UNIVERSIDADE FEDERAL DO PARANÁ

LUCAS OLINI

# DESENVOLVIMENTO DE ABORDAGEM DE MAPEAMENTO DE ESQUEMAS EM CENÁRIO DE DADOS ABERTOS EDUCACIONAIS

CURITIBA

2019

**LUCAS OLINI**

# DESENVOLVIMENTO DE ABORDAGEM DE MAPEAMENTO DE ESQUEMAS EM CENÁRIO DE DADOS ABERTOS EDUCACIONAIS

Trabalho apresentado como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação no curso de Ciência da Computação, Setor de Ciências Exatas da Universidade Federal do Paraná.
Orientador: Prof. Dr. Marcos Didonet Del Fabro

**CURITIBA**

**2019**

**TERMO DE APROVAÇÃO**

**LUCAS OLINI**

**DESENVOLVIMENTO DE ABORDAGEM DE MAPEAMENTO DE ESQUEMAS EM CENÁRIO DE DADOS ABERTOS EDUCACIONAIS**

Trabalho apresentado como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação no curso de Ciência da Computação, Setor de Ciências Exatas da Universidade Federal do Paraná., pela seguinte banca examinadora:

---

**Prof. Dr. Marcos Didonet Del Fabro**
**Orientador**

---

Professor
UFPR

---

Professor

---

Professor

Curitiba, Dezembro de 2019.

*Este trabalho é dedicado ao pequeno Lucas Olini,*
*que desde criança, em suas brincadeiras e mundos fictícios,*
*sonhou em ser um cientista.*

**AGRADECIMENTOS**

Os agradecimentos iniciais são dedicados a todos os professores que conheci e que se disponibilizaram e se importaram em ensinar os conteúdos lecionados da melhor forma possível, sem eles não teria chegado até aqui. Em seguida, meus agradecimentos são para:

- aos meus pais, Denise do Rocio Olini e Odair Olini, que sempre acreditaram em mim e me incentivaram a ir cada vez mais longe, sempre provendo o necessário para que pudesse dar meu melhor;
- ao meu professor orientador, Prof. Dr. Marcos Didonet Del Fabro, por acreditar e me ajudar durante toda a execução deste trabalho, sempre se mostrando disponível para ajudar, além de ter mostrado um grande conhecimento e profissionalismo, dos quais pude aprender muito;
- à minha namorada, Karine Martins Alves, que tive o presente de conhecer durante a graduação e me torna uma pessoa melhor a cada dia;
- ao meu colega e grande amigo, Felipe Shi Iu Wu, que compartilhou comigo toda a graduação e seus desafios, assim como me presenteou com sua amizade e companheirismo;
- às amizades que criei durante toda a graduação, são pessoas que compartilharam seus dias comigo e que quero levar para a vida toda.

*“A educação é a arma mais poderosa que você pode usar para mudar o mundo.“ (Nelson Mandela)*

**RESUMO**

A quantidade de dados produzida e armazenada cresce a cada ano e com isso surgem inúmeras análises possíveis que podem gerar valor para empresas e para a sociedade. No ambiente de dados abertos, uma grande quantidade de dados é disponibilizada para a sociedade para que possa ser utilizada por quem tiver interesse. No Brasil e em outros países, como os Estados Unidos, os governos disponibilizam dados governamentais sobre a população e o país como um método de transparência para com a sociedade. Porém, estes dados disponibilizados não são necessariamente de fácil uso, não seguem um padrão e não possuem uma documentação acessível para o entendimento dos dados e de seus possíveis mapeamentos com outros conjuntos. Sendo assim, para que análises envolvendo diversos conjuntos de dados ao mesmo tempo sejam feitas, é necessário uma forma de utilização que possibilite o uso integrado destes dados diversos garantindo o correto mapeamento entre eles.

Uma abordagem para resolver este desafio é a de integração de dados, unindo conjuntos de dados diversos através de atributos com mesmo contexto, realizando mapeamentos que possibilitem a correlação entre bases de dados diversas. Técnicas e estudos nesta área de integração de dados se tornam mais relevantes para o contexto de dados abertos e é neste cenário que o presente trabalho se apresenta, propondo uma abordagem de *schema matching* semi-automatizada utilizando os valores dos conjuntos de dados como meio para achar os mapeamentos corretos. A técnica de *schema matching* consiste na prática de realizar o mapeamento entre *schemas* diferentes fazendo a correlação entre os atributos destes, sabendo então quais colunas possuem o mesmo conteúdo.

Esta abordagem proposta foi implementada e testada com conjuntos de dados abertos brasileiros disponibilizados pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP), entidade governamental brasileira. Para colunas tipo texto, foram obtidos mapeamentos corretos com probabilidades entre 100% e 99,6%, para colunas tipo numérico contínuo, mapeamentos corretos com probabilidades entre 98,9% e 92,1%, e para colunas tipo numérico discreto, mapeamento corretos com probabilidades entre 100% e 0%.

Além da implementação e dos resultados obtidos, o presente trabalho mostra os desafios da área de integração de dados para dados abertos, assim como sua importância. É feita também uma comparação dos resultados obtidos com os resultados de uma outra abordagem atual, utilizando as mesmas bases de dados. Sendo assim, o trabalho contribui com a área de estudos com a releitura de uma abordagem mais antiga, tendo os resultados comparados com uma abordagem mais atual. Por fim, são apresentados trabalhos futuros a serem desenvolvidos para que a abordagem proposta obtenha melhores resultados.

**Palavras-chaves**: Integração de dados. *Schema matching*. Dados abertos

**ABSTRACT**

The amount of data produced and stored grows each year, creating countless possible analyses which can generate value for companies and for the society. In the context of open data, a huge amount of data is provided to the society so they can be used by anyone interested. In Brazil and other countries, like the United States of America, governments provide governmental data about the population and the country as a transparency initiative with the society. Although, these provided data are not necessarily easy to use, they do not follow a standard and do not have a proper documentation which is accessible for a better understanding of the data and the possible mappings between datasets. So, for analyses involving multiple datasets at the same time to be done, it is necessary a way to use these multiple datasets in an integrated way that the mappings between them are correct.

An approach to solve this challenge is data integration, aggregating diverse datasets through attributes with the same context, creating mappings that make possible the correlation between diverse datasets. Techniques and researches about data integration become more relevant to the context of open data and it is in this scenario that this work is, proposing a semi automatized schema matching approach using the data values from the datasets as a way to make correlations between the bases. The technique of schema matching consists of realizing the mapping between different schemas making the correlations between the attributes of each other, knowing then which columns have the same content.

This proposed approach was implemented and tested with Brazilian open data datasets provided by Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP), a Brazilian governmental entity. For columns of text type, looking the correct mappings, it was obtained probabilities between 100% and 99,6%, for columns of type continuous numeric, looking the correct mappings, it was obtained probabilities between 98,9% and 92,1%, and for columns of type discrete numeric, looking the correct mappings, it was obtained probabilities between 100% and 0%.

Besides the implementation and the obtained results, this work shows the challenges of data integration for open data, as well as its importance. It is also done a comparison between the obtained results with the results from another current approach, using the same datasets. So, the present work contributes with the area of study with the modification of a dated approach, having results compared with a recent approach. Finally, the future improvements to be developed are presented so the proposed approach can obtain better results.

**Key-words**: Data Integration. Schema Matching. Open Data

## LISTA DE ILUSTRAÇÕES

## LISTA DE CÓDIGOS

# SUMÁRIO

# 1 INTRODUÇÃO

O uso de dados tem crescido cada vez mais no mundo atual, a sociedade como um todo tem utilizado ferramentas que manipulam estes dados gerados e extraem cada vez mais informações inteligentes destes. Com isso, tem-se criado uma grande dependência das pessoas com os dados, que se tornaram parte do dia a dia de todos (DAVID REINSEL, 2018). Quanto mais dados são gerados, armazenados e utilizados, maior se torna nossa dependência. Estima-se que, se somado todos os dados que temos em *datacenters*, infra-estruturas de empresas e dispositivos finais de usuários, como celulares e *notebooks*, tínhamos 33 Zettabytes (ZB) em 2018, e a previsão de crescimento é que teremos 175 ZB até 2025 (DAVID REINSEL, 2018).

Com esta popularização e crescimento, os dados abertos também se tornaram mais presentes e relevantes. Dados abertos são dados acessíveis por todos, disponibilizados para obtenção de qualquer pessoa que tenha interesse (OFFICE, 2012), o que não necessariamente significa que são fáceis de utilizar. Este tipo de dado tem sido muito usado e produzido por entidades governamentais como uma medida de transparência para com a população (DING et al., 2011), possibilitando várias oportunidades de análises e descobertas, as quais podem ter valor para a sociedade. Porém, os dados abertos geralmente não seguem um padrão, os conjuntos de dados disponibilizados podem ter estruturas diferentes, tipos de valores distintos, salvos e disponibilizados em formatos diferentes, dificultando a análise conjunta de múltiplas bases de dados ao mesmo tempo e/ou combinadas. Sendo assim, baseado nas dificuldades e na necessidade de extração de valor de grandes quantidades de dados para empresas e/ou sociedades (LV et al., 2017), pesquisas e ferramentas nas áreas de integração de dados e *schema matching* se tornaram ainda mais relevantes, principalmente no contexto de dados abertos. A técnica de *schema matching* consiste na prática de realizar o mapeamento entre *schemas* diferentes fazendo a correlação entre os atributos destes, sabendo então quais colunas possuem o mesmo conteúdo.

Estudos como (BERLIN; MOTRO, 2002) e (NARGESIAN et al., 2018) apresentam técnicas e abordagens para realização de *schema matching* e integração de dados, focando no objetivo de correlacionar conjuntos de dados distintos em uma base de conhecimento única que possibilite a análise de todos os dados em conjunto de forma correta. Mais especificamente, (NARGESIAN et al., 2018) propõe modelos de integração de dados com o objetivo final de fazer buscas em repositórios de dados abertos que contêm diversas tabelas, mostrando como o assunto está em evidência em pesquisas e estudos mais recentes na área de integração de dados. Reafirmando a relevância do tema, (MILLER, 2018) apresenta o futuro da área de integração de dados e como ela é essencial para o aumento do impacto e manutenção das atividades das áreas de ciência de dados e análise de dados no futuro, visto que repositórios de dados estão em constante crescimento.

Porém, estes estudos apresentam algumas lacunas que o presente trabalho busca trabalhar, como aplicação no cenário brasileiro de dados abertos, disponibilização de um *framework* acessível e de fácil utilização, tendo seu código modularizado e de uso generalista, para que possa ser usado em diversos cenários e ocasiões distintas, e também tratamento específico para atributos do tipo numérico.

A solução proposta neste trabalho se insere neste contexto de dados abertos e na necessidade de integração dos mesmos de forma eficaz e automatizada. Utilizando os estudos e pesquisas já desenvolvidos na área, é proposto um sistema que realiza o mapeamento (*matching*) de colunas entre duas bases distintas que podem estar em formatos diferentes de forma semiautomatizada, provendo ao usuário recomendações de correlações entre colunas das duas bases. Além disso, a solução proposta busca solucionar os problemas citados no parágrafo anterior, provendo um *framework* modularizado e de fácil acesso e uso, que foi testado no cenário de dados abertos brasileiros. A mesma não requer entradas manuais especializadas no contexto e também propõe uma forma clara de lidar com parte dos dados numéricos. A ideia desta solução surgiu a partir de um processo manual realizado pelo C3SL (Centro de Computação Científica e Software Livre) (DIRENE et al., 2016), grupo de pesquisa do Departamento de Informática da Universidade Federal do Paraná (UFPR), de integrar, anualmente, os Microdados do INEP (INEP, 2019) em uma base comum para consolidação dos dados históricos e tornar possível a análise histórica e completa dos dados.

Dentre as contribuições do presente trabalho para a área de estudo, pode-se destacar a releitura de uma abordagem datada de mapeamento de dados, realizando modificações e aplicando testes sobre a mesma, apresentando resultados e os comparando com os resultados obtidos com os mesmos conjuntos de dados por uma abordagem atual proposta por outro trabalho.

O presente trabalho está organizado na seguinte estrutura: a Seção II aborda os conceitos de bancos de dados, integração de dados, *schema matching*, dados abertos e formato de arquivo CSV necessários para o entendimento completo da solução proposta. A Seção III percorre por trabalhos relacionados e que foram referência para a concepção e implementação da solução proposta. A Seção IV descreve detalhadamente a solução implementada, mostrando seu funcionamento e aspectos específicos de implementação. Na Seção V são apresentados os resultados obtidos ao utilizarmos a solução implementada em um exemplo real. Por fim, a conclusão deste presente artigo e uma breve discussão sobre próximos passos são colocados na Seção VI.

# 2 REFERENCIAL TEÓRICO

Esta seção aborda os conceitos explorados pelos trabalhos relacionados e que são utilizados na solução proposta no presente trabalho.

## 2.1 BANCO DE DADOS

Um banco de dados possui a função de armazenar dados, os quais são valores puros que podem refletir informações quando vistos da maneira correta. Pensando em um sistema, os dados armazenados pelo banco de dados seriam dados operacionais pertinentes para o correto e completo funcionamento do sistema, sendo diferente dos dados de entrada (entrada do usuário, por exemplo) e dos dados de saída (relatórios, por exemplo) (DATE, 2003).

Entre os dados puros e os usuários, existe uma camada de *software* que realiza o intermédio entre as duas partes, que é chamada de sistema de gerenciamento do banco de dados (SGBD, ou DBMS, em inglês). Este sistema é responsável por receber comandos dos usuários e realizar as manipulações necessárias nos dados puros (DATE, 2003), como por exemplo uma consulta aos dados ou até mesmo uma alteração na estrutura de armazenamento dos mesmos.

### 2.1.1 Banco de dados relacional

Os dados podem ser armazenados de forma tabular, ou seguindo o modelo relacional, que consiste de tabelas, colunas e linhas. As tabelas são conjuntos de dados que possuem colunas e linhas para armazenamento e organização dos dados. Cada coluna, ou atributo, contém tipos de dados distintos entre si, mudando o contexto do dado (podendo representar um nome, um valor numérico, um ID, uma data, entre outros) e o tipo do mesmo (podendo ser número inteiro, texto, formato de data, entre outros). Já as linhas representam tuplas (termo técnico mais correto para *linhas* ou *registros*) (DATE, 2003), que ligam os dados de todas as colunas linha a linha, criando relação entre os mesmos. Como este modelo de banco de dados consiste na relação entre os valores de uma mesma tupla, o mesmo pode ser visto em forma de tabela. Com isso, temos que cada célula da tabela é referente à uma linha e à uma coluna.

Seguindo o exemplo da Figura 1, a tabela *Pessoas* possui 3 colunas: *ID* (do tipo número inteiro), *Nome* (do tipo texto) e *Idade* (do tipo inteiro). Possui também 3 linhas, cada uma representando uma relação entre as colunas diferentes (a linha de ID 1 representa o João que tem 26 anos, a linha de ID 2 representa a Maria que tem 35 anos e a linha de ID 3 representa o José que tem 48 anos).

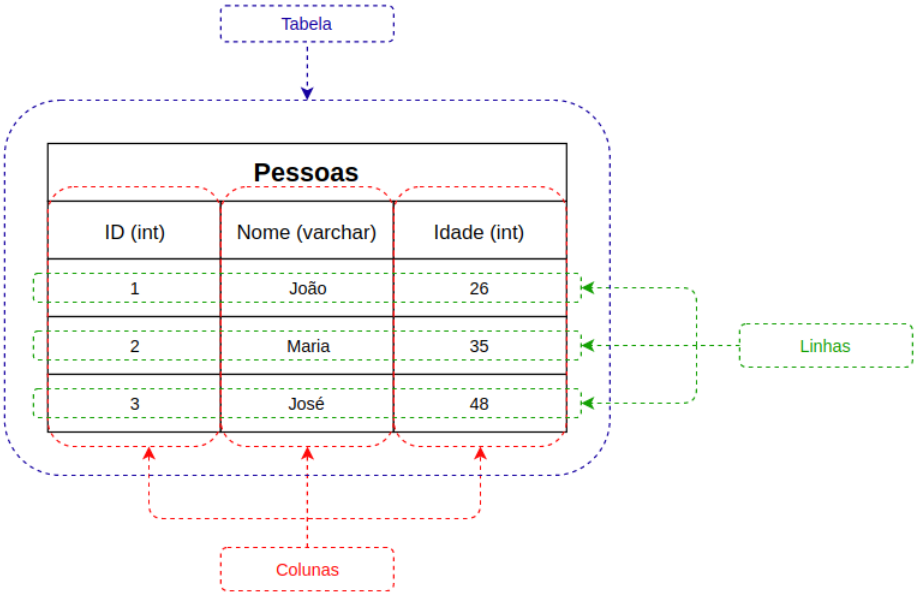


FIGURA 1 – Exemplo de tabela de um banco de dados relacional com suas colunas e linhas

## 2.2 INTEGRAÇÃO DE DADOS

Quando há a necessidade de relacionar diferentes fontes de dados, é necessário realizar alguma forma de unificação, para permitir o acesso a estas fontes. Abordagens de integração de dados permitem essa unificação. As abordagens existentes permitem combinar diferentes fontes de dados em uma visão unificada destes dados para o usuário (LENZERINI, 2002), realizando o ato de agregar informações de diferentes fontes e fazê-las trabalharem juntas (ULLMAN, 1997). A integração de dados é uma parte importante na tarefa de análise e manipulação de dados, visto que ela obtém, manipula e armazena dados oriundos de fontes diferentes em uma base unificada para consultas e buscas (WANG; HAAS; MELIOU, 2018). As primeiras soluções na área visavam a integração de bancos de dados relacionais, porém este conceito foi expandido para diferentes fontes de dados, como arquivos em formato XML, texto e CSV (DIDONET DEL FABRO, 2007).

Na década de 80, uma das abordagens centrais para integração de dados era o uso de federações de dados, estes podendo ser distribuídos ou centralizados, com uma visão unificada dos dados. Porém, com a popularização do uso da internet e o aumento crescente de dados disponibilizados, as abordagens evoluíram para tratamento de dados com visões integradas ou parciais. Uma abordagem possível é através do conceito conhecido como *Data Exchange*, que consiste na tradução de diferentes fontes de dados para uma fonte centralizada, porém fazendo a junção apenas dos conjuntos de dados que se tem interesse e desenvolvendo uma organização específica para determinada junção (MILLER, 2018). Outra abordagem recente são os *polystores*, que consistem na utilização de dados de fontes diferentes ao mesmo tempo através da utilização de diferentes sistemas de gerenciamento de banco de dados (SGBD) com camadas integradoras, sendo que estas fontes podem ter estruturas de armazenamento e tipos

de dados diferentes entre si (GADEPALLY et al., 2016).

Para realizar a integração de dados de diferentes fontes, é necessário realizar o mapeamento dos dados (DIDONET DEL FABRO, 2007), ou seja, definir qual atributo de uma fonte de dados possui o mesmo contexto e valores de um atributo de uma outra fonte de dados. Fazendo este processo para todos os atributos de todas as fontes de dados, teremos todas as ligações possíveis entre elas, completando então o mapeamento.

Uma arquitetura comum para integração de dados é a mostrada pela Figura 2, na qual temos três bancos de dados diferentes (BD-1, BD-2 e BD-3) que são acessados por um ponto de acesso comum. Este ponto de acesso tem o conhecimento das três fontes de dados e, portanto, consegue realizar mapeamentos (M1, M2 e M3) entre dados requisitados por consultas/buscas e os dados reais nas fontes. Estas consultas/buscas são realizas pelo usuário no ponto de acesso, que representa a visão unificada dos dados de todas as fontes para o usuário.

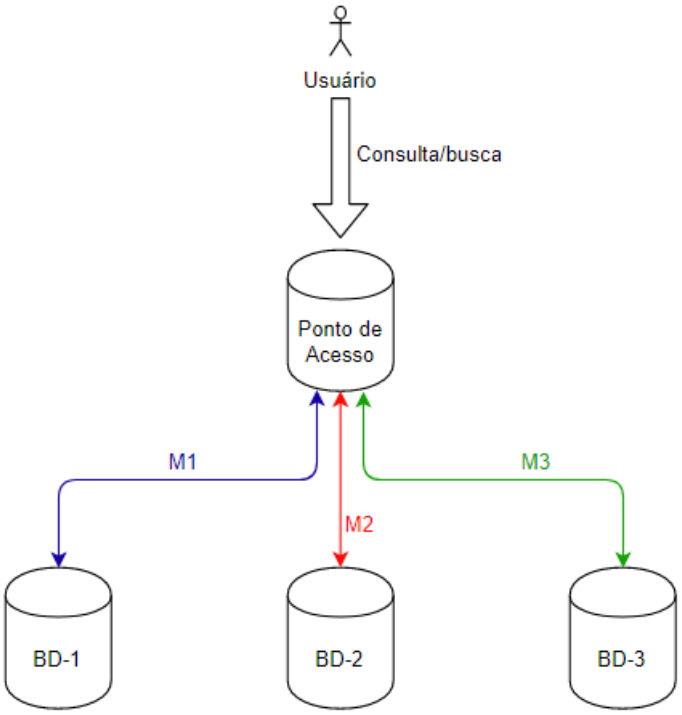


FIGURA 2 – Exemplo de integração entre três bancos de dados diferentes

## 2.3 SCHEMA MATCHING

Quando as diferentes fontes de dados tem formato heterogêneo, isto é, quando há uma diferença na estrutura dos esquemas, é necessário realizar o alinhamento dos mesmos, em um processo conhecido como *schema matching*, que consiste na prática de realizar o mapeamento entre *schemas* diferentes fazendo a correlação entre os atributos destes, sabendo então quais colunas possuem o mesmo conteúdo. O termo *schema* se refere ao modelo que uma fonte de dados utiliza para modelar e guardar seus dados (MILO; ZOHAR, 1998), por exemplo, um banco

de dados utiliza seus *schemas* para modelar as instâncias do banco, um arquivo em formato CSV possui seu *schema* seguindo o próprio modelo que possui regras de separação de valores, entre outras. E o termo *matching*, derivado de *match*, se refere à realização do mapeamento entre os elementos dos dois *schemas* de entrada (RAHM; BERNSTEIN, 2001).

O *schema matching* é o processo de criação de mapeamentos, ou relacionamentos, entre elementos de *schemas* diferentes (DIDONET DEL FABRO, 2007), mas ele não é usado somente para isso, podendo também ser explorado em cenários de construção de *data warehouses*, por exemplo (RAHM; BERNSTEIN, 2001).

Seguindo o exemplo da Figura 3, temos um mapeamento entre duas tabelas de bancos de dados relacionais (poderiam ser duas fontes de dados de outro tipo, como arquivos em formato CSV). O mapeamento *M1* relaciona a coluna *Cidade* da *Tabela 1* com a coluna *Município* da *Tabela 2*, o mapeamento *M2* relaciona a coluna *Nome* da *Tabela 1* com a coluna *Habitante* da *Tabela 2* e o mapeamento *M3* relaciona a coluna *Idade* da *Tabela 1* com a coluna *Idade* da *Tabela 2*. É possível ver que os mapeamentos foram feitos baseados no contexto e no tipo de dado das colunas.

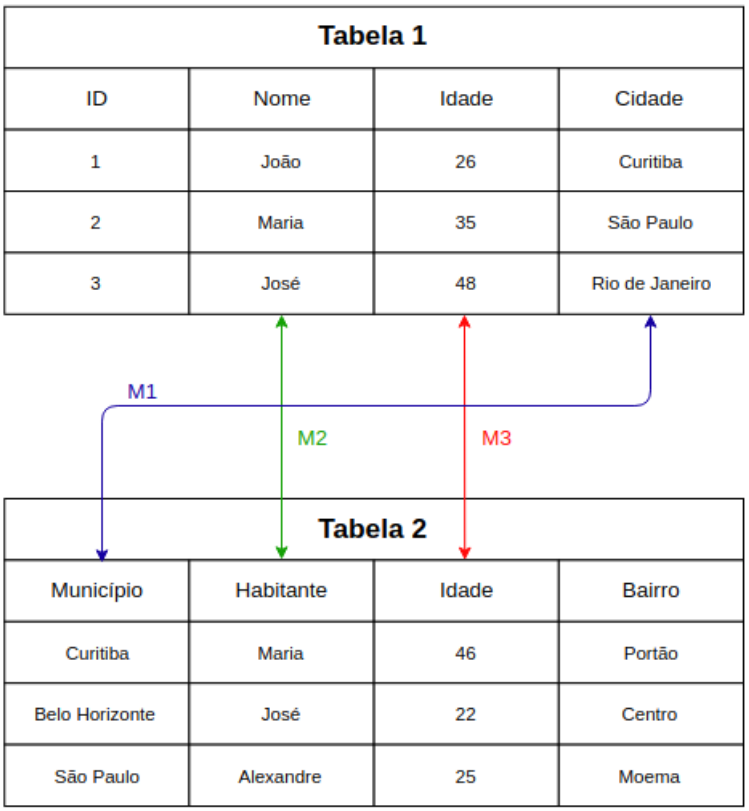


FIGURA 3 – Exemplo de mapeamento de atributos entre duas tabelas distintas

O problema é que esse processo de mapeamento normalmente é feito de forma manual, sendo então bastante demorado e não escalável. Existem formas de automatização parcial ou completa desse processo, utilizando informações destes *schemas* para tal (RAHM; BERNSTEIN, 2001). Uma das formas é utilizando os valores *schema-level*, ou seja, metadados do *schema*, como

nome das colunas, tipo de dados das colunas, tamanho dos dados das colunas, entre outras. Uma outra forma é utilizando os valores *instance-level*, ou seja, os próprios valores contidos nas colunas. Também existe a possibilidade de realizar a automatização deste processo de forma hibrida, ou seja, utilizando ambas as técnicas descritas acima combinadas para um resultado mais completo e preciso (RAHM; BERNSTEIN, 2001). Além destas formas, existem várias outras abordagens, como *element-level* e *structure-level* (RAHM; BERNSTEIN, 2001).

## 2.4 DADOS ABERTOS

O *schema matching* pode ser utilizado no contexto de dados abertos, em que existem diversos conjuntos de dados diferentes que podem ser utilizados em conjunto para derivar análises mais profundas e ricas. Porém, estes conjuntos não estão previamente correlacionados e informações que detalhem explicitamente e de forma direta os possíveis mapeamentos entre esses conjuntos não são encontradas facilmente. Diante disso, a técnica de *schema matching* pode ajudar.

Segundo (OFFICE, 2012), dados abertos são dados que:

- são acessíveis (idealmente pela internet) sem limitações relacionadas à permissões para usuários ou permissões para determinados objetivos, o único custo existente é custo próprio de manipulação destes dados;
- estão disponibilizados digitalmente e em um formato que seja possível a leitura por uma máquina, para que os mesmos possam ser integrados com outros dados de fontes diferentes; e
- livre de restrições no uso e redistribuição em seu termo de uso.

O atual estado dos dados abertos requer que pessoas busquem conjuntos destes dados e façam análises sobre eles, fazendo então uma análise manual referente à relevância destes dados (PARNIA, 2014). Estes conjuntos são buscados em portais ou pontos de acesso de instituições que liberam estes dados e possibilitam a análise dos mesmos por terceiros (PARNIA, 2014).

Nos anos recentes, temos observado um crescimento de conjuntos de dados abertos governamentais disponibilizados, ou seja, dados obtidos e disponibilizados por entidades governamentais. Estes conjuntos possuem uma variedade de informações relevantes para a população, como custo de saúde regional, gastos governamentais, entre outros. A disponibilização destes conjuntos tem sido usada como um canal de comunicação entre o governo e sua população (DING et al., 2011).

No contexto de dados abertos, o tipo de arquivo CSV é o mais utilizado para publicação de conjuntos de dados. Utilizando como exemplo os conjuntos de dados disponibilizados pelo governo brasileiro em (DADOS ABERTOS, 2019), dos 6.920 conjuntos disponibilizados

(consultado em: 18 de agosto de 2019), 5.854 destes estão em formato CSV. Analisando também o portal de dados abertos do governo estadunidense (U.S. GENERAL SERVICES ADMINISTRATION, 2019), todos os 236.303 conjuntos disponibilizados (consultado em: 06 de outubro de 2019) estão em formato CSV ou EXCEL. Com isso, conseguimos ver um grande uso e grande tendência do uso deste formato para disponibilização de dados abertos.

Para os conjuntos de dados deste contexto, não são encontradas facilmente informações que detalhem os possíveis relacionamentos entre conjuntos, ou até mesmo informações diretas que facilitem no mapeamento entre eles.

## 2.5 FORMATO DE ARQUIVO CSV

O tipo de arquivo CSV foi muito usado para troca de dados entre planilhas digitais, como Microsoft Excel. Ele consiste na separação de cada registro em uma linha, e a separação dos atributos de cada registro é feito através de vírgulas (caso tenha uma vírgula no meio do valor do campo, este deve ser salvo com aspas duplas no início e fim). Além disso, em alguns arquivos em formato CSV, a primeira linha contém o cabeçalho, que consiste no nome de cada coluna (REPICI, 2004).

Para exemplificar um arquivo em formato CSV, podemos observar no Código 2.5.1 um pedaço de um arquivo em formato CSV figurativo. Neste pedaço, a primeira linha se refere ao cabeçalho e as outras duas à registros de valores. Todos os valores são separados por vírgula, e em situações de valores que possuem uma vírgula em seu valor, o mesmo deve ser demarcado por aspas duplas em seu início e fim, como podemos ver com o valor do segundo registro na coluna *ENDEREÇO*.

```
1 NOME, IDADE, CIDADE, ENDEREÇO
2 Lucas, 23, Curitiba, Rua Carlos Machado 123
3 João, 32, São Paulo, "Rua Machado Carlos, 321"
```

CÓDIGO 2.5.1: Exemplo de arquivo em formato CSV

É importante destacar que alguns arquivos em formato CSV não utilizam vírgulas como separadores de valores, mas outros sinais como o ponto-e-vírgula, por exemplo. Isto não é um problema, desde que no momento de leitura do arquivo seja especificado qual o separador correto para a correta interpretação dos valores contidos.

Um processo de *schema matching* entre dois arquivos CSV diferentes se dá da mesma forma que com tabelas de um banco de dados relacional, por exemplo. Mapeamentos são criados ligando atributos dos dois arquivos que possuem o mesmo contexto. Para melhor visualização, temos a Figura 4, na qual o mapeamento *M1* relaciona o atributo *Cidade* do arquivo à esquerda com o atributo *Município* do arquivo à direita, o mapeamento *M2* relaciona o atributo *Nome* do arquivo à esquerda com o atributo *Habitante* do arquivo à direita e o mapeamento *M3* relaciona

o atributo *Idade* do arquivo à esquerda com o atributo *Idade* do arquivo à direita. É possível ver que os mapeamentos foram feitos baseados no contexto e no tipo de dado dos atributos.

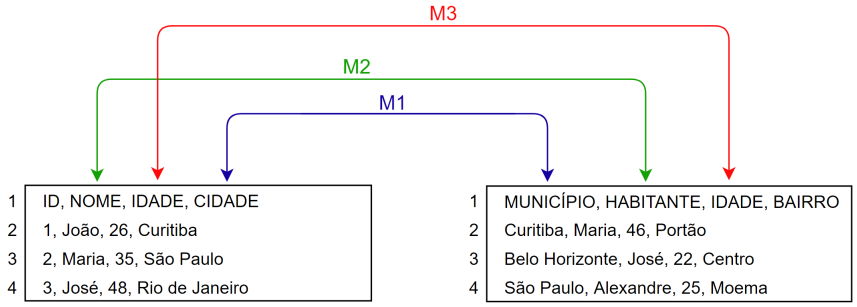


FIGURA 4 – Exemplo de mapeamento de atributos entre dois arquivos CSV distintos

# 3 TRABALHOS RELACIONADOS

Com a popularização dos dados abertos e a necessidade de extração de valor de grandes quantidades de dados para negócios e/ou sociedades (LV et al., 2017), pesquisas nas áreas de integração de dados e *schema matching* se tornaram relevantes para este contexto. Porém, estudos nestas área já eram relevantes antes, como (RAHM; BERNSTEIN, 2001), que apresenta diversas abordagens para realizar o *schema matching*. Uma delas chamada de *instance-level* se baseia em instâncias das bases de dados, utilizando os valores dos dados para realizar os mapeamentos e não a estrutura que estes dados estão armazenados. Esta abordagem por instâncias é a utilizada neste presente trabalho.

Em (BERLIN; MOTRO, 2002), os pesquisadores propõem uma metodologia de *schema matching* na qual é realizado o mapeamento dos atributos de duas fontes de dados diferentes através do uso de uma base de conhecimento comum contendo informações dos atributos, chamada de dicionário de atributos. Para a época, este trabalho tinha uma grande vantagem por propor um sistema automatizado, visto que grande parte dos trabalhos se baseavam no trabalho manual de mapeamento. Além disso, com o dicionário de atributos criado, era possível o mapeamento entre diversos conjuntos de dados sem todo o processamento inicial, o que se mostrou uma grande vantagem em um ambiente em que cada mapeamento manual entre duas bases de dados diferentes não podia ser usado para o mapeamento de outras bases.

O processo de *matching* se dá da seguinte forma: inicialmente, as duas fontes são lidas e o mapeamento entre atributos se inicia. Este mapeamento será feito para toda possível combinação de dois atributos de fontes diferentes, ou seja, a tentativa de mapeamento será feita para todos os atributos de uma fonte com todos os atributos da segunda fonte. Para isso, são feitas tentativas de mapeamento de todos os atributos de ambas as fontes com todos os atributos presentes no dicionário de atributos. Nestas tentativas de mapeamento, um *score* individual para determinada combinação será gerado, o qual é calculado utilizando a fórmula 1

$$M(X, A) = \frac{P(A)}{P(V)} . \prod_{k=1}^{n} P(v_k|A) \tag{1}$$

para atributos texto. Esta fórmula se baseia no Teorema de Bayes e para o contexto deste trabalho, *A* é o conjunto que representa o dicionário de atributos, *X* é o conjunto que representa uma das fontes por vez e *v* são os valores de *X*. As probabilidades apresentadas na fórmula são:

- *M(X,A)*: A probabilidade a posteriori do conjunto *X* mapear para o conjunto *A*, pois ela reflete a probabilidade de um mapeamento de *X* para *A* ser correto após observar os valores de *X*.
- *P(A)*: A probabilidade de ocorrência de *A*, que é a probabilidade a priori do conjunto *X*

mapear para o conjunto *A*, pois ela reflete a probabilidade de um mapeamento de *X* para *A* ser correto antes de observar qualquer valor de *X*.

- *P(V)*: A probabilidade a priori do conjunto *V* estar em *X*.
- $P(v_k|A)$: A probabilidade de um valor *v* pertencente a *X* dado *A*, ou seja, estar em *A*, partindo do pressuposto de que o mapeamento entre *X* e *A* é correto.

Sendo assim, para o cálculo dos *scores*, utiliza-se a fórmula acima com os valores de *P(A)*, *P(¬A)*, *P(v|A)* e *P(v|¬A)*, sem precisar calcular *P(V)*, visto que *M(X,A) + M(X,¬A) = 1*.

Já para atributos numéricos, é assumida uma distribuição normal e então utilizada uma função de densidade normal. Neste cálculo, são utilizados os valores contidos nas fontes e os valores contidos no dicionário de atributos. Este dicionário de atributos consiste em um conjunto de possíveis valores e suas respectivas probabilidades de aparição para cada atributo contido no dicionário.

Esses *scores* fonte-dicionário são combinados com o intuito de gerar *scores* fonte-fonte. Supondo que um atributo A da fonte de dados 1 tem *score* P1 com o atributo D do dicionário de dados, e que um atributo B da fonte de dados 2 tem *score* P2 com o mesmo atributo D do dicionário de dados, conseguimos combinar os *scores* e ter o mapeamento do atributo A com o atributo B com *score* P1 + P2. Tendo isso, esses mapeamentos fonte-fonte podem ser agrupados e combinados para gerar mapeamento *schema-schema*, no qual conseguimos realizar todos os mapeamentos possíveis entre atributos de uma fonte de dados com atributos de uma segunda fonte de dados. Este resultado final pode ser visto na Figura 5, na qual *A1*, *A2* e *A3* são atributos do dicionário de atributos, *B1* e *B2* são atributos da fonte de dados (*schema*) *R1*, *C1* e *C2* são atributos da fonte de dados (*schema*) *R2*, e *w1-w12* são as *scores* entre atributos.

Na mesma Figura 5 podemos ver que um único atributo de uma fonte mapeia e tem um *score* com todos os atributos do dicionário, existindo então diversas possibilidades de combinação para mapeamento com a segunda fonte de dados. Para isso, (BERLIN; MOTRO, 2002) citam a técnica de buscar sempre os *scores* com maior valor, formando sempre as maiores duplas possíveis em termos de valor de *score*. Porém, esse modelo pode gerar mapeamentos ambíguos, ou seja, um atributo de uma fonte mapeando para mais de um atributo numa segunda fonte. Sendo assim, o autor propõe uma segunda forma de realizar esse agrupamento entre mapeamentos, no qual cada atributo do dicionário de atributos só pode mapear para um atributo de cada fonte, e estes mapeamentos fonte-dicionários serão escolhidos buscando a maior soma total de todos os *scores* de todos os mapeamentos escolhidos. Nesta estratégia, alguns atributos de uma fonte podem ficar sem correspondentes na outra fonte, e vice-versa, mas este é um efeito esperado, já que alguns atributos podem ser exclusivos e únicos de determinada fonte.

Em (NARGESIAN et al., 2018), os autores também propõem uma solução de encontrar atributos que possuem o mesmo contexto e conteúdo em fontes de dados diversas com o intuito

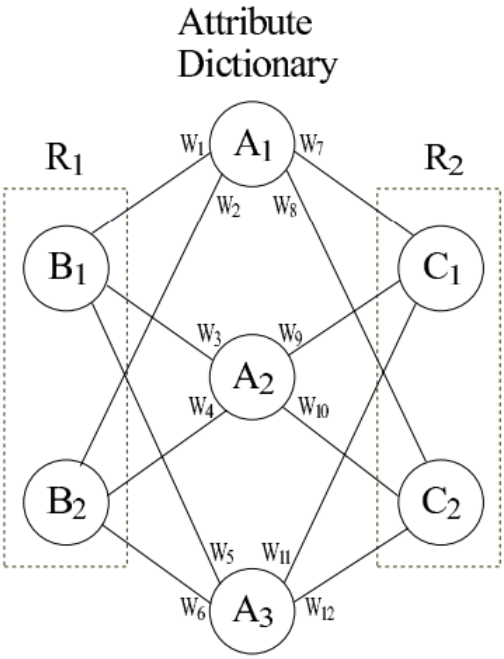


FIGURA 5 – Mapeamento entre atributos e dicionário de atributos. Retirado de (BERLIN; MOTRO, 2002).

de fazer buscas em repositórios de dados abertos que contém diversas tabelas, necessitando então achar correlação entre as tabelas para que uma única consulta possa relacionar dados de fontes diferentes. Neste trabalho, são introduzidas três formas de avaliar a corretude de união entre atributos, sendo cada abordagem utilizada em um tipo específico de atributo. A primeira é a denominada *Set Domains*, que é utilizada para atributos que possuem valores que se repetem em ambas as fontes. Esta abordagem utiliza o teste de distribuição hipergeométrica para calcular a probabilidade de se ter *k* sucessos em *n* escolhas dentro de um grupo de valores, sendo um sucesso quando se escolhe um valor que existe nos atributos de ambas as fontes, e o conjunto total de valores é a união de todos os valores presentes nos atributos de ambas as fontes. Com isso, este cálculo é realizado para todos os valores presentes na interseção de valores dos atributos das duas fontes e os valores são somados. O resultado final da soma representa a probabilidade dos dois atributos possuírem o mesmo contexto.

A segunda abordagem é a denominada *Semantic Domains*, que é utilizada para atributos que não possuem valores puros iguais, porém representam o mesmo contexto semântico, como por exemplo atributos de cidade, país, localização. Nesta abordagem, são criados conjuntos de valores auxiliares para cada atributo contendo as classes semânticas de cada valor do atributo. Por exemplo, para um atributo *localização*, seus valores [Brasil, Curitiba, Paraná] poderiam ser representados pelas classes [país, cidade, estado]. Tendo estes conjuntos auxiliares contendo as classes, o mesmo cálculo utilizando distribuição hipergeométrica e posteriormente o somatório é utilizado, dando como entrada os conjuntos contendo as classes. O resultado final da soma representa a probabilidade dos dois atributos possuírem o mesmo contexto.

Já a terceira abordagem é a denominada *Natural Language Domains*, que é utilizada

para atributos que não possuem valores puros iguais e nem possuem valores que poderiam ser classificados em classes distintas de contexto, como nomes, títulos ou descrições de produtos. Sendo assim, para atributos que possuem valores que pertencem ao domínio da linguagem natural (não são IDs únicos, URLs ou IDs de produtos, por exemplo), é utilizada a técnica *word embedding*. Esta técnica transforma toda palavra em um vetor denso de alta dimensionalidade, e palavras que possuem maior probabilidade de compartilharem o mesmo contexto possuem estes vetores próximos de acordo com a distância Euclidiana. Sendo assim, os autores assumem que palavras que possuem este mesmo contexto, também possuem o mesmo contexto considerando a visão de linguagem natural. Para isto, foi utilizada a *fastText*, que é uma forma de realizar o *word embedding* baseado em artigos do Wikipedia, que provê para a abordagem de *Natural Language Domains* um modelo já treinado baseado em conteúdos externos escritos em linguagem natural.

Em (MILLER, 2018), a autora apresenta um estudo sobre o futuro da integração de dados e como o avanço desta técnica está relacionada aos avanços e necessidades da área de análise de dados. A nova abordagem apresentada considera um cenário com repositórios de dados massivos e que estão em constante crescimento, alinhando então o foco da integração de dados para atividades de exploração e descoberta de dados, já que a área de ciência e análise de dados necessita de um sistema que busque e agregue ou una dados ou tabelas de diferentes fontes de maneira precisa. Com isso, o trabalho propõe uma máquina de busca que seja capaz de encontrar tabelas juntáveis através de pelo menos um atributo (tabelas que possuem pelo menos um atributo em comum com os mesmo valores puros e que possam ser então juntadas em uma tabela com a união dos atributos de ambas), e que também seja capaz de encontrar tabelas que possam ser somadas (tabelas que podem ser concatenadas, mantendo o número de atributos mas tendo uma união das linhas de ambas).

Para encontrar tabelas juntáveis através de pelo menos um atributo, a autora analisa o problema como um problema de busca de domínio, no qual cada atributo de uma tabela é tratado como um domínio. Com isso, é utilizado uma estrutura de índices distribuídos, *LSH (locality sensitive hashing) Ensemble*, que indexa eficientemente todos os domínios de uma tabela em um *data lake* e suporta consultas rápidas de busca de domínio. Para calcular a relevância de comparação entre atributos (A e B, por exemplo) de conjuntos de dados diferentes, é utilizada uma combinação da fórmula de *containment* 2

$$containment(A, B) = \frac{|A \cap B|}{|A|} \tag{2}$$

com a fórmula do Índice de Similaridade de Jaccard 3

$$jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3}$$

tendo então a fórmula 4

$$jaccard(A, B) = \frac{containment(A, B)}{\frac{|B|}{|A|} + 1 - containment(A, B)} \tag{4}$$

como a fórmula resultante. Esta combinação é feita pois o Índice de Similaridade de Jaccard tende a favorecer tabelas com poucos domínios e desfavorecer tabelas com muitos domínios, o que é um problema para o cenário de dados abertos no qual existem tabelas com uma grande quantidade de atributos. Porém, este mesmo índice pode ser eficientemente computado para um grande número de domínios utilizando *LSH* e *MinHash*. Além disso, os conjuntos de dados utilizados são distribuídos entre núcleos de processamento diferentes, tendo então um grupo de indexações *LSH* distribuídas, o que a autora chama de *LSH Ensemble*.

Analisando os três trabalhos, podemos ver a relevância de uma boa forma de integração de dados, que seja precisa e que também seja escalável, conseguindo trabalhar com uma grande quantidade de dados. Vemos também que a solução proposta por (BERLIN; MOTRO, 2002) pode ser usada para o problema de encontrar tabelas somáveis (como visto em (NARGESIAN et al., 2018) e (MILLER, 2018)) e também para o problema de encontrar tabelas juntáveis através de pelo menos um atributo, como visto em (MILLER, 2018), porém também com algumas limitações.

### 3.0.1 Considerações

Apesar dos três trabalhos apresentarem abordagens bastante diferentes, a abordagem dos três quanto à forma de classificação é a mesma, todos utilizam mapeamentos baseados nas instâncias dos *schemas*, ou seja, todos utilizam os valores das colunas para realizar os mapeamentos (abordagem *instance-level* (RAHM; BERNSTEIN, 2001)). (BERLIN; MOTRO, 2002) utilizam aprendizado Bayesiano para detectar similaridades de valores entre colunas, necessitando de uma base de conhecimento prévia, visto que o Teorema de Bayes utiliza as probabilidades a priori para então calcular as probabilidades a posteriori. Além disso, é assumido que as colunas não tem correlação de probabilidade entre si e são independentes, o que não necessariamente é verdade, visto que os contextos das colunas são independentes, mas os mapeamentos são dependentes, já que se espera correlações de um para um. O modelo apresentado por (NARGESIAN et al., 2018) utiliza três formas diferentes de correlação entre colunas, não apenas buscando similaridades de valores puros entre as colunas, como (BERLIN; MOTRO, 2002), com teste de distribuição hipergeométrica mas também analisando se os valores das colunas podem ser abstraídas para classes definidas (realizando então o mesmo cálculo para correlações por valores puros). E se os valores possuem mesmo contexto baseado na linguagem natural, utilizando *word embedding* para encontrar este tipo de similaridade. E por fim, (MILLER, 2018) utiliza o mesmo modelo de (NARGESIAN et al., 2018) para detecção de tabelas somáveis, mas propõe um novo modelo específico para detecção de tabelas juntáveis, que procura por colunas que tenham os valores puros iguais. Este modelo utiliza uma combinação da fórmula de *containment* e Índice de Similaridade de Jaccard, sem necessitar também de uma base de conhecimento prévia, assim como o teste de distribuição hipergeométrica de (NARGESIAN et al., 2018). Como esta fórmula combinada é utilizada para cada possível combinação de colunas, (MILLER, 2018) propõe uma técnica de clusterização processamento e indexação das colunas para um acesso

mais rápido.

Porém, nenhum destes estudos aplica e faz uma análise de suas propostas sob o cenário brasileiro de dados abertos. Para os conjuntos de dados deste contexto, não são encontradas facilmente informações que detalhem os possíveis relacionamentos entre conjuntos, ou até mesmo informações diretas que facilitem no mapeamento entre eles. Além disso, nenhum destes trabalhos apresenta um *framework* acessível e de fácil utilização, tendo seu código modularizado e de uso generalista, para que possa ser usado em diversos cenários e ocasiões distintas. Analisando as soluções propostas, nenhum possui uma proposta clara e explícita de como tratar colunas com valores numéricos. E em (BERLIN; MOTRO, 2002), a solução proposta utiliza uma base de conhecimento construída previamente, que requere entradas manuais especializadas no contexto em questão, exigindo a interação com um humano capacitado para o completo e correto funcionamento da mesma.

Considerando as lacunas e fatores apresentados acima, o próximo capítulo aborda uma proposta de sistema de mapeamento de esquemas que propõe-se a atuar nestes pontos, trazendo o desenvolvimento de uma abordagem para tal.

# 4 ABORDAGEM DE MAPEAMENTO DE ESQUEMAS EM CENÁRIO DE DADOS ABERTOS EDUCACIONAIS

## 4.1 CONTEXTO E MOTIVAÇÃO

O cenário de dados abertos têm importância no contexto atual, e com a quantidade de dados disponíveis, a possibilidade de análises fica ainda maior. Pensando em dados abertos governamentais, o impacto pode ser ainda maior, visto que as análises podem gerar valor para toda a sociedade e melhorar a vida de muitas pessoas. Porém, para que análises mais completas e efetivas sejam feitas, é preferível que tenhamos a possibilidade de correlacionar corretamente estes dados presentes em diversos conjuntos de dados distintos. Além disso, é importante que sejam armazenados de forma padronizada e documentada os dados divulgados de ano em ano, ou de mês em mês, para que análises históricas possam ser feitas, trazendo então outras possibilidades de análises e resultados das mesmas.

Inserido no cenário descrito acima, este trabalho tem objetivo de atuar em um processo realizado atualmente pelo C3SL (DIRENE et al., 2016) de integração anual dos Microdados do INEP (INEP, 2019) em um *schema* padrão. O intuito do sistema desenvolvido é facilitar e automatizar o mapeamento entre colunas deste processo de integração, além de colaborar com todo o cenário de integração de dados abertos. Este processo de integração realizado pelo C3SL (DIRENE et al., 2016) é feito utilizando a ferramenta *HOTMapper* (EHRENFRIED et al., 2019), que lê de um arquivo com um padrão definido os mapeamentos entre bases de dados e realiza a integração das mesmas. O problema é que este arquivo com os mapeamentos é construído manualmente, sendo assim, uma forma automatizada de realizar a integração dos mesmos dados tem valor.

Outra motivação do presente trabalho é atuar nas lacunas e problemas identificados nos trabalhos relacionados na seção anterior, buscando o desenvolvimento de uma abordagem que englobe todos os pontos e consiga então contribuir com a área de estudos, trazendo implementações pertinentes e aplicações de testes para mensuração de resultados.

Um dos grandes desafios do sistema apresentado é a capacidade de ser modularizado e abstrato o suficiente para que seja acessível e utilizável em diversos cenários com diferentes conjuntos de dados, não apenas com os utilizados para a elaboração e testes do sistema, os quais fazem parte da motivação da criação do mesmo. Além disso, é essencial que os resultados de saída do sistema sejam confiáveis para que possibilite a automatização do processo sem perder a qualidade dos mapeamentos.

## 4.2 ARQUITETURA

Tendo a necessidade de possuir um sistema modularizado e abstrato o suficiente para que seja acessível e utilizável em diversos cenários com diferentes conjuntos de dados, uma arquitetura clara e com módulos bem definidos é essencial. A arquitetura da solução proposta é classificada em componentes internos e externos, sendo os internos os componentes que fazem parte do sistema mapeador de dados e os externos os componentes de entrada e de saída. Na sequência, os componentes internos e externos serão abordados detalhadamente.

Como componentes externos, destacamos os arquivos CSV ou as tabelas de um banco de dados relacional que são usados como entradas e as probabilidades de mapeamento que são retornadas como saída. Como componentes internos, podemos destacar os módulos de leitura de arquivos CSV ou de tabelas de bancos de dados relacionais, o módulo que cria o dicionário de atributos, o próprio dicionário de atributos e o módulo de correlação entre colunas, que realiza as correlações e retorna as probabilidades ao usuário. Podemos visualizar esta arquitetura e o fluxo de informações na Figura 6.

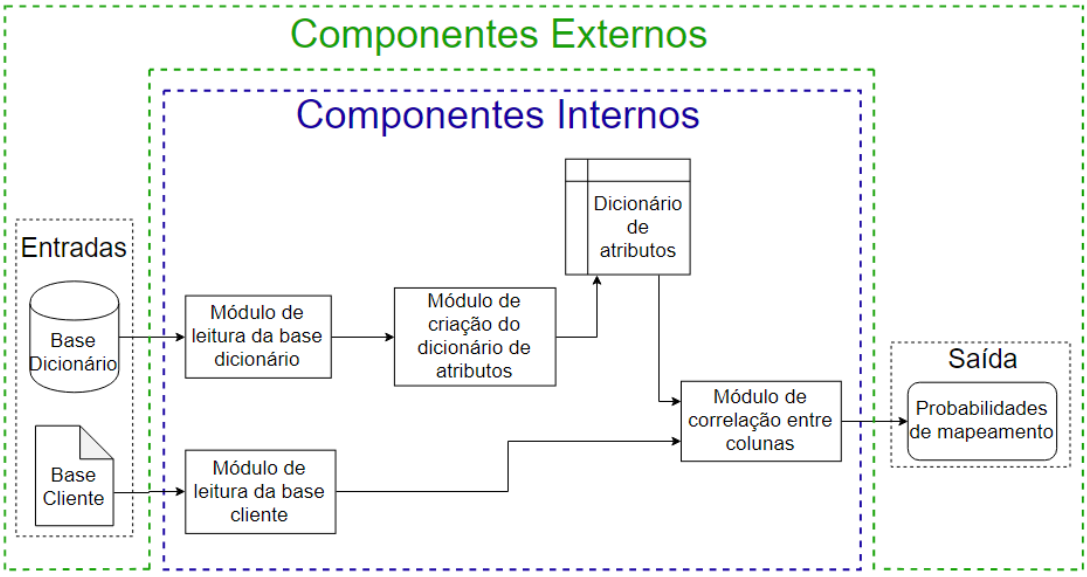


FIGURA 6 – Arquitetura e fluxo de dados da solução

### 4.2.1 Componentes externos

Como entradas, temos a possibilidade de leitura de arquivos CSV e de tabelas de um banco de dados relacional. Serão lidos dois conjuntos de dados:

- Base dicionário: representa a estrutura desejada, será utilizada para construção do dicionário de atributos.
- Base cliente: contém os dados que se deseja integrar na base dicionário.

Isso é ilustrado na Figura 7, sendo que as entradas são uma tabela de um banco de dados relacional como base dicionário e um arquivo CSV como base cliente.

Os atributos destas bases são interpretados de duas formas diferentes baseado no tipo de valor que possuem: valores tipo texto ou tipo numérico discreto (chamados neste trabalho de atributos do tipo 1); e valores tipo numérico contínuo (chamados neste trabalho de atributos do tipo 2). Atributos com valores *booleanos* são desconsiderados e não são mapeados.

**Base Dicionário**

| Tabela Pessoas | | | | | |
|---|---|---|---|---|---|
| ID | Nome | Idade | Cidade | Estado | Altura |
| 1 | João | 26 | Curitiba | Paraná | 1,70 |
| 2 | Maria | 35 | São Paulo | São Paulo | 1,80 |
| 3 | José | 48 | Rio de Janeiro | Rio de Janeiro | 1,75 |
| 3 | Juliane | 45 | Jundiaí | São Paulo | 1,87 |
| 3 | João | 67 | Curitiba | Paraná | 1,65 |

**Base Cliente**

```
1 NOME, IDADE, CIDADE, ENDEREÇO
2 Lucas, 23, Curitiba, "Rua Machado, 20"
3 José, 48, Rio de Janeiro, "Rua Silva, 12"
4 Juliana, 33, São Paulo, Rua Silveira 456
```

FIGURA 7 – Exemplos de formatos para as entradas do sistema

Como saída, temos várias probabilidades de mapeamento entre pares de atributos de bases diferentes. Estes números representam a probabilidade de o par de atributos ser corretamente juntável, ou seja, possuírem o mesmo contexto. A saída com as probabilidades tem o formato apresentado no Código 4.2.1. Os cálculos utilizados para obter estes valores são apresentados na Seção 4.3.

```
1  ==================================
2  Coluna Cidade
3  Mapeamento com atributos do dicionário:
4  Atributo dicionário - UNIVERSIDADE: 0%
5  Atributo dicionário - MUNICÍPIO: 98,8%
6  Atributo dicionário - ESTADO: 78%
7
8  ==================================
9  Coluna Universidade
10 Mapeamento com atributos do dicionário:
11 Atributo dicionário - UNIVERSIDADE: 99%
12 Atributo dicionário - MUNICÍPIO: 0%
13 Atributo dicionário - ESTADO: 0%
```

CÓDIGO 4.2.1: Exemplo de saída do sistema

### 4.2.2 Componentes internos

O módulo de leitura da base dicionário é responsável pela leitura do arquivo CSV ou da tabela de um banco de dados relacional que será usado(a) como a base dicionário. O processo de leitura difere baseado no tipo da coluna:

- colunas compostas de valores texto ou numéricos discretos (tipo 1): os dados texto são sanitizados (retirada dos acentos e passagem de todos os caracteres para caixa baixa) e é feita a contagem de aparições de cada valor por coluna. Após isso, é armazenado, para cada coluna, os valores e suas respectivas contagens de aparições na base dicionário. Esta contagem é utilizada posteriormente para os cálculos de probabilidade de mapeamento.
- colunas compostas de valores numéricos contínuos (tipo 2): todos os valores lidos de cada coluna são armazenados, separados por coluna de aparição.

O módulo de leitura da base cliente é responsável pela leitura do arquivo CSV ou da tabela de um banco de dados relacional que será usado(a) como a base cliente. No processo de leitura, se a coluna for composta de valores texto, os dados são sanitizados e então são armazenados, por coluna, todos os valores distintos entre si. Se a coluna for composta de valores numéricos, é apenas feita a armazenagem, por coluna, de todos os valores distintos entre si.

O dicionário de atributos representa a base de conhecimento das colunas construído através da base dicionário. Ele contém as informações, por coluna, que serão utilizadas na etapa de mapeamento e correlação com a base cliente. Para colunas que possuem valores texto ou valores numéricos discretos (tipo 1), o dicionário de atributos contém 80% dos valores mais encontrados em cada coluna, juntamente com o número de vezes que cada valor aparece (contagem realizada na leitura da base dicionário). Para colunas que possuem valores numéricos contínuos (tipo 2), o dicionário de atributos contém os valores da média e desvio padrão dos seus valores, para que uma distribuição normal possa ser traçada na etapa de correlação entre colunas. O módulo de criação do dicionário de atributos utiliza os dados lidos e armazenados pelo módulo de leitura da base dicionário para criar o dicionário de atributos. O formato de um dicionário de atributos pode ser visualizado na Figura 8, o qual possui três atributos de valores texto e um atributo de valor numérico contínuo. Para os atributos tipo texto, são armazenados os valores seguidos de suas contagens de aparições na base dicionário, e para o atributo tipo numérico contínuo, é armazenado a média e o desvio padrão dos valores encontrados na base dicionário.

O módulo de correlação entre colunas utiliza o dicionário de atributos e os dados lidos e armazenados pelo módulo de leitura da base cliente para calcular as probabilidades de corretude de mapeamento entre pares de atributos (um de cada fonte) e então retornar estes

| Dicionário de Atributos | | | |
|---|---|---|---|
| **UNIVERSIDADE** | **MUNICÍPIO** | **ESTADO** | **IDADE** |
| UFPR: 240 | Curitiba: 280 | PR: 330 | Média: 35<br>Desvio padrão: 3,4 |
| UFMG: 200 | Florianopolis: 180 | SP: 240 | |
| UFBA: 180 | Rio de Janeiro: 150 | RS: 140 | |
| ⋮ | ⋮ | ⋮ | |

FIGURA 8 – Formato de um dicionário de atributos hipotético

valores para o usuário. Os cálculos utilizados para obter estas probabilidades são apresentados na Seção 4.3.

## 4.3 IMPLEMENTAÇÃO

Para desenvolvimento desta solução, foi utilizado como base o trabalho (BERLIN; MOTRO, 2002), que propõe uma forma de pontuação para mapeamento entre colunas de bases distintas. O motivo de escolha deste trabalho e abordagem como base é a tentativa de comparação de uma abordagem mais datada com abordagens mais atuais, atualizando então e contribuindo com a área de estudos sobre esta abordagem em cenários atuais.

Diferente deste trabalho base, na solução proposta no presente trabalho, existirão apenas um dicionário de atributos e uma outra fonte de dados. Podemos visualizar na Figura 9 esta mudança de perspectiva, na qual a base cliente possui dois atributos (*B1* e *B2*) e o dicionário de atributos possui três atributos (*A1*, *A2* e *A3*). As probabilidades de ligação entre esses atributos são os pesos *W1*, *W2*, *W3*, *W4*, *W5* e *W6*.

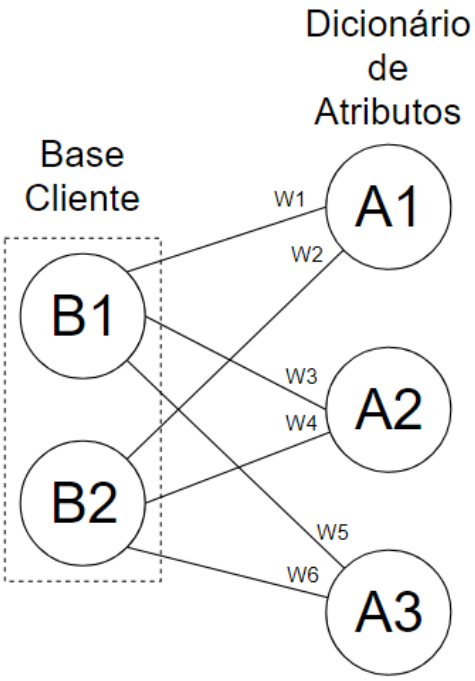


FIGURA 9 – Mapeamento entre atributos e dicionário de atributos da solução proposta

Partindo da forma de cálculo dos *scores* de colunas tipo texto em (BERLIN; MOTRO, 2002), a fórmula utilizada neste presente trabalho para colunas tipo texto e colunas tipo numérico discreto é a 1

$$M(X,A) = \frac{P(A)}{P(V)}.\sum_{k=1}^{n} P(v_k|A) \tag{1}$$

na qual as probabilidades dos valores *v* estarem em *A*, partindo do pressuposto de que o mapeamento entre *X* e *A* está correto, são somadas e não multiplicadas, como no trabalho base. Com isso, é assumido que as probabilidades de correlação são independentes entre si (fato não necessariamente verdadeiro, mas esta assunção é válida para este problema e não compromete o desempenho (BERLIN; MOTRO, 2002)).

Para o cálculo de *P(V)*, sabendo que *M(X,A) + M(X,¬A) = 1*, é utilizada a fórmula 2 (BERLIN; MOTRO, 2002)

$$P(V) = (P(A).\sum_{k=1}^{n} P(v_k|A)) + (P(\neg A).\sum_{k=1}^{n} P(v_k|\neg A)) \tag{2}$$

No contexto do presente trabalho, definimos que *A* representa o dicionário de atributos e *X* representa a outra fonte de dados (base cliente) que se têm como objetivo encontrar mapeamentos com o dicionário de atributos. As probabilidades presentes nas fórmulas são interpretadas e utilizadas da seguinte forma:

- *M(X,A)*: A probabilidade a posteriori do conjunto *X* mapear para o conjunto *A*, pois ela reflete a probabilidade de um mapeamento de *X* para *A* ser correto após observar os valores de *X*.
- *P(A)*: A probabilidade a priori do conjunto *X* mapear para o conjunto *A*, assumindo que o mapeamento de *X* para *A* é correto antes de observar qualquer valor de *X*. Neste presente trabalho, esta probabilidade é estimada pela proporção de exemplos existentes na base dicionário que efetivamente fazem parte do dicionário de atributos. Esta proporção é 80%, pois representa os valores mais representativos ao mesmo tempo que diminui o número de valores no dicionário de atributos. Sendo assim, *P(A)* assume sempre o valor 0.8.
- *P(¬A)*: A probabilidade a priori do conjunto *X* mapear para o conjunto *A*, assumindo que o mapeamento de *X* para *A* não é correto antes de observar qualquer valor de *X*. Por ser a negação do campo anterior, utiliza-se 20% para esta probabilidade (0.2 aplicado no cálculo).
- $P(v_k|A)$: A probabilidade de um valor $v_k$ pertencente a *X* estar em *A* partindo do pressuposto de que o mapeamento entre *X* e *A* é correto. As probabilidades utilizadas são adquiridas calculando o racional entre o número de ocorrências de determinado valor

para determinado atributo no dicionário de atributos sobre o número total de ocorrências de valores para determinado atributo no dicionário de atributos. O somatório destas probabilidades é inicializado como *0,0*.

- $P(v_k|\neg A)$: A probabilidade de um valor $v_k$ pertencente a *X* estar em *A* partindo do pressuposto de que o mapeamento entre *X* e *A* não é correto. Como não se sabe em tempo de execução se um valor é realmente do mesmo contexto de determinado atributo do dicionário de atributos, é assumido que sim e o racional calculado para $P(v_k|A)$ é subtraído do somatório de $P(v_k|\neg A)$. Devido a isto, este somatório é inicializado como *1,0*.

Neste trabalho, o somatório de $P(v_k|A)$ é inicializado como *0.0* e as probabilidades vão sendo somadas a este valor. Já o somatório de $P(v_k|\neg A)$ é inicializado como *1.0* e as probabilidades vão se subtraindo deste valor. É feito desta maneira pois toda vez que se encontra um valor $v_k$ em *A*, aumenta a probabilidade da correlação (iniciada em *0.0*) entre os atributos respectivos de *A* e *X* ser correta, assumindo que o mapeamento entre *A* e *X* é correto. E ao mesmo tempo, diminui a probabilidade da correlação (iniciada em *1.0*) entre os atributos respectivos de *A* e *X* ser correta, assumindo que o mapeamento entre *A* e *X* é incorreto. Esta forma de cálculo foi validada experimentalmente neste trabalho, a validação teórica da mesma é objeto de estudo para um trabalho futuro.

(BERLIN; MOTRO, 2002) também cita a forma de mapeamento entre colunas do tipo numérico, na qual é assumida uma distribuição normal dos valores e então utilizada a função densidade de probabilidade (PDF) para cálculo das probabilidades. Para este trabalho, este procedimento é usado apenas para colunas que possuem valores do tipo numérico contínuo, pois estes sim podemos assumir uma distribuição normal. Já colunas que possuem valores do tipo numérico discreto, utiliza-se o mesmo procedimento que para atributos texto, já que são números distintos sem uma relação entre si. Sendo assim, para colunas do tipo numérico contínuo, é utilizada a fórmula 3

$$M(X, A) = \sum_{k=1}^{n} PDF(v_k|A) \tag{3}$$

Sabendo que *A* representa o dicionário de atributos e *X* representa a outra fonte de dados (base cliente) que se têm como objetivo encontrar mapeamentos com o dicionário de atributos, as probabilidades presentes na fórmula são interpretadas e utilizadas da seguinte forma:

- *M(X,A)*: A probabilidade do conjunto *X* mapear para o conjunto *A*.
- *PDF*$(v_k|A)$: A probabilidade do valor $v_k$ estar presente na distribuição normal dos valores do conjunto *A*.

Será usado um cenário real para exemplificar as etapas do sistema. Este cenário utiliza duas bases dos microdados do Inep (INEP, 2019): o conjuntos de dados referente ao Censo da

Educação Superior de 2015 (será nossa base dicionário); e o conjunto de dados referente ao Censo da Educação Superior de 2016 (será nossa base cliente). Destes conjuntos, utilizaremos o arquivo *DM_DOCENTE.CSV*, que está presente em ambas as bases e representa os dados de docentes para cada ano, respectivamente. Nosso objetivo é gerar recomendações de mapeamentos entre os atributos das duas bases.

### 4.3.1 Construção do dicionário de atributos

Para a construção do dicionário de atributos, usa-se uma base dicionário (podendo ser um arquivo .csv ou até mesmo uma tabela de um banco de dados relacional), que será percorrida linha a linha, coluna a coluna, sendo que cada coluna estará presente no dicionário de atributos. Nesta etapa ocorre a única entrada manual deste sistema, na qual o usuário deve identificar quais colunas possuem valores texto, quais colunas possuem valores numéricos contínuos e quais colunas possuem valores numéricos discretos. Colunas que possuem valores *booleanos* não são tratadas pelo sistema. Sendo assim, no dicionário, cada coluna estará representada de diferentes formas dependendo do tipo de dado que a mesma contém, separando em dois grupos:

- campos texto ou campos numéricos discretos (tipo 1): para estas colunas, o dicionário contém 80% dos valores mais encontrados na mesma, juntamente com o número de vezes que cada valor aparece.
- campos numéricos contínuos (tipo 2): para estas colunas, o dicionário contém os valores da média e desvio padrão dos seus valores, para que uma distribuição normal possa ser traçada.

O Pseudo Código 4.3.1, com o auxílio do Pseudo Código 4.3.2, apresenta a implementação de todo este processo. O mesmo se inicia armazenando todos os nomes das colunas da base dicionário, os quais serão os nomes dos atributos no dicionário de atributos. Começa então a leitura linha a linha da base dicionário, fazendo a sanitização dos valores, a contagem de todos os valores existentes e o armazenamento dos valores com suas contagens para colunas do tipo 1. E fazendo a armazenagem de todos os valores existentes para colunas do tipo 2. Após a leitura, para colunas do tipo 1, os valores são ordenados de forma decrescente baseado no número de ocorrências de cada valor e são mantidos e armazenados apenas os valores que constituem 80% do valor total de aparições e a contagem total destes valores para cada coluna. A lógica de seleção dos valores que constituem 80% do valor total de aparições de determinada coluna pode ser vista no Pseudo Código 4.3.2. Para colunas do tipo 2, é calculada a média e o desvio padrão considerando todos os valores, armazenando então no dicionário de atributos apenas estes dois valores para cada coluna deste tipo. Após todo este processo, o dicionário de atributos está construído e pronto para o uso.

```
1: function constroiDicionarioAtributos
2:     abre arquivo da base dicionário baseDic
3:     valoresCols = [ ]
4:     colsDicAtr = [ ]
5:     for nomeColuna in cabecalho(baseDic) do ▷ armazenamento de todos os nomes das
   colunas da base dicionário
6:         insere nomeColuna em colsDicAtr[ ]
7:     for linha in linhas(baseDic) do
8:         for i in tamanho(colsDicAtr) do
9:             if i in indices_colunas_tipo_texto_num_disc then
10:                valor = sanitizacao(linha[i])                    ▷ sanitização dos valores
11:                if valor in valoresCols[i] then    ▷ armazenamento dos valores com suas
   contagens
12:                    valoresCols[i][valor]+ = 1
13:                else
14:                    valoresCols[i][valor] = 1
15:            else
16:                insere linha[i] em valoresCols[i]    ▷ armazenamento de todos os valores
   existentes
17:    dicAtr = {}
18:    totalOcorrDicAtr = [ ]
19:    for i in tamanho(colunas) do
20:        if i in indices_colunas_tipo_texto_num_disc then
21:            ordena decrescentemente valoresCols[i] pelo número de ocorrências
22:            valoresAtrDic, numTotalOcorr = getTop80(valoresCols[i])
23:            dicAtr[colsDicAtr[i]] = valoresAtrDic
24:            insere numTotalOcorr em totalOcorrDicAtr[i]
25:        else
26:            media = media(valoresCols[i])
27:            desvioPadrao = desvioPadrao(valoresCols[i])
28:            dicAtr[colsDicAtr[i]][media] = media
29:            dicAtr[colsDicAtr[i]][desvioPadrao] = desvioPadrao
30:    return dicAtr, totalOcorrDicAtr
```

CÓDIGO 4.3.1: Pseudo código do processo de construção do dicionário de atributos

```
1:  function getTop80(dic)
2:      numTotalOcorr = soma(valores(dicionario))
3:      numTotalOcorrContabilizadas = 0.0
4:      dicAux = {}
5:      while chave in chaves(dic) do
6:          dicAux[chave] = dic[chave]
7:          numTotalOcorrContabilizadas+ = dic[chave]
8:          remove chave de dic
9:      numTotalOcorrAtrDic = soma(valores(dicAux))
10:     return dicAux, numTotalOcorrAtrDic
```

CÓDIGO 4.3.2: Pseudo código do processo de obtenção de apenas 80% dos valores mais relevantes de um dicionário

Executando este processo para nosso exemplo com as bases do INEP (INEP, 2019) para Censo da Educação Superior, utilizaremos a base de 2015 como nossa base dicionário. O resultado do processo, ou seja, o dicionário de atributos, pode ser visto no Código 4.3.3, que contém um extrato reduzido do dicionário de atributos inteiro. Nele pode-se ver que o atributo *NO_IES* do tipo texto possui os valores *universidade paulista*, com 6628 aparições no total, *universidade de sao paulo*, com 6418 aparições no total, entre outros. Também pode-se ver o atributo *NU_ANO_DOCENTE_NASC* do tipo numérico contínuo, que contém a média e o desvio padrão de seus valores (1971.03 e 10.96, respectivamente).

```
{
    'CO_IES': {
        '322': 6628,
        '55': 6418,
        '56': 4310,
        [...]
    },
    'NO_IES': {
        'universidade paulista': 6628,
        'universidade de sao paulo': 6418,
        'universidade estadual paulista julio de mesquita filho': 4310,
        [...]
    },
    [...]
    'NU_ANO_DOCENTE_NASC': {
        'media': 1971.0285946389101,
        'dv_padrao': 10.960257719023923
    },
    [...]
}
```

CÓDIGO 4.3.3: Extrato do dicionário de atributos do exemplo utilizado

### 4.3.2 Leitura da base cliente

Com o dicionário construído, é feita então a leitura da base cliente (podendo ser um arquivo .csv ou até mesmo uma tabela de um banco de dados relacional). Percorrendo esta base linha a linha, fazendo o tratamento (retirada dos acentos e passagem de todos os caracteres para caixa baixa) para colunas do tipo texto e armazenando para cada coluna todos os valores distintos presentes em cada, independente do tipo da mesma.

### 4.3.3 Correlação entre colunas

Tendo os dois passos anteriores concluídos, a fase de correlação entre colunas é iniciada. O objetivo desta etapa é identificar quais colunas da base cliente possuem o mesmo conteúdo de alguma coluna do dicionário de atributos. Para esta identificação, todas as colunas do tipo 1 da base cliente serão comparadas com todas as colunas do tipo 1 do dicionário de atributos, e todas as colunas do tipo 2 da base cliente serão comparadas com todas as colunas do tipo 2 do dicionário de atributos. O Pseudo Código 4.3.4 apresenta a implementação de todo este processo. A comparação entre colunas se dá de forma diferente baseado no tipo das colunas, sendo:

- campos texto ou campos numéricos discretos (tipo 1), nas linhas 9 à 23 do Pseudo Código 4.3.4: para estas colunas, o processo se inicia com duas variáveis para cada combinação entre coluna da base cliente e atributo do dicionário de atributos, sendo elas:
  - *somatorio_pv_A*: é iniciada com o valor 0.
  - *somatorio_pv_notA*: é iniciada com o valor 1.

  Então, para cada valor distinto de determinada coluna da base cliente, este é procurado em todas as colunas de mesmo tipo do dicionário de atributos. Caso for encontrado, pega-se a probabilidade de determinado valor ser encontrado naquela coluna do dicionário de atributos e a soma na variável *somatorio_pv_A* e a diminui na variável *somatorio_pv_notA* referentes à determinada relação coluna base cliente com coluna dicionário de atributos. Após todos os valores distintos de todas as colunas da base cliente terem sido verificados contra as colunas do mesmo tipo do dicionário de atributos, é feita a seguinte conta para cada relação coluna da base cliente com coluna do dicionário de atributos (sendo a primeira fórmula abaixo a aplicação da fórmula 2, e a segundo fórmula abaixo a aplicação da fórmula 1):

$$pv = (0,8 * somatorio\_pv\_A) + (0,2 * somatorio\_pv\_notA)$$

$$resultado = (0,8/pv) * somatorio\_pv\_A$$

  A variável *resultado* conterá a probabilidade com que determinada coluna da base cliente possui o mesmo conteúdo de determinada coluna do dicionário de atributos, sendo que

teremos uma variável *resultado* para cada combinação, podendo escolher a combinação com maior probabilidade.

- campos numéricos contínuos (tipo 2), nas linhas 4 à 8 do Pseudo Código 4.3.4: para estas colunas, o processo se inicia com apenas uma variável, *resultado*, para cada combinação entre coluna da base cliente e atributo do dicionário de atributos. Então, para cada valor distinto de determinada coluna da base cliente, este é buscado na distribuição normal de cada atributo do mesmo tipo do dicionário de atributos, e o valor resultante é somado à variável *resultado* referente à determinada combinação coluna da base cliente e coluna do dicionário de atributos. Esta busca na distribuição normal, na linha 8 do Pseudo Código 4.3.4, é feita utilizando a função densidade de probabilidade (PDF), que retorna a probabilidade do valor buscado cair na faixa de valores da distribuição normal. Sendo assim, após todos os valores distintos de todas as colunas da base cliente terem sido verificados contra as colunas do mesmo tipo do dicionário de atributos, temos a variável *resultado* para cada combinação, podendo escolher a combinação com maior probabilidade para ditar a correlação correta.

```
1:  function correlacionaAtributos(colsBaseCliente, valoresColsBaseCliente, colsDicAtr, di-
    cAtr, totalOcorrDicAtr)
2:      for i in tamanho(colsBaseCliente) do
3:          resultados = [ ]
4:          if i in indices_colunas_tipo_num_con_base_cliente then
5:              for valorBuscado in valoresColsBaseCliente[i] do
6:                  for j in tamanho(colsDicAtr) do
7:                      if j in indices_colunas_tipo_num_con_dic_atr then
8:                          resultados[j]+ = norm(dicAtr[j]).pdf(valorBuscado)
9:          else
10:             resultadosNegados = [ ]
11:             for j in tamanho(colsDicAtr) do
12:                 inicializa resultados[j] = 0.0
13:                 inicializa resultadosNegados[j] = 1.0
14:             for valorBuscado in valoresColsBaseCliente[i] do
15:                 for coluna in colsDicAtr do
16:                     if valorBuscado in dicAtr[coluna] then
17:                         ocorrencias = dicAtr[coluna][valorBuscado]
18:                         relacaoAoTotal = ocorrencias/totalOcorrDicAtr[coluna]
19:                         resultados[coluna]+ = relacaoAoTotal
20:                         resultadosNegados[coluna]− = relacaoAoTotal
21:             for j in tamanho(colsDicAtr) do
22:                 pv = (0.8 ∗ resultados[j]) + (0.2 ∗ resultadosNegados[coluna])
23:                 resultados[j] = (0.8/pv) ∗ resultados[j]
24:         imprime resultados para a coluna índice i da base cliente
25:     return
```

CÓDIGO 4.3.4: Pseudo código do processo de correlação entre atributos

Com isso, ao final deste processo, teremos uma relação de todas as colunas da base cliente que possuem o mesmo conteúdo que alguma coluna do dicionário de atributos, mostrando qual coluna do dicionário de atributos é e com qual probabilidade as duas possuem o mesmo conteúdo.

# 5 RESULTADOS EXPERIMENTAIS

Para validar o sistema e testar sua eficácia, utilizamos as mesmas duas bases dos microdados do Inep (INEP, 2019) utilizadas como exemplo no capítulo 4: o conjuntos de dados referente ao Censo da Educação Superior de 2015 (será nossa base dicionário); e o conjunto de dados referente ao Censo da Educação Superior de 2016 (será nossa base cliente). Destes conjuntos, utilizaremos o arquivo *DM_DOCENTE.CSV*, que está presente em ambas as bases e representa os dados de docentes para cada ano, respectivamente. A Tabela 1 apresenta algumas métricas de ambas as bases. Como ambas têm a mesma origem e apresentam os mesmos dados, as métricas são muito parecidas, inclusive o nome das colunas são iguais entre si. As análises apresentadas se limitam à qualidade dos resultados de saída, não são feitas análises de desempenho, como tempo de execução, visto que o processamento do mesmo é em lotes e construído para cenários de aplicabilidade sazonal e não constante.

| | Base Dicionário (2015) | Base Cliente (2016) |
|---|---|---|
| Número de linhas | 401.299 | 397.611 |
| Número de colunas | 50 | 50 |
| Número de colunas tipo texto | 9 | 9 |
| Número de colunas tipo numérico discreto | 13 | 13 |
| Número de colunas tipo numérico contínuo | 4 | 4 |
| Número de colunas tipo booleano (0/1) | 24 | 24 |

TABELA 1 – Métricas das bases utilizadas

O sistema foi implementado utilizando a linguagem *Python* (versão 3.7.4), e além das bibliotecas padrões, também foram utilizadas as bibliotecas *Unidecode* (versão 1.1.1) e *SciPy* (versão 1.3.1).

O código utilizado para estes testes e do sistema como um todo pode ser visto em https://github.com/olini/TG, no arquivo *main_docentes_tg_ex.py*. O código está modularizado em três funções: *main*, *get_top_80* e *show_probabilidades*. A primeira realiza a leitura das duas bases, realizando as manipulações e armazenamentos necessários, e também chama as outras duas funções. A *get_top_80* é chamada após a leitura da base que se torna o dicionário de atributos para seleção dos 80% valores mais representativos para cada atributo do tipo 1(texto e numérico discreto). Já a função *show_probabilidades* é chamada após todo o processo de leitura, tratamento e armazenamento das duas bases, com o objetivo de calcular as probabilidades de mapeamento e retornar para o usuário os valores encontrados.

Analisando toda a saída do sistema, podemos ver um êxito do sistema em reconhecer e saber diferenciar corretamente colunas do tipo texto, como por exemplo as colunas *NO_IES*, *DS_REGIME_TRABALHO* e *DS_NACIONALIDADE_DOCENTE*, das quais podemos ver um extrato de

seus respectivos resultados na saída do sistema no Código 5.0.1. Para o atributo *DS_REGIME_TRABALHO*, podemos analisar que o mesmo teve uma probabilidade de correlação com as colunas *CO_UF_NASCIMENTO* e *CO_MUNICIPIO_NASCIMENTO* do dicionário de atributos de aproximadamente 57%. Porém, a probabilidade de correlação com o atributo *DS_REGIME_TRABALHO* do dicionário de atributos, que é a correlação correta, é de 100%, garantindo a corretude na saída do sistema. Isto ocorreu pois as três colunas possuem como um de seus possíveis valores uma *string* vazia, gerando então esta probabilidade apresentada. É importante notar que, para os resultados das colunas *CO_UF_NASCIMENTO* e *CO_MUNICIPIO_NASCIMENTO* da base cliente, o atributo do dicionário de atributos *DS_REGIME_TRABALHO* aparece com 0% de probabilidade de correlação, isto ocorre pois o valor *string* vazia não entrou no conjunto de valores deste atributo no dicionário de atributos, já que não pertence ao grupo dos valores mais representativos do atributo (80% dos valores da base dicionário para determinado atributo, selecionando os com mais ocorrências).

```
[...]
==================================
Coluna NO_IES
Mapeamento com atributos do dicionario:
Atributo dicionario - CO_IES: 0.0
Atributo dicionario - NO_IES: 0.9961334503010552
Atributo dicionario - CO_CATEGORIA_ADMINISTRATIVA: 0.0
Atributo dicionario - DS_CATEGORIA_ADMINISTRATIVA: 0.0
[...]

==================================
Coluna DS_REGIME_TRABALHO
Mapeamento com atributos do dicionario:
Atributo dicionario - CO_IES: 0.0
Atributo dicionario - NO_IES: 0.0
[...]
Atributo dicionario - DS_REGIME_TRABALHO: 1.0
[...]
Atributo dicionario - CO_UF_NASCIMENTO: 0.5702863911332966
Atributo dicionario - CO_MUNICIPIO_NASCIMENTO: 0.5737552003348075
[...]

==================================
Coluna DS_NACIONALIDADE_DOCENTE
Mapeamento com atributos do dicionario:
Atributo dicionario - CO_IES: 0.0
Atributo dicionario - NO_IES: 0.0
[...]
Atributo dicionario - DS_NACIONALIDADE_DOCENTE: 1.0
Atributo dicionario - CO_UF_NASCIMENTO: 0.0
[...]
```

CÓDIGO 5.0.1: Extrato da saída do sistema mostrando os resultados de campos texto

Olhando para todos os resultados de colunas tipo texto da base cliente, temos que:

- para recomendações corretas, a maior probabilidade foi 100%, para várias colunas, e a menor probabilidade foi 99,6%, para a coluna *NO_IES*.
- para recomendações erradas, a maior probabilidade foi 57,3%, para a coluna *DS_REGIME_TRABALHO*, e a menor probabilidade foi 0%, para diversas colunas.

Analisando as colunas do tipo numérico contínuo *NU_ANO_DOCENTE_NASC*, *NU_MES_DOCENTE_NASC*, *NU_DIA_DOCENTE_NASC* e *NU_IDADE_DOCENTE*, as quais possuem seus respectivos resultados na saída do sistema no Código 5.0.2, podemos observar que os valores de saída apresentados são corretos, com exceção dos resultados do campo *NU_DIA_DOCENTE_NASC*, que teve uma maior probabilidade de correlação com o campo do dicionário *NU_MES_DOCENTE_NASC* de forma errônea. De toda forma, o campo tem uma alta taxa de probabilidade de correlação com o campo do dicionário *NU_DIA_DOCENTE_NASC*, que é correta. Este ocorrido pode ser explicado devido à semelhança de valores entre os campos, sendo que um representa um dia e o outro um mês.

```
1  ==================================
2  Coluna NU_ANO_DOCENTE_NASC
3  Mapeamento com atributos do dicionario:
4  Atributo dicionario - NU_ANO_DOCENTE_NASC: 0.9899549320943859
5  Atributo dicionario - NU_MES_DOCENTE_NASC: 0.0
6  Atributo dicionario - NU_DIA_DOCENTE_NASC: 0.0
7  Atributo dicionario - NU_IDADE_DOCENTE: 0.0
8  ==================================
9  Coluna NU_MES_DOCENTE_NASC
10 Mapeamento com atributos do dicionario:
11 Atributo dicionario - NU_ANO_DOCENTE_NASC: 0.0
12 Atributo dicionario - NU_MES_DOCENTE_NASC: 0.9217491323260961
13 Atributo dicionario - NU_DIA_DOCENTE_NASC: 0.3157421939487459
14 Atributo dicionario - NU_IDADE_DOCENTE: 0.002000496324840789
15 ==================================
16 Coluna NU_DIA_DOCENTE_NASC
17 Mapeamento com atributos do dicionario:
18 Atributo dicionario - NU_ANO_DOCENTE_NASC: 0.0
19 Atributo dicionario - NU_MES_DOCENTE_NASC: 0.9604785754109493
20 Atributo dicionario - NU_DIA_DOCENTE_NASC: 0.9220947421991433
21 Atributo dicionario - NU_IDADE_DOCENTE: 0.1274675608069768
22 ==================================
23 Coluna NU_IDADE_DOCENTE
24 Mapeamento com atributos do dicionario:
25 Atributo dicionario - NU_ANO_DOCENTE_NASC: 0.0
26 Atributo dicionario - NU_MES_DOCENTE_NASC: 6.67395945445003e-05
27 Atributo dicionario - NU_DIA_DOCENTE_NASC: 0.33307961217830023
28 Atributo dicionario - NU_IDADE_DOCENTE: 0.987240311067033
```

CÓDIGO 5.0.2: Extrato da saída do sistema mostrando os resultados de campos numéricos contínuos

Olhando para todos os resultados de colunas tipo numérico contínuo da base cliente, temos que:

- para recomendações corretas, a maior probabilidade foi 98,9%, para a coluna *NU_ANO_DOCENTE_NASC*, e a menor probabilidade foi 92,1%, para a coluna *NU_MES_DOCENTE_NASC*.
- para recomendações erradas, a maior probabilidade foi 96%, para a coluna *NU_DIA_DOCENTE_NASC*, e a menor probabilidade foi 0%.

Analisando as colunas do tipo numérico discreto, como por exemplo *CO_IES*, *CO_CATEGORIA_ADMINISTRATIVA* e *CO_ORGANIZACAO_ACADEMICA*, podemos verificar pelos seus respectivos resultados na saída do sistema no Código 5.0.3 que o sistema não foi eficaz em fazer a diferenciação e correlação correta entre as colunas, já que elas tem probabilidades de 100% para correlação com múltiplas outras colunas de forma errônea. Isto ocorre pois estas colunas são compostas de valores numéricos discretos porém sequenciais, ou seja, cada coluna possui sua própria contagem de valores iniciando em 1 e completando o intervalo necessário, ocorrendo então superposição de valores iguais entre colunas que não representam o mesmo contexto. Caso as colunas possuíssem sequências distintas entre si, o sistema poderia retornar as correlações corretas entre as colunas, pois assim poderia diminuir a quantidade de valores sobrepostos. De toda forma, podemos interpretar estes resultados como uma limitação do sistema.

```
1  ================================
2  Coluna CO_IES
3  Mapeamento com atributos do dicionario:
4  Atributo dicionario - CO_IES: 1.0
5  Atributo dicionario - NO_IES: 0.0
6  Atributo dicionario - CO_CATEGORIA_ADMINISTRATIVA: 1.0
7  [...]
8  ================================
9  Coluna CO_CATEGORIA_ADMINISTRATIVA
10 Mapeamento com atributos do dicionario:
11 [...]
12 Atributo dicionario - CO_CATEGORIA_ADMINISTRATIVA: 1.0
13 Atributo dicionario - DS_CATEGORIA_ADMINISTRATIVA: 0.0
14 Atributo dicionario - CO_ORGANIZACAO_ACADEMICA: 1.0
15 [...]
16 ================================
17 Coluna CO_ORGANIZACAO_ACADEMICA
18 Mapeamento com atributos do dicionario:
19 [...]
20 Atributo dicionario - CO_CATEGORIA_ADMINISTRATIVA: 1.0
21 Atributo dicionario - DS_CATEGORIA_ADMINISTRATIVA: 0.0
22 Atributo dicionario - CO_ORGANIZACAO_ACADEMICA: 1.0
23 [...]
```

CÓDIGO 5.0.3: Extrato da saída do sistema mostrando os resultados de campos numéricos discretos

Como um exemplo de campos numéricos discretos com valores não sequenciais, temos as colunas *CO_UF_NASCIMENTO* e *CO_MUNICIPIO_NASCIMENTO*, as quais apresentam resultados

coerentes com o mapeamento correto, como podemos ver no Código 5.0.4. Isto ocorre pois o atributo *CO_UF_NASCIMENTO* contém os DDDs dos estados, e o atributo *CO_MUNICIPIO_NASCIMENTO* contém um código para cada município, mas sempre contendo como sufixo o DDD do estado correspondente, possibilitando então a diferenciação correta do sistema.

```
1  ==================================
2  Coluna CO_UF_NASCIMENTO
3  Mapeamento com atributos do dicionario:
4  Atributo dicionario - CO_IES: 0.14338563308642663
5  Atributo dicionario - NO_IES: 0.0
6  Atributo dicionario - CO_CATEGORIA_ADMINISTRATIVA: 0.0
7  [...]
8  Atributo dicionario - CO_DOCENTE: 0.00019932230416583606
9  [...]
10 Atributo dicionario - CO_UF_NASCIMENTO: 1.0
11 Atributo dicionario - CO_MUNICIPIO_NASCIMENTO: 0.5737552003348075
12 [...]
13
14 ==================================
15 Coluna CO_MUNICIPIO_NASCIMENTO
16 Mapeamento com atributos do dicionario:
17 Atributo dicionario - CO_IES: 0.0
18 Atributo dicionario - NO_IES: 0.0
19 [...]
20 Atributo dicionario - CO_UF_NASCIMENTO: 0.5702863911332966
21 Atributo dicionario - CO_MUNICIPIO_NASCIMENTO: 0.9999999999999999
22 [...]
```

CÓDIGO 5.0.4: Extrato da saída do sistema mostrando os resultados das colunas *CO_UF_NASCIMENTO* e *CO_MUNICIPIO_NASCIMENTO*

Outro resultado interessante para este tipo de coluna é o da *CO_PAIS_DOCENTE*, no qual o sistema retornou como probabilidade 0% a correlação que deveria ser a correta. Esta coluna contém o código do país de nascimento ou de naturalização do docente, porém o código de um mesmo país mudou da base de 2015 (base dicionário) para a base de 2016 (base cliente), impossibilitando o sistema de realizar a correta correlação. Com isso, conseguimos detectar mais uma limitação do sistema apresentado, a inabilidade de reconhecer mudanças de valores para colunas e atributos do mesmo contexto.

Olhando para todos os resultados de colunas tipo numérico discreto da base cliente, temos que:

- para recomendações corretas, a maior probabilidade foi 100%, para várias colunas, e a menor probabilidade foi 0%, para a coluna *CO_PAIS_DOCENTE*.
- para recomendações erradas, a maior probabilidade foi 100%, para várias colunas, e a menor probabilidade foi 0%.

A Tabela 2 apresenta os resultados dos três tipos de colunas juntos. Nela podemos ver que, para colunas contendo valores de texto, o sistema fez recomendações corretas partindo de 99,6% até 100% de probabilidade de correlação. Observando as recomendações erradas do sistema, as probabilidades partem de 0% até 57,3%. Para colunas contendo valores numéricos contínuos, o sistema fez recomendações corretas partindo de 92,1% até 98,9% de probabilidade de correlação. Analisando as recomendações erradas do sistema, as probabilidades partem de 0% até 96%. E por fim, para colunas contendo valores numéricos discretos, o sistema fez recomendações corretas partindo de 0% até 100% de probabilidade de correlação. Já para recomendações erradas do sistema, as probabilidades partem de 0% até 100%.

| | Recomendações corretas - Maior probabilidade | Recomendações corretas - Menor probabilidade | Recomendações erradas - Maior probabilidade | Recomendações erradas - Menor probabilidade |
|---|---|---|---|---|
| Coluna tipo texto | 100% | 99,6% | 57,3% | 0% |
| Coluna tipo numérico contínuo | 98,9% | 92,1% | 96% | 0% |
| Coluna tipo numérico discreto | 100% | 0% | 100% | 0% |

TABELA 2 – Resultados gerais dos tipos de colunas tratados da abordagem proposta pelo presente trabalho

## 5.1 COMPARAÇÃO DE RESULTADOS

A abordagem proposta por (MILLER, 2018) para encontrar tabelas juntáveis utilizando uma combinação entre as fórmulas (fórmula 4) de *containment* (fórmula 2) e Índice de Similaridade de Jaccard (fórmula 3) também pode ser aplicada para o contexto que utilizamos o sistema proposto e apresentado por este trabalho.

A abordagem é aplicada de forma igual para todos os atributos, sem distinção entre colunas do tipo texto e tipo numérico. Porém, para melhor comparação com a abordagem apresentada pelo presente trabalho, os resultados são apresentados seguindo a mesma separação utilizada nos resultados do presente trabalho: colunas tipo texto, colunas tipo numérico contínuo e colunas tipo numérico discreto. Os resultados obtidos podem ser visualizados na Tabela 3.

| | Recomendações corretas - Maior probabilidade | Recomendações corretas - Menor probabilidade | Recomendações erradas - Maior probabilidade | Recomendações erradas - Menor probabilidade |
|---|---|---|---|---|
| Coluna tipo texto | 100% | 85,7% | 11,1% | 0% |
| Coluna tipo numérico contínuo | 100% | 92,5% | 46,1% | 0% |
| Coluna tipo numérico discreto | 100% | 6,5% | 100% | 0% |

TABELA 3 – Resultados gerais dos tipos de colunas tratados da abordagem proposta por (MILLER, 2018)

Para as colunas tipo texto, tivemos os resultados:

- para recomendações corretas, a maior probabilidade foi 100%, para várias colunas, e a menor probabilidade foi 85,7%, para a coluna *DS_COR_RACA_DOCENTE*.
- para recomendações erradas, a maior probabilidade foi 11,1%, para a coluna *DS_REGIME_TRABALHO*, e a menor probabilidade foi 0%, para diversas colunas.

Para este tipo de coluna, temos o caso da coluna *DS_REGIME_TRABALHO* que obteve uma probabilidade de correlação de 57% com outra coluna de forma errônea nos resultados do presente trabalho, nesta abordagem esta probabilidade foi para 11%. Isto ocorre pois não são definidos pesos para os valores e os conjuntos só possuem valores únicos, sendo assim, o valor *string* vazia não gera tanta similaridade entre as colunas que não possuem relação.

Para as colunas tipo numérico contínuo, tivemos os resultados:

- para recomendações corretas, a maior probabilidade foi 100%, para as colunas *NU_MES_DOCENTE_NASC* e *NU_DIA_DOCENTE_NASC*, e a menor probabilidade foi 92,5%, para a coluna *NU_IDADE_DOCENTE*.
- para recomendações erradas, a maior probabilidade foi 46,1%, para a coluna *NU_MES_DOCENTE_NASC*, e a menor probabilidade foi 0%, para algumas colunas.

Para este tipo de coluna, os resultados se mostraram melhores que os do sistema apresentado por este trabalho, visto que não houve erro de correlação para a coluna *NU_DIA_DOCENTE_NASC*. Este resultado melhor para a abordagem de (MILLER, 2018) é causado pelo fato da abordagem utilizada pelo presente trabalho dar pesos para os valores da colunas, podendo gerar erros mais significativos se há valores de grande peso para um coluna presente em outra. Este problema não acontece para a abordagem de (MILLER, 2018), visto que não são definidos pesos para os valores e os conjuntos só possuem valores únicos.

Para as colunas tipo numérico discreto, tivemos os resultados:

- para recomendações corretas, a maior probabilidade foi 100%, para várias colunas, e a menor probabilidade foi 6,5%, para a coluna *CO_PAIS_DOCENTE*.
- para recomendações erradas, a maior probabilidade foi 100%, para várias colunas, e a menor probabilidade foi 0%.

Para este tipo de coluna, também ocorreram erros devido ao fato das colunas terem valores sequenciais iguais. Porém, assim como nos resultados apresentados pela solução deste trabalho, os atributos *CO_UF_NASCIMENTO* e *CO_MUNICIPIO_NASCIMENTO* foram classificados e mapeados corretamente, como visto no Código 5.1.1. Além disso, a coluna *CO_PAIS_DOCENTE* apresentou o mesmo problema de correlação, sendo também causado pela troca dos valores entre uma base e outra.

```
1  ==================================
2  Coluna CO_UF_NASCIMENTO
3  Mapeamento com atributos do dicionario:
4  Atributo dicionario - CO_IES: 0.007582139848357203
5  Atributo dicionario - NO_IES: 0.0
6  Atributo dicionario - CO_CATEGORIA_ADMINISTRATIVA: 0.0
7  [...]
8  Atributo dicionario - CO_DOCENTE: 4.248160546483373e-05
9  [...]
10 Atributo dicionario - CO_UF_NASCIMENTO: 1.0
11 Atributo dicionario - CO_MUNICIPIO_NASCIMENTO: 0.00022614201718679328
12 [...]
13
14 ==================================
15 Coluna CO_MUNICIPIO_NASCIMENTO
16 Mapeamento com atributos do dicionario:
17 Atributo dicionario - CO_IES: 0.0
18 Atributo dicionario - NO_IES: 0.0
19 [...]
20 Atributo dicionario - CO_UF_NASCIMENTO: 0.00022825838849577725
21 Atributo dicionario - CO_MUNICIPIO_NASCIMENTO: 0.9437902688291493
22 [...]
```

CÓDIGO 5.1.1: Extrato da saída da abordagem de (MILLER, 2018) mostrando os resultados das colunas *CO_UF_NASCIMENTO* e *CO_MUNICIPIO_NASCIMENTO*

Estes testes foram executados no mesmo ambiente utilizado para os testes do presente trabalho, utilizando os mesmos recursos de *hardware* e *software*. O código utilizado para estes testes pode ser visto em https://github.com/olini/TG, no arquivo *jaccard_docentes_tg_ex.py*

# 6 CONCLUSÃO

O presente trabalho apresenta um sistema prático e acessível para o *matching* de conjuntos de dados distintos. Em um contexto de dados abertos, iniciativas e ferramentas que trabalham neste contexto de *schema matching* e integração são importantes para que possa ser retirado real valor da grande quantidade de dados que temos hoje em dia. Para que uma análise de dados seja completa e o melhor resultado possível seja entregue, é essencial que os dados dispostos estejam em boa qualidade e corretamente agrupados, aumentando as possibilidades de análises. E é baseado nesta necessidade que o presente trabalho se enquadra, propondo e aplicando em um contexto real um sistema que realiza o *matching* de conjuntos de dados diferentes baseado nas colunas e nos valores desta colunas das bases distintas.

O sistema proposto oferece um *framework* acessível e modularizado que realiza o *matching* de duas bases distintas através dos valores de suas colunas, propondo recomendações com probabilidades de quais colunas entre as bases possuem o mesmo contexto. Por ser modularizado e generalista, o sistema pode ser utilizado em cenários com bases diferentes das utilizadas neste trabalho. A única entrada manual no sistema é a classificação dos tipos das colunas de cada base, todo o resto do processo é feito de forma automatizada, não necessitando um conhecimento prévio dos contextos de cada uma das colunas. Além disso, apresenta um modelo para correlação de colunas do tipo texto e do tipo numérico. O sistema é uma adaptação de uma abordagem existente (BERLIN; MOTRO, 2002), contendo algumas modificações e sendo utilizado para um contexto de dados abertos.

Como vimos com os resultados apresentados, para colunas contendo valores de texto, podemos concluir que o sistema funciona para este tipo de coluna, apresentando corretas classificações e distinções entre colunas. Para colunas contendo valores numérico contínuos, também verificamos uma correta classificação e distinção entre colunas com exceção de uma coluna que obteve sua maior probabilidade de correlação com uma coluna errada.Já para colunas contendo valores numéricos discretos, pelo fato do método utilizado ser o mesmo para colunas contendo valores de texto e o fato das colunas possuírem os mesmos números entre si, o sistema fica impossibilitado de realizar a diferenciação. Porém, é importante destacar que, caso as colunas não possuíssem valores sequencias iguais, o sistema conseguiria realizar a diferenciação corretamente, e portanto, conseguiria mapear as colunas corretamente.

Pensando em melhorias no sistema e na abordagem proposta e trabalhos futuros, visando uma maior automatização do processo, a detecção automática do tipo de cada coluna seria muito importante, visto que este é o único passo manual e que requere entradas de humanos no sistema atual inteiro. Outro passo futuro seria, além de realizar a análise apenas utilizando os valores das colunas para realizar o mapeamento entre os conjuntos de dados distintos, realizar também a análise utilizando os nomes das colunas e outros possíveis metada-

dos, podendo detectar semelhanças entre os nomes das colunas, por exemplo. Além disso, uma análise mais prática, sobre outros possíveis métodos de mapeamento e como eles podem atuar em conjunto com o modelo proposto neste trabalho, pode gerar uma melhoria dos resultados atuais e até a implementação de funções que consigam realizar a detecção e mapeamento corretos das colunas com as características que o modelo proposto não conseguiu realizar corretamente.

Para uma melhor visualização da comparação entre os resultados obtidos pela abordagem proposta e por outras abordagens, deve-se organizar os resultados em um mesmo gráfico, possibilitando a leitura dos valores de ambas as abordagens ao mesmo tempo, facilitando a comparação. A visualização em tabelas diferentes, como foi utilizada neste trabalho, não é a ideal para este tipo de análise.

# REFERÊNCIAS


BERLIN, Jacob; MOTRO, Amihai. Database Schema Matching Using Machine Learning with Feature Selection. In: PROCEEDINGS of the 14th International Conference on Advanced Information Systems Engineering. London, UK, UK: Springer-Verlag, 2002. (CAiSE '02), p. 452–466. ISBN 3-540-43738-X. Disponível em: <http://dl.acm.org/citation.cfm?id=646090.680403>.

DADOS ABERTOS, Portal Brasileiro de. Conjuntos de dados. In: disponível em: <http://dados.gov.br/dataset>.

DATE, Christopher J. **Introdução a sistemas de bancos de dados 8ª Edição**. Rio de Janeiro: Elsevier, 2003. 803 p.

DAVID REINSEL John Gantz, John Rydning. The Digitization of the World From Edge to Core. In: disponível em: <https://www.seagate.com/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf>.

DIDONET DEL FABRO, Marcos. **Gestion de métadonnées utilisant tissage et transformation de modèles**. Set. 2007. Theses – Université de Nantes. Disponível em: <https://tel.archives-ouvertes.fr/tel-00481520>.

DING, Li et al. TWC LOGD: A portal for linked open government data ecosystems. **Journal of Web Semantics**, v. 9, n. 3, p. 325–333, 2011. Semantic Web Dynamics Semantic Web Challenge, 2010. ISSN 1570-8268. DOI: https://doi.org/10.1016/j.websem.2011.06.002. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1570826811000382>.

DIRENE, Alexandre Ibrahim et al. **C3SL – From Education to Public Transparency, Fifteen Years Developing Computer Systems for the Brazilian Society**. [S.l.]: SciTePress, 2016. ISBN 978-989-758-207-3. DOI: 10.5220/0007902900500072.

EHRENFRIED, Henrique et al. HOTMapper: Historical Open Data Table Mapper. In: DOI: 10.5441/002/edbt.2019.53.

GADEPALLY, V. et al. The BigDAWG polystore system and architecture. In: 2016 IEEE High Performance Extreme Computing Conference (HPEC). [S.l.: s.n.], set. 2016. p. 1–6. DOI: 10.1109/HPEC.2016.7761636.

INEP. Microdados. In: disponível em: <http://portal.inep.gov.br/microdados>.

LENZERINI, Maurizio. Data Integration: A Theoretical Perspective. In: p. 233–246. DOI: 10.1145/543613.543644.

LV, Z et al. Next-Generation Big Data Analytics: State of the Art, Challenges, and Future Research Topics. In: 4. v. 13, p. 1891–1899. DOI: 10.1109/TII.2017.2650204.

MILLER, Renée J. Open Data Integration. **Proc. VLDB Endow.**, VLDB Endowment, v. 11, n. 12, p. 2130–2139, ago. 2018. ISSN 2150-8097. DOI: 10.14778/3229863.3240491. Disponível em: <https://doi.org/10.14778/3229863.3240491>.

MILO, Tova; ZOHAR, Sagit. Using Schema Matching to Simplify Heterogeneous Data Translation. In: PROCEEDINGS of the 24rd International Conference on Very Large Data Bases. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998. (VLDB '98), p. 122–133. ISBN 1-55860-566-5. Disponível em: <http://dl.acm.org/citation.cfm?id=645924.671326>.

NARGESIAN, Fatemeh et al. Table Union Search on Open Data. **Proc. VLDB Endow.**, VLDB Endowment, v. 11, n. 7, p. 813–825, mar. 2018. ISSN 2150-8097. DOI: 10.14778/3192965.3192973. Disponível em: <https://doi.org/10.14778/3192965.3192973>.

OFFICE, United Kingdom Cabinet. Open Data White Paper: Unleashing the Potential. In: ISBN 9780101835329. Disponível em: <https://www.gov.uk/government/publications/open-data-white-paper-unleashing-the-potential>.

PARNIA, Armin. User Interface design for open data platforms. In:

RAHM, Erhard; BERNSTEIN, Philip A. A Survey of Approaches to Automatic Schema Matching. **The VLDB Journal**, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 10, n. 4, p. 334–350, dez. 2001. ISSN 1066-8888. DOI: 10.1007/s007780100057. Disponível em: <http://dx.doi.org/10.1007/s007780100057>.

REPICI, Dominic John. HOW-TO: The Comma Separated Value (CSV) File Format. In: disponível em: <http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm>.

ULLMAN, Jeffrey D. Information integration using logical views. In: _____. **Database Theory — ICDT '97**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997. p. 19–40. ISBN 978-3-540-49682-3.

U.S. GENERAL SERVICES ADMINISTRATION, Technology Transformation Service. Data.gov. In: disponível em: <https://www.data.gov/metrics>.

WANG, Xiaolan; HAAS, Laura; MELIOU, Alexandra. Explaining Data Integration. **IEEE Data Engineering Bulletin**, v. 41, n. 2, p. 47–58, jun. 2018.