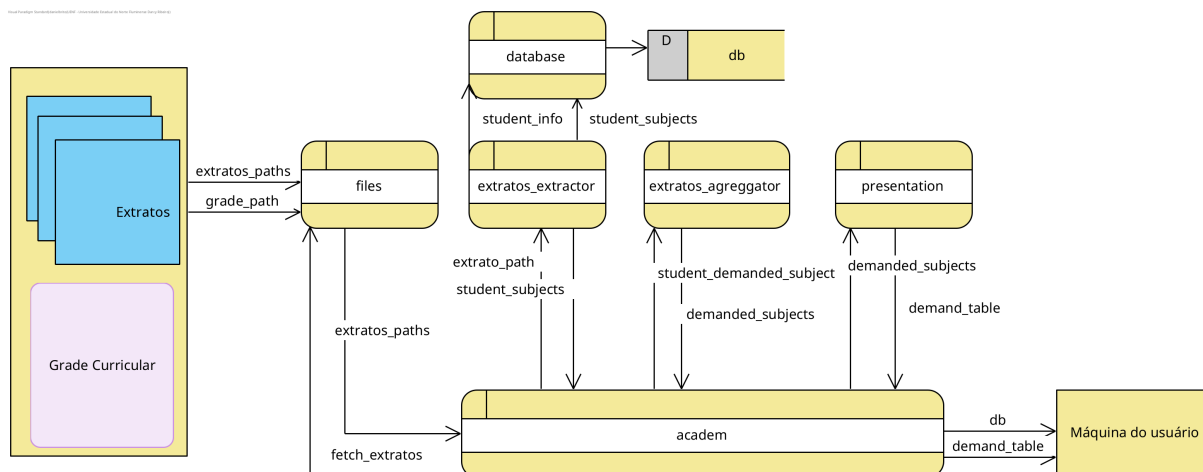


Figura 3 – Diagrama de Fluxo de Dados

Em ambas as figuras vemos que as entradas do sistema são os extratos e a grade curricular, enquanto suas saídas são o banco de dados dos dados extraídos e a demanda por cada disciplina. Mas na Figura 3 podemos observar os diversos módulos que compõem o sistema bem como os dados que são recebidos e retornado por cada um deles.

Listamos a seguir cada módulo de nossa implementação com uma breve explicação do papel que ele desempenha em nosso programa.

- **academ**: módulo principal do programa, ele se encarrega de orquestrar os processos internos para que o programa execute sua tarefa.
- **files**: módulo responsável por fornecer endereços dos arquivos a serem lidos, lê-los, escrevê-los, e direcionar o corretamente o arquivo necessário.
- **extratos_extractor**: módulo responsável por extrair os dados do pdf. Nele estão as funções necessárias para ler o pdf em uma string, encontrar, limpar e organizar os dados nessa string e formatá-los de modo propício a futuras manipulações.
- **extratos_agreggator**: módulo que se encarrega de agregar os dados dos extratos individuais. Nesse caso, sua principal função é a de consolidar as matérias demandadas por cada aluno nas demanda total alunos por matéria.
- **presentation**: módulo responsável por estruturar os dados de modo mais propício a apresentação. Principalmente na construção da tabela de resultado final.
- **database**: módulo responsável pela persistência dos dados. Ele constrói o banco de dados local, e nele armazena os dados extraídos dos extratos.

4.2.3.2 Banco de dados

Para desenvolver o banco de dados de nosso programa utilizamos muitos dos conceitos aprendidos nas disciplinas de Banco de Dados da universidade, bem como o livro Elmasri e Navathe (2017).

Construímos o modelo entidade-relacionamento apresentado na Figura 4 a partir da análise qualitativa dos dados presentes nos extratos acadêmicos dos alunos. Esse diagrama apresenta as entidades de um banco de dados, seus atributos, chaves primária e estrangeira, tipo de dado, além das relações entre as entidades e o tipo dessas relações.

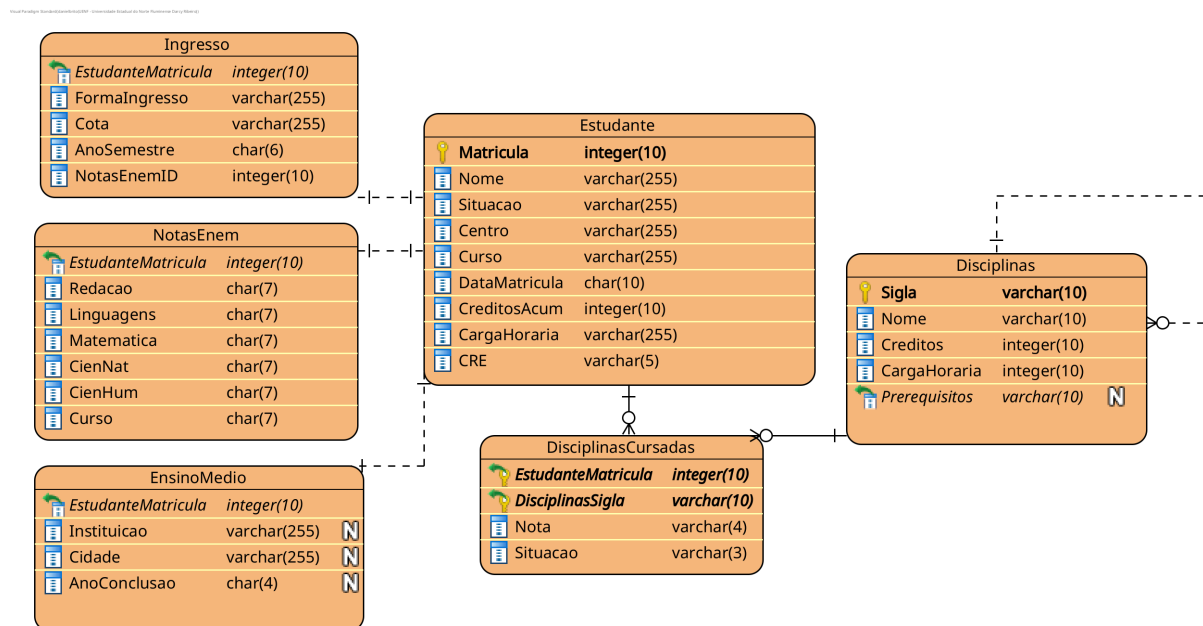


Figura 4 – Diagrama Entidade Relacionamento

Podemos observar no diagrama que nossos dados podem ser atribuídos às entidades Estudante, Ingresso, NotasEnem, EnsinoMedio, DisciplinasCursadas e Disciplinas. A seguir detalhamos o que cada uma representa.

- **Estudante:** reúne os dados do estudante com a matrícula como chave primária e informações como o nome do aluno, seu centro, situação no curso, data de matrícula, créditos acumulados e principalmente o seu coeficiente de rendimento efetivo (CRE)
- **Ingresso:** apresenta a matrícula do estudante como chave estrangeira e primária em uma relação com a entidade Estudante. Nos atributos dessa entidade encontramos a forma de ingresso, ano e cota do estudante.
- **NotasEnem:** entidade que abarca as notas que o estudante obteve no Exame Nacional do Ensino Médio, logo, também apresenta sua matrícula como chave primária e estrangeira.

- **EnsinoMedio:** tem como atributo a instituição, cidade e ano em que o estudante concluiu o ensino médio, também apresenta sua matrícula como chave primária e estrangeira.
- **DisciplinasCursadas:** contém os cada disciplina cursada por cada aluno e sua respectiva nota e situação. Como chave primária temos a composta das chaves estrangeiras Matrícula e Sigla, respectivamente referentes à Estudante e Disciplinas.
- **Disciplinas:** é a entidade responsável pelos dados das disciplinas, como seu nome, créditos, carga horária e pré-requisitos. Tem a Sigla da disciplina como chave primária, e o atributo Prerequisitos referencia a própria entidade de modo que os pré-requisitos de uma disciplina pode ser zero ou muitas disciplinas.

As entidades Ingresso, NotasEnem e EnsinoMedio apresentam uma relação do tipo “um para um” com Estudante. Ou seja, cada estudante têm exclusivamente uma tupla em cada uma dessas tabelas. Já em DisciplinasCursadas cada tupla representa uma disciplina cursada por um aluno, com a sua respectiva nota e situação, ou seja, temos uma relação “um-para-muitos” na qual cada Estudante tem diversas DisciplinasCursadas. Finalmente a entidade Disciplinas tem uma relação de um para um com o DisciplinasCursadas, mas uma relação um para muitos consigo mesma em seu atributo “Pre-requisitos”, ou seja, cada DisciplinaCursada está vinculada a exclusivamente uma Disciplina, que por sua vez pode ter muitas outras disciplinas como Pré-Requisito.

A partir desse modelo nós construímos um arquivo de banco de dados local, e o preenchemos com os dados extraídos e processados por nosso programa.

Conforme explicamos na subseção 4.2.2, selecionamos o sqlite como banco de dados de nossa aplicação por ser leve, acessível, performático e plenamente adequado para o nosso fim de obter persistência e organização relacional dos dados.

4.2.3.3 Estrutura de pastas

A estrutura de pastas e arquivos do AcaDem reflete a modelagem apresentada pelos diagramas de fluxo de dados na subseção 4.2.3.1. Podemos ver na Figura 5 que cada módulo do programa é representado por um arquivo python.

— README.md	<- Apresentação do programa
— academ	<- Pasta principal
— __init__.py	<- Arquivo de sinalização python
— academ.py	<- Módulo principal
— extratos_aggregate.py	<- Módulo de agregação da demanda
— extratos_extractor.py	<- Módulo de extração dos dados
— files.py	<- Módulo de manipulação de arquivos
— presentation.py	<- Módulo de apresentação visual
— config	<- Pasta de configuração
— disciplinas_do_curso.csv	<- Grade curricular do curso
— extratos_academicos	<- Pasta com os extratos acadêmicos
— extrato1.pdf	
— extrato2.pdf	
— ...	
— results	<- Pasta com os resultados
— academ.db	<- Banco de dados local
— demanda_disciplinas_22-05-02.csv	<- Tabela da demanda por disciplina
— requirements.txt	<- Pacotes necessários para instalação

Figura 5 – Estrutura de Pastas e Arquivos

Vemos ainda que os extratos escolares de cada aluno devem ser armazenados na pasta “extratos_academicos”, a grade curricular é representada pelo arquivo “disciplinas_do_curso.csv” conforme exemplificamos na Figura 6.

	A	B	C
1	Sigla	Nome	Prerequisitos
31	INF01115	Redes de Computadores	INF01204
32	INF01116	Banco de Dados I	INF01202
33	INF01117	Linguagens Formais e Teoria da Computação	INF01207, INF01113
34	LES04514	Metodologia do Trabalho Científico	
35	INF01119	Engenharia de Software	INF01201
36	INF01124	Introdução à Computação Gráfica	MAT01208, MAT01107, FIS01103
37	INF01212	Compiladores	INF01117
38	INF01206	Banco de Dados II	INF01116
39	INF01205	Inteligência Artificial	INF01202
40	INF01210	Paradigma OO para Desenvolvimento de Software	INF01203, INF01119
41	INF01211	Pesquisa Operacional	MAT01208
42	INF01211	Testes de Software	INF01210
43	INF01123	Interface Homem-Máquina	INF01205
44	LES04536	Computação e Sociedade	
45	INF01122	Sistemas Distribuídos	INF01115
46	PRO01540	Empreendedorismo	INF01211
47	INF01130	Projeto de Monografia	PRO01540, INF01124, INF01122, INF01212, INF01206, INF01123, INF01121
48	INF01131	Monografia	INF01130
49	INF01127	Estágio Supervisionado	INF01130

Figura 6 – Arquivo *disciplinas_do_curso.csv*

Finalmente, a tabela de resultados é construída na pasta “results” junto ao banco de dados local. A tabela tem o título de “demanda_disciplinas_{data em que foi construída}”, podemos ver um exemplo na Figura 7.

	A	B	C	
1	Sigla	Nome	Demanda (alunos)	
2	FIS01109	Laboratório de Física Gera III	4	
3	INF01201	Análise e Projeto de Sistemas	4	
4	INF01115	Redes de Computadores	4	
5	INF01116	Bancode Dados I	4	
6	INF01117	Linguagens Formais e Teoriada Computação	4	
7	LES04514	Metodologia do Trabalho Científico	4	
8	INF01205	Inteligência Artificial	4	
9	MAT01201	Estatística E Probabilidades	3	
10	PRO01540	Empreendedorismo	3	
11	FIS01103	Física Gera III	2	
12	LES04536	Computação e Sociedade	2	
13	MAT01105	Cálculo Diferencial e Integra IIII	1	
14	MAT01106	Método Matemático	1	
15	LEL04102	Inglês Instrumenta II	1	
16				

Figura 7 – Arquivo *demanda_disciplinas.csv*

4.2.3.4 Diagrama de sequência

A Figura 8 apresenta as principais etapas do comportamento do AcaDem. Nela podemos observar que a função *get_subjects_demand* (subseção 4.2.3.4.1) orquestra a execução das funções necessárias para **obter** os dados da grade curricular e dos extratos de cada aluno, **processar** esses dados para obter as demandas, **agregar** as demandas e **retornar** uma tabela com o código de cada disciplina e o número de alunos que a demandam. Em seguida detalhamos o funcionamento de cada uma dessas funções.

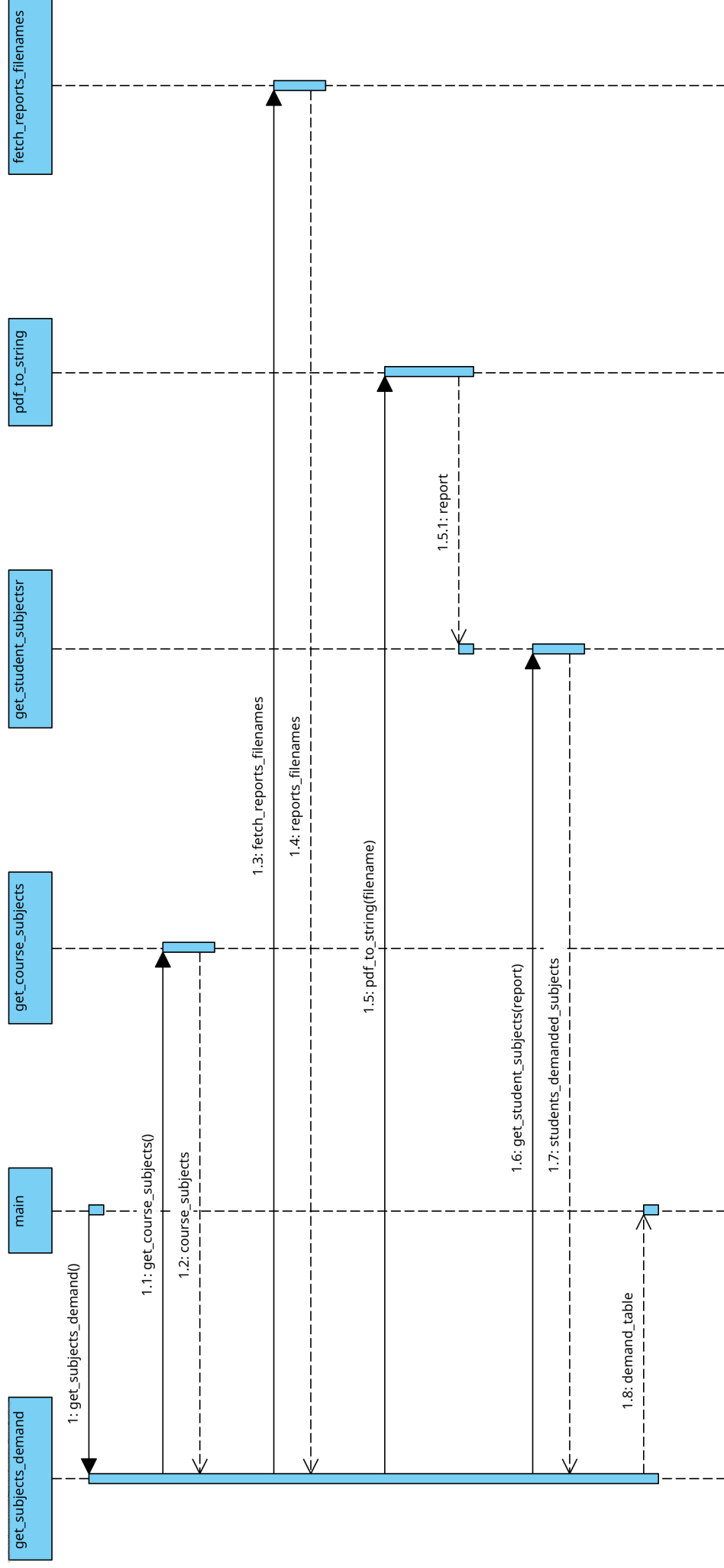


Figura 8 – Diagrama de sequência principal

4.2.3.4.1 **get_subjects_demand**

Função que orquestra a principal funcionalidade do programa: obter o número de estudantes que demanda cada disciplina. Apresentamos sua implementação no Código 1. Podemos observar a função executa as seguintes tarefas:

1. Armazena na variável *course_subjects* os resultados da função ***files.get_course_subjects*** (subseção 4.2.3.4.2). Essa função está presente no módulo "■*files*" e retorna o conteúdo da grade curricular: código, nome e pré-requisitos de cada disciplina do curso.

Armazena na variável *students_demanded_subjects* a agregação das demandas dos estudantes. Para tanto, utilizamos a técnica de "list_compreehension" da linguagem python (PYTHON, 2021). Essa técnica é considerada uma boa prática por ser mais idiomática e mais performática que um laço de repetição. Nesse caso, a expressão:

1. Percorre os nomes dos arquivos dos extratos acadêmicos que a função ***fetch_reports_filenames*** (subseção 4.2.3.4.3) retorna.
2. Para cada nome de arquivo a expressão constrói seu endereço e passa para a função ***ee.PDF_to_string*** (subseção 4.2.3.4.4) que retorna todo o texto do extrato acadêmico referente aquele endereço.
3. O texto do extrato acadêmico e o conteúdo da grade curricular são passados como parâmetros para a função ***ee.get_student_subjects*** (subseção 4.2.3.4.5) que retorna um dicionário com as disciplinas aprovadas e demandadas daquele aluno. Desse dicionário selecionamos a lista de disciplinas demandadas daquele aluno por meio da chave "[demanded]".
4. Finalmente, todo esse processo é repetido para cada nome de arquivo de extrato, gerando uma lista onde cada elemento é a lista das disciplinas demandadas por cada aluno, representado por cada extrato acadêmico. Essa lista aninhada é armazenada na variável *students_demanded_subjects*.

Armazena a demanda agregada dos alunos na variável "subject_demand_count". Essa demanda é calculada pela função ***ea.count_subjects_demand*** (subseção 4.2.3.4.6) a partir da variável *subject_demand_count*.

Finalmente, a tabela de demanda final é gerada pela função ***present.subjects_demand_to_table*** (subseção 4.2.3.4.7), a partir da contagem de alunos demandando cada disciplina, e dos dados da grade curricular.

```
1 def get_subjects_demand():
2     course_subjects = files.get_course_subjects()
3     students_demanded_subjects =
4         ↪ [ee.get_student_subjects(ee.PDF_to_string(f'extratos_academicos/{filename}'),
5         ↪     course_subjects)['demanded']]
6         for filename in files.fetch_reports_filenames()]
7     subject_demand_count = ea.Count_subjects_demand(students_demanded_subjects)
8     demand_table =
9     ↪ present.subjects_demand_to_table(subject_demand_count, course_subjects)
10
11     return demand_table
```

Código 1: Função `get_course_subjects`

4.2.3.4.2 `files.get_course_subjects`

Essa é a função responsável por obter os dados do arquivo “disciplinas_do_curso.csv” que representa a grade curricular do curso. Ela executa tal tarefa, conforme apresentamos no Código 2, da seguinte maneira:

1. A primeira sub-tarefa é a leitura do arquivo “disciplinas_do_curso.csv” encontrado na pasta “config”. Isto é feito pela função ***read_csv***, que lê cada linha do arquivo csv e retorna uma tupla (PYTHON..., 2022) em que cada elemento é uma linha da tabela com a sigla, nome e prerequisites de cada disciplina.
2. A função ***format_read_course_subjects*** transforma a tupla gerada no passo anterior em um dicionário com a sigla da disciplina como chave e seus respectivos nome e prerequisites como valores.
3. Finalmente, a função principal retorna o dicionário das disciplinas do curso.

```

1 import csv
2
3 def get_course_subjects():
4     return format_read_course_subjects(
5         read_csv('config/disciplinas_do_curso.csv'))
6
7 def format_read_course_subjects(course_subject_tuple):
8     return {subject_id: {'name': name, 'prerequisites':
9         ↪ _treat_prerequisite_read(prerequisites)}
10            for subject_id, name, prerequisites in course_subject_tuple[1:]}
11
12 def read_csv(filename):
13     with open(filename, mode='r') as csv_file:
14         csv_reader = csv.reader(csv_file, delimiter=';')
15         return tuple(csv_reader)
16
17 def _treat_prerequisite_read(prerequisite_str):
18     return [] if prerequisite_str == '' else prerequisite_str.replace(' ',
19     ↪ '').split(',')

```

Código 2: Função files.get_course_subjects

4.2.3.4.3 files.fetch_reports_filenames

Essa função retorna os nomes de todos os arquivos presentes na pasta “extratos_academicos”. O objetivo de coletar os nomes de cada arquivo referente ao extrato acadêmico de cada aluno do curso. Para isso, vemos no Código 3 que ela utiliza a função “os.listdir(dir)” da biblioteca “os” que está encapsulada na função “fetch_filenames”.

```

1 import os
2
3 def fetch_reports_filenames():
4     reports_names = fetch_filenames('extratos_academicos')
5     return reports_names
6
7 def fetch_filenames(dir):
8     return os.listdir(dir)

```

Código 3: Função fetch_reports_filenames

4.2.3.4.4 ee.PDF_to_string

Essa é a função responsável por ler o arquivo pdf do extrato acadêmico recebido como argumento “doc_path” e retornar uma string com todo o seu conteúdo. Para tanto, essa função utiliza a biblioteca “pypdf”. Podemos observar no Código 4 que utilizamos a função “Document” para ler o pdf, e as funções “TextControl” e “TextOutput” da biblioteca para configurar essa leitura, e finalmente transformar o conteúdo em string com a “Page_iterator”.

```
1 from pyxpdf import Document
2 from pyxpdf.xpdf import TextOutput, TextControl, page_iterator
3
4 def PDF_to_string(doc_path):
5     doc = Document(doc_path)
6     control = TextControl(mode = "table")
7     text_out = TextOutput(doc, control)
8     return "\n".join(page_iterator(text_out))
```

Código 4: Função `ee.PDF_to_string`

4.2.3.4.5 `ee.get_student_subjects`

Esta é uma das funções mais importantes do AcaDem. Ela é a responsável por construir o dicionário com todas as disciplinas cursadas, aprovadas e demandadas de determinado estudante.

Seus argumentos são:

- `report`: a string do extrato acadêmico do estudante extraída pela função “`ee.PDF_to_string`” subseção 4.2.3.4.4
- `course_subjects`: dicionário com as disciplinas do curso e seus prerrequisitos, gerado pela função “`get_course_subjects`” subseção 4.2.3.4.2
- `subjects_equivalences`: esse argumento opcional é um dicionário de equivalências de disciplinas que permite que disciplinas fora da grade sejam consideradas com a sigla equivalente da grade. Por exemplo o dicionário {“PROD12049”: “MAT01201”} indica que o estudante cursou Introdução a Estatística e Probabilidade com a sigla PROD12049, mas essa disciplina equivale a MAT01201 presente em sua grade.

Podemos observar no Código 5 que essa função atua da seguinte forma:

1. Constrói “`taken_subjects`” com um par de funções compostas cuja primeira vai localizar na string do extrato acadêmico todas as linhas com informações de disciplina utilizando expressões regulares, as linhas devolvidas são recebidas pela função que percorre cada uma delas reorganizando os dados em um dicionário com sigla e suas respectivas informações.
2. Constrói “`approved_subjects`” filtrando o “`taken_subjects`” por todas as disciplinas com situação “APR” ou “CVD”
3. Constrói “`demandated_subjects`” que localiza todas as disciplinas presentes na grade das quais o estudante ainda não cursou e dispõem dos prerrequisitos.
4. Finalmente, todos esses dados são armazenados em um dicionário que é retornado pela função.

```

1  import re
2
3  def get_student_subjects(report, course_subjects, subjects_equivalences={}):
4      taken_subjects = build_taken_subjects_dict(fetch_rows_with_subject_info(report))
5      approved_subjects = list_approved_subjects(taken_subjects, subjects_equivalences)
6      demanded_subjects = list_demanded_subjects(approved_subjects, course_subjects)
7      return {"taken": taken_subjects,
8              "approved": approved_subjects,
9              "demanded": demanded_subjects}
10
11 def build_taken_subjects_dict(subject_rows):
12     taken_subjects_dict = {subject[:8] : subject_to_dict(subject) for subject in
13                             ↪ subject_rows}
14     return taken_subjects_dict
15
16 def fetch_rows_with_subject_info(report):
17     exp = r"[A-Z]{3}\d{5}.*"
18     subject_rows = re.findall(exp, report, re.M)
19     return subject_rows
20
21 def list_approved_subjects(taken_subjects, subjects_equivalences={}):
22     approved_subjects = {subject for subject in taken_subjects.keys() if
23                           (taken_subjects[subject]["situation"] in ["APR", "CVD"])}
24     subjects_with_equivalences =
25         ↪ set.intersection(set(approved_subjects), set(subjects_equivalences.keys()))
26     equivalent_subjects = {subjects_equivalences['subject'] for subject in
27                             ↪ subjects_with_equivalences}
28
29     return set.union(approved_subjects, equivalent_subjects)
30
31 def list_demanded_subjects(approved_subjects, course_subjects):
32     demanded_subjects = [subject for subject in course_subjects if
33                           (subject not in approved_subjects
34                            and
35                            ↪ set(course_subjects[subject]['prerequisites']).issubset(set(approved_subjects)))]
36     return demanded_subjects
37
38 def subject_to_dict(subject_row):
39     name = subject_row[8:100].strip()
40     data = subject_row[100:].strip().split()
41     if len(data) == 2:
42         """Subjects taken within Covid special period"""
43         subject_dict =
44         ↪ {"name": name, "credit": "-", "workload": data[1], "grade": data[0], "situation": "CVD"}
45     else:
46         subject_dict = {"name": name,
47                           "credit": data[0],
48                           "workload": data[1]+data[2],
49                           "grade": data[3],
50                           "situation": data[4]}
51     return subject_dict

```

Código 5: Função ee.get_student_subjects

4.2.3.4.6 **ea.count_subjects_demand**

Essa é a função responsável por transformar as demandas individuais de cada aluno na demanda agregada de cada disciplina. Como podemos observar no Código 6, utilizamos as bibliotecas nativas de python “collections” e “itertools” para computar esses cálculos e retornar o número de alunos demandantes por cada disciplina.

```
1 from collections import Counter
2 from itertools import chain
3
4 def Count_subjects_demand(subject_demand_list):
5     return Counter(chain.from_iterable(subject_demand_list)).most_common()
```

Código 6: Função ea.count_subjects_demand

4.2.3.4.7 **present.subjects_demand_to_table**

Essa é a função responsável por reestruturar a tabela de demanda em um formato de apresentação mais adequado. Ela recebe a tabela de demandas e as disciplinas da grade para construir a tabela de resultados final do programa Código 7.

```
1 def subjects_demand_to_table(subjects_demand, course_subjects):
2     header = [('Sigla', 'Nome', 'Demanda (#alunos)')]
3     demand_table = [(subject_id, course_subjects[subject_id]['name'], subject_demand)
4                      for subject_id, subject_demand in subjects_demand]
5
6     return header + demand_table
7
8 def present_demand_table(demand_table):
9     for row in demand_table:
10         print(*row)
11     print()
```

Código 7: Função present.subjects_demand_to_table

5 Resultados

Os resultados desse ano de pesquisa são de diversas naturezas. O principal deles é o próprio AcaDem, um programa escrito em python com todo o seu código disponível em nosso repositório online ¹ (BRITO, 2022a). Neste seção apresentamos a execução do “AcaDem”, os testes de seu desempenho, alguns exemplos com resultados e sugestões de análises que seriam possíveis caso pudéssemos aplica-lo em um conjunto representativo de extratos dos estudantes. Lembrando que temos apenas quatro extratos voluntariamente cedido pelo bolsista e seus colegas para desenvolver esse projeto.

5.1 Instalação

Para instalar o AcaDem, basta seguir as instruções fornecidas em seu repositório online. O processo pode ser resumido nos seguintes passos:

1. Obter a linguagem Python, versão 3.7
2. Efetuar o download do pasta AcaDem
3. Executar o script `setup.py`
4. Pronto! Agora basta executar o script `academ/academ.py`, conforme apresentado na seção 5.2

5.2 Execução

Uma vez com o AcaDem instalado, basta abrir um terminal em sua pasta principal e executar o comando apresentado no Código 5.1

```
1 python academ/academ.py
```

Código 5.1 – Comando para executar o AcaDem

O próprio terminal logo apresentará a tabela de resultados, conforme a figura 9.

¹ https://github.com/dbs-97/demanda_academica

```
(base) [danielbrito@fedora demanda_academica]$ python3 academ/academ.py
Sigla Nome Demanda (#alunos)
FIS01109 Laboratóriode Física Geral II 4
INF01201 Análisee Projetode Sistemas 4
INF01115 Redesde Computadores 4
INF01116 Bancode Dados I 4
INF01117 Linguagens Formaise Teoriada Computação 4
LES04514 Metodologiado Trabalho Científico 4
INF01205 Inteligência Artificial 4
MAT01201 Estatística EProbabilidades 3
PRO01540 Empreendedorismo 3
FIS01103 Física Geral II 2
LES04536 Computaçãoe Sociedade 2
MAT01105 Cálculo Diferenciale Integral III 1
MAT01106 Método Matemático 1
LEL04102 Inglês Instrumental I 1
```

Figura 9 – Execução do AcaDem

Logo após o resultado apresentado diretamente no terminal, o programa cria um arquivo csv com a tabela apresentada na pasta “results” com o nome “demanda_disciplinas_{data-de-criação-do-arquivo}.csv”, conforme apresentamos na Figura 7.

Além do arquivo csv, o AcaDem também constrói o banco de dados local. Podemos observar na figura 10 que o banco de dados gerado está plenamente funcional, e organizado de acordo com o modelo que apresentamos na seção 4.2.3.2.

```
(base) [danielbrito@fedora demanda_academica]$ sqlite3 results/academ.db
SQLite version 3.37.0 2021-11-27 14:13:22
Enter ".help" for usage hints.
sqlite> .tables
Disciplinas          EnsinoMedio          Ingresso
DisciplinasCursadas  Estudante            NotasEnem
sqlite> SELECT * FROM DisciplinasCursadas;
119110351|INF01101|8,6|APR
119110351|INF01105|9,5|APR
119110351|INF01106|5,0|APR
119110351|MAT01101|6,4|APR
119110351|MAT01104|7,2|APR
119110351|MAT01117|5,3|APR
119110351|FIS01202|8,3|CVD
119110351|FIS01204|7,6|APR
119110351|INF01104|6,8|APR
119110351|INF01207|5,0|APR
```

Figura 10 – Consultando o banco de dados criado

5.3 Desempenho

Para avaliar o desempenho do nosso programa, o executamos com o sistema de logs (LOGGING..., 2022) registrando a hora exata na qual ocorreu cada operação principal. A partir dessa informação de horário é possível determinar o tempo dispendido pela diferença nos horários. A figura 11 apresenta o resultado dessa execução temporizada com os quatro extratos acadêmicos que temos disponíveis. Nela podemos perceber que as duas operações que mais demandam tempo são a leitura dos dados e a construção do banco de dados, que respectivamente duram 70 (19:34:23,208 - 19:34:23,138) e 297 (19:34:23,505 - 19:34:23,208) milissegundos.

```
(base) [danielbrito@fedora demanda_academica]$ time python academ/academ.py
2023-01-11 19:34:23,138 - Lendo os dados...

2023-01-11 19:34:23,208 - Construindo tabela de demandas...

2023-01-11 19:34:23,208 - Resultados:

Sigla Nome Demanda (#alunos)
FIS01109 Laboratóriode Física Geral II 4
INF01201 Análisee Projetode Sistemas 4
INF01115 Redesde Computadores 4
INF01116 Bancode Dados I 4
INF01117 Linguagens Formaise Teoriada Computação 4
LES04514 Metodologiado Trabalho Científico 4
INF01205 Inteligência Artificial 4
MAT01201 Estatística EProbabilidades 3
PRO01540 Empreendedorismo 3
FIS01103 Física Geral II 2
LES04536 Computaçãoe Sociedade 2
MAT01105 Cálculo Diferenciale Integral III 1
MAT01106 Método Matemático 1
LEL04102 Inglês Instrumental I 1

2023-01-11 19:34:23,208 - Salvando os resultados...
2023-01-11 19:34:23,208 - Arquivo 'results/demanda_disciplinas_2023-01-11.csv' salvo com sucesso!
2023-01-11 19:34:23,208 - Construindo o banco de dados...
2023-01-11 19:34:23,505 - Banco de dados construído com sucesso!
2023-01-11 19:34:23,505 - Script finalizado.
```

Figura 11 – Execução temporizada com quatro extratos

Para poder fazer uma melhor análise do código desenvolvido, sabendo que com o número de extratos que contamos não é possível fazer nenhuma previsão em quanto a eficiência do sistema; efetuamos dez cópias dos nossos quatro extratos, totalizando 40 extratos. Executamos o programa a partir desses 40, duplicamos os extratos e executamos o programa novamente. Repetimos esse processo quatro vezes para construir a tabela 1. Ela apresenta o tempo dispendido em milissegundos para ler os extratos e construir o banco de dados, de acordo com o número de extratos processados (4, 40, 80, 160, 320, 640 e 1280). Desse modo, a tabela nos permite inferir o desempenho aproximado do programa para conjuntos de dados de tamanhos diferentes.

# Extratos	Leitura Dados	Construção BD
4	70	297
40	729	1.291
80	1.562	3.791
160	2.947	5.930
320	6.027	11.083
640	10.213	22.503
1280	22.778	48.561

Tabela 1 – Tempo em milissegundos dispendido na leitura dos extratos e construção do banco de dados de acordo com o número de extratos

Podemos observar que a construção do banco de dados se mostra a operação mais demandante. Esse fato é compreensível se considerarmos que essa etapa requer percorrer todos os dados coletados, e levanta uma possível melhoria ao refatorar o programa de modo que ele preencha os dados enquanto os processa.

Assim, considerando a complexidade das operações efetuadas, acreditamos que o desempenho do AcaDem é satisfatório para o que ele se propõem a fazer.

5.4 Análise

Além do desempenho, levantamos algumas possíveis análises e visualizações que seriam possíveis caso tivéssemos acesso aos dados completos.

- Poderíamos comparar a “nota de curso” que o aluno obteve no Enem com o seu CRE na universidade. Seria interessante analisar se há alguma relação, e caso haja qual seria a sua natureza.
- Poderíamos utilizar um cálculo de Informação Mutua para encontrar quais dos dados dos estudantes apresentam maior correlação com o seu CRE, ou qual seria a variável mais relacionada ao número de reprovações em disciplinas.
- Poderíamos avaliar a variação de notas de cada aluno ao longo dos anos na universidade e ver se a sua tendência é positiva, negativa ou próxima de constante.
- Seria possível encontrar as disciplinas com notas médias mais destoantes das outras.
- Com o devido acesso e tempo de pesquisa seria talvez fosse possível sugerir quais matérias poderiam ser ofertadas para otimizar o número de alunos matriculados, permitindo a ponderação pela taxa de conclusão do curso, por exemplo. Ainda nesse caso seria possível analisar programaticamente os conflitos de horários e as características dos alunos que demandam cada grupo de disciplinas.

Estas são apenas algumas das possíveis análises caso os dados fossem acessíveis. Elas ressaltam a capacidade que a ciência de dados tem de gerar perguntas interessantes, e trazer conhecimento que pode ser relevante para as partes interessadas.

6 Conclusão

O objetivo desse segundo ano de pesquisa foi definir e executar um projeto de dados para abordar um problema real e relevante. De modo que pudéssemos aplicar os conceitos e diretrizes estudados no primeiro ano de iniciação científica e aprofundar nas áreas de banco de dados, visualização de dados e modelos preditivos.

Nesse sentido, o projeto escolhido, intitulado “AcaDem” foi pensado como resposta a uma problemática comum entre alunos e coordenadores, decorrente do desafio de estruturar a oferta de disciplinas de maneira satisfatória. Percebemos que o pináculo desse problema é a inexistência de ferramentas que informem aos coordenadores e chefes de laboratório exatamente qual é a demanda por cada disciplina em cada período letivo na UENF.

Desse modo, estruturamos um projeto de Ciência de Dados para abordar essa questão, de acordo com as diretrizes estudadas anteriormente e o executamos com sucesso. Ou seja, primeiramente identificamos um problema a ser abordado com um projeto de dados. Logo adiante, seguindo o modelo de trabalho DSWM (CRISAN; FIORE-GARTLAND; TORY, 2021) nós definimos as necessidades do projeto, coletamos os dados e os transformamos conforme nosso objetivo. Também desenvolvemos atividades de análise de dados, e comunicação. Revisitando nesse processo os fundamentos da ciência de dados, além de conceitos fundamentais do desenvolvimento de software. Nesse sentido, criamos um script personalizado para a extração de dados de arquivos pdfs, construção da tabela com as demandas dos alunos e do banco de dados com os dados extraídos. Em seguida realizamos uma sequência de testes para avaliar o desempenho e funcionamento do programa.

Concluimos que um projeto prático é uma excelente maneira de solidificar conteúdo teórico, tanto ao observar o que a teoria propõe funcionando, quanto quando nos deparamos com as limitações frente a complexidade de problemas reais.

Assim, entendemos que este segundo ano de pesquisa concluiu com sucesso os seus principais objetivos: executar um projeto real de ciência de dados. Podemos concluir que temos uma primeira versão do “AcaDem” que pode ser executada conforme explicitado no capítulo seção 5.1.

7 Perspectivas de continuidade

Tendo em vista o tempo necessário e a importância do assunto, considerando ainda que não foi possível completar essa parte da pesquisa nessa etapa, conforme explicado no Capítulo 1, propomos que a próxima etapa de pesquisa seja exclusivamente dedicada a área de modelos preditivos.

Além da relevância do tópico para a área de ciência de dados (DONOHO, 2017b; WICKHAM; GROLEMUND, 2016) ao iniciar os estudos nessa área o bolsista entrou em contato com o conceito de Inferência Ativa. Esse conceito teórico busca oferecer uma estrutura teórica para explicar e permitir a implementação de agentes que tomam decisões racionais. Segundo a Inferência Ativa, a ação e a racionalidade podem ser explicadas e reproduzidas justamente por modelos preditivos (FRISTON; SAMOTHRAKIS; MONTAGUE, 2012).

Essa perspectiva indica uma linha de pesquisa muito promissora como continuidade deste projeto de Iniciação Científica. Especialmente considerando os avanços recentes nesta área (HEINS et al., 2022). Assim, propomos continuar nossa pesquisa exploratória da ciência de dados, dessa vez explorando um tópico teórico no estado da arte da pesquisa em modelos preditivos. O plano de trabalho em anexo, explicitará os detalhes desta pesquisa.

8 Participação em congressos e trabalhos publicados ou submetidos e outras atividades acadêmicas




CERTIFICADO


Certificamos que

Daniel Brito dos Santos

participou como congressista no XIV Congresso Fluminense de Iniciação Científica e Tecnológica / VII Congresso Fluminense de Pós-Graduação, ocorrido virtualmente no período de 20 a 24 de junho de 2022.


Dra. Ana Maria Almeida da Costa
Diretora do ESR/UFF Campos


Dr. José Augusto Ferreira da Silva
Pró-Reitor de Pesquisa e Pós-Graduação
IFF


Dra. Maura Da Cunha
Pró-Reitora de Pesquisa e Pós-Graduação
UENF



CERTIFICADO

Certificamos que o trabalho intitulado:

Contornos da ciência de dados: uma meta-revisão conceitual


De autoria de

Daniel Brito dos Santos, Annabell D.r. Tamariz

foi apresentado por **Daniel Brito dos Santos** na categoria Iniciação Científica-Poster durante o XIV Congresso Fluminense de Iniciação Científica e Tecnológica / VII Congresso Fluminense de Pós-Graduação, ocorrido virtualmente no período de 20 a 24 de junho de 2022.


Dra. Ana Maria Almeida da Costa
Diretora do ESR/UFF Campos


Dr. José Augusto Ferreira da Silva
Pró-Reitor de Pesquisa e Pós-Graduação
IFF


Dra. Maura Da Cunha
Pró-Reitora de Pesquisa e Pós-Graduação
UENF






CERTIFICADO

Certificamos que **Daniel Brito dos Santos** participou do minicurso intitulado **"A relação entre forma e função nos seres vivos: o olhar natural e o olhar humano"** com carga horária de **4 horas** durante o **XIV Congresso Fluminense de Iniciação Científica e Tecnológica / VII Congresso Fluminense de Pós-Graduação**, ocorrido no período de 20 a 24 de junho de 2022.


Dra. Ana Maria Almeida da Costa
Diretora do ESR/UFF Campos


Dr. José Augusto Ferreira da Silva
Pró-Reitor de Pesquisa e Pós-Graduação
IFF


Dra. Maura Da Cunha
Pró-Reitora de Pesquisa e Pós-Graduação
UENF



9 Data e assinatura do bolsista (assinatura digitalizada)

Donel Brito dos Santos

23/01/2023

10 Data e assinatura do orientador (assinatura digitalizada)

Annabell D.R. Tamariz

26/01/2023

Referências

- BRITO, D. *Pesquisa sobre fundamentos da ciência de dados*. [S.l.]: GitHub, 2021. [⟨https://github.com/dbs-97/ds_fundamentals_research⟩](https://github.com/dbs-97/ds_fundamentals_research). 11
- BRITO, D. *AcaDem*. [S.l.]: GitHub, 2022. [⟨https://github.com/dbs-97/demanda_academica⟩](https://github.com/dbs-97/demanda_academica). 10, 11, 26
- BRITO, D. *Introdução à Ciência de Dados*. [S.l.]: GitHub, 2022. [⟨https://github.com/dbs-97/introds⟩](https://github.com/dbs-97/introds). 9, 11
- CRISAN, A.; FIORE-GARTLAND, B.; TORY, M. Passing the Data Baton : A Retrospective Analysis on Data Science Work and Workers. *IEEE Transactions on Visualization and Computer Graphics*, v. 27, n. 2, p. 1860–1870, fev. 2021. ISSN 1077-2626, 1941-0506, 2160-9306. Disponível em: [⟨https://ieeexplore.ieee.org/document/9222030/⟩](https://ieeexplore.ieee.org/document/9222030/). 5, 8, 31
- DOMINGO-FERRER, J.; SÁNCHEZ, D.; SORIA-COMAS, J. Database anonymization: privacy models, data utility, and microaggregation-based inter-model connections. *Synthesis Lectures on Information Security, Privacy, & Trust*, Morgan & Claypool Publishers, v. 8, n. 1, p. 1–136, 2016. 9
- DONOHO, D. 50 years of data science. *Journal of Computational and Graphical Statistics*, Taylor Francis, v. 26, n. 4, p. 745–766, 2017. Disponível em: [⟨https://doi.org/10.1080/10618600.2017.1384734⟩](https://doi.org/10.1080/10618600.2017.1384734). 9
- DONOHO, D. 50 Years of Data Science. *Journal of Computational and Graphical Statistics*, v. 26, n. 4, p. 745–766, out. 2017. ISSN 1061-8600, 1537-2715. Disponível em: [⟨https://www.tandfonline.com/doi/full/10.1080/10618600.2017.1384734⟩](https://www.tandfonline.com/doi/full/10.1080/10618600.2017.1384734). 32
- ELMASRI, R.; NAVATHE, S. *Fundamentals of Database Systems*. Pearson/Addison Wesley, 2017. ISBN 9789332582705. Disponível em: [⟨https://books.google.com.br/books?id=s5d8zwEACAAJ⟩](https://books.google.com.br/books?id=s5d8zwEACAAJ). 15
- FRISTON, K.; SAMOTHRAKIS, S.; MONTAGUE, R. Active inference and agency: optimal control without cost functions. *Biological Cybernetics*, v. 106, n. 8, p. 523–541, out. 2012. ISSN 1432-0770. Disponível em: [⟨https://doi.org/10.1007/s00422-012-0512-8⟩](https://doi.org/10.1007/s00422-012-0512-8). 10, 32
- HEINS, C. et al. pymdp: A Python library for active inference indiscrete state spaces. *Journal of Open Source Software*, v. 7, n. 73, p. 4098, maio 2022. ISSN 2475-9066. Disponível em: [⟨https://joss.theoj.org/papers/10.21105/joss.04098⟩](https://joss.theoj.org/papers/10.21105/joss.04098). 32
- HIPP, *SQLite: self-contained, serverless, zero-configuration, transactional SQL database engine*. [S.l.]: sqlite.org, 2000. 13
- ISO Central Secretary. *Document management — Portable document format — Part 2: PDF 2.0*. Geneva, CH, 2020. ICS: 35.240.30, 37.100.99. Disponível em: [⟨https://www.iso.org/standard/75839.html⟩](https://www.iso.org/standard/75839.html). 11

KLUYVER; RAGAN-KELLEY; PÉREZ. Jupyter notebooks – a publishing format for reproducible computational workflows. In: F. LOIZIDES AND B. SCHMIDT (Ed.). *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. [S.l.], 2016. p. 87 – 90. 10

KOTONYA, G.; SOMMERVILLE, I. *Requirements engineering: processes and techniques*. [S.l.]: Wiley Publishing, 1998. 11

LOGGING - logging facility for python. 2022. Disponível em: [⟨https://docs.python.org/3.9/library/logging.html⟩](https://docs.python.org/3.9/library/logging.html). 28

PIP documentation v22.3.1.2022. 2022. Disponível em: [⟨https://pip.pypa.io/en/stable/⟩](https://pip.pypa.io/en/stable/). 10

PYPA. *Python packaging user guide*. 2023. Disponível em: [⟨https://github.com/pypa/packaging.python.org⟩](https://github.com/pypa/packaging.python.org). 10

PYTHON, R. *When to use a list comprehension in Python*. Real Python, 2021. Disponível em: [⟨https://realpython.com/list-comprehension-python/⟩](https://realpython.com/list-comprehension-python/). 20

PYTHON Sequence Types — list, tuple, range. 2022. Disponível em: [⟨https://docs.python.org/3.9/library/stdtypes.html#typeseq⟩](https://docs.python.org/3.9/library/stdtypes.html#typeseq). 21

SHAFRANOVICH, Y. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. [S.l.], 2005. [⟨http://www.rfc-editor.org/rfc/rfc4180.txt⟩](http://www.rfc-editor.org/rfc/rfc4180.txt). Disponível em: [⟨http://www.rfc-editor.org/rfc/rfc4180.txt⟩](http://www.rfc-editor.org/rfc/rfc4180.txt). 12

SQLITE.ORG. *Most Widely Deployed and Used Database Engine*. 2021. Disponível em: [⟨https://sqlite.org/mostdeployed.html⟩](https://sqlite.org/mostdeployed.html). 13

Stack Overflow. *Stack Overflow Developer Survey Results 2020*. 2020. Disponível em: [⟨https://insights.stackoverflow.com/survey/2020⟩](https://insights.stackoverflow.com/survey/2020). 12

VAN ROSSUM GUIDO AND DRAKE, *Python 3 Reference Manual*. [S.l.]: CreateSpace, 2009. ISBN 1441412697. 12

WICKHAM, H.; GROLEMUND, G. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly Media, 2016. ISBN 9781491910344. Disponível em: [⟨https://books.google.com.br/books?id=I6y3DQAAQBAJ⟩](https://books.google.com.br/books?id=I6y3DQAAQBAJ). 9, 32