UNIVERSIDADE FEDERAL FLUMINENSE

LUCAS BERTELLI MARTINS

DIMPLY: Uma Abordagem para Privacidade Diferencial em Sistemas Polystore

NITERÓI

2021

UNIVERSIDADE FEDERAL FLUMINENSE

LUCAS BERTELLI MARTINS

DIMPLY: Uma Abordagem para Privacidade Diferencial em Sistemas Polystore

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Ciência da Computação.

Orientador:
Prof. D.Sc. DANIEL CARDOSO MORAES DE OLIVEIRA

NITERÓI

2021

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

M379d Martins, Lucas Bertelli
DIMPLY: Uma Abordagem para Privacidade Diferencial em Sistemas Polystore / Lucas Bertelli Martins ; Daniel Cardoso Moraes de Oliveira, orientador. Niterói, 2021.
82 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Niterói, 2021.

DOI: http://dx.doi.org/10.22409/PGC.2021.m.06085942702

1. Privacidade Diferencial. 2. Polystores. 3. Base de dados. 4. Saúde. 5. Produção intelectual. I. Oliveira, Daniel Cardoso Moraes de, orientador. II. Universidade Federal Fluminense. Instituto de Computação. III. Título.

CDD -

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

LUCAS BERTELLI MARTINS

`DIMPLY`: Uma Abordagem para Privacidade Diferencial em Sistemas *Polystore*

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Ciência da Computação.

Aprovada em outubro de 2021.

BANCA EXAMINADORA

Prof. D.Sc. Daniel Cardoso Moraes de Oliveira - Orientador, IC/UFF

Prof. Dr. Javam de Castro Machado - UFC

Prof. D.Sc. José Viterbo Filho, IC/UFF

Prof. D.Sc. Victor Ströele de Andrade Menezes, UFJF

Niterói

2021

Aos meus pais, Ricardo e Luzia, que sempre me deram todo o apoio e amor.

Agradecimentos

Primeiramente eu gostaria de agradecer à Deus e a Nossa Senhora da Água Santa pela intercessão por todas as bênçãos ao longo da minha trajetória.

Ao meu pai Ricardo e a minha mãe Luzia que foram fundamentais na minha vida, sempre me apoiando de todas as formas e me ofertando muito amor. Agradecer também a minha amada irmã Rafaella.

Ao professor Daniel de Oliveira, por toda dedicação, apoio, conselhos, uma contribuição que vai muito além da sua orientação responsável nessa dissertação. Muito obrigado por todo sua contribuição em minha formação acadêmica, considero você um exemplo! Sou realmente muito grato por todo apoio ao longo dessa trajetória, todos os ensinamentos e conversas. Trajetória essa que começou há bastante tempo, desde que foi meu professor no Ensino Técnico de Informática no CEFET-RJ, depois na Graduação em Sistemas de Informação, já na UFF, o qual também me orientou em meu TCC, e agora no Mestrado. Sempre ministrando excelentes aulas, à disposição dos alunos para esclarecer dúvidas, e olha que até mesmo as que não eram da sua matéria hein! Além das boas conversas que tivemos durante todos esses anos.

Gostaria de agradecer à todos os professores do Instituto de Computação da UFF pelas excelentes aulas ministradas, por todas as contribuições para minha formação acadêmica.

Gostaria de agradecer também aos professores Dr. Javam de Castro Machado, D.Sc. José Viterbo Filho e D.Sc. Victor Ströele de Andrade Menezes por aceitarem fazer parte da banca examinadora. E a professora D.Sc. Aline Marins Paes Carvalho por aceitar ser suplente.

Por fim, gostaria de agradecer à CAPES pela bolsa concedida.

Resumo

As organizações (sejam elas públicas ou privadas) detém dados de seus clientes e funcionários, seja para pesquisa ou como forma de obter vantagens competitivas. Apesar de representarem um potencial ganho para as organizações, as consultas e o aprendizado realizado sobre esses dados podem apresentar riscos à privacidade dos indivíduos cujos dados se encontram nos *datasets*, podendo vazar dados sensíveis, como valor do salário, endereço de residência, *etc*. É de responsabilidade dos donos dos dados garantir a privacidade dos indivíduos cujos dados se encontram no *dataset* em questão. Desde que a Lei Geral de Proteção aos Dados (LGPD) entrou em vigor no Brasil, em agosto de 2020, qualquer organização deve cumprir obrigações legais em relação à privacidade de dados em território brasileiro. Existem diversas técnicas e abordagens para garantir a privacidade de dados, em especial dos dados armazenados em Sistemas de Gerência de Banco de Dados (SGBDs), sejam eles relacionais ou não. Entretanto, nos dias atuais muitas organizações optam por armazenar seus dados em formato bruto em *Data Lakes*. Tais dados podem ser encontrados em múltiplos formatos (*e.g.*, JSON, XML, CSV, bancos relacionais, *etc*). Para conseguir consultar esses dados heterogêneos de forma integrada, os sistemas *Polystore* vêm sendo utilizados. Esses sistemas permitem que o usuário submeta uma consulta que integra dados em formatos heterogêneos por meio de uma sintaxe de consulta única. Entretanto, os sistemas *Polystore* não consideram questões de privacidade até o momento, delegando essa responsabilidade para os SGBDs que eles consultam. Mesmo que os SGBDs que um sistema *Polystore* consulta ofereçam recursos de privacidade, a integração posterior dos dados pode trazer ameaças. Somente remover os identificadores explícitos (*e.g.*, nome, CPF, *etc*) e semi-identificadores (*e.g.*, CEP) ou disponibilizar apenas resultados agregados pode não proporcionar proteção suficiente. Nessa dissertação, propomos uma abordagem chamada `DIMPLY` para integrar mecanismos de privacidade em sistemas *Polystore*. Os usuários deste *middleware* submetem consultas na sintaxe do *Polystore* e recebem os resultados já anonimizados, sem depender dos SGBDs subjacentes. Como técnica de privacidade, escolhemos a Privacidade Diferencial (com os mecanismos de Laplace, Gaussiano e Resposta Randômica). Consideramos um contexto de consultas interativas e buscamos preservar a utilidade dos dados. Para avaliar o `DIMPLY`, utilizamos um *dataset* de exames de casos suspeitos do Vírus da Zika (ZIKV) no Brasil, extraído do sistema do SUS GAL (Gerenciador de Ambiente Laboratorial). Os resultados indicaram que a solução proposta foi capaz de anonimizar o retorno das consultas submetidas ao sistema *Polystore*, preservando a utilidade dos dados.

Palavras-chave: Privacidade Diferencial, *Polystores*, Banco de Dados, Anonimização, Saúde.

Abstract

Organizations (whether public or private) retain the data of their customers and employees, either for research or as a way to gain competitive advantages. Although they represent a potential gain for organizations, the consultations and the learning carried out on these data can pose risks to the privacy of individuals whose data are in the datasets, and sensitive data such as salary value, residence address, etc. may be leaked. It is the responsibility of the data owners to guarantee the privacy of the individuals whose data are found in the data in question. Since the General Data Protection Law (LGPD) entered is valid in Brazil in August 2020, any organization must comply with legal obligations concerning data privacy in Brazilian territory. There are several techniques and approaches to ensure data privacy, especially data stored in Database Management Systems (DBMS), whether relational or not. However, nowadays many organizations choose to store their data in raw format in Data Lakes. Such data can be found in multiple formats (*e.g.*, JSON, XML, CSV, relational databases, *etc*). To be able to query this heterogeneous data in an integrated way, Polystore systems have been used. These systems allow for the user to submit a query that integrates data in heterogeneous formats through a single query syntax. However, Polystore systems do not consider privacy issues so far, delegating this responsibility to the DBMS they consult. Even though the DBMS that a Polystore system queries offer privacy features, further data integration can bring threats. Simply removing the explicit identifiers (*e.g.*, name, social security number, *etc*) and semi-identifiers (*e.g.*, zip code) or providing only aggregated results may not provide sufficient protection. In this dissertation, we propose an approach called `DIMPLY` to integrate privacy mechanisms into Polystore systems. Users of this middleware submit queries in Polystore syntax and receive the results already anonymized, without depending on the underlying DBMS. As a privacy technique, we chose Differential Privacy (with Laplace, Gaussian, and Randomized Response mechanisms). We consider a context of interactive queries and seek to preserve the usefulness of the data. To evaluate `DIMPLY`, we used a dataset of examinations of suspected cases of the Zika Virus (ZIKV) in Brazil, extracted from the SUS GAL (Laboratory Environment Manager) system. The results indicated that the proposed solution was able to anonymize the return of queries submitted to the Polystore system, preserving the usefulness of the data.

Keywords: Differential Privacy, *Polystores*, Databases, Anonymization, Health.

Sumário

1 Introdução **1**

1.1 Motivação 4

1.2 Contribuições 5

1.3 Organização do Trabalho 5

2 Referencial Teórico **6**

2.1 Classificação de Dados 6

2.2 Privacidade de Dados 7

2.2.1 Privacidade Diferencial 7

2.2.2 Resposta Randômica 8

2.2.3 Sensibilidade Global da Consulta Δf 9

2.2.4 Laplace 10

2.2.5 Gaussiano 12

2.2.6 Utilidade dos Dados 13

2.2.7 Composição 13

2.2.8 O *Privacy Budget* 14

2.3 Sistemas *Polystore* 15

2.3.1 BigDAWG 16

2.3.2 Apache Drill 19

2.4 Considerações Finais 23

3 Trabalhos Relacionados **24**

3.1 Anonimização de Consultas com Privacidade Diferencial 24
3.1.1 PINQ e wPINQ 24
3.1.2 FLEX 26
3.1.3 CHORUS 28
3.1.4 *Shrinkwrap* 30
3.1.5 APEx 31
3.1.6 Discussão 32
3.2 Escolha do melhor mecanismo para um cenário 32

4 Abordagem Proposta: `DIMPLY` 34

4.1 Arquitetura da Solução 34
4.2 Detalhes Técnicos da Solução 36
4.2.1 Escopo da Solução 37
4.2.2 Modelo de Custo 38
4.2.2.1 Definição do Modelo de Custo 39
4.2.2.2 Implementação do Modelo de Custo 41
4.2.3 Cálculo da Sensibilidade Global da Consulta Δf 42
4.2.4 Implementação do Resposta Randômica 43
4.2.5 Cálculo do Erro Relativo de cada Mecanismo 44
4.2.6 Limitações do `DIMPLY` 44

5 Avaliação Experimental 45

5.1 Estudo de Caso 45
5.2 Descrição do Ambiente 47
5.3 Configuração dos Parâmetros 47
5.4 Resultados 49
5.4.1 Experimento 49

6 Conclusão e Trabalhos Futuros **62**

6.1 Trabalhos Futuros 64

Referências **65**

Apêndice A – Resultados com outros valores de ϵ **69**

Capítulo 1

Introdução

A privacidade de dados é uma responsabilidade legal das organizações [13, 47]. Em 2016, a Comissão Europeia aprovou uma legislação que trata da privacidade das informações dos países e cidadãos europeus. A GDPR (*General Data Protection Regulation* ou Regulamento Geral sobre Proteção de Dados) [47] entrou em vigor em 25 de Maio de 2018. No Brasil, a Lei Geral de Proteção de Dados Pessoais (LGPD) [13] entrou em vigor em Setembro de 2020, porém as punições relativas ao seu descumprimento entraram em vigor a partir de 1 de Agosto de 2021 [14]. De forma geral, ambas estabelecem que os processos que lidam com dados pessoais são obrigados a seguir medidas que respeitem os princípios da proteção de dados. Tais medidas definem os direitos dos indivíduos presentes em um conjunto de dados, regras em relação à coleta e tratamento dos dados, necessidade de obtenção do consentimento do indivíduo e especificação explícita da finalidade de uso dos dados.

Além de obrigações de segurança para garantir que não ocorram vazamentos, como por exemplo, que os dados sejam armazenados utilizando *pseudonimização* ou *anonimização* [13] e usando as mais elevadas configurações de privacidade por padrão, os dados não podem ser disponibilizados sem o consentimento explícito do titular e o mesmo pode revogar essa autorização a qualquer momento. Não deve ser possível identificar os indivíduos em um *dataset* sem uma chave secreta adicional que deverá ser armazenada em separado do *dataset*. As organizações cuja atividade principal envolvam manipular dados pessoais são obrigadas a terem um cargo específico responsável por assegurar que o tratamento está de acordo com a lei [13, 47]. As organizações são ainda obrigadas a comunicar dentro de um prazo definido qualquer violação de dados que tenha qualquer efeito adverso na privacidade do indivíduo [47]. Dado o exposto, é uma prioridade para as organizações que os dados sejam anonimizados de alguma forma para evitar os problemas supracitados.

A anonimização de dados é uma técnica que objetiva preservar a privacidade dos indivíduos contidos em um *dataset* através da alteração dos valores originais dos atributos. Contudo, essa modificação implica em uma certa perda de informação e, consequentemente, pode diminuir a utilidade dos dados. Logo, o desafio é anonimizar os dados ao mesmo tempo em que procuramos garantir a utilidade dos mesmos. Lidar com esse *trade-off*, de forma que a privacidade dos indivíduos seja protegida, enquanto a utilidade dos dados é mantida, pode não ser trivial.

Ao longo das últimas décadas, diversas técnicas de anonimização de dados foram propostas na literatura [9]. Os Modelos Sintáticos foram os primeiros, os mais conhecidos da literatura são k-anonimato [48], l-diversidade [37], t-proximidade [36] e δ-presença [42]. Porém, esses modelos são vulneráveis à ataques maliciosos que utilizam de conhecimento prévio, e podem ser capazes de identificar os indivíduos presentes quando utilizadas estas técnicas. Muito esforço foi despendido na busca de técnicas que garantissem um nível de privacidade forte, como alcançado pelas técnicas RAPPOR (*Randomized Aggregatable Privacy-Preserving Ordinal Response*) [21] e a Privacidade Diferencial [18] que objetivam resolver esse problema.

Em especial, a Privacidade Diferencial [18] se tornou uma das técnicas mais utilizadas, sendo uma técnica de anonimização de dados com um forte rigor matemático. A Privacidade Diferencial permite análises estatísticas sobre conjuntos de dados enquanto preserva a privacidade dos indivíduos. Ela assegura que qualquer resposta para uma determinada consulta tenha ocorrência igualmente possível, independente da presença ou ausência de um indivíduo no *dataset*. A técnica de Privacidade Diferencial foi proposta inicialmente no contexto de consultas interativas, nas quais o usuário submete uma consulta e recebe a resposta de forma anonimizada. Apesar disso, também existem formas de utilizar a Privacidade Diferencial sobre todo um *dataset* para poder publicar os dados anonimizados [12].

Para anonimizar os dados, a Privacidade Diferencial utiliza um mecanismo responsável por inserir um ruído aleatório nos dados do *dataset* ou no retorno da consulta. O mecanismo mais usado atualmente para consultas estatísticas é o de Laplace [19]. Outros mecanismos comuns são o Gaussiano [19], que aplica uma distribuição simétrica a de Laplace, e o de Resposta Randômica [50], inicialmente proposta como uma técnica de pesquisa social com intuito que os participantes respondessem questões sensíveis, como o uso de drogas ilícitas, sem serem identificados. Cada um desses mecanismos possui vantagens e desvantagens, e pode ser mais adequado para uma determinada consulta. Avaliar

a utilidade de cada um deles para uma determinada consulta se torna uma importante tarefa a ser desempenhada.

Atualmente, as organizaçõe (sejam elas privadas ou públicas) possuem *datasets* armazenados em diferentes formatos. É uma tendência que se armazene todos os dados em seus formatos originais (*i.e.*, formato bruto), em uma estrutura do tipo *Data Lake* [40]. Um *Data Lake* pode ser imaginado como um imenso repositório com todos os *datasets* e arquivos da organização armazenados em seu formato bruto (*i.e.*, sem pré-processamento), com ferramentas de gerência, indexação, controle de acesso, entre outras administrativas. Existem sistemas que permitem que esses dados sejam consultados e combinados por meio de uma única linguagem de consulta, e sem exigir um esquema único, chamados de sistemas *Polystore* [15]. Por meio de um sistema *Polystore*, pode-se trabalhar com vários formatos diferentes, como por exemplo, combinar um *dataset* em formato JSON armazenado no sistema de arquivos com uma tabela armazenada em um SGBD relacional como o PostgreSQL[1]. Um sistema *Polystore* atua como uma camada sobre SGBDs existentes, submetendo consultas para múltiplos SGBDs e integrando o resultado.

As organizações precisam consultar e garantir a privacidade desses dados em *Data Lakes*. Porém, na revisão da literatura realizada no contexto dessa dissertação (apresentada na Seção 3.1) não encontramos trabalhos relacionados a como consultar de forma anonimizada dados em sistemas *Polystore*. Em geral, os sistemas *Polystore* delegam essa responsabilidade para os SGBDs que se encontram na camada subjacente. Como cada SGBD pode utilizar técnicas de privacidade diferentes, fica difícil mensurar a perda de privacidade total de uma consulta integrada submetida. Além disso, podem ser consultados arquivos ou SGBDs que não possuem anonimização e trazer ameaças. Diante dessa lacuna, propomos nesse trabalho um *Middleware* de Privacidade Diferencial, chamado `DIMPLY`. Os usuários do `DIMPLY` submetem consultas na sintaxe do sistema *Polystore* utilizado e recebem os resultados anonimizados. Para garantir a privacidade dos indivíduos presentes nos *datasets*, utilizamos no `DIMPLY` a técnica de Privacidade Diferencial e os mecanismos de Resposta Randômica, Laplace e Gaussiano.

Consideramos um contexto de consultas interativas e buscamos preservar a utilidade dos dados. O escopo do `DIMPLY` é anonimizar consultas estatísticas com as funções `SUM`, `AVG`, `VARIANCE`, `STDEV`, `MIN` e `MAX`. Temos como limitação não suportar o `COUNT` na versão atual. Além disso, consideramos que o dono dos danos é confiável e que as parcelas de dados manipuladas na anonimização e consulta cabem em memória principal.

[1] https://www.postgresql.org/

Para avaliar o `DIMPLY`, utilizamos um *dataset* real contendo casos suspeitos do Vírus da Zika (ZIKV) no Brasil. Este *dataset* foi extraído do sistema do SUS chamado GAL (Gerenciador de Ambiente Laboratorial). O GAL tem como objetivo proporcionar a gerência das rotinas laboratoriais e o acompanhamento das etapas para realização dos exames, possibilitando profissionais da área de saúde consultar, extrair e inferir conhecimento a partir dos dados. Tais dados exportados pelo GAL são fundamentais para identificar a circulação, distribuição e epidemiologia de diversos vírus no país. Porém, essa exportação pode levar à riscos a privacidade dos indivíduos, já que nos *datasets* existem dados sensíveis. Somente remover os identificadores explícitos (*i.e.*, nome, CPF, *etc.*) não é suficiente para proteger a privacidade dos indivíduos, devido à existência dos semi-identificadores (*i.e.*, CEP) [48]. Além disso, disponibilizar apenas os resultados agregados também pode não proporcionar proteção suficiente, devido a um possível e imprevisível conhecimento prévio do atacante obtido de outras fontes de informações [5, 49]. Sendo assim, é necessário o uso de técnicas mais elaboradas de privacidade de dados, como a Privacidade Diferencial nesse contexto. A área saúde tem sido alvo de frequentes ataques maliciosos aos dados sensíveis [11], tornando-se de extrema importância a anonimização dos dados.

1.1 Motivação

Diante do cenário apresentado anteriormente, observamos uma exigência crescente para que as organizações protejam a privacidade dos participantes de suas bases de dados, seja por exigência de seus próprios participantes ou de regulações governamentais, como a Lei Geral de Proteção de Dados (LGPD) e GDPR na Europa.

Nessa dissertação, propomos combinar uma técnica de privacidade de dados com sistemas *Polystore*. Dessa forma o *middleware* de privacidade proposto anonimiza a consulta a todos os *datasets* existentes e futuros da organização. Em suma, os fatores motivadores do trabalho são:

- Regulações Governamentais. Exigências Governamentais de privacidade de dados, *e.g.*, GDPR e LGPD.
- *Datasets* em diferentes formatos. Os usuários precisam analisar e combinar *datasets* de diferentes formatos.
- Acesso a dados em diferentes armazenamentos. Os usuários precisam consultar de forma integrada, eficiente e simplificada dados armazenados em diferentes SGBDs.

- Escolha do mecanismo a ser utilizado. Escolher automaticamente o melhor mecanismo para cada consulta, após aprendizado de qual manteve maior utilidade naquele tipo de consulta estatística.

1.2 Contribuições

Como contribuições desta dissertação propomos o `DIMPLY`, um *Middleware* de Privacidade Diferencial sobre sistemas *Polystore*, que tem como objetivo prover respostas de consultas anonimizadas aplicando a técnica de Privacidade Diferencial sobre dados. Para maximizar a utilidade dos dados, o `DIMPLY` escolhe o mecanismo com o menor Erro Relativo para a consulta estatística submetida. A medida do Erro Relativo avalia o quão distantes as respostas anonimizadas estão das originais. Em síntese, essa dissertação tem como contribuições:

- Um modelo de custo para escolher automaticamente o mecanismo de Privacidade Diferencial para cada consulta submetida de forma a maximizar utilidade dos dados.
- O `DIMPLY`, o *Middleware* desenvolvido capaz de prover acesso diferencialmente privado aos dados em sistemas *Polystore* mantendo a utilidade dos mesmos.

1.3 Organização do Trabalho

Além da introdução, esta dissertação é composta por mais 5 capítulos. O Capítulo 2 apresenta as definições teóricas necessárias para melhor compreensão da abordagem proposta, além de aspectos das tecnologias utilizadas para apoiar a solução desenvolvida. No Capítulo 3 são exibidos e analisados os trabalhos relacionados. O Capítulo 4 descreve a abordagem proposta para anonimização de dados em sistemas *Polystore*, chamada de `DIMPLY` e detalha como foi projetado e sua implementado. O Capítulo 5 relata como foi a avaliação experimental, suas configurações, descrição do *dataset* utilizado e apresenta os resultados. Finalmente, o Capítulo 6 conclui esta dissertação.

Capítulo 2

Referencial Teórico

Neste capítulo são abordados os principais conceitos utilizados ao longo dessa dissertação. Na Seção 2.1 apresentamos como classificamos os dados no trabalho. Uma visão geral acerca da privacidade de dados é apresentada na Seção 2.2. Na Seção 2.2.1 definimos o conceito de Privacidade Diferencial e explicamos os mecanismos que foram utilizados no trabalho, além de definirmos utilidade de dados e como a mensuramos por meio do Erro Relativo. Na Seção 2.3 é apresentado o conceito de sistemas *Polystore* e duas implementações atuais desse conceito, sendo na Subseção 2.3.2 o utilizado em nosso trabalho. E por fim as considerações finais na Seção 2.4.

2.1 Classificação de Dados

De acordo com [8], existem diferentes tipos de dados: (i) Dados Relacionados, (ii) Dados de Transações, (iii) Dados de Grafo e (iv) Dados de Trajetória. E a escolha da técnica de anonimização mais adequada pode variar de acordo com o tipo de dado em questão. Nessa dissertação vamos considerar apenas Dados Relacionados, que é o modelo mais comum de se armazenar dados. Pode-se pensar nesse tipo como um conjunto de dados relacionais representado por uma tabela, onde cada coluna corresponde a um atributo e cada linha uma tupla. Em geral, esse tipo de dado tem um conjunto fixo de atributos que são comuns em uma coleção de tuplas $t_1, t_2, ..., t_n$. Quatro tipos de atributos podem existir em um conjunto de dados desse tipo [22]: (i) identificadores explícitos, (ii) semi-identificadores, (iii) atributos sensíveis e (iv) atributos não sensíveis. Podemos descrever esses tipos mais detalhadamente como:

- Identificadores explícitos: São atributos capazes de identificar unicamente indiví-

duos, tais como: "nome", "CPF", "e-mail". Estes devem ser sempre removidos dos resultados.

- Semi-identificadores: São todos aqueles atributos que não são identificadores explícitos, mas podem potencialmente identificar um indivíduo, em geral quando relacionados com outros conjuntos de dados. São exemplos de semi-identificadores em dados relacionais: "data de nascimento", "cor da pele" e "CEP".

- Atributos sensíveis: São os atributos que contém informações sensíveis dos indivíduos, como "salário", uma "doença", uma "opção religiosa".

- Atributos não sensíveis: São todos os atributos que não se encaixam em nenhum dos tipos anteriores, como o "estado de residência do indivíduo", a "cidade", o "país".

2.2 Privacidade de Dados

Ao longo dos anos, diversos modelos foram propostos para lidar com a questão da privacidade de dados [9]. Inicialmente, surgiram os Modelos Sintáticos, sendo os mais conhecidos da literatura o k-anonimato [48], l-diversidade [37], t-proximidade [36] e δ-presença [42]. Porém, nestes modelos um usuário malicioso, com conhecimento prévio, pode ser capaz de identificar os indivíduos presentes na base de dados. Surgiram então modelos capazes de prover fortes garantias de privacidade, como o RAPPOR (*Randomized Aggregatable Privacy-Preserving Ordinal Response*) [21] implementado pela Google em seu navegador Google Chrome, e que é baseado no modelo de Resposta Randômica [50]. O RAPPOR é capaz de coletar dados estatísticos entregando uma forte garantia de privacidade para o indivíduo, limitando a informação privada divulgada através do limiar ϵ, entregando assim uma privacidade ϵ-diferencial. Na Subseção 2.2.1 a seguir apresentamos o conceito de Privacidade Diferencial, um modelo interativo com perdas de privacidade limitadas pelo parâmetro ϵ e que será adotado na solução proposta nessa dissertação.

2.2.1 Privacidade Diferencial

A Privacidade Diferencial [18] é um modelo matemático capaz de permitir análises estatísticas sobre um conjunto de dados, sem comprometer a privacidade dos indivíduos. Com fortes garantias de privacidade, ela assegura que qualquer resposta à uma determinada consulta, tem ocorrência igualmente possível e independe da presença, ou ausência, de um indivíduo no conjunto de dados. Ela consegue isso por meio de um mecanismo que

insere um ruído aleatório na resposta das consultas. Este modelo foi projetado em um ambiente interativo, sendo essa a forma de Privacidade Diferencial mais utilizada, para consultas interativas e que retornam informações estatísticas [10]. Neste tipo de ambiente, os usuários submetem as consultas a um conjunto de dados, que por sua vez responde a consulta anonimizada por um mecanismo de aleatoriedade.

Com a Privacidade Diferencial é possível mensurar matematicamente o *tradeoff* entre utilidade e privacidade, e calibrar, por meio do parâmetro ϵ, de acordo com as necessidades e exigências da legislação vigente, visto que pode ser mensurada uma possível perda de privacidade na liberação dos dados. A Privacidade Diferencial também é imune a pós-processamento, mesmo considerando poder computacional, conhecimento prévio e outras fontes de dados para realizar ataques de cruzamentos de dados. Em particular, nessa dissertação utilizamos os mecanismos de Resposta Randômica, Laplace e Gaussiano. Nas subseções seguintes discutimos com mais detalhes cada um destes mecanismos e os seus parâmetros de configuração.

2.2.2 Resposta Randômica

O mecanismo de Resposta Randômica [50] foi originalmente uma técnica desenvolvida nas ciências sociais para coletar informações estatísticas sobre dados sensíveis, garantindo que os indivíduos participantes da pesquisa não pudessem ser identificados. Dados sensíveis são aqueles que, se descobertos, podem impactar diretamente na vida daquele indivíduo, como um comportamento constrangedor ou ilegal, um problema de saúde, uma opção política ou religiosa, ser um usuário de drogas ilícitas, entre outras informações sensíveis que podem ser utilizadas para prejudicar de alguma forma o indivíduo que tem sua privacidade violada.

O mecanismo de Resposta Randômica [19] consegue garantir a privacidade dos participantes por meio do processo exemplificado na Figura 2.1. O processo funciona da seguinte forma: para cada pergunta, o participante joga mentalmente uma moeda, caso o lançamento dessa moeda der "COROA" ele sempre responderá a "VERDADE", seja ela "SIM" ou "NÃO". Agora, se no primeiro lançamento der "CARA", o participante deve lançar uma segunda moeda e responder "SIM" se der "CARA" e "NÃO" se der "COROA". A intuição por trás da Resposta Randômica é que o mecanismo sempre fornece uma "negação plausível", de forma que mesmo que o indivíduo venha a ser identificado como participante da pesquisa, não será possível saber se a sua resposta dada foi verdadeira ou falsa. Sendo que esta versão do Resposta Randômica garante privacidade (ln 3, 0)-diferencial como

provado no *Claim* 3.5 no livro [19] dos autores Dwork e Roth [18].

Definição 1 (O mecanismo de Resposta Randômica)**.** *Um algoritmo de Resposta Randômica [19]* $\mathcal{M}$ *com domínio* A *e intervalo discreto* B *está associado a um mapeamento* $M : A \to \Delta(B)$*. Para uma entrada* $a \in A$*, o algoritmo* $\mathcal{M}$ *produz uma saída* $\mathcal{M}(a) = b$ *com probabilidade* $(M(a))_b$ *para cada* $b \in B$*. O espaço de probabilidade é sobre os lançamentos de moeda do algoritmo* $\mathcal{M}$*.*

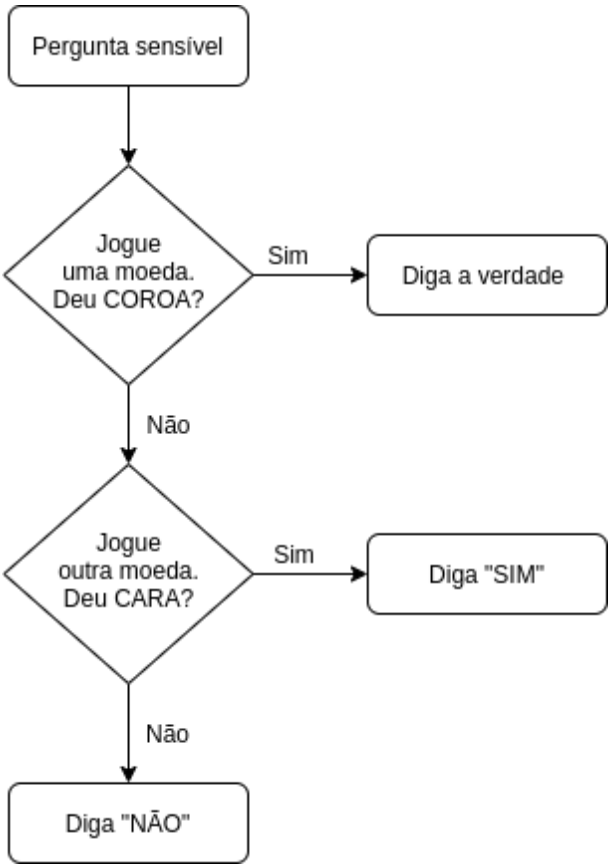

Figura 2.1: Fluxo do processo do Resposta Randômica

2.2.3 Sensibilidade Global da Consulta Δf

Antes de falarmos sobre os outros dois mecanismos utilizados no trabalho, o Laplace e o Gaussiano, precisamos explicar um conceito fundamental para que eles tenham um desempenho satisfatório com relação à utilidade dos dados de suas respostas anonimizadas, que é a sensibilidade global da consulta, que denotamos por Δf. Ela é a responsável por estabelecer a magnitude do ruído que precisa ser inserido para garantir a privacidade ϵ-diferencial. Para tal, primeiro precisamos definir o conceito de conjuntos de dados vizinhos, conforme apresentado a seguir.

Definição 2 (Conjuntos de Dados Vizinhos)**.** *Dado um conjunto de dados D, todos os conjuntos de dados D_i decorrentes da remoção de um indivíduo i do conjunto de dados original D são definidos como conjuntos vizinhos [9].*

Definição 3 (Sensibilidade Global da Consulta $\boldsymbol{\Delta f}$)**.** *Seja D o domínio de todos os conjuntos de dados. Seja f uma função de consulta que mapeia conjuntos de dados a vetores de números reais. A sensibilidade global da função f é:*

$$\Delta f = \max_{x,y \in D} ||f(x) - f(y)||_1$$

para todo x, y diferindo de no máximo um elemento, ou seja, vizinhos [9].

Onde $||.||_1$ denota a diferença com L1 Norm, sendo o Δf utilizado no Laplace [18]. Existe ainda a $||.||_2$, que é a diferença com L2 Norm, e consequentemente, a definição do $\Delta_2 f$ que é o utilizado para o Gaussiano [19]. Porém, como vai ser mais detalhado na Subseção 2.2.5, para o nosso cenário de uma dimensão o valor do $\Delta f = \Delta_2 f$ [19].

A definição de sensibilidade global da consulta Δf objetiva encontrar a menor quantidade de ruído necessária para fazer uma consulta ser diferencialmente privada. Sendo o Δf o maior impacto causado ao remover ou adicionar um indivíduo no conjunto de dados para o resultado de uma determinada consulta em todos os conjuntos vizinhos possíveis, de forma que não seja possível determinar se um indivíduo está ou não presente no conjunto de dados. Este conceito está diretamente ligado à consulta, e assim, cada consulta tem o seu próprio Δf. Quanto maior for o valor do Δf, mais ruído é necessário ser adicionado para mascarar a presença dos indivíduos. A quantidade de ruído do Δf garante que a entrada ou saída do indivíduo do conjunto de dados não vai alterar substancialmente o valor da consulta. Por exemplo, ainda que um atacante suspeite que um indivíduo está presente no conjunto de dados, e tenha uma informação externa que o indivíduo vai deixar o conjunto de dados, e realize uma consulta antes e depois da saída do indivíduo, ele não conseguirá determinar com precisão a contribuição daquele indivíduo na consulta.

2.2.4 Laplace

O mecanismo de Laplace [18] oferece privacidade diferencial para consultas que retornam valores reais (vetoriais), sendo o mais utilizado para consultas estatísticas como as do nosso trabalho [19]. Este mecanismo introduz um ruído aleatório na resposta original com base na distribuição estatística de Laplace. O ruído se baseia em dois parâmetros:

- ϵ
- sensibilidade global Δf.

Sendo o ϵ um parâmetro para definir o nível de privacidade do algoritmo. Já a sensibilidade global Δf representa o máximo de impacto que um indivíduo pode ter na resposta, sendo utilizada para garantir que a entrada ou saída de indivíduos no conjunto de dados não afete substancialmente a resposta [19]. Logo, a sensibilidade varia de acordo com o domínio da consulta.

Definição 4 (A Distribuição de Laplace)**.** *A distribuição de Laplace [19] tem média μ e escala b, sendo que na geração das nossas variáveis aleatórias utilizamos uma distribuição de Laplace centralizada em 0, ou seja, com média $\mu = 0$. A função densidade de probabilidade da distribuição de Laplace é dada por:*

$Lap(z|b) = \frac{1}{2b} exp(-\frac{|z|}{b})$.

A variância da distribuição é $\sigma^2 = 2b^2$. $Lap(b)$ denota uma distribuição de Laplace com escala b, e $X \sim Lap(b)$ será usado para denotar uma variável aleatória gerada a partir da distribuição de Laplace. A distribuição de Laplace é uma distribuição simétrica a distribuição exponencial. O mecanismo de Laplace calcula o valor da função f e adiciona um ruído aleatório que segue a distribuição de Laplace com escala $b = \frac{\Delta f}{\epsilon}$, a seguir apresentamos sua definição.

Definição 5 (O Mecanismo de Laplace)**.** *Para qualquer dada função $f : \mathbb{N}^{|x|} \rightarrow \mathbb{R}^k$*

O mecanismo de Laplace [19] é definido como:

$M_L(x, f, \epsilon) = f(x) + (Y_1 ... Y_k)$ *, onde*

$Y_i \sim Lap(\frac{\Delta f}{\epsilon})$ *, i.i.d. (variáveis aleatórias independentes e identicamente distribuídas)*

.

Portanto, a quantidade de ruído adicionado depende da sensibilidade global Δf e do parâmetro de privacidade ϵ. Quanto maior for o valor da sensibilidade, maior será a quantidade de ruído adicionada. Além disso, é válido destacar que o mecanismo de Laplace utiliza a distância de Manhattan, ou L1 Norm, no cálculo da sua sensibilidade global Δf [19]. O que será debatido com mais detalhes na sua comparação com o Gaussiano, na Seção 2.2.5.

2.2.5 Gaussiano

O mecanismo Gaussiano consiste em adicionar um ruído aleatório que segue a distribuição Gaussiana, ou também chamada de distribuição normal, sendo uma alternativa principalmente para o mecanismo de Laplace. Porém, diferentemente do mecanismo de Laplace, o mecanismo gaussiano não satisfaz a privacidade pura, também chamada de privacidade ϵ-diferencial ou (ϵ,0)diferencial, mas satisfaz a privacidade (ϵ, δ)-diferencial, onde δ é um fator de relaxamento.

Apesar de utilizar o fator relaxamento δ, Dwork e Roth no livro [19] provam que usando ruído gaussiano com variância calibrada para $\Delta f \ ln(1/\delta)/\epsilon$, pode-se obter privacidade (ϵ, δ)diferencial, além de demonstrar que os dois mecanismos se comportam de forma semelhante sobre composição em seu Teorema 3.20 - *Advanced Composition.*

Outro ponto importante a ser destacado é que o mecanismo Gaussiano utiliza L2 Norm, ou seja, na hora de calcular a sensibilidade global Δf da consulta, a diferença entre os resultados das consultas sobre conjuntos vizinhos utiliza a distância Euclidiana que é menor do que distância L1 Norm, distância de Manhattan, utilizada pelo Laplace. Isso significa que, para altas dimensões, a diferença de vetores de múltiplas dimensões terá valores menores em comparação com os do Laplace, e consequentemente obtendo um Δf menor o mecanismo Gaussiano inserirá menos ruído nesse cenário. Porém, o contexto do nosso trabalho se aplica apenas a uma dimensão e com isso é normal observar que nesse cenário o mecanismo Gaussiano tende a inserir mais ruído nos dados do que o Laplace. Dwork e Roth [19] discutem que no cenário de uma única dimensão, o Δf calculado com L1 Norm é igual ao com L2 Norm, ou seja, $\Delta f = \Delta_2(f)$. O que é bom para diminuir drasticamente o *overhead* do `DIMPLY`, visto que só precisamos calcular um único Δf para utilizar os dois mecanismos.

Definição 6 (A Distribuição Gaussiana)**.** *Sua função de densidade de probabilidade é dada por:*

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

onde, μ é a média, σ o desvio padrão e σ^2 a variância.

Definição 7 (O Mecanismo Gaussiano)**.** *Para qualquer função $d-$dimensional arbitrária, seja $f : \mathbb{N}^{|x|} \rightarrow \mathbb{R}^d$*

O mecanismo de Gaussiano [19] é definido como:

$$M_G(x, f, \epsilon) = f(x) + (Y_1 ... Y_k) \ , \ onde$$

$Y_i \sim Gau(c\Delta_2(f)/\epsilon)$, i.i.d. (variáveis aleatórias independentes e identicamente distribuídas) .

E como seguimos o estado da arte [19] do mecanismo, temos, para o nosso cálculo de privacidade (ϵ, δ)diferencial, um ruído que segue a distribuição de probabilidade acima centralizada em 0, ou seja, com $\mu = 0$. Além disso, temos um desvio padrão $\sigma \geq c\Delta_2(f)/\epsilon$ que fornece privacidade (ϵ, δ)diferencial segundo o Teorema 3.22 [19], para valores de ϵ no intervalo de [0,1], intervalo que consideramos nos experimentos do `DIMPLY`, e com valores de $c^2 > 2ln(1.25/\delta)$.

2.2.6 Utilidade dos Dados

No processo de anonimização dos dados, os mecanismos inserem ruído aleatório para preservar a privacidade dos indivíduos presentes no conjunto de dados. Essa modificação nos dados originais por sua vez implica em perda de informação e, consequentemente, diminui a utilidade dos dados. Para realizar comparações sobre os resultados das anonimizações dos mecanismos vamos utilizar a métrica do Erro Relativo, que já foi utilizada por diversos autores para mensurar utilidade dos dados [12, 45]. O Erro Relativo procura verificar o quão distantes os resultados anonimizados estão dos resultados das consultas originais, sendo definido como:

Definição 8 (Erro Relativo)**.** *Erro Relativo* $= \frac{|x-x'|}{x}$ *, onde x representa o valor original e x' o valor com ruído.*

2.2.7 Composição

O problema da composição descreve que o risco a privacidade dos indivíduos pertencentes a uma base de dados anonimizada se acumula a cada nova análise disponibilizada com os dados do indivíduo, sendo uma verdade que independe da técnica de privacidade de dados utilizada [51]. Em nosso trabalho não é diferente, ainda que em um ambiente interativo, no qual, não há a divulgação da base de dados anonimizada, divulguemos apenas respostas de agregações estatísticas anonimizadas com o uso de Privacidade Diferencial, não estamos imunes ao problema da composição.

No entanto, é válido destacar que este problema não é exclusivo de análises diferencialmente privadas. Porém, dentre as técnicas de privacidade de dados atuais a única que consegue mensurar com alto rigor matemático a perda de privacidade acumulada devido a

composição é a Privacidade Diferencial. Sendo essa, uma das características mais poderosas da privacidade diferencial, sua robustez sob composição. Através do uso do parâmetro ϵ podemos quantificar a perda de privacidade de um indivíduo em cada análise diferencialmente privada disponibilizada. Além disso, é possível verificar um limite máximo da perda de privacidade acumulada por várias análises diferencialmente privadas [51]. Suponha que duas análises diferencialmente privadas sejam divulgadas, uma calculada com $\epsilon_1 = 0{,}01$ e outra com $\epsilon_2 = 0{,}02$. O risco acumulado a privacidade de um indivíduo será de no máximo $\epsilon_1 + \epsilon_2 = 0{,}01 + 0{,}02 = 0{,}03$.

Por ser mensurável, com Privacidade Diferencial podemos apoiar a decisão de indivíduo em contribuir para a nossa pesquisa, já que é possível estimar os riscos máximos que a sua participação na pesquisa pode lhe ocasionar. Por exemplo, suponha que será feita uma análise diferencialmente privada sobre saúde de um grupo populacional de idosos. Um idoso deseja participar do estudo, porém está com medo da seguradora ter acesso aos resultados do estudo, e com isso o pagamento mensal do seu prêmio de vida aumentar. Com Privacidade Diferencial é possível mensurar, uma vez que o idoso decida participar do estudo, o valor máximo que a seguradora aumentará no valor do seu pagamento. E assim o idoso pode decidir se vai participar ou não com uma informação clara. É importante destacar que, ainda que o idoso não participe da pesquisa, o seu prêmio pode aumentar do mesmo jeito. A seguradora pode concluir que por ele pertencer a mesma faixa etária do estudo ele tem os mesmos riscos. Outro ponto a se destacar, é que não necessariamente a seguradora vai aumentar caso ele participe, o que a Privacidade Diferencial garante é que caso ele participe do estudo o máximo valor que a seguradora conseguiria aprender com a sua participação [51].

Concluímos que apesar de não ser imune a lei fundamental que o risco de privacidade aumenta quando múltiplas análises são realizadas nos dados do mesmo indivíduo, a privacidade diferencial consegue oferecer garantias que esse risco vai se acumular de forma limitada. Independente da quantidade de combinações de análises diferencialmente privadas a perda de privacidade nunca vai ser desproporcional ao risco de privacidade associado a cada uma das análises isoladamente [51].

2.2.8 O *Privacy Budget*

Conhecido na literatura como "*privacy budget*", nada mais é que o orçamento máximo de perda de privacidade, sendo uma das estratégias para lidar com o problema da composição descrito na Subseção 2.2.7. Pode ser definido como uma determinação da proteção geral da

privacidade fornecida por um análise diferencialmente privada. Intuitivamente, determina o "quanto" da privacidade de um indivíduo a análise pode utilizar, ou seja, o quanto de risco a privacidade de um indivíduo pode aumentar quando ele decide participar da análise. Conceitualmente, valores baixos implicam em maior proteção da privacidade, ou seja, menos risco para a privacidade dos indivíduos presentes na base de dados analisada. Por outro lado, valores altos implicam em menos proteção da privacidade, ou seja, um maior risco a privacidade dos indivíduos presentes. Em particular, $\epsilon = 0$ implica em perfeita garantia de privacidade, ou seja, não aumenta o risco a privacidade de nenhum indivíduo presente. Infelizmente, análises que satisfaçam a privacidade diferencial com $\epsilon = 0$ vão ignorar completamente seus dados de entrada e portanto são inúteis [51].

Felizmente, uma série de teoremas de composição foram desenvolvidos para privacidade diferencial [19]. Estes em particular, afirmam que a composição de duas análises diferencialmente privadas resultam em uma perda de privacidade que é limitada pela soma das perdas de privacidade de cada uma das análises. Por exemplo, suponha o cenário que um especialista precisa configurar o `DIMPLY` para atender a uma legislação vigente que define que o dono dos dados pode ter uma perda de privacidade de no máximo 1,0, ou seja, seu *privacy budget* = 1,0. Esse especialista sabe que segundo os teoremas de como a Privacidade Diferencial se comporta sobre composição [19] a perda de privacidade a cada nova consulta se somam, ou seja, se a consulta 1 foi anonimizada com $\epsilon_1 = 0{,}01$ e a consulta 2 tem $\epsilon_2 = 0{,}02$, ao disponibilizá-las ele terá uma perda de privacidade de no máximo $\epsilon 1 + \epsilon 2 = 0{,}03$, que é menor que o seu *privacy budget*. Assim, ele pode definir quais valores de ϵ vai utilizar para cada consulta, podendo balancear o *tradeoff* entre utilidade e privacidade, e bloquear novas consultas ao sistema quando consumir todo o seu *privacy budget*.

2.3 Sistemas *Polystore*

É um fato bem conhecido que as organizações possuem dados em diferentes formatos, e para consultá-los de forma eficiente é despendido um esforço para mapear esses dados em *schemas* bem definidos, independente se seguem o modelo relacional ou não. Além disso, durante este processo de mapeamento dos dados, alguns dados podem ser perdidos para se adequar ao *schema* estabelecido. Outro problema é quando precisamos realizar consultas que cruzem informações armazenadas em diferentes *schemas* ou Sistemas de Gerência de Banco de Dados (SGBDs), o que pode ter uma latência muito alta ou mesmo nem ser possível em alguns casos.

Para resolver esse problema de integração de dados onde os dados se encontram armazenados seguindo diferentes modelos e SGBDs, os sistemas *Polystore* foram propostos. Os sistemas *Polystore* tem como premissa básica a ideia do "*One size does not fit all*", onde assume-se que existem SGBDs que atendem melhor determinado tipo de dado (ou consulta) e idealmente o dado deve ser armazenado nesse SGBD ou modelo [20].

Um sistema de banco de dados *Polystore* pode ser definido como qualquer sistema capaz de manter dados de múltiplos SGBDs que seguem modelos heterogêneos, sejam eles relacionais, orientados à coluna, orientado à grafos, chave-valor, *etc.* Ao mesmo tempo, os sistemas *Polystore* são capazes de consultar esses dados de forma integrada por meio de uma interface única e com alto desempenho. É importante distinguir sistemas *Polystore* de SGBDs federados, que também integram diferentes SGBDs e possibilitam consultas com uma interface única, porém estes somente integram bases de dados com modelos homogêneos, *e.g.*, todos seguindo o modelo relacional.

Nos sistemas *Polystore*, os dados podem ser consultados a partir de formatos otimizados ou mesmo serem consultados em seus formatos originais. Estes sistemas permitem realizar consultas em *datasets* heterogêneos, com uma linguagem de consulta única e sem a necessidade de definição de um *schema* padrão para os dados, o que é ótimo para ser usado em conjunto com a tendência de se armazenar todos os dados em seus formatos originais em um *Data Lake* [40]. Os *Data Lakes* são sistemas baseados no conceito *schema-on-read*, de forma que os arquivos são carregados sem associar nenhum *schema* a eles, e só no momento da consulta é definida a sua estrutura a partir dos seus metadados armazenados.

2.3.1 BigDAWG

O BigDAWG é um sistema *Polystore* criado pelo Centro de Ciência e Tecnologia da Intel (ISTC) para explorar os desafios associados com a construção de banco de dados federados sobre múltiplos modelos de dados, com mecanismos de armazenamento especializados e com foco em visualização para *Big Data* [15].

Atualmente existem diversos SGBDs, cada qual com linguagens de consulta e mecanismos de armazenamento diferentes, ou seja, além dos já consolidados bancos de dados relacionais, existem hoje diversas soluções NoSQL com modelos de dados diferentes e diferentes formas de consultar. Cada uma dessas soluções é mais adequada para um determinado contexto de aplicação da organização, e o que se tem feito é utilizar múltiplas soluções. Partindo-se dessa premissa, temos a consequente necessidade de gerenciar dados

em múltiplos modelos.

O BigDAWG se propõe a resolver essas questões mencionadas anteriormente, ou seja, unificar consultas sobre vários modelos de dados e SGBDs, com três objetivos: *Location Transparency* - evitar gargalos nas consultas a diferentes sistemas, *Semantic Completeness* - garantir que perdas não ocorram com a adoção do *Polystore* e possibilitar que usuários acessem objetos armazenados de um único SGBD com múltiplas *Ilhas* [15].

Uma *ilha* é a definição de um *modelo de dados* e uma *linguagem de consulta* que representa um *tipo de dado*. Por exemplo, você pode usar SQL para submeter consultas para a ilha *relacional* porque é a *linguagem de consulta* definida por ela. Como o *modelo de dados* e a *linguagem de consulta* definidos por uma *ilha* podem ser diferentes daqueles implementados em um SGBD subjacente, [15] introduz o conceito de *shims*.

O operador *shim* é responsável por traduzir o *modelo de dados* e as construções de consulta definidas por uma *ilha* para o *modelo* e as construções suportadas pelo SGBD subjacente. Além disso, os *shims* podem navegar por diferentes *ilhas*, *i.e.*, os usuários podem recuperar dados usando construtos de consulta de *ilhas* diferentes do banco de dados que pertence.

A arquitetura do BigDAWG, representada na Figura 2.2, utiliza fortemente os conceitos de *Ilha*, *Consulta Inter Ilha* e os *Shim*.

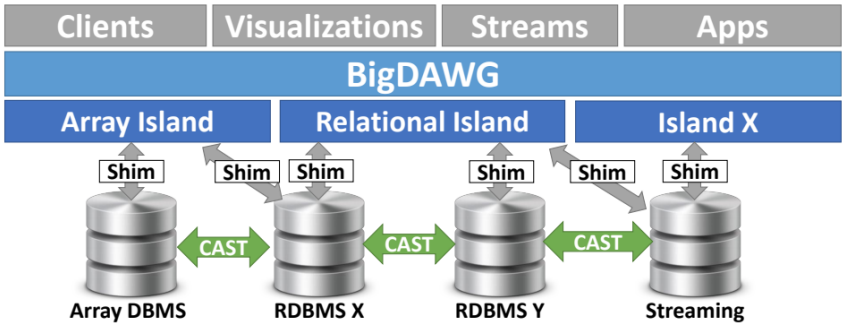

Figura 2.2: Arquitetura do BigDAWG. Figura extraída do artigo original [15].

Em especial, no BigDAWG, as *Ilhas* são não somente SGBD, mas também sistemas de armazenamentos que possuem uma mesma linguagem de consulta (*e.g.*, MapReduce, Thrift [46], SQL, *etc*), o mesmo modelo de dados (*Array*, Colunar, Relacional, *etc*). Existe ainda, um sistema de gerência do conjunto de dados que é responsável por executar as consultas escritas nas respectivas *Ilhas* [15].

O processo de execução de uma consulta é realizado da seguinte forma: o sistema recebe a consulta, cria uma AST (Árvore de Sintaxe Abstrata), faz otimizações dividindo

as AST em outras sub-consultas menores, encaminha para os *Shims* que são responsáveis por fazer a tradução da consulta para a linguagem da determinada *Ilha*, e após isso as consultas são realizadas e os resultados são acumulados [15].

As *Consultas Inter Ilha* são componentes que suportam o conceito de *location transparency* com o uso de um *Shim* para cada sistema de armazenamento, ou seja, os objetos são copiados entre os diferentes SGBDs subjacentes, e esse processo é chamado de CAST [15]. Para otimizar as consultas no BigDAWG, são utilizadas três abordagens de planejamento de execução, *Single Island Planning*, *Workload Monitoring* e *Multi-Island Planning*.

O *Single Island Planning* é utilizado para as consultas que executam somente em uma *Ilha*, sendo os passos de otimização da consulta divididos em:

- Diminuição dos dados que trafegam entre diferentes sistemas de armazenamento (mas ainda dentro de uma mesma *Information Island*), a fim de evitar custos de conversões de dados e tráfegos desnecessários de rede. A ideia é computar os dados o máximo possível dentro de um mesmo sistema de armazenamento, antes de se comunicar com outros sistemas daquela mesma *Ilha* [15].
- No caso de consultas complexas, as otimizações buscam realocar os dados para os SGBDs onde é possível um alto desempenho, de modo que o custo da realocação compense em relação ao tempo gasto na execução da consulta [15].

Na otimização por *Workload Monitoring*, para adquirir rapidamente as informações, existem três modos de operação: *Training*, *Optimized* e *Opportunistic*. No modo *Training*, o otimizador tem o perfil abrangente e tem acesso tanto a AST, mas também a uma lista de preferência para determinar o mecanismo mais adequado para cada sub-consulta. Assim novas sub-consultas correspondentes às características desta sub-consulta não vão precisar de experimentação adicional. No *Optimized* o sistema verifica se a sub-consulta já está na lista de preferências criada pelo *Training* e executa segundo o que está definido nela. Caso não esteja, é executado o *Training* para essa sub-consulta de modo a ser uma nova entrada na lista. E por fim, no modo *Opportunistic*, a ideia é tentar prever qual sub-consulta será executada novamente, e armazená-la como se fosse em um *cache*, de modo que no momento de uma nova submissão ela será executada mais rapidamente [15].

Concluindo as otimizações, temos o *Multi-Island Planning*, ou seja, para consultas que envolvam diferentes *Ilhas*, ele busca identificar erros de semântica e locais onde a consulta pode ser convertida para a otimização tradicional. Além disso, ele identifica áreas de

interseção entre vários *Shims* associados ao mesmo SGBD para serem executados em conjunto, e procura por equivalências de gramática entre diferentes *Ilhas* [15].

Além de otimizações de consultas, existe um conceito de *Data Placement* implementado nos sistemas de bancos de dados *Polystore*, que garante, por exemplo, que ao mover dados não seja necessário mudar a lógica das aplicações que o utilizam. Em resumo, nesse conceito de *Data Placement*, as características desejáveis são a transparência de sistemas distribuídos, como por exemplo, a transparência de migração [15].

2.3.2 Apache Drill

O Apache Drill[1] pode ser considerado uma solução que possibilita a integração de variadas fontes de dados acessíveis da organização, e também possibilita explorar *schemas* de fontes sob demanda. Podemos utilizar o Apache Drill como um *Middleware* para se consultar de forma integrada diferentes SGBDs relacionais ou não-relacionais, sistemas de arquivos distribuídos ou não, estejam eles em uma máquina local ou na nuvem, além de alguns formatos de arquivos específicos como CSV, JSON e Parquet[2].

O Apache Drill é capaz de processar os dados no local (*in-situ*) sem exigir que os usuários definam *schemas* ou transformem os dados. Ele possibilita aos usuários explorar e analisar esses dados sem sacrificar a flexibilidade e agilidade oferecidas por cada mecanismo de armazenamento ou SGBD subjacente. Com o Drill, é possível analisar dados estruturados ou semi-estruturados em larga escala, armazenados de forma distribuída. Algumas das principais vantagens da ferramenta é a redução da latência nas respostas das consultas.

Para garantir um bom desempenho no processamento das consultas, o Apache Drill faz uso de meios de processamento em memória principal. O serviço *Drillbit* é o responsável por aceitar requisições, processá-las e retornar os resultados. Quando um serviço *Drillbit* é executado, a ferramenta maximiza a localidade dos dados durante a execução, e assim evita a necessidade de transportar dados na rede ou movê-los entre os nós, desta forma a análise é feita localmente poupando esforços de recursos de banda, por exemplo. Para realizar o processamento distribuído e verificar a integridade do *cluster*, o Apache Drill deve ser configurado com o uso do ZooKepeer[3] [4]. Embora projetado para ser distribuído,

[1] https://drill.apache.org/

[2] O Apache Parquet é um formato de armazenamento colunar disponível para qualquer projeto no ecossistema Hadoop, independentemente da escolha da estrutura de processamento de dados, modelo de dados ou linguagem de programação.

[3] https://zookeeper.apache.org/

o Apache Drill também pode ser configurado em uma única máquina, para isso devemos utilizar o seu modo *Embedded* [26], sendo este o modo utilizado em nossos experimentos.

O Apache Drill é uma solução de código aberto desenvolvida em [26]Java que foi baseada no Google Dremel [39]. Apesar de compatível com SGBDs convencionais, o principal foco do Apache Drill é possibilitar acesso e consultas integradas a bancos de dados não relacionais, além do *stack* Hadoop e armazenamentos em nuvem. Atualmente são suportados armazenamentos dos seguintes *softwares* do *stack* Hadoop: Apache Hadoop[4], MapR[5], CDH - Cloudera Distributed Hadoop[6] e Amazon EMR[7]. Além disso, o Drill oferece apoio para os seguintes SGBDs NoSQL: MongoDB[8] e HBase[9]. E por fim nos armazenamentos em nuvem: Amazon S3[10], Google Cloud Storage[11], Azure Blob Storage[12] e OpenStack Swift[13]. A Figura 2.3 apresenta a arquitetura de cada *Drillbit* sendo cada módulo[14] explicado individualmente a seguir [3].

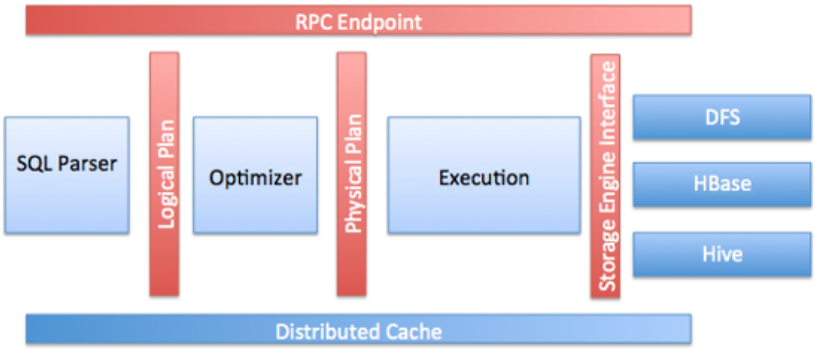

Figura 2.3: Arquitetura do Apache Drill. Figura extraída do site oficial [4][15]

1. *RPC Endpoint*: O Apache Drill utiliza um protocolo RPC para se comunicar com clientes com baixa sobrecarga. Além disso, também é possível se comunicar via APIs C++ ou Java. Os clientes podem se comunicar com um *Drillbit* específico ou descobrir um conjunto de *Drillbits* disponíveis por meio do ZooKeeper. É recomendado utilizar a segunda opção por realizar todo o processo de forma transparente.

[4]https://hadoop.apache.org/
[5]https://mapr.com/try-mapr/
[6]https://www.cloudera.com/products/open-source/apache-hadoop/hdfs-mapreduce-yarn.html
[7]https://aws.amazon.com/pt/emr/
[8]https://www.mongodb.com/
[9]https://hbase.apache.org/
[10]https://aws.amazon.com/pt/s3/
[11]https://cloud.google.com/storage
[12]https://azure.microsoft.com/pt-br/services/storage/blobs/
[13]https://www.swiftstack.com/product/open-source/openstack-swift
[14]https://drill.apache.org/docs/core-modules/

2. *SQL Parser*: O Apache Drill utiliza o Apache Calcite[16] para transformar as consultas recebidas em um plano lógico que representa a consulta. Esse plano lógico é independente da linguagem que foi escrita a consulta.

3. *Optimizer*: A otimização é realizada com base em regras, custo, localidade dos dados e outros itens. A saída do otimizador é o plano físico da consulta a ser realizada de forma distribuída.

4. *Execution Engine*: O Apache Drill utiliza um mecanismo de execução MPP (*Massive Parallel Processing*) para processar as consultas distribuídas entre os nós.

5. *Storage Plugin Interfaces*: Os *plugins* de armazenamento representam abstrações que o Apache Drill utiliza para se comunicar com as bases de dados. Podendo conter:

   - Metadados disponíveis na fonte;
   - Localização dos dados e um conjunto de regras de otimização para execução eficiente;
   - Interface para ler e gravar dados.

6. *Distributed Cache*: Utilizado para gerenciar metadados e informações de configurações em vários nós do *cluster*.

Quando um cliente ou aplicativo submete uma consulta ao Apache Drill, ela é recebida por um nó *Drillbit* do *cluster*, esse nó passa a ser chamado de *Foreman*. A partir de então, este nó fica encarregado pela condução da consulta inteira, ou seja, o *Foreman* tem a responsabilidade de coordenar, planejar e distribuir a consulta no *cluster* maximizando a localidade dos dados. Essa arquitetura de comunicação entre clientes, aplicativos e *Drillbits* descrita anteriormente é exibida na Figura 2.4[17].

[16] https://calcite.incubator.apache.org/
[17] https://drill.apache.org/docs/drill-query-execution/

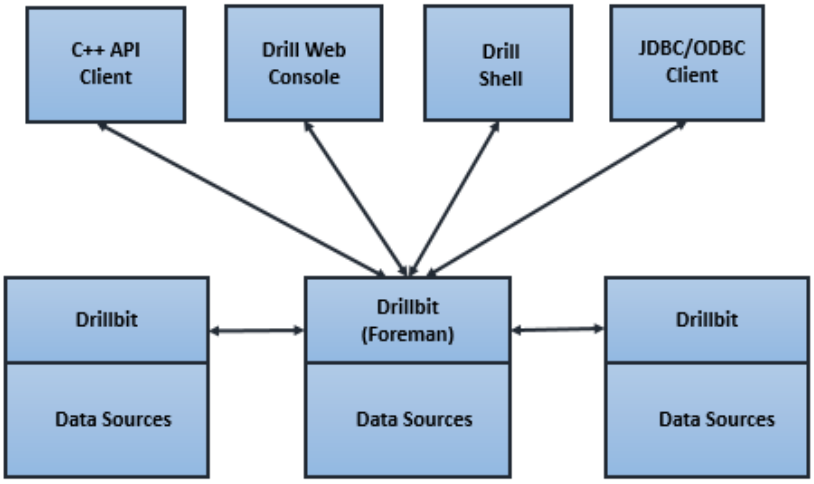

Figura 2.4: Comunicação entre clientes, aplicativos e *Drillbits* no Apache Drill. Figura extraída do site oficial [1][18]

O processo de execução da consulta internamente no *Foreman* acontece conforme a Figura 2.5. Nesse processo o *Foreman* transforma a consulta SQL recebida em um plano lógico e envia o plano lógico para o otimizador. Então, o otimizador lê o plano lógico e aplica vários tipos de regras para reorganizar os operadores e funções da consulta em um plano ideal baseado nos custos. Por fim, o otimizador converte o plano lógico em um plano físico que descreve como executar a consulta de forma mais otimizada.

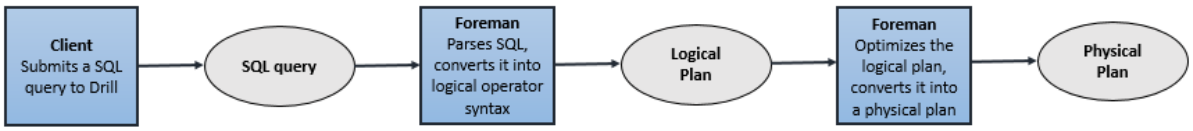

Figura 2.5: Processo de execução da consulta internamente no *Foreman*. Figura extraída do site oficial [1][19]

Concluindo os passos de execução de uma consulta, temos o paralelizador do *Foreman*. O paralelizador é responsável por transformar o plano físico em várias fases, que são chamadas de fragmentos maiores e menores. Esses fragmentos juntos criam uma árvore de execução multinível que reescreve a consulta e a executa em paralelo nas fontes de dados configuradas. Ao término, o resultado é enviado de volta ao cliente ou aplicativo. Esse processo descrito anteriormente é exibido na Figura 2.6.

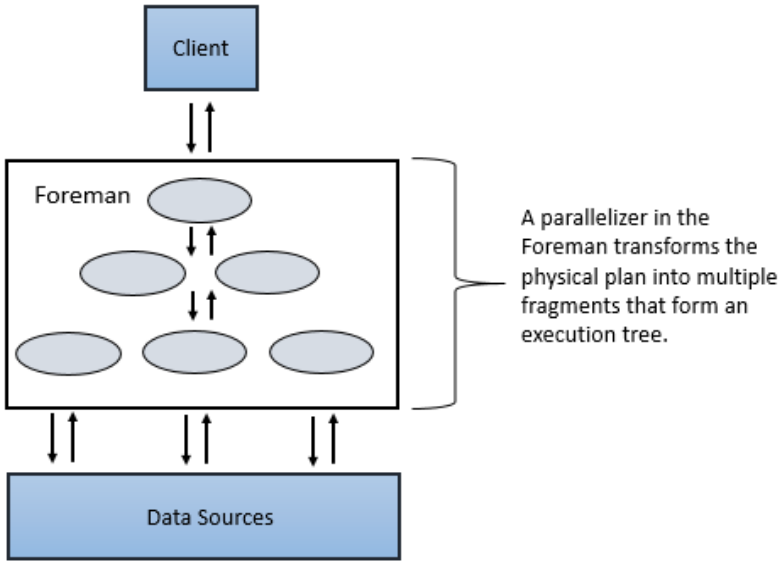

Figura 2.6: Transformação do plano físico em fragmentos que formam a árvore de execução da consulta. Figura extraída do site oficial [1][20]

2.4 Considerações Finais

Neste capítulo apresentamos os conceitos básicos necessários para o entendimento do trabalho, como as definições de privacidade de dados; da técnica de Privacidade Diferencial, dos mecanismos de Privacidade Diferencial utilizados no `DIMPLY` e seus parâmetros; do conceito da lei da composição; e do *privacy budget*. Além disso, foi apresentada a definição de sistemas *Polystore*, as implementações existentes para esse conceito, sendo o Apache Drill a escolhida para esta versão do `DIMPLY`. Sendo importante destacar que os sistemas *Polystores* são projetos ainda em desenvolvimento, e logo possuem limitações e um certo grau de dificuldade em sua instalação e configuração.

Capítulo 3

Trabalhos Relacionados

Até o momento não foram encontrados na literatura trabalhos que abordem o uso de Privacidade Diferencial em conjunto com Sistemas *Polystore*. Então, analisamos trabalhos que resolveram questões similares com as da nossa pesquisa, ainda que não aplicados no mesmo contexto. Assim, a Seção 3.1 descreve trabalhos que anonimizam consultas SQL com Privacidade Diferencial. Na Seção 3.2 apresentamos um trabalho que busca fazer a escolha do melhor mecanismo de Privacidade Diferencial para anonimizar uma consulta submetida de acordo com alguns critérios.

3.1 Anonimização de Consultas com Privacidade Diferencial

3.1.1 PINQ e wPINQ

McSherry [38] propõe o PINQ (*Privacy Integrated Queries*), uma implementação baseada no LINQ do C#, que é uma extensão integrada ao .NET. Apesar de não serem exatamente consultas na linguagem SQL, o LINQ é uma linguagem declarativa, que se assemelha muito ao SQL. O PINQ é uma plataforma de consultas integradas para a preservação da privacidade nas tarefas de análise de dados. Ele fornece aos analistas uma interface de acesso a dados não criptografados, ao mesmo tempo, que garante a privacidade de dados. Na plataforma do PINQ, o dono dos danos pode estabelecer e controlar definições de privacidade individualmente para cada analista, por meio do PINQAgent, fixando um valor de *privacy budget*. Antes da execução de cada agregação, o PINQ invoca o método *Alert* do PINQAgent associando a este o valor do ϵ, que verifica se o usuário pode realizar a consulta com aquele valor de ϵ. Se for permitido, a resposta da consulta é anonimizada

com o mecanismo de Laplace.

A plataforma do PINQ controla as requisições verificando se o analista tem *privacy budget* suficiente para executar a sua requisição com aquele valor de ϵ. Um analista que usa PINQ não tem certeza se sua requisição vai ser aceita ou rejeitada, ele deve esperar que os PINQAgents subjacentes aceitem todas as suas solicitações de acesso. Para resolver isso, existe uma outra forma de consultar, um método que o analista tenta "alocar" uma quantidade de *privacy budget* previamente. Se sua solicitação for aprovada, um novo PINQueryable com este *budget* é conectado ao seu agente de consultas. Ao terminar suas consultas, e seu agente for destruído, é liberada a quantidade do *budget* não utilizada [38].

Em sua implementação, os donos dos dados são capazes de usar o PINQ para agrupar dados LINQ de diferentes fontes atribuindo uma especificação de privacidade para cada analista. Assim, os analistas podem escrever seus códigos em C# para acessar fontes de dados compatíveis com o LINQ, e de forma transparente o PINQ faz a ponte entre essas consultas e as fontes de dados, uma camada fina de anonimização com Privacidade Diferencial. Ao mesmo tempo, o *design* da linguagem de análise do PINQ e sua implementação cuidadosa fornece garantias formais de Privacidade Diferencial para todo e qualquer uso da plataforma [38].

As garantias estruturais incondicionais do PINQ fazem com que não seja necessário ter confiança nos analistas para disponibilizar o acesso aos dados e nem que eles entendam de privacidade de dados. Eles podem ser programadores comuns, e a plataforma do PINQ (configurada pelo dono dos dados) se encarrega de controlar todas as requisições e as garantias de privacidade para cada analista individualmente, o que aumenta consideravelmente o escopo de aplicação do PINQ. O PINQ fornece uma fina camada protetora em frente às fontes de dados existentes, apresentando uma interface que parece ser a dos próprios dados brutos, ou seja, quase como acessar as fontes com o LINQ somente. As restrições de linguagem do PINQ e suas verificações em tempo de execução garantem que os requisitos de privacidade diferencial estabelecidos pelo do dono dos dados são respeitados [38].

Além de permitir que os programadores escrevam códigos arbitrariamente, e ainda assim, garantir que estes códigos sejam diferencialmente privados, como PINQ mantém os recursos do LINQ, é possível também utilizar o conceito de herança, e reutilizar códigos privados para gerar novos códigos baseados no conceito privado do pai. Isso expande o conjunto de usuários capazes de acessar os dados confidenciais, aumenta a portabilidade de algoritmos de preservação de privacidade e o escopo das análises de dados sensíveis [38].

Existe ainda uma extensão proposta por Proserpio *et al.* em 2014, o wPINQ (*Weighted PINQ*) [44] que estende o PINQ com apoio para consultas com junções do tipo *equijoin* (*i.e.*, junções entre tabelas que utilizam apenas o operador de igualdade - que o PINQ não apoiava). O wPINQ funciona atribuindo um peso a cada tupla no banco de dados e, em seguida, reduz os pesos das linhas em juntando-as para garantir uma sensibilidade geral de 1. Com os wPINQ, o resultado de uma consulta de contagem é a soma dos pesos dos registros contados mais o ruído retirado da distribuição de Laplace. Essa abordagem permite que o wPINQ apoie todos os três tipos de junções.

Comparando o PINQ e o wPINQ com o a `DIMPLY`, a principal diferença é que eles não executam consultas utilizando sistemas *Polystore*. Ambos acessam as fontes de dados apoiadas pelo LINQ. O PINQ apresenta novos operadores que não existem no SQL padrão, então a abordagem não é compatível com os SGBDs relacionais tradicionais. Outra diferença é que o PINQ e o wPINQ oferecem apoio a anonimização de consultas de contagem, enquanto o `DIMPLY` somente para as consultas de soma, média, desvio padrão, variância, mínimo e máximo. O PINQ e wPINQ anonimizam apenas com o mecanismo de Laplace, enquanto o `DIMPLY` oferece atualmente três opções de mecanismo diferentes. O PINQ e wPINQ não realizam o cálculo da sensibilidade global Δf da consulta, a magnitude do ruído deve ser configurada pelo dono dos dados. Como eles anonimizam apenas com um mecanismo, consequentemente outra diferença é que eles não se propõem a escolher um mecanismo ideal para anonimizar cada consulta.

3.1.2 FLEX

Noah Johnson *et al.* [30] conduzem o maior estudo empírico conhecido sobre consultas reais na sintaxe SQL, utilizando um base de dados com 8,1 milhões de consultas no total. Os autores expõem os resultados de suas análises sobre essas consultas, como quais o tipos de junção mais frequentes, quais as funções de agregação mais utilizadas, a fração das consultas estatísticas em relação as consultas a dados brutos. A partir dessas análises eles concluem que o tipo de junção mais utilizado em 76% das consultas é o *equijoin*, junções entre tabelas que utilizam apenas o operador de igualdade, que a função de agregação mais utilizada é a de contagem com 51% e que 34% das consultas da base são consultas estatísticas com agregação. Diante disso, eles se propõem a desenvolver uma solução que seja capaz de anonimizar consultas de contagem com *equijoin*, porque 34% é uma porcentagem alta das consultas da base que serão beneficiadas pela anonimização com privacidade diferencial. Diante destes resultados, os autores mostram que as consultas

usadas em trabalhos anteriores para avaliar os mecanismos diferencialmente privados não são representativas para consultas do mundo real e propõem novos requisitos para uma privacidade diferencial prática.

Para conseguir atender aos novos requisitos de privacidade diferencial prática propostos no trabalho, os autores propõem a sensibilidade elástica, uma aproximação do conceito de sensibilidade local da consulta [17, 43] que suporta consultas com junções do tipo *equijoin* mais comuns e que pode ser calculado de forma eficiente usando apenas a própria consulta, e um conjunto de métricas do banco de dados pré-computadas. Os autores provam que a sensibilidade elástica é um limite superior da sensibilidade local e que pode, portanto, ser usada para reforçar a privacidade diferencial para qualquer mecanismo baseado em sensibilidade local da consulta.

Os autores implementam o FLEX, um sistema de privacidade diferencial ponta a ponta para consultas SQL com base na *sensibilidade elástica* definida por eles. A arquitetura do FLEX pode ser vista na Figura 3.1. O sistema anonimiza com o mecanismo proposto FLEX que é uma implementação diferencialmente privada baseada no mecanismo de Laplace proposto por Nissim *et al.* [43] que utiliza a sensibilidade *Smooth* da consulta, que por sua vez, é baseado em sensibilidade local da consulta. Os autores provam que o mecanismo FLEX garante privacidade (ϵ, δ)diferencial.

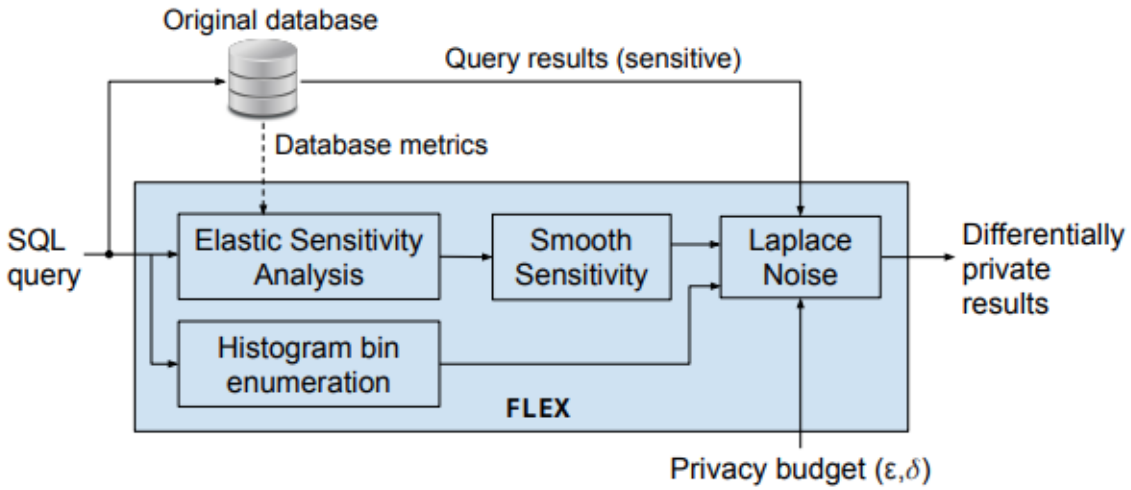

Figura 3.1: Arquitetura do FLEX. Figura extraída do artigo original [30].

O FLEX em seus experimentos demonstra que é compatível com todos os SGBDs que apoiam SQL existentes. Porém, não é compatível com bancos de dados NoSQL, e sistemas de arquivos locais ou em nuvem. Sendo este o principal diferencial entre o FLEX e o `DIMPLY`, que por ser desenvolvido com o uso de um sistema *Polystore* permite

consultas anonimizadas que integram esses diferentes tipos de armazenamentos e modelos heterogêneos.

Além da diferença principal, existem outras diferenças entre as abordagem FLEX e `DIMPLY`. O FLEX utiliza uma aproximação da sensibilidade local da consulta [17, 43], a sensibilidade elástica, para dimensionar a magnitude do ruído do mecanismo de Laplace para uma dada consulta, enquanto que o `DIMPLY` utiliza a sensibilidade global da consulta, o Δf. Outro ponto, é que o FLEX utiliza apenas o mecanismo de Laplace para anonimização, enquanto o `DIMPLY` oferece atualmente três opções de mecanismos e também se propõe a escolher um mecanismo mais adequado para novas consultas. O FLEX suporta consultas estatísticas com agregações de contagem e histogramas, enquanto o `DIMPLY` suporta consultas de estatísticas com as agregações de soma, média, desvio padrão, variância, mínimo e máximo. Por fim, nos experimentos do FLEX os autores relatam que ele não lida tão bem com conjuntos de dados pequenos, inserindo muito ruído nos dados, isso se deve ao fato de como funciona a sensibilidade local da consulta que se baseia no conjunto de dados original para dimensionar o ruído, como o `DIMPLY` utiliza a sensibilidade global da consulta Δf, o `DIMPLY` não é tão impactado igual o FLEX para conjuntos pequenos.

3.1.3 CHORUS

Noah Johson *et al.* [29] apontam que muito se avançou na parte teórica de Privacidade Diferencial nos últimos anos, mas seu uso prático ainda é um desafio em aberto. De acordo com os autores, isso acontece devido aos protótipos das pesquisas não satisfazerem aos requisitos de escalabilidade para implantações em produção.

Diante deste cenário, os autores propõe o CHORUS um *framework* para a construção de mecanismos diferencialmente privados escaláveis que se baseia na forte integração entre o próprio mecanismo e um SGBD de alto desempenho. O CHORUS apoia a integração de dados em qualquer SGBD que ofereça suporte a SQL padrão.

No trabalho é demonstrado o uso de CHORUS para construir as primeiras implementações altamente escaláveis de mecanismos complexos como o *Weighted PINQ* [44], o MWEM [25] e o Matrix [35]. Os autores também relatam a sua experiência no processo de implantação do CHORUS na Uber e avaliam a sua escalabilidade para consultas do mundo real.

Os mecanismos no CHORUS são funções na linguagem Scala, e os analistas podem

definir novos mecanismos no *framework*. A biblioteca de pós-processamento CHORUS disponibilizada fornece uma série de utilitários, incluindo o mecanismo de Laplace, o mecanismo Gaussiano, o mecanismo Exponencial e várias formas de *clipping*. A estrutura do CHORUS visa apoiar a implementação de mecanismos de forma cooperativa, sendo os seus principais componentes: *rewriting* - que modifica a consulta para executar as funções como *clipping*; *analysis* - que analisa as consultas para determinar propriedades necessárias, como a quantidade de ruído necessária para garantir a Privacidade Diferencial; e *post-processing* para processar os resultados das consultas em execução.

A API do CHORUS fornece duas interfaces para contabilizar o *privacy budget*: o *PrivacyCost*, para representar os custos de privacidade, e a *PrivacyAccountant*, para monitorar o custo total de vários mecanismos sobre composição. É possível configurar a auditoria a uma pessoa confiável responsável por estabelecer o *privacy budget*, e então bloquear a execução de consultas após o *budget* se esgotar.

O CHORUS também permite a implementação de seleção automática de mecanismo (como "mecanismos" que usam propriedades da consulta para selecionar de uma lista de mecanismos a serem executados). Por exemplo, eles implementam um mecanismo com CHORUS que executa cada um dos três mecanismos individuais usando uma simples abordagem baseada em regras.

Em comparação com o `DIMPLY`, a principal diferença é que o CHORUS é fortemente atrelado a um SGBD, de forma que ele ganha em desempenho e escalabilidade, mas perde em flexibilidade para o `DIMPLY` que utiliza um sistema *Polystore* que permite consultas integradas em diferentes SGBDs, sistemas de arquivos em nuvem ou local, e SGBDs NoSQL.

O CHORUS permite programar escolhas automáticas dentre os mecanismos disponíveis baseados em informações da consulta, mas não propõe nenhum Modelo de Custo para escolha do mecanismo ideal como feito pelo `DIMPLY`. Além disso, a informação da consulta que ele sugere armazenada no *PrivacyCost* seria apenas o custo de privacidade da consulta, ou seja, o ϵ, não mantendo outras informações relevantes para escolha de um mecanismo. No `DIMPLY`, além do ϵ, armazenamos o histórico das consultas executadas com tempo de execução decorrido na anonimização com cada mecanismo disponíveis e o Erro Relativo da consulta para cada função de agregação da consulta submetida. Dessa forma, o analista pode configurar pesos no Modelo de Custo proposto, priorizando entre um menor tempo de execução ou um menor Erro Relativo.

3.1.4 *Shrinkwrap*

Bater *et al.* [6] propõem o *Shrinkwrap*, um algoritmo de ponta a ponta para responder a uma consulta SQL em uma federação de dados privados. Uma federação de dados privados é um conjunto de bancos de dados autônomos que compartilham uma interface de consulta unificada. Por questões de privacidade, esses sistemas não tem um coletor de dados confiável que possa acessar todos os dados e aprender sobre eles. Abordagens anteriores, conseguem garantir isso por meio de computação multipartidária segura, que oculta a cardinalidade de resultado intermediário de cada operador da consulta e vai preenchendo-a exaustivamente.

Assim, os autores propõem um mecanismo diferencialmente privado que revela a cardinalidade intermediária de cada operador. O aumento exponencial na cardinalidade de saída requer significativamente mais acessos de E/S, o que causa a grande desaceleração no desempenho. O *Shrinkwrap* aplica privacidade diferencial para reduzir o tamanho da saída em cada junção, reduzindo a magnitude de cada saída de junção. Embora o *Shrinkwrap* ainda veja um significativo crescimento no tempo de execução das funções de contagem com junções, o desempenho geral é ordens de grandeza menor do que outras estratégias de privacidade para bancos de dados federados.

Em sua essência, o *Shrinkwrap* é um sistema que aplica Privacidade Diferencial em toda a execução de uma consulta SQL para reduzir os tamanhos de resultados intermediários e melhorar o desempenho geral. Para tal, ele utiliza sua própria versão do mecanismo de Laplace, o *Truncated Laplace Mechanism*, que utiliza uma privacidade relaxada por δ. O *Shrinkwrap* é capaz de prover privacidade (ϵ, δ)diferencial para consultas de contagem e *distinct* com junções sobre bancos de dados federados.

Os autores também propõem três estratégias de *baseline* para alocar o *budget* de privacidade para operadores individuais na árvore de execução da consulta. *Eager*: Esta abordagem aloca todo o *budget* para o primeiro operador na árvore de comando. *Uniforme*: a segunda abordagem divide o *budget* de privacidade uniformemente na árvore de comando, resultando em parâmetros de privacidade iguais para cada operador. *Modelo de Custo*: a terceira abordagem usa um modelo de custo de execução como uma função objetivo e aplica a otimização convexa para determinar a estratégia de divisão do *budget* ideal.

3.1.5 APEx

Ge *et al.* [23] apresentam uma forma inversa de se pensar sistemas que possibilitam consultar de forma diferencialmente privada conjuntos de dados. Para contextualizar, reflita sobre a carga excessiva que os sistemas atuais colocam sobre os analistas de dados, para realizar suas consultas eles precisam entender a Privacidade Diferencial, gerenciar seu *privacy budget* e até mesmo implementar novos algoritmos para responder a consultas com ruído. Além disso, os sistemas atuais não oferecem qualquer garantia ao analista de dados sobre a qualidade com que ele se preocupam, nomeadamente a precisão das respostas às consultas.

Como uma alternativa as abordagens dos sistemas atuais, os autores propõem o APEx, um novo sistema que permite aos analistas de dados apresentar sequências de consultas escolhidas de forma adaptativa junto com os seus limites de precisão exigidos. O APEx então traduz as consultas dos analistas com limites de precisão passados em mecanismos diferencialmente privados com o mínimo de perda de privacidade, e ainda garantindo que o *privacy budget* definido pelo dono dos dados não seja violado. Então, o APEx retorna as respostas das consultas ao analista de dados que atendem aos limites de precisão passados e ainda prova ao dono dos dados que todo o processo de exploração de dados é diferencialmente privado.

A partir de um estudo experimental com conjuntos de dados reais e *benchmarks*, os autores demonstraram que o APEx pode responder a uma variedade de consultas com precisão, com perda de privacidade moderada a pequena e pode apoiar a exploração de dados para resolução de entidade com alta precisão para configurações de privacidade razoáveis. Contudo, no momento o APEx se concentra apenas em um tipo particular de consulta, as de contagem linear [34, 35]. Para anonimizar, o APEx escolhe um dos três mecanismos que tem o seu estado da arte implementado no sistema: o mecanismo de Laplace [16]; o mecanismo Data-Aware/Workload-Aware (DAWA) [33] e o mecanismo Matrix [35].

Podemos perceber que o APEx e o `DIMPLY` diferem em diversos aspectos, desde a sua concepção invertida, na qual, a entrada do sistema é a precisão desejada pelo analista, até as funções de agregações compatíveis. O APEx contempla atualmente apenas consultas de contagem linear e o `DIMPLY` as consultas com funções de agregação de soma, média, desvio padrão, variância, mínimo e máximo, além do foco do `DIMPLY` em sistemas *Polystore*.

3.1.6 Discussão

Os trabalhos analisados propõem consultas escritas em formato SQL padrão, ou muito similar, e aplicam técnicas de Privacidade Diferencial para anonimizar os dados e garantir privacidade dos indivíduos presentes nos conjuntos de dados. Alguns tem especificidades de configurações e opções de entradas adicionais, como limites superiores e inferiores, sensibilidade global ou local da consulta, ou precisão da consulta desejada pelo analista. Porém, o que vale destacar da análise é que nenhum deles se propõe a anonimizar dados com o uso de Privacidade Diferencial no contexto de sistemas *Polystore*. Apesar de alguns serem compatíveis com diversos SGBDs SQL existentes, não oferecem a flexibilidade do `DIMPLY`, capaz de prover uma consulta integrada diferencialmente privada sejam quaisquer bancos de dados SQL ou NoSQL, sistemas de arquivos distribuídos ou não, armazenamentos locais ou em nuvem e formatos sem *schema* definido compatíveis com o sistema *Polystore* do Apache Drill.

3.2 Escolha do melhor mecanismo para um cenário

Nesta seção vamos analisar a questão: "Como escolher automaticamente o melhor mecanismo para anonimizar uma dada consulta de acordo com os critérios desejados?", ou seja, dada uma consulta, um conjunto de possíveis mecanismos de Privacidade Diferencial, o *tradeoff* entre utilidade e privacidade, o tempo para anonimizar com o mecanismo, e o Erro Relativo do mecanismo, como escolher o melhor mecanismo automaticamente.

Geng *et al.* [24] apresentam provas matemáticas e demonstrações de teoremas para justificar como é escolhido o melhor mecanismo na sua abordagem. Se propõem a escolher de acordo com as classes de mecanismos de Privacidade Diferencial desconsiderando o banco de dados e a sensibilidade global das consultas. Em seu modelo proposto é considerado a escolha do *tradeoff* entre utilidade e privacidade e fazem derivações matemáticas para encontrar o que seria o mecanismo diferencialmente privado ideal para cada função única de agregação da consulta, de forma a maximizar a utilidade ou minimizar custos.

Por outro lado, na abordagem do `DIMPLY` a escolha do melhor mecanismo se dá através de um modelo de custo, que apesar de não ter sido modelado como um problema de otimização, se propõe a escolher o melhor mecanismo baseado nos critérios de maximizar a utilidade e minimizar os custos também. A consulta do modelo de custo verifica a base do histórico das consultas anteriormente submetidas pelos usuários na busca por consultas com agregações semelhantes, parâmetro de privacidade ϵ iguais, se não houver

iguais, ao menos ϵ dentro do intervalo definido pelo *Threshold*, podendo ainda atribuir pesos na escolha entre minimizar o Erro Relativo e minimizar o tempo de execução da anonimização por cada mecanismo.

Capítulo 4

Abordagem Proposta: DIMPLY

Diante do cenário apresentado na introdução, esta dissertação propõe como solução o DIMPLY, um *Middleware* que recebe as consultas estatísticas do usuários, contempladas no escopo Subseção 4.2.1, e as responde de forma anonimizada por meio da aplicação de técnicas de Privacidade Diferencial. Neste capítulo apresentaremos os detalhes de como a solução foi projetada e implementada.

4.1 Arquitetura da Solução

Conforme mencionado anteriormente, o DIMPLY tem como objetivo atuar como uma camada entre o sistema *Polystore* e o usuário, de forma a prover um resultado anonimizado. O DIMPLY considera múltiplos mecanismos de Privacidade Diferencial para anonimizar os dados. Um diferencial da proposta é que o DIMPLY auxilia na escolha do mecanismo adequado para a consulta submetida. Essa escolha é uma tarefa importante e tem relação direta com a qualidade do resultado. Considerando isso, nossa abordagem inicialmente anonimiza o resultado das consultas submetidas com todos os mecanismos disponíveis, que no momento são o Resposta Randômica, Laplace, Gaussiano e calcula e armazena o Erro Relativo de cada uma. Dessa forma, temos uma base representativa de múltiplas consultas, ou seja, registros do Erro Relativo para cada um dos três mecanismos sobre cada consulta estatística proposta na Seção 4.2.1.

Em sua versão atual, o DIMPLY foi construído sobre o sistema *Polystore* Apache Drill[1], e está limitado aos tipos de armazenamentos compatíveis com o mesmo, que podem ser vistos na Subseção 2.3.2. Mas a ideia do trabalho é valida para outros sistemas *Polystore*

[1] https://drill.apache.org/

que podem ser adicionados ao DIMPLY no futuro. A arquitetura proposta é apresentada na Figura 4.1.

Consideremos o fluxo principal de utilização do DIMPLY como: (i) o usuário submete uma consulta para o *Middleware*; (ii) o DIMPLY verifica se a consulta é uma consulta estatística dentro do escopo, definido na Subseção 4.2.1, e envia a consulta para o sistema *Polystore*; (iii) ao receber a resposta, o DIMPLY anonimiza com Privacidade Diferencial a resposta recebida do sistema *Polystore*; (iv) entrega o resultado anonimizado com o mecanismo que manteve maior utilidade dos dados para o usuário, ou de outra forma, com o menor Erro Relativo para aquele tipo de consulta estatística no *dataset* em questão; e por fim, (v) salva as informações relevantes sobre a execução da consulta atual no banco de dados de estatísticas.

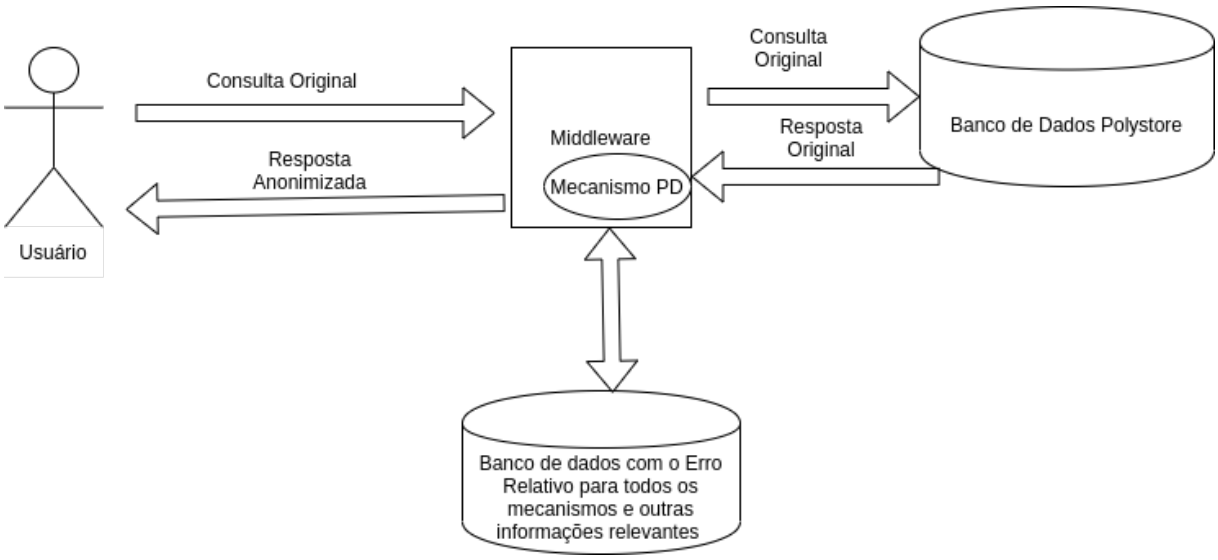

Figura 4.1: Fluxo Principal de utilização do DIMPLY.

Observe que enquanto o DIMPLY ainda não possui informações em seu banco de dados de estatísticas, cada consulta submetida é anonimizada com todos os mecanismos disponíveis no DIMPLY, que hoje são três mecanismos (Resposta Randômica, Laplace e Gaussiano), calculado o Erro Relativo de cada mecanismo para cada função de agregação da consulta e armazenado em uma tabela no banco de dados junto com outras informações relevantes da execução atual desta consulta, como por exemplo, o tempo de execução decorrido na anonimização com cada mecanismo.

Em segundo momento, após o DIMPLY já ter armazenado informações suficientes sobre a organização, ele se torna capaz de escolher um mecanismo de Privacidade Diferencial dentre os disponíveis para anonimizar novas consultas que forem submetidas, e assim, otimizar o processo de anonimização das consultas, anonimizando a partir de então com

apenas um mecanismo, o escolhido pelo Modelo de Custo, proposto na Subseção 4.2.2. Dessa forma, diminuímos o *overhead* do tempo de execução do `DIMPLY`.

Além do fato descrito anteriormente, as consultas mais significativas também devem ser executadas previamente, de preferência fora do pico de uso do sistema, para que seus Δf sejam calculados, e assim, também diminua o *overhead* do `DIMPLY` em comparação com a execução de consultas não anonimizadas.

O limiar da quantidade de informações suficientes para o `DIMPLY` começar a escolher o melhor mecanismo automaticamente é configurável e vai variar de acordo com a organização e suas consultas mais significativas. Como uma definição proposta para esse limiar, consideramos um valor mínimo de pelo menos uma consulta para cada tipo de função de agregação definida no escopo Subseção 4.2.1. E as consultas mais significativas sendo aquelas mais executadas no dia a dia da organização.

4.2 Detalhes Técnicos da Solução

O `DIMPLY` foi desenvolvido em Python 3. Como o Apache Drill não possui suporte nativo para a linguagem Python, utilizamos um *wrapper* para consultar via API REST o Apache Drill. Portanto, em sua versão atual, o `DIMPLY` se encontra limitado a requisições com a máxima latência do protocolo HTTP. O *wrapper* utilizado no desenvolvimento foi o Pydrill[2].

Inicialmente pesquisamos as implementações *open source* de mecanismos de Privacidade Diferencial, e por um dado momento nos propormos a utilizar a biblioteca da IBM ibmdiffpriv [27][3]. Porém, ao avançar com as implementações do `DIMPLY` observamos que haviam otimizações nas implementações e incompatibilidades com nossa abordagem. Outras bibliotecas encontradas, como a da Google[4] , não permitem que a escolha do melhor mecanismo seja customizada, o que também é incompatível com a abordagem proposta. Diante disso, por não terem sido encontradas outras implementações *open source* do estado da arte dos mecanismos de Privacidade Diferencial escolhidos, decidimos por implementar os mecanismos previamente definidos.

Como sistema de arquivos acoplado ao `DIMPLY`, utilizamos o sistema de arquivos distribuído HDFS do *stack* Apache Hadoop[5]. Porém, é importante destacar que poderia ser

[2]https://github.com/PythonicNinja/pydrill
[3]https://github.com/IBM/differential-privacy-library
[4]https://github.com/google/differential-privacy
[5]http://hadoop.apache.org/

usado qualquer armazenamento compatível com o Apache Drill. Por fim, para armazenar todas as informações relevantes para o funcionamento do DIMPLY (*e.g.*, os resultados de Erro Relativo para cada consulta estatística e mecanismo, as consultas já submetidas, os *diffs* dos arquivos contemplados pelas consultas, e os devidos Δf calculados de cada consulta, mais especificamente o Δf para cada função de agregação de cada consulta), utilizamos tabelas no SGBD PostgreSQL[6]. A Figura 4.2. exibe a solução com os detalhes técnicos descritos.

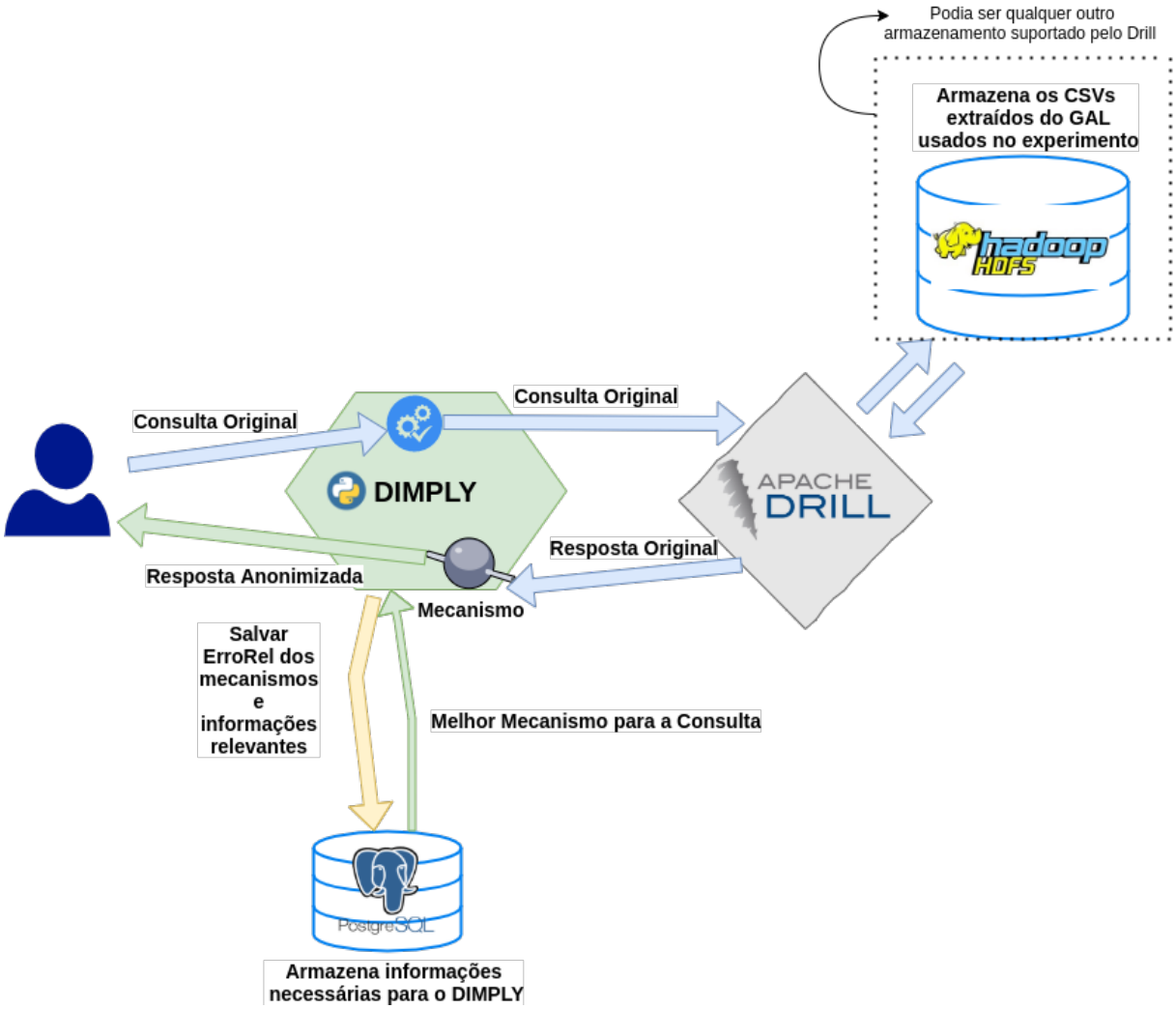

Figura 4.2: Arquitetura Detalhada do DIMPLY.

4.2.1 Escopo da Solução

De forma a garantir a privacidade, o *Middleware* proposto só permite que sejam realizadas consultas estatísticas, (*i.e.*, aquelas que não retornam indivíduos) sendo os operadores de agregação possíveis na versão atual do DIMPLY:

[6]https://www.postgresql.org/

- `SUM`
- `AVG`
- `VARIANCE`
- `STDEV`
- `MIN`
- `MAX`

A versão atual do `DIMPLY` tem uma limitação de não oferecer a função de contagem `COUNT`. Isso se deve a dois problemas, o principal deles é que a função `COUNT` é sobrecarregada na sintaxe de consulta do Apache Drill, no sentido que existem dois tipos diferentes de `COUNT` escritos da mesma forma, um `COUNT` para contagem normal do SQL e um `COUNT` aplicado ao atributo entre parênteses que trás como retorno a quantidade de valores diferentes do domínio [2]. Nós obtemos as colunas envolvidas a partir do plano de execução da consulta retornado pelo Drill, mas não é possível diferenciar esses dois `COUNT`. O segundo problema são os `COUNT(*)`, nesse caso não é possível identificar a coluna que se precisa calcular a sensibilidade global Δf e como a ideia do `DIMPLY` é que usuário digite uma de consulta similar ao SQL e o próprio sistema obtenha as informações a partir dessa consulta, optamos por não pedir adicionalmente ao usuário a informação da coluna envolvida.

Outras definições do escopo do trabalho são que: consideramos que o dono dos danos é confiável e que as parcelas de dados manipuladas no processo anonimização e consulta cabem em memória principal.

Quando uma consulta é submetida ao `DIMPLY`, extraímos e quantificamos os operadores. Uma vez identificados, são armazenamos no banco de dados informações sobre a consulta e seus respectivos operadores envolvidos. Essas informações são utilizadas pelo Modelo de Custo, Subseção 4.2.2, para encontrar consultas que tenham as mesmas funções de agregação que a nova consulta submetida que se pretende recomendar um mecanismo para anonimizar os resultados.

4.2.2 Modelo de Custo

Nesta subseção apresentamos um modelo de custo que tem como objetivo auxiliar na escolha do melhor mecanismo para uma determinada classe de consulta. Em um momento

inicial, o `DIMPLY` precisa anonimizar com todas as opções de mecanismos de Privacidade Diferencial disponíveis para obter informações do desempenho de cada mecanismo de Privacidade Diferencial para cada classe de consulta. Tal desempenho é calculado a partir do tempo de execução e do Erro Relativo. Sendo, o Erro Relativo utilizado para mensurar a utilidade dos resultados anonimizados pelo mecanismo. O Modelo de Custo entra em funcionamento em um segundo momento, que é quando o `DIMPLY` já possui informações suficientes em seu banco de dados de estatísticas. Então, neste momento o `DIMPLY` suspende a anonimização com todos os mecanismos disponíveis e passa escolher um mecanismo mais adequado para cada nova consulta submetida. Com isso, o *overhead* de tempo de execução diminui, passando a ser correspondente ao do mecanismo selecionado.

4.2.2.1 Definição do Modelo de Custo

O modelo de custo proposto leva em consideração duas dimensões simultaneamente para minimização (biobjetivo), atribuindo pesos a cada um dos critérios em sua escolha:
(i) Tempo de Execução e (ii) Erro Relativo.

Antes de realizar a escolha baseada nos critérios supracitados, o Modelo de Custo filtra quais consultas anteriormente executadas possuem o mesmo padrão da nova consulta submetida. Os critérios de semelhança considerados foram: (i) Valor igual do parâmetro ϵ, e caso não encontre um valor igual, considere ϵ dentro do intervalo dado pelo *threshold* e (ii) Mesmo identificador único de conjunto de funções de agregação.

Onde ϵ é o parâmetro de privacidade que o mecanismo utilizou na anonimização daquela consulta e o identificador único de conjunto de funções de agregação é referente ao identificador único atribuído pelo `DIMPLY` para um conjunto de funções de agregação quantificadas contidas naquela consulta. Por exemplo, caso uma determinada consulta utilize as funções de agregação `SUM` e `AVG`, um ID interno é criado no `DIMPLY` para representar essa combinação. Considere as seguintes variáveis definidas na Tabela 4.1.

O modelo de custo proposto é, então, baseado no tempo de execução das consultas e no Erro Relativo. Para definir o mecanismo mais adequado para anonimizar uma consulta submetida com base nestes dois objetivos, utilizamos um modelo de custo ponderado já amplamente adotado em problemas de escalonamento [7] em que o usuário tem de informar o peso de cada critério (que varia de usuário para usuário). A vantagem de oferecer um modelo de custo ponderado é que o usuário pode realizar uma sintonia fina de critérios

Tabela 4.1: Variáveis do Modelo de Custo Definido

Variáveis	***Descrição***
M	Mecanismo de Privacidade Diferencial analisado
D_M	Conjunto com todos os mecanismos de Privacidade Diferencial existentes no `DIMPLY`
q_j	Consulta a ser anonimizada
q_i	Consulta semelhante a q_j encontrada no banco de dados de estatísticas
Q	Conjunto de todas as consultas semelhantes a q_j, *i.e.*, $q_i \in Q$
$\overline{t_{exec}}(M, q_j)$	Média do tempo de execução das consultas semelhantes a q_j para um dado mecanismo M
$\overline{ERel}(M, q_j)$	Média do Erro Relativo para um dado mecanismo M e consultas semelhantes a consulta q_j
α	Peso definido para um determinado critério
ϵ_j	Parâmetro de entrada ϵ para a consulta q_j
ϵ_{q_i}	Valor de ϵ utilizado na anonimização das consultas semelhantes a q_j
θ	*Threshold* do ϵ_i
$id_{agreg}(q_j)$	Identificador atribuído pelo `DIMPLY` para um conjunto de funções de agregação quantificadas contidas na consulta submetida
$id_{agreg}(Q)$	Identificador atribuído pelo `DIMPLY` para um conjunto de funções de agregação quantificadas contidas nas consultas semelhantes a q_j
$M_{ideal}(q_j)$	Mecanismo de Privacidade Diferencial ideal para a consulta submetida

de forma simples, sem que tenha que eleger um critério principal e outro "subordinado". Desta forma, para cada consulta q_j submetida, o `DIMPLY` identifica um conjunto Q de consultas semelhantes a q_j. Tal semelhança depende de quais funções de agregação são utilizadas nas consultas. Uma vez identificada uma consulta $q_i \equiv q_j$ é, então, realizada uma pesquisa para descobrir o melhor mecanismo M na lista de mecanismos disponíveis D_M para anonimizar q_j seguindo o modelo de custo de dois objetivos. Dada uma consulta q_j, temos de encontrar $M \in D_M$ que minimize a função de custo a seguir:

$$f(M, q_j) = \alpha\ \overline{t_{exec}}(M, q_j) + (1 - \alpha)\overline{ERel}(M, q_j) \tag{4.1}$$

De acordo com:

$$(\epsilon_j - \theta \leq \epsilon_{q_i} \leq \epsilon_j + \theta)\ \mid\ \theta \geq 0 \tag{4.2}$$

$$id_{agreg}(Q) = id_{agreg}(q_j) \tag{4.3}$$

4.2.2.2 Implementação do Modelo de Custo

Cada consulta submetida ao `DIMPLY` tem suas informações relevantes coletadas e armazenadas no banco de dados. Ficam armazenadas as informações extraídas da consulta e os resultados da execução da anonimização da mesma com cada um dos mecanismos disponíveis. No banco de dados estão contidas as informações relevantes para o funcionamento do `DIMPLY` e para suprir a função a ser minimizada. Algumas dessas informações relevantes armazenadas são: identificação única da consulta, seu plano lógico de execução, quais e quantos tipos de agregações ela possui, o tempo de execução decorrido na anonimização de cada consulta com cada um dos mecanismos. Para uma dada consulta, armazenamos o tempo de anonimização com cada mecanismo, o seu Erro Relativo e o seu Δf global. Por exemplo, uma consulta submetida com as funções de agregação `SUM` e `AVG`, tem o Erro Relativo e o Δf diferentes para `SUM` e o `AVG`, e essa informação é armazenada no banco de dados.

Diante dessas informações, ao atingir o limiar definido, o `DIMPLY` para de anonimizar com todos os mecanismos existentes e escolhe o mais apropriado a partir do modelo de custo definido. Para selecionar o mecanismo de Privacidade Diferencial que será aplicado sobre os dados, executamos uma função de custo ponderada pelos parâmetros de entrada α e *threshold* θ. Sendo α o peso desejado na escolha entre priorizar o menor Erro Relativo ou o menor tempo de execução no modelo. E o θ define que caso não seja encontrada consulta com exatamente o mesmo ϵ_j, pode ser considerada uma consulta com um valor de ϵ_{q_i} dentro do intervalo $[\epsilon_j - \theta, \epsilon_j + \theta]$. O parâmetro de entrada $id_{agreg}(q_j)$ é o identificador único do conjunto de funções de agregação compatível com a consulta submetida. Independente da ordem que apareçam na consulta, por exemplo, uma consulta na qual o `AVG` e o `SUM` apareçam nesta ordem, vai ter o mesmo identificador de agregação único que uma consulta em que as funções de agregação aparecem na ordem `SUM` e depois `AVG`. Assim, o parâmetro $id_{agreg}(q_j)$ é utilizado pelo modelo de custo para considerar apenas as consultas que sejam semelhantes em relação ao tipo e a quantidade de cada função de agregação. Por fim, o parâmetro de entrada ϵ_j é o valor de ϵ que está configurado no `DIMPLY` no momento da execução da consulta submetida.

Observe no código a seguir, que a consulta que define o mecanismo apropriado primeiramente busca por consultas semelhantes, ou seja, que tenham o mesmo parâmetro $id_{agreg}(q_j)$ ($I_AGREGACAO$) e que foram anonimizadas com o mesmo valor de ϵ que o configurado atualmente em ϵ_j ($I_EPSILON$). Caso ele encontre, ele calcula a média do tempo de execução e do Erro Relativo delas para cada mecanismo e então seleciona

o mecanismo de acordo com a prioridade definida em α (I_ALFA) entre tempo de execução e Erro Relativo. Caso o DIMPLY não encontre consultas anteriormente executadas com exatamente o mesmo valor definido em ϵ_j, ele utiliza o θ ($TRESHOLD$) em busca de consultas dentro do intervalo $[\epsilon_j - \theta, \epsilon_j + \theta]$. Caso encontre, o DIMPLY escolhe o mecanismo da mesma forma explicada anteriormente. Existe ainda o cenário em que o modelo de custo não encontra na base de dados nenhuma consulta correspondente aos parâmetros de entrada, ou seja, nenhuma com o mesmo valor de $id_{agreg}(q_j)$ ou nenhum ϵ igual ou dentro do intervalo, neste caso, o DIMPLY anonimiza com todos os mecanismos disponíveis e retorna a resposta do mecanismo com o menor Erro Relativo apresentado.

```
 SELECT mecanismo, (I_ALFA*AVG_TEMPO + (1-I_ALFA)*AVG_ERRO) AS CALCULO
FROM (
SELECT mecanismo, AVG(tempo_exec) AS AVG_TEMPO, AVG(erro_relativo) AS AVG_ERRO
FROM dimply_historico
WHERE
(CASE WHEN EXISTS (select 1 from dimply_historico as ex where ex.epsilon =
    I_EPSILON AND ex.agregacao = I_AGREGACAO limit 1) THEN epsilon = I_EPSILON
   ELSE ((epsilon >= (I_EPSILON - I_THRESHOLD) AND epsilon <= I_EPSILON) OR
       (epsilon >= I_EPSILON AND epsilon <= (I_EPSILON + I_THRESHOLD)))
END)
AND agregacao = I_AGREGACAO
GROUP BY mecanismo) AS t1
WHERE AVG_TEMPO > 0
ORDER BY CALCULO
LIMIT 1;
```

4.2.3 Cálculo da Sensibilidade Global da Consulta Δf

Esta Subseção tem por objetivo descrever como foi a implementação prática do cálculo da sensibilidade global da consulta Δf, definido na Subseção 2.2.3. A cada nova consulta submetida, calculamos o Δf. Para explicar como este cálculo foi implementado no DIMPLY, suponha que seja submetida a consulta Q3: Média de idade dos pacientes com Zika no Rio de Janeiro no ano de 2016 sobre um CSV com ocorrências de Zika no Brasil (explicado no Capítulo 5):

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`limpo-final-RJ-2016.csv` where ILIKE(agravo, '%Zika%')
```

Primeiramente verificamos se é necessário calcular o Δf. O DIMPLY busca no banco de dados e verifica se é uma nova consulta ou se houve alterações nos dados dos arquivos envolvidos nesta consulta submetida. A seguir, se houver a necessidade de calcular o Δf, o DIMPLY extrai as colunas que precisam ser anonimizadas e os arquivos envolvidos na consulta. São desconsiderados os filtros realizados nas cláusulas WHERE e WHERE_FREE_COLUMNS. Assim, temos uma consulta que envolve todos os dados que precisam ser anonimizados e evitamos que um atacante limite o conjuntos de dados que serão anonimizados para um pequeno conjunto restrito no qual ele deseja obter informações. A consulta resultante para o cálculo do Δf é:

```
select CAST(idadenotif AS DOUBLE) FROM dfs.zikadb.‘limpo-final-RJ-2016.csv‘
```

Esta consulta montada pelo DIMPLY tem apenas as colunas que precisam ser anonimizadas, trazendo assim menos dados para a memória principal, e envolve todos os dados dos arquivos relacionados na cláusula FROM. Então, esta consulta é executada no Apache Drill e o Δf calculado em memória sobre os dados retornados pelo Apache Drill. Ao término do cálculo, o Δf é armazenado no banco de dados do DIMPLY.

4.2.4 Implementação do Resposta Randômica

O Resposta Randômica é essencialmente um mecanismo que se aplica sobre dados binários, dada a forma como ele foi concebido [50]. Diante disso, para implementa-lo para dados não binários existem algumas abordagens para codificar de forma a tomar decisões binárias mantendo as probabilidades originais do Resposta Randômica [21].

No DIMPLY codificamos o mecanismo de Resposta Randômica definido por [19], ou seja, utilizamos apenas um lançamento de moeda e distribuímos as probabilidades. Primeiramente, descobrimos o que seria a “VERDADE” de cada registro envolvido na consulta, esse passo é bem custoso, pois pegamos todos os registros de todos arquivos envolvidos e aplicamos o filtro presente na cláusula WHERE e assim obtemos a “VERDADE”, ou seja, se o registro atual está ou não contido na resposta original da consulta. Após, termos a “VERDADE”, aplicamos o fluxo condicional codificado com as probabilidades, caso entre na opção de falar “VERDADE” o registro é incluído ou não no cálculo de acordo com a sua “VERDADE”. Caso entre na opção dizer “SIM” o registro é incluído no cálculo, e caso contrário ele não é incluído no cálculo da consulta estatística submetida.

4.2.5 Cálculo do Erro Relativo de cada Mecanismo

Para mensurar a utilidade dos dados, utilizamos a métrica de erro relativo, conforme descrito na Subseção 2.2.6. No `DIMPLY` para cada consulta única submetida armazenamos no banco de dados o Erro Relativo para cada função de agregação contida na consulta utilizando todos os mecanismos de Privacidade Diferencial propostos na dissertação. Essa informação é consumida na escolha do mecanismo pelo modelo de custo.

4.2.6 Limitações do `DIMPLY`

É válido ressaltar algumas limitações da abordagem. No `DIMPLY` é permitido que o usuário realize apenas consultas estatísticas. Existem apenas permissões de `SELECT` no SGBD, e o `DIMPLY` verifica a sintaxe da consulta submetida. Além disso, outras considerações que devem ser seguidas para a consulta não ser bloqueada pelo `DIMPLY`, são:

- As colunas devem respeitar a mesma ordem de aparição no `SELECT` e no `WHERE`.
- Só são permitidas no `WHERE` colunas contidas no `SELECT`.
- Não são permitidas consultas aninhadas, ou seja, apenas uma cláusula `FROM`.
- A função de agregação `COUNT` não está disponível na versão atual do `DIMPLY`.
- Não é permitido repetir colunas de mesmo nome no `SELECT`. Separe em consultas diferentes.
- Apenas consultas estatísticas são permitidas.
- Colunas que não possuem funções de agregações no `SELECT` devem ser declaradas no campo `WHERE_FREE_COLUMNS` e devem ter sido cadastradas pelo administrador como liberadas por não possuírem risco a privacidade, *e.g.*, Estado, Município, Nacionalidade.

Capítulo 5

Avaliação Experimental

Para avaliarmos o `DIMPLY`, utilizamos um *dataset* real retirado do sistema GAL do SUS, e um conjunto de consultas mais significativas, que será definido na Subseção 5.4.1. Nesta seção apresentamos as características deste *dataset* e quais foram os resultados obtidos ao submeter as consultas por meio do `DIMPLY`.

5.1 Estudo de Caso

Utilizamos um *dataset* contendo originalmente 1.846.602 tuplas exportadas do sistema GAL do SUS. Esse *dataset* possui 104 atributos com informações relativas a exames realizados por pacientes suspeitos e confirmados com o Zika vírus. A amostra corresponde a 16 valores de UF distintos, ou seja, abrange 16 estados da federação brasileira e 464 municípios distintos. A Tabela 5.1, apresentada a seguir, descreve os principais atributos do *dataset* utilizado no experimento, seus tipos associados e seus domínios.

Realizamos um pré-processamento no *dataset*, e removemos as tuplas onde o campo `UFSOL` (Unidade Federativa) e `DATASOL` (Data da Solicitação do Exame) eram vazios. Removemos os atributos identificadores (atributos que identificam unicamente indivíduos) e mantivemos semi-identificadores (atributos que não são identificadores explícitos, mas podem identificar um indivíduo), atributos sensíveis e não sensíveis os quais pretendemos anonimizar. Após essa etapa, restaram 1.840.198 tuplas. O *dataset* foi exportado para o formato CSV, dividindo os dados por UF e Mês/Ano (no padrão zikadb-UF-mês-ano.csv). Ao final, obtivemos um total de 795 arquivos armazenados no HDFS. Por fim, armazenamos os arquivos no HDFS instalado em uma máquina *commodity* detalhada na Seção 5.2.

Tabela 5.1: Atributos mais relevantes. Pré-Proc. R: Removido, L: Livre e A: Anonimizado

Atributo	***Descrição***	***Valores***	***Pré-Proc.***
`nomepaciente`	Nome do paciente	`Texto livre`	`R`
`sexo`	Sexo do paciente	`(MASCULINO, FEMININO)`	`L`
`racacor`	Raça/Cor do paciente	`(BRANCA, PARDA, PRETA, AMARELA)`	`L`
`etinia`	Etnia do paciente	`(INDÍGENA, BRANCO, NEGRO, etc)`	`L`
`cpf`	CPF do paciente	`Texto livre`	`R`
`idadenotif`	Idade do paciente	`Numérico`	`A`
`datasol`	Data de início do atendimento	`Data`	`L`
`datasintomas`	Data de início dos sintomas	`Data`	`L`
`nomeprofsaude`	Nome do médico que prestou atendimento	`Texto livre`	`R`
`conselhoprofsaude`	CRM do médico que prestou atendimento	`Texto livre`	`R`
`municsol`	Município onde ocorreu o atendimento	`Texto livre`	`L`
`ufsol`	Unidade da Federação do local de atendimento	`Valores Fixos`	`L`
`nacionalidade`	Nacionalidade do paciente	`Valores Fixos`	`L`
`endereco`	Endereço do paciente	`Texto livre`	`R`
`localizacao`	Zona de residência do paciente	`(URBANA, RURAL)`	`L`
`ibgemunic`	Código do IBGE do município	`Valores Fixos`	`L`
`agravosinam`	Sintomas que a paciente teve	`(VÔMITOS, MIALGIA, CEFALEIA, etc)`	`L`
`agravo`	Diagnóstico do paciente	`(Zika, Dengue, Febre amarela, Chikungunya, etc)`	`L`
`examesol`	Exames solicitados ao paciente	`Texto livre`	`L`
`idadegestacional`	Número do trimestre da gravidez	`(1º TRIMESTRE, 2º TRIMESTRE, 3º TRIMESTRE ou nulo)`	`L`
`totalpessoas`	Total de habitantes no município segundo o IBGE	`Numérico`	`A`

Para avaliar o DIMPLY, executamos as consultas mais significativas, definidas na Subseção 5.4.1, e aplicamos a anonimização com Privacidade Diferencial com todos os mecanismos disponíveis sobre o resultado original dessas consultas. Depois calculamos o Erro Relativo entre resposta original e a anonimizada para mensurar a utilidade do dados. Sendo o Erro Relativo calculado separadamente para cada função de agregação presente na consulta. Além disso, também armazenamos o tempo decorrido no processo de anonimização para cada consulta.

5.2 Descrição do Ambiente

A avaliação experimental foi realizada em um notebook *commodity* com 8 GB de RAM DDR4, um processador Intel Core i5-7200U 2.50 GHz x 4, com placa de vídeo Geforce 940MX, com HDD Sata e sem SSD. O sistema operacional utilizado foi Ubuntu 18.04.5 LTS 64 bits com GNOME versão 3.28.2. Neste equipamento foram instalados os *softwares* necessários para o experimento, sendo as versões utilizadas: Python versão 3.6.9, Apache Drill 1.17.0, Apache HDFS e PostgreSQL. Além de bibliotecas comuns Pyhton, sendo elas, math, numpy, random, time, re e json. É importante detalhar que o Apache Drill foi configurado para utilizar no máximo 5 GB de RAM. A comunicação do DIMPLY com o Apache Drill é feita via protocolo HTTP e API Restfull utilizando o *wrapper* PyDrill 0.3.4, uma vez que o Apache Drill não fornece conexão nativa para Python.

5.3 Configuração dos Parâmetros

Todo mecanismo de privacidade diferencial é limitado pelo parâmetro ϵ, daí o termo privacidade ϵ-diferencial. Tal parâmetro determina a quantidade máxima de informações que um atacante pode aprender sobre um indivíduo estudando a saída do mecanismo. Conceitualmente, utilizar um valor baixo fornece mais garantias de privacidade, porém isso costuma implicar em maior adição de ruído nos dados e, consequentemente, diminuição na utilidade dos mesmos.

Determinar o valor adequado do grau de similaridade requer equilibrar os interesses de duas partes com objetivos conflitantes: o analista de dados, que deseja aprender algo sobre os dados, e o indivíduo alvo, que deve decidir se permite que seus dados sejam incluídos na análise. Portanto, resolver esse *tradeoff* entre utilidade e privacidade é uma

tarefa difícil, e para ajudar na escolha desse valor foram surgindo diversas heurísticas ao longo dos anos [28,32,41]. Já na proposta de Kohli *et al.* [31] os próprios indivíduos votam os ϵ desejados.

Diante dessa complexidade da escolha do ϵ ideal, optamos por seguir uma abordagem empírica definida por Wood *et al.* [51]. Wood *et al.* definem os seguintes valores para $\epsilon = \{0,01 \mid 0,05 \mid 0,1 \mid 0,25 \mid 0,5 \mid 1\}$ no caso dos mecanismos de Laplace e Gaussiano. Além disso, definem $\delta = 0$ para o Laplace e $\delta = 1e - 9 = 0{,}000000001$ para o Gaussiano. Este valor de δ é o equivalente a considerar o cenário de uma liberação de privacidade em um bilhão de consultas para Gaussiano [51]. Não existe privacidade forte, privacidade (ϵ,0)-diferencial, para o mecanismo Gaussiano. Porém, é demonstrado em [19] que para valores muito baixos o Gaussiano se aproxima da privacidade forte, a (ϵ,0)-diferencial. Além disso, ambas as distribuições do Laplace e Gaussiano foram centralizados com média $\mu = 0$ nos mecanismos.

Dessa forma, pretendemos apoiar um especialista na escolha do ϵ ideal para o seu contexto. Lembrando que esse contexto deve considerar o orçamento de Privacidade Diferencial, conforme explicado na Subseção 2.2.8. Pois a cada nova consulta executada uma parte da privacidade correspondente ao valor de ϵ pode ser perdida. Isso se deve ao problema da composição, Subseção 2.2.7, que descreve que o risco a privacidade dos indivíduos pertencentes a uma análise diferentemente privada se acumula com múltiplas consultas. É importante destacar que essa propriedade da composição se aplica a todas as técnicas de privacidade. Porém, a Privacidade Diferencial é a única que permite mensurar com forte rigor matemático a perda de privacidade que irá ocorrer a cada nova consulta.

Por exemplo, suponha o cenário em que um especialista precisa configurar o `DIMPLY` para atender a uma legislação vigente que define que o dono dos dados pode ter uma perda de privacidade de no máximo 1,0, ou seja, seu *privacy budget* = 1,0. Esse especialista sabe que segundo os teoremas de como a Privacidade Diferencial se comporta sobre composição [19] a perda de privacidade a cada nova consulta se somam, ou seja, se a consulta 1 foi anonimizada com $\epsilon_1 = 0{,}01$ e a consulta 2 tem $\epsilon_2 = 0{,}02$, ao final ele terá uma perda de privacidade de no máximo $\epsilon 1 + \epsilon 2 = 0{,}03$, que é menor que o seu *privacy budget*. Assim, ele pode definir quais valores de ϵ vai utilizar para cada consulta, podendo balancear o *tradeoff* entre utilidade e privacidade, e bloquear novas consultas ao sistema quando consumir todo o seu *privacy budget*.

Para o mecanismo Resposta Randômica não é necessário configurar nenhum parâmetro. Em relação aos parâmetros configuráveis na função objetivo do modelo de custo,

Subseção 4.2.2, utilizamos $\alpha = 0,6$. Lembrando que o parâmetro α serve para dar peso para o menor erro relativo ou o menor tempo de execução na escolha feita pelo modelo. E para o parâmetro $\theta = 0,4$, que define que caso não seja encontrada consulta com exatamente o mesmo ϵ, pode ser considerada uma consulta com ϵ dentro do intervalo de $[\epsilon-\theta, \epsilon+\theta]$. Os valores de α e θ foram escolhidos arbitrariamente para testar a abordagem do `DIMPLY` e são configuráveis a critério da organização.

É importante observar que o ϵ não é uma medida absoluta de privacidade, mas sim uma medida relativa. Ou seja, um mesmo valor de ϵ oferece garantias de privacidade diferentes com base no domínio do atributo em questão e nas consultas suportadas.

5.4 Resultados

Nessa seção apresentamos os resultados da avaliação experimental considerando um conjunto de consultas estatísticas com os operadores descritos na Seção 4.2.1. Fazemos uma análise da anonimização com o uso da técnica de Privacidade Diferencial para os mecanismos Resposta Randômica, Laplace e Gaussiano aplicados nos resultados das consultas a uma base de dados real de saúde em um sistema *Polystore*.

5.4.1 Experimento

Neste experimento tivemos como objetivos principais avaliar o *overhead* do `DIMPLY` sobre tempo de execução da consulta e analisar a utilidade das respostas anonimizadas das consultas mais significativas para cada mecanismo disponível no `DIMPLY` atualmente. Para avaliar o *overhead* inserido no processo de anonimização, aplicamos a Privacidade Diferencial sobre o resultado das consultas para cada mecanismo 10 vezes e calculamos a média do tempo de execução, em segundos. No caso do erro relativo, consideramos também 10 execuções e calculamos a média do Erro Relativo para cada mecanismo. A média do tempo de execução é comparada com a média de 10 execuções de cada uma das consultas sem nenhum processo de anonimização dos dados para avaliar o *overhead* do `DIMPLY`. Além desses objetivos principais, nesse experimento também analisamos os custos de tempo em segundos do cálculo da sensibilidade global Δf para cada uma das consultas mais significativas analisadas, na Tabela 5.3.

Para realizar a avaliação do `DIMPLY`, definimos as consultas mais significativas de forma a contemplar todas as funções de agregações definidas no escopo Subseção 4.2.1 e consideramos um conjunto de consultas que fossem classificados nas seguintes classes:

- C1 - Agregação “Pura"
- C2 - Agregação com seleção simples
- C3 - Agregação com seleção múltipla
- C4 - Agregação com Junção
- C5 - Agregação com Junção e Seleção

Dessa forma, as consultas mais significativas utilizadas nesse experimento e as suas respectivas transcrições para a sintaxe do Apache Drill, que é semelhante ao SQL, são exibidas a seguir:

[Q1] Qual a idade do paciente mais velho com Zika no estado do Rio de Janeiro no ano de 2016?

```
select max(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.‘limpo-final-RJ-2016.csv‘ where ILIKE(agravo, ’%Zika%’)
```

[Q2] Qual a idade do paciente mais novo com Zika no estado do Rio de Janeiro no ano de 2016?

```
select min(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.‘limpo-final-RJ-2016.csv‘ where ILIKE(agravo, ’%Zika%’)
```

[Q3] Qual a média de idade dos pacientes com Zika no Rio de Janeiro no ano de 2016?

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.‘limpo-final-RJ-2016.csv‘ where ILIKE(agravo, ’%Zika%’)
```

[Q4] Qual a média de idade das grávidas com Zika no Rio de Janeiro no ano de 2016?

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.‘limpo-final-RJ-2016.csv‘ where ILIKE(agravo, ’%Zika%’) AND
    ILIKE(idadegestacional, ’%TRIMESTRE%’)
```

[Q5] Qual a média de idade das crianças com Zika no Rio de Janeiro no ano de 2016?

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`limpo-final-RJ-2016.csv` where ILIKE(agravo, '%Zika%') AND
    CAST(idadenotif AS DOUBLE)<12
```

[Q6] Qual a média de idade dos idosos com Zika no Rio de Janeiro no ano de 2016?

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`limpo-final-RJ-2016.csv` where ILIKE(agravo, '%Zika%') AND
    CAST(idadenotif AS DOUBLE)>=60
```

[Q7] Qual o desvio padrão da idade dos pacientes com Zika no Rio de Janeiro no ano de 2016?

```
select stddev(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`limpo-final-RJ-2016.csv` where ILIKE(agravo, '%Zika%')
```

[Q8] Qual a variância da idade dos pacientes com Zika no Rio de Janeiro no ano de 2016?

```
select variance(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`limpo-final-RJ-2016.csv` where ILIKE(agravo, '%Zika%')
```

[Q9] Qual a média de idade dos pacientes com Zika em municípios com mais de 100.000 habitantes no estado do RJ no ano de 2016?

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`final2-join-limpo-RJ-2016-e-ibgedb.csv` where
CAST(totalpessoas AS DOUBLE)>100000 AND ILIKE(agravo, '%Zika%')
```

[Q10] Qual a média de idade das grávidas com Zika em municípios com menos de 50 mil habitantes no estado do RJ no ano de 2016?

```
select avg(CAST(idadenotif AS DOUBLE)) FROM
    dfs.zikadb.`final2-join-limpo-RJ-2016-e-ibgedb.csv` where
CAST(totalpessoas AS DOUBLE)<50000 AND ILIKE(agravo, '%Zika%') AND
    ILIKE(idadegestacional, '%TRIMESTRE%')
```

A Tabela 5.2 e a Figura 5.1 exibem os resultados da média do tempo de execução, em segundos, de 10 consultas sem a anonimização dos resultados e com anonimização,

para cada mecanismo disponível atualmente no `DIMPLY`. Dessa forma, é possível analisar o *overhead* do passo de anonimização realizado pelo `DIMPLY`. Esses valores consideram que o Δf já foi calculado previamente. Para facilitar a análise das tabelas, as células foram coloridas da seguinte forma:

- Com a cor verde, destacamos os melhores resultados em comparação com a sua linha da tabela.
- Com a cor amarela, destacamos os resultados intermediários em comparação com a sua linha da tabela.
- Com a cor vermelha, destacamos os piores resultados em comparação com a sua linha da tabela.
- Com a cor cinza, informações adicionais não comparáveis com a sua linha da tabela.

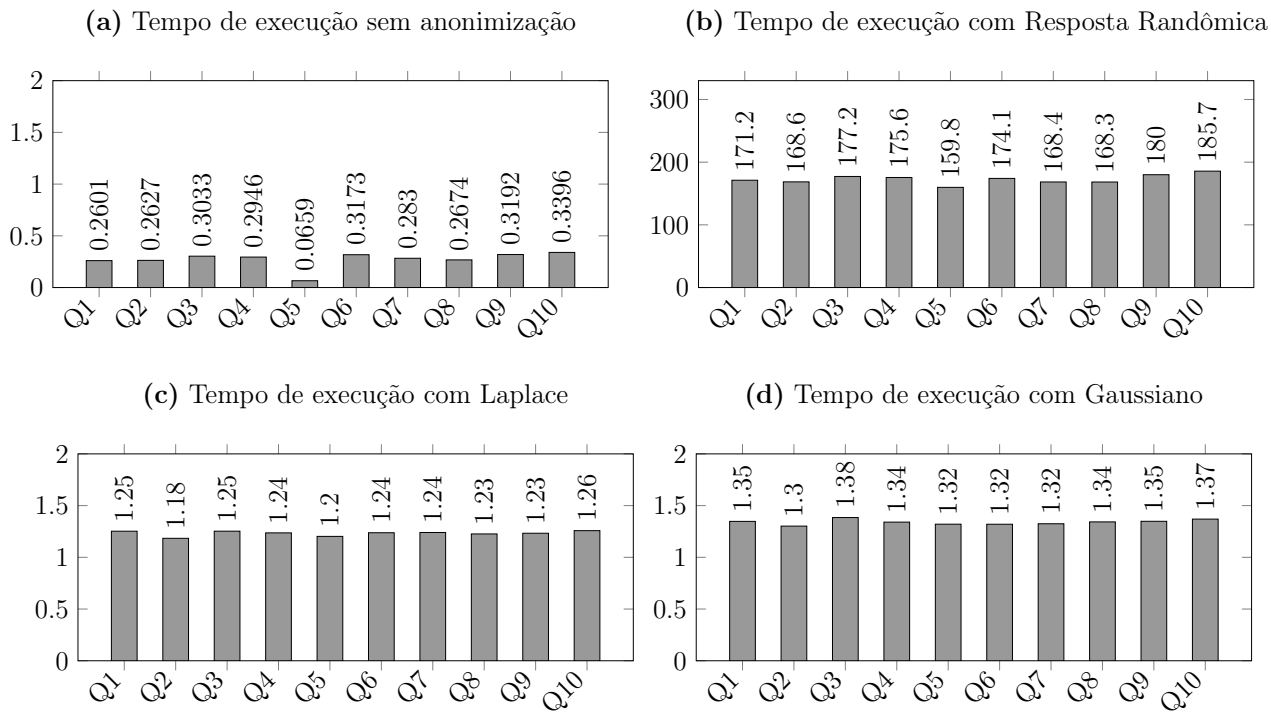

Figura 5.1: Comparação gráfica entre os tempos de execução em segundos decorridos das consultas anonimizadas pelos mecanismos e os tempos de execução das consultas sem anonimização. (a) Tempo de execução sem anonimização. (b) Tempo de execução com Resposta Randômica. (c) Tempo de execução com Laplace. (d) Tempo de execução com Gaussiano.

A partir da análise da Tabela 5.2 e da Figura 5.1, é possível concluir que com o cálculo do Δf já tendo sido feito anteriormente, o *overhead* para os mecanismos de Laplace e

Consulta	Não Anonimizado	RR	Laplace	Gaussiano
Q1	0,2600960255	171,1859307	1,252578831	1,347488165
Q2	0,2626921415	168,5694322	1,183857775	1,30183382
Q3	0,3032758951	177,1871768	1,252785993	1,384660172
Q4	0,2946297646	175,5626013	1,23642695	1,340360689
Q5	0,06590607166	159,7977958	1,202420831	1,320435786
Q6	0,3173264742	174,085975	1,237671328	1,319834399
Q7	0,2830294371	168,4043618	1,240108252	1,324499607
Q8	0,2673701525	168,3448241	1,226130939	1,342548203
Q9	0,3191612244	179,9821284	1,23279314	1,34826324
Q10	0,3396401882	185,6691124	1,257799268	1,369547534

Tabela 5.2: Comparação entre os tempos de execução em segundos das consultas sem nenhuma anonimização e o tempo de execução das consultas com o processo de anonimização dos dados para cada mecanismo. Esses valores de tempo não consideram o tempo de calcular o Δf, ou seja, consideramos um cenário que o Δf das consultas mais significativas já foi calculado previamente.

Gaussiano foi aceitável, com um atraso em torno de cinco vezes o tempo de execução da consulta não anonimizada (porém, a consulta era relativamente rápida, *i.e.*, $\leq 0,5$ segundos). O mecanismo Laplace apresentou um tempo de execução ligeiramente menor que o Gaussiano para todas as consultas e o Resposta Randômica foi ordens de grandeza pior que ambos para todas as consultas analisadas.

O mecanismo de Resposta Randômica não utiliza Δf em seus passos de anonimização dos dados, e consequentemente não se beneficia da otimização do `DIMPLY` de utilizar um Δf previamente calculado. Com isso, o tempo decorrido na sua anonimização foi alto, no geral mais de 600 vezes o tempo da consulta não anonimizada. Além disso, na versão atual do `DIMPLY` consultamos o Apache Drill para descobrir qual seria a resposta "verdadeira" para cada linha dos arquivos envolvidos na consulta, e então executamos a etapa de anonimização do Resposta Randômica e respondemos de acordo com mecanismo. Ou seja, se a resposta verdadeira descoberta vai ser computada nos resultados ou não. Com isso, temos um *overhead* de N consultas ao Apache Drill, onde N vai ser igual ao número total de linhas contidas em todos os arquivos envolvidos na consulta submetida.

Contudo, é válido ressaltar a importância do Resposta Randômica para consultas novas e fora do escopo das consultas mais significativas, visto que, como pode ser observado na Tabela 5.3, o cálculo do Δf é bem demorado, e o Resposta Randômica para essas consultas pode passar a ser mais rápido que o Laplace e Gaussiano se considerarmos o tempo de cálculo de Δf, já que o número de conjuntos vizinhos pode ser alto. Lembrando

que, o número de conjuntos vizinhos está diretamente ligado à quantidade de registros envolvidos na consulta. Como explicado na Subseção 4.2.3, considere D o conjunto de resposta da consulta, os conjuntos vizinhos D' se refere à todos os conjuntos que diferem em apenas um registro, ou seja, considerando o cenário que aquele indivíduo não está presente na análise.

Consulta	Tempo de Cálculo Δf
Q1	263,8262436
Q2	233,3958898
Q3	270,5135016
Q4	268,3816931
Q5	269,6764607
Q6	269,8084908
Q7	259,2733197
Q8	257,72294
Q9	272,8725243
Q10	272,2379997

Tabela 5.3: Tempo decorrido no cálculo da sensibilidade global Δf de cada consulta.

Ao analisar em conjunto a Tabela 5.2 e a Tabela 5.3, podemos observar que, para as consultas analisadas, se o cálculo do Δf for realizado em tempo de execução, o mecanismo de Resposta Randômica ficaria em torno de 100 segundos mais rápido do que os mecanismos de Laplace e Gaussiano para todas as consultas. Porém, o cenário de uso geral do `DIMPLY` é que as consultas mais significativas serão previamente executadas em um horário oportuno, como por exemplo, fora do pico de uso do sistema, e assim o Δf já estaria calculado.

Em um momento inicial, o `DIMPLY` anonimiza com todos os mecanismos disponíveis para alimentar a sua base de dados com informações sobre o histórico de execução das consultas com cada mecanismo, e assim, quando tiver informações suficientes sobre o contexto da organização, pode ser utilizado o modelo de custo para escolher o mecanismo adequado para anonimizar as novas consultas. Dessa forma, o `DIMPLY` passaria a anonimizar apenas com um mecanismo, o escolhido pelo modelo de custo, otimizando assim o tempo de execução e diminuindo o seu *overhead* de anonimização em comparação com uma consulta não anonimizada.

Apesar disso, ainda existem duas situações que o `DIMPLY` precisa anonimizar nova-

mente com todos mecanismos disponíveis e calcular o Δf. Uma delas é se houver modificação nos arquivos envolvidos na consulta submetida, o que é raro de acontecer no nosso cenário, mas pode acontecer. Aqui cabe uma análise que, para um cenário em que haja muita modificação nos dados, o desempenho do `DIMPLY` seria ruim, pois teria que recalcular o Δf constantemente. Além desse cenário de dados modificados, para consultas não consideradas previamente também é necessário recalcular o Δf. Em ambas situações, o cálculo do Δf precisa ser realizado e não pode ser evitado. Porém, o processo de anonimização pode ser otimizado definindo um limiar, de forma que ao ser alcançado, não mais seja feita a anonimização com todos os mecanismos existentes para novas consultas, usando apenas o mecanismo proposto pelo modelo de custo.

Analisemos agora a utilidade dos dados, ou seja, o quão precisas podem ser as conclusões tomadas a partir de cada consulta anonimizada. Para avaliar essa parte vamos utilizar o Erro Relativo, definido na Subseção 2.2.6, que é utilizado para mensurar a utilidade dos dados, visto que, quanto maior o Erro Relativo, mais ruído teve quer ser inserido nos dados para preservar a privacidade de dados dos pacientes pertencentes ao *dataset*. Esse valor está diretamente ligado a distribuição dos dados no conjunto, ou seja, o quão distantes os dados estão entre si distribuídos e o quanto a sua ausência impacta no resultado da consulta submetida. Essa informação é medida a partir do Δf, que também está exibido nas tabelas referentes ao Erro Relativo desta seção e nas do Apêndice A.

Dessa forma, podemos observar na Tabela 5.4 que para altos valores de Δf, temos consequentemente mais Erro Relativo para os mecanismos que utilizam o Δf na anonimização, que na nossa versão atual do `DIMPLY` são o Laplace e o Gaussiano. O algoritmo de Resposta Randômica não utiliza o Δf, então apesar do Δf também aparecer na mesma linha da tabela que o Resposta Randômica, ele não deve ser considerado como um fator que aumentou o Erro do Relativo para o caso do Resposta Randômica.

		Erro Relativo com $\epsilon = 0{,}01$		
Consulta	Δf	RR	Laplace	Gaussiano
Q1	3	6,333333333	2243,58315	6701,591157
Q2	0	0	0	0
Q3	0,0161411213	0,1278876646	0,04216150561	0,1305625606
Q4	0,0161411213	0,09305420118	0,05282914462	0,1510908518
Q5	0,0161411213	0,0682542837	1,323828181	1,357299488
Q6	0,0161411213	0,4884946106	0,4954532283	2,353529154
Q7	0,04211905169	0,1552295681	1,438709108	17,96700085
Q8	1,065945978	5,278416829	23100,23121	471066,971
Q9	0,0161411213	0,07802085417	0,03088776395	0,1828309104
Q10	0,0161411213	0,9590842659	0,1861811917	1,069494631

Tabela 5.4: Tabela comparativa do Erro Relativo dos mecanismos para todas as consultas. Considerando um $\epsilon = 0{,}01$ para o Laplace e o Gaussiano. Um $\delta = 1e-9$ para o Gaussiano. Também é exibido a sensibilidade global Δf de cada consulta. Para outros valores de ϵ veja o Apêndice A.

Observe que para a consulta Q2, que se refere a menor idade de quem teve Zika, o Δf foi de zero, pois existe mais de um registro distinto com a idade mínima, que nesse caso era de um ano de idade. Observe ainda que para as consultas Q3, Q4, Q5, Q6, Q9 e Q10 o Δf foi o mesmo, isso acontece pois são consultas semelhantes, mesma função de agregação sobre a mesma coluna, executadas sobre os mesmos arquivos, e estes arquivos não foram alterados durante a execução delas, suas particularidades descritas nas classes de consulta como "Seleção Simples" e "Seleção Múltipla" não afetam o cálculo do Δf. Apesar das consultas Q9 e Q10 realizarem uma junção com os dados do IBGE, os dados resultantes continuaram com o mesmo conjunto das outras, só que com as colunas a mais do IBGE, ou seja, ainda eram referentes ao Rio de Janeiro no ano de 2016, com a mesma função de agregação, a média. Para mais detalhes de como foi implementado o cálculo do Δf no `DIMPLY` veja a Subseção 4.2.3.

De forma a facilitar comparações, a Figura 5.2 exibe o gráfico do erro relativo para as consultas de médias. As consultas que utilizam outras funções de agregação foram omitidas do gráfico para manter uma mesma escala no gráfico e assim possibilitar comparações. As consultas omitidas foram as Q1 (Maior Valor), Q2 (Menor Valor), Q7 (Desvio Padrão) e Q8 (Variância).

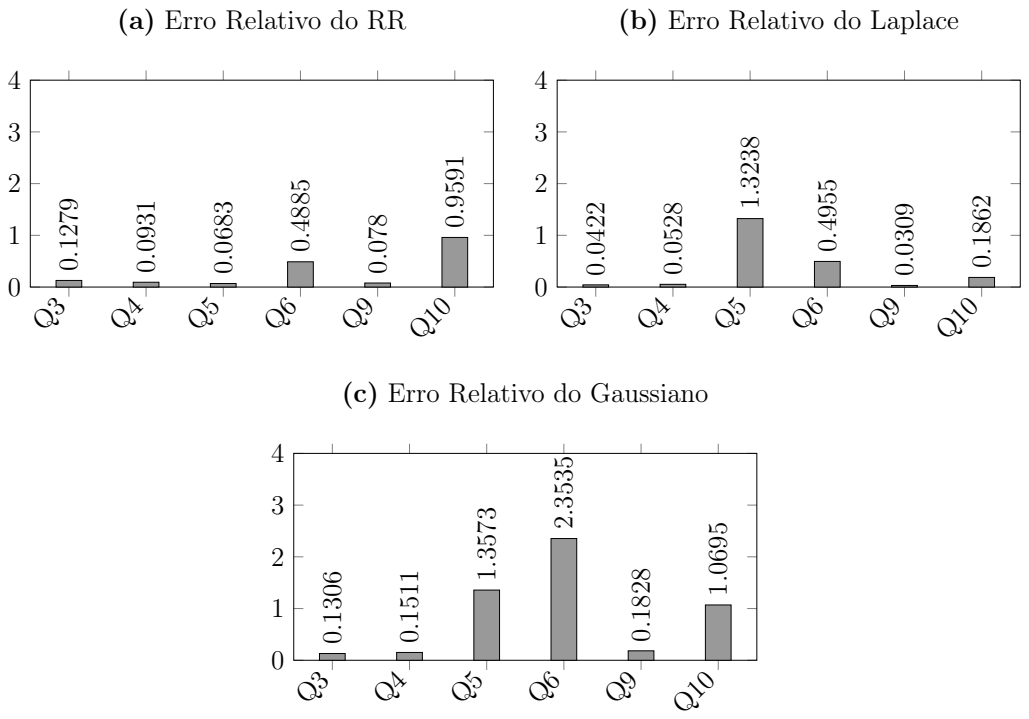

Figura 5.2: Comparativo do Erro Relativo dos mecanismos para as consultas de média. (a) Erro Relativo do mecanismo Resposta Randômica. (b) Erro Relativo do mecanismo Laplace com $\epsilon = 0{,}01$. (c) Erro Relativo do mecanismo Gaussiano com $\epsilon = 0{,}01$ e $\delta = 1e - 9$.

Além da relação entre o Δf e o erro relativo descrito anteriormente, a partir da Tabela 5.4 e dos gráficos da Figura 5.2 também podemos comparar o desempenho dos mecanismos em relação ao erro relativo. É possível observar como a natureza da consulta impacta no resultado do erro relativo, apesar de terem o mesmo Δf, no caso do Laplace e Gaussiano que utilizam o Δf, observe que a consulta Q5 apresentou um erro relativo maior do que a Q3 e Q4, o que indica que seus dados tem valores mais espaçados, no caso, o filtro da Q5 é "que eram crianças", do que os filtros das consultas Q4: "que estavam grávidas" e Q3 que não possui filtros específicos, é apenas a média de idade dos pacientes.

Outra análise possível é verificar se o mecanismo Gaussiano realmente inseriu mais ruído do que o de Laplace, o que seria o comportamento esperado para o nosso contexto de apenas uma dimensão. Destacando que em Q1, Q6, Q7, Q8 e Q10 o Gaussiano inseriu bem mais ruído que o Laplace. Em Q3, Q4 e Q9 o ruído inserido ainda foi alto, mas dentro de uma ordem de grandeza. E por fim, para Q5 foi bem próximo do valor do Laplace. Como explicado na Subseção 2.2.5, esse é um comportamento esperado, uma vez que o Gaussiano começa a inserir menos ruído que o Laplace para cenários com mais dimensões, pois como ele utiliza a L2 Norm, distância Euclidiana, que é menor que a L1

Norm, distância de Manhattan, utilizada pelo Laplace, e assim terá menores valores de Δf nesses casos. Porém, no nosso cenário de uma dimensão o Δf é igual para L1 Norm e L2 Norm [19].

Concluindo as análises acerca da Tabela 5.4 e da Figura 5.2, observamos que para todas as consultas consideradas, o mecanismo Gaussiano inseriu mais ruído comparado com todos os outros mecanismos disponíveis no `DIMPLY`. Já o Resposta Randômica teve um erro relativo ordens de grandeza menor que todos nas consultas Q1 e Q8, isso se deve ao fato dele não utilizar o Δf, e nestas consultas que o valor do Δf foi alto, os mecanismos de Laplace e Gaussiano acabaram inserindo mais ruído para garantir a privacidade de dados. Além dessas consultas, o Resposta Randômica também teve o menor erro relativo para as consultas Q5, Q6 e Q7. Por fim, o mecanismo de Laplace teve o menor erro relativo para as consultas Q3, Q4, Q9 e Q10.

Depois de avaliar os resultados do `DIMPLY` em relação ao seu *overhead* no tempo de execução das consultas mais significativas e em relação a utilidade dos resultados anonimizados dessas consultas para cada mecanismo disponível, analisamos como o erro relativo se comporta com a variação dos valores de ϵ no intervalo $[0,01, 0,05, 0,1, 0,25, 0,5, 1]$ para cada consulta e para os mecanismos que o utilizam, sendo a Tabela 5.5 referente ao mecanismo de Laplace e a Tabela 5.6 referente ao mecanismo Gaussiano.

		ϵ					
Consulta	Δf	0,01	0,05	0,1	0,25	0,5	1,0
Q1	3,000000	2.243,583150	364,338911	183,620967	43,730680	6,017790	2,279558
Q2	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
Q3	0,016141	0,042162	0,005337	0,002861	0,001512	0,000912	0,000218
Q4	0,016141	0,052829	0,007538	0,007098	0,001717	0,001189	0,000413
Q5	0,016141	1,323828	0,413744	0,319265	0,249796	0,296934	0,370228
Q6	0,016141	0,495453	0,312381	0,395607	0,340911	0,344729	0,336834
Q7	0,042119	1,438709	0,062976	0,010226	0,003479	0,001853	0,000788
Q8	1,065946	23.100,231210	923,747562	228,696930	35,135940	9,222948	2,291632
Q9	0,016141	0,030888	0,006012	0,003016	0,001185	0,000536	0,000270
Q10	0,016141	0,186181	0,039830	0,026318	0,008565	0,003803	0,002516

Tabela 5.5: Erro Relativo da anonimização do resultado de cada uma das consultas com o mecanismo Laplace para vários valores de ϵ. Onde para cada ϵ o mecanismo Laplace foi executado dez vezes e o valor exibido é a média do Erro Relativo dessas execuções.

A partir da análise da Tabela 5.5 e da Tabela 5.6 podemos confirmar o que foi explicado na Subseção 2.2.8, que valores baixos de ϵ apesar de garantir mais privacidade

aos indivíduos presentes na análise implica em menos utilidade para os dados. Observe que a medida que aumentamos o ϵ, o erro relativo diminui para todas as consultas, e isso independe do mecanismo utilizado, seja ele no nosso caso Laplace ou Gaussiano. Assim, um especialista de privacidade de dados se faz necessário no período de configuração do `DIMPLY` para definir qual valor de ϵ será utilizado para anonimizar as consultas, de forma a gerenciar esse *tradeoff* entre utilidade e privacidade de acordo com o contexto e exigências da sua organização, como por exemplo, regulações governamentais, qual o seu *privacy budget* disponível.

Agora analisando especificamente a Tabela 5.5, destacamos em vermelho os valores de erro relativo que foram muito altos, como para a Q1 e Q8. Os resultados dessas consultas só vão ter alguma utilidade com o valor de $\epsilon = 1$, e ainda sim, apresentam um erro relativo alto. Destacamos a consulta Q7 de rosa claro para o $\epsilon = 0{,}01$, pois apesar de não ser um valor de erro relativo alto o suficiente para ser classificada como vermelho, optamos por destacar que o erro relativo diminui quase duas ordens de grandeza para o próximo $\epsilon = 0{,}05$. Por fim, destacamos alguns em amarelo que diminuíram uma ordem de grandeza em relação ao valor de ϵ anterior.

		ϵ					
Consulta	Δf	0,01	0,05	0,1	0,25	0,5	1,0
Q1	3,000000	6.701,591157	1.324,648502	636,819255	214,743462	80,889040	23,710930
Q2	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
Q3	0,016141	0,130563	0,031255	0,012133	0,005881	0,001899	0,000826
Q4	0,016141	0,151091	0,041702	0,007605	0,006364	0,004964	0,001808
Q5	0,016141	1,357299	1,098258	0,468009	0,307247	0,235683	0,336283
Q6	0,016141	2,353529	0,600366	0,583416	0,311039	0,357326	0,342671
Q7	0,042119	17,967001	1,192552	0,283338	0,047204	0,014423	0,004168
Q8	1,065946	471.066,971000	19.116,013500	4.744,909146	772,836054	188,012573	47,680072
Q9	0,016141	0,182831	0,038602	0,017111	0,006699	0,002287	0,001020
Q10	0,016141	1,069495	0,287755	0,135901	0,046959	0,031026	0,005755

Tabela 5.6: Erro Relativo da anonimização do resultado de cada uma das consultas com o mecanismo Gaussiano para vários valores de ϵ e com $\delta = 1e - 9$. Onde para cada ϵ o mecanismo Gaussiano foi executado dez vezes e o valor exibido é a média do Erro Relativo dessas execuções.

Para a Tabela 5.6 referente ao Gaussiano, note que colorimos de vermelho os valores de erro relativo que foram muito altos, como para a Q1, Q7 e Q8. De forma que os resultados das consultas Q1 e Q8 para esses valores analisados de ϵ anonimizados com Gaussiano são inúteis e recomendamos anonimizar estas consultas com Resposta Randômica para

este nosso *dataset*. Lembrando que o modelo de custo proposto já é capaz de fazer essa escolha automaticamente. Já para a consulta Q7, valores a partir de $\epsilon = 0{,}1$ já apresentam resultados com utilidade. Os destacados em amarelo das consultas Q5 e Q6 com $\epsilon = 0{,}01$ apresentam um erro relativo mediano, e não comprometem totalmente a análise dos resultados. Apesar disso, se for possível para a organização recomendamos os valores de $\epsilon = 0{,}05$ e $\epsilon = 0{,}1$ respectivamente para estas consultas, que diminuem significativamente o ruído nos dados. Por fim, destacamos alguns em verde que tiveram uma diminuição de pelo menos de uma ordem de grandeza no erro relativo em comparação com o valor de ϵ anterior.

Consulta	Conjunto 1	Conjunto 2	Conjunto 3	Conjunto 4	Conjunto 5	Conjunto 6
Q1	6,333333	0,000000	6,333333	4,333333	4,666667	0,000000
Q2	0,000000	0,000000	0,000000	0,000000	0,000000	0,000000
Q3	0,127888	0,143100	0,194020	0,234873	0,119120	0,082374
Q4	0,093054	0,176988	0,084601	0,097060	0,148610	0,060889
Q5	0,068254	0,252901	0,376025	0,255469	0,057781	0,216585
Q6	0,488495	0,408080	0,814734	0,529738	0,339901	0,108987
Q7	0,155230	0,078142	0,165868	0,272680	0,122857	0,158153
Q8	5,278417	2,446415	3,790033	6,944586	3,896859	2,845182
Q9	0,078021	0,290413	0,135161	0,210663	0,119621	0,205092
Q10	0,959084	1,348687	0,763824	0,676354	0,367760	0,270064

Tabela 5.7: Erro Relativo da anonimização do resultado de cada uma das consultas com o mecanismo Resposta Randômica para vários conjuntos de execuções. Onde em cada conjunto o mecanismo Resposta Randômica foi executado dez vezes e o valor exibido é a média do Erro Relativo dessas execuções.

Finalizando, temos na Tabela 5.7 o erro relativo da anonimização do resultado de cada uma das consultas com o mecanismo Resposta Randômica para vários conjuntos de execuções, onde em cada consulta o mecanismo Resposta Randômica foi executado dez vezes e o valor exibido é a média do erro relativo dessas execuções. Observe que os valores do erro relativo do Resposta Randômica são mais aleatórios se comparados com os dos mecanismos de Laplace e Gaussiano. De forma, que é difícil saber se anonimização vai manter a utilidade dos dados ao longo do tempo. É possível observar na tabela os coloridos como são valores distantes. Esse comportamento pode ser explicado pela própria implementação do estado da arte do Resposta Randômica, no qual, ele garante a privacidade de dados, porém ele não dimensiona um limiar para o ruído inserido, visto que ele não utiliza, por exemplo, uma sensibilidade global Δf da consulta para dimensionar a

quantidade de ruído necessária para se garantir privacidade de dados. Por outro lado, ele tem a sua flexibilidade de aplicabilidade em qualquer cenário de consultas, inclusive em campos categóricos, e para *datasets* muito grandes, ele pode ser escolhido caso precisemos calcular o Δf em tempo de execução da consulta do usuário final.

Capítulo 6

Conclusão e Trabalhos Futuros

As organizações detém dados dos indivíduos, seja para pesquisas científicas ou para outros fins. É impossível se pensar hoje em uma organização que não faça uso de seus dados. Porém, é de responsabilidade legal dos donos dos danos garantir a privacidade dos participantes de sua base de dados. Essas obrigações legais já estão em vigor no Brasil com a LGPD desde agosto de 2020, e suas punições no caso de descumprimento estão em vigor desde 1 agosto de 2021. Na Europa, a GDPR já está em vigor desde 2016.

Para conseguir trabalhar de forma eficiente com o grande volume de dados atual, muitas organizações passaram a ter dados armazenados em diferentes SGBDs, ter sistemas de armazenamentos locais ou em nuvem e com modelos heterogêneos. De forma a oferecer o suporte adequado para cada cenário da organização, priorizando a flexibilidade e o desempenho das aplicações, existe uma tendência que as organizações armazenem seus dados em seus formatos originais, ou seja, sem definir esquemas para eles. A ideia é se beneficiar de arquiteturas de *Data Lakes*. Porém, se faz necessário consultar esses dados de forma integrada, eficiente e com uma interface de consulta unificada. Para resolver este problema surgiram os Sistemas *Polystore* que são capazes de consultar os dados armazenados sem esquemas, em diferentes formatos, diferentes arquiteturas de armazenamento e com uma sintaxe única de consulta combinar esses dados.

Na revisão da literatura, não identificamos trabalhos que apliquem a técnica de Privacidade Diferencial em consultas sobre sistemas *Polystore*. Diante disso, nos propomos a elaborar uma solução para essa lacuna. Além disso, a escolha do mecanismo de Privacidade Diferencial mais adequado e seus parâmetros é um tarefa complexa, pois existe um *trade-off* entre utilidade e privacidade, ou seja, sempre que anonimizamos uma base de dados através de técnicas que inserem ruídos nos dados estamos consequentemente diminuindo a utilidade dos mesmos.

Assim, como contribuições do nosso trabalho, projetamos e implementamos um *Middleware* de Privacidade Diferencial para sistemas *Polystore*, chamado `DIMPLY`, que tem como objetivo prover respostas anonimizadas aplicando o modelo de Privacidade Diferencial sobre o retorno das consultas. Além disso, o `DIMPLY` escolhe em tempo real qual o melhor mecanismo para cada tipo de consulta estatística submetida. Inicialmente, o `DIMPLY` anonimiza com todos os mecanismos disponíveis e responde com o de menor erro relativo, ou seja, a resposta com maior utilidade. E após adquirir conhecimento suficiente sobre o cenário da organização, o `DIMPLY` passa a escolher automaticamente com o uso do seu modelo de custo um dos mecanismos de Privacidade Diferencial disponíveis para anonimizar cada nova consulta. Dessa forma, buscamos diminuir o *overhead* de ter que anonimizar com todos os mecanismos e eliminamos a necessidade de um especialista ter que anonimizar a base de dados para cada novo arquivo de contexto semelhante adicionado. Com a ressalva que se forem ser adicionados arquivos de contexto diferentes ou um novo mecanismo de Privacidade Diferencial, é recomendado retornar o `DIMPLY` para o modo de treinamento até que ele possa aprender sobre as modificações, e assim, o modelo de custo possa refletir em sua resposta a mudança. Em síntese, as contribuições dessa dissertação são:

- Um estudo sobre a anonimização de dados em sistemas *Polystore*.
- Análise dos mecanismos Resposta Randômica, Laplace e Gaussiano para uma base de dados real de saúde em um sistema *Polystore*.
- Um modelo de custo que se propõe a escolher dinamicamente o mecanismo de Privacidade Diferencial para cada nova consulta submetida de forma a maximizar utilidade dos dados.

A avaliação experimental do `DIMPLY` foi realizada com um *dataset* de dados de exames de pacientes com suspeita de Zika vírus extraído do GAL, e consideramos um conjunto de consultas significativas e o parâmetro $\epsilon = 0,01$ para os mecanismos de Laplace e Gaussiano, com suas distribuições centralizadas em 0. Além disso, definimos $\delta = 1e - 9$ para o Gaussiano. E a sensibilidade global Δf das consultas mais significativas foram calculadas previamente. Dessa forma, obtivemos os seguintes resultados:

- O mecanismos de Resposta Randômica foi o que inseriu menos Erro Relativo, garantindo assim mais utilidade para os resultados das consultas Q1, Q5, Q6, Q7, Q8. Porém, considerando o Δf já calculado previamente, o tempo de execução do Resposta Randômica foi ordens de grandeza maior que os outros mecanismos.

- O mecanismo de Laplace foi o que obteve os menores valores de Erro Relativo para as consultas Q3, Q4, Q9 e Q10. Consequentemente mais utilidade para os resultados anonimizados delas.
- O mecanismo Gaussiano foi o que teve os maiores valores de Erro Relativo para todas as consultas mais significativas.

6.1 Trabalhos Futuros

A seguir, apresentamos uma proposta de trabalhos futuros sobre os tópicos desse trabalho:

- Avaliar a inclusão de outros mecanismos de Privacidade Diferencial.
- Generalizar o *Middleware* para outros Sistemas *Polystore* além do Apache Drill.
- Propor novas métricas de escolhas do melhor mecanismo e permitir que o usuário priorize também o sistema *Polystore* que prefere.
- Elaborar uma interface para o *Middleware*, permitindo o usuário customizar mais fácil os parâmetros dos algoritmos, como o grau de privacidade desejado.
- Avaliar uma outra abordagem diferente da do Modelo de Custo utilizado para a escolha do mecanismo mais adequado para cada consulta, como por exemplo, utilizando um modelo de aprendizado de máquina.

Referências

[1] Documentação oficial do apache drill sobre como ocorre a execução de consultas. `https://drill.apache.org/docs/drill-query-execution/`. Último Acesso: 07/07/2021.

[2] Documentação oficial do apache drill sobre funções de agregação. `https://drill.apache.org/docs/aggregate-and-aggregate-statistical/`. Último Acesso: 25/10/2021.

[3] Documentação oficial do apache drill sobre seus módulos principais. `https://drill.apache.org/docs/core-modules/`. Último Acesso: 07/07/2021.

[4] Documentação oficial do apache drill sobre sua arquitetura. `https://drill.apache.org/docs/architecture-introduction/`. Último Acesso: 07/07/2021.

[5] Backstrom, L.; Dwork, C.; Kleinberg, J. Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web* (2007), pp. 181–190.

[6] Bater, J.; He, X.; Ehrich, W.; Machanavajjhala, A.; Rogers, J. Shrinkwrap: Differentially-private query processing in private data federations. *arXiv preprint arXiv:1810.01816* (2018).

[7] Boeres, C.; Sardiña, I. M.; Drummond, L. M. An efficient weighted bi-objective scheduling algorithm for heterogeneous systems. *Parallel Computing 37*, 8 (2011), 349–364. Follow-on of ISPDC'2009 and HeteroPar'2009.

[8] Brito, F. T. Uma abordagem distribuída para preservação de privacidade na publicação de dados de trajetória. 2016. 66 f. dissertação (mestrado em computação)-universidade federal do ceará, fortaleza-ce, 2016.

[9] Brito, F. T.; Machado, J. C. Preservação de privacidade de dados: Fundamentos, técnicas e aplicações. *Jornadas de A tualização em Informática* (2017).

[10] Chen, R.; Mohammed, N.; Fung, B. C.; Desai, B. C.; Xiong, L. Publishing set-valued data via differential privacy. *Proceedings of the VLDB Endowment 4*, 11 (2011), 1087–1098.

[11] Dagher, G. G.; Mohler, J.; Milojkovic, M.; Marella, P. B. Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology. *Sustainable cities and society 39* (2018), 283–297.

[12] de Oliveira, D.; Rodrigues, E.; Costa, S.; Amora, P.; Caldas, A.; Horta, M.; de Fillippis, A. M.; Ocaña, K.; Vidal, V.; Machado, J. Um estudo comparativo de mecanismos de privacidade diferencial sobre um dataset de ocorrências do zikv no brasil. In *Anais do XXXIV Simpósio Brasileiro de Banco de Dados* (2019), SBC, pp. 253–258.

[13] de Proteção de Dados Pessoais (LGPD), L. G. LGPD lei geral de proteção de dados pessoais (lgpd), 2021.

[14] do Brasil, G. F. LGPD Sanções lgdp sanções administrativas, 2021.

[15] Duggan, J.; Elmore, A. J.; Stonebraker, M.; Balazinska, M.; Howe, B.; Kepner, J.; Madden, S.; Maier, D.; Mattson, T.; Zdonik, S. The bigdawg polystore system. *ACM Sigmod Record 44*, 2 (2015), 11–16.

[16] Dwork, C. Differential privacy. In *Automata, Languages and Programming* (Berlin, Heidelberg, 2006), M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds., Springer Berlin Heidelberg, pp. 1–12.

[17] Dwork, C.; Lei, J. Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing* (2009), pp. 371–380.

[18] Dwork, C.; McSherry, F.; Nissim, K.; Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference* (2006), Springer, pp. 265–284.

[19] Dwork, C.; Roth, A., et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science 9*, 3–4 (2014), 211–407.

[20] Elmore, A. J.; Duggan, J.; Stonebraker, M.; Balazinska, M.; Cetintemel, U.; Gadepally, V.; Heer, J.; Howe, B.; Kepner, J.; Kraska, T., et al. A demonstration of the bigdawg polystore system. *Proceedings of the VLDB Endowment 8*, 12 (2015), 1908.

[21] Erlingsson, Ú.; Pihur, V.; Korolova, A. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security* (2014), pp. 1054–1067.

[22] Fung, B. C.; Wang, K.; Chen, R.; Yu, P. S. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys (Csur) 42*, 4 (2010), 1–53.

[23] Ge, C.; He, X.; Ilyas, I. F.; Machanavajjhala, A. Apex: Accuracy-aware differentially private data exploration. In *Proceedings of the 2019 International Conference on Management of Data* (2019), pp. 177–194.

[24] Geng, Q.; Viswanath, P. The optimal noise-adding mechanism in differential privacy. *IEEE Transactions on Information Theory 62*, 2 (2015), 925–951.

[25] Hardt, M.; Ligett, K.; McSherry, F. A simple and practical algorithm for differentially private data release. *arXiv preprint arXiv:1012.4763* (2010).

[26] Hausenblas, M.; Nadeau, J. Apache drill: interactive ad-hoc analysis at scale. *Big data 1*, 2 (2013), 100–104.

[27] Holohan, N.; Braghin, S.; Aonghusa, P. M.; Levacher, K. Diffprivlib: The ibm differential privacy library, 2019.

[28] Hsu, J.; Gaboardi, M.; Haeberlen, A.; Khanna, S.; Narayan, A.; Pierce, B. C.; Roth, A. Differential privacy: An economic method for choosing epsilon. In *2014 IEEE 27th Computer Security Foundations Symposium* (2014), IEEE, pp. 398–410.

[29] Johnson, N.; Near, J. P.; Hellerstein, J. M.; Song, D. Chorus: Differential privacy via query rewriting. *arXiv preprint arXiv:1809.07750* (2018).

[30] Johnson, N.; Near, J. P.; Song, D. Towards practical differential privacy for sql queries. *Proceedings of the VLDB Endowment 11*, 5 (2018), 526–539.

[31] Kohli, N.; Laskowski, P. Epsilon voting: Mechanism design for parameter selection in differential privacy. In *2018 IEEE Symposium on Privacy-Aware Computing (PAC)* (2018), IEEE, pp. 19–30.

[32] Lee, J.; Clifton, C. How much is enough? choosing ε for differential privacy. In *International Conference on Information Security* (2011), Springer, pp. 325–340.

[33] Li, C.; Hay, M.; Miklau, G.; Wang, Y. A data-and workload-aware algorithm for range queries under differential privacy. *arXiv preprint arXiv:1410.0265* (2014).

[34] Li, C.; Hay, M.; Rastogi, V.; Miklau, G.; McGregor, A. Optimizing linear counting queries under differential privacy. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (2010), pp. 123–134.

[35] Li, C.; Miklau, G.; Hay, M.; McGregor, A.; Rastogi, V. The matrix mechanism: optimizing linear counting queries under differential privacy. *The VLDB journal 24*, 6 (2015), 757–781.

[36] Li, N.; Li, T.; Venkatasubramanian, S. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering* (2007), IEEE, pp. 106–115.

[37] Machanavajjhala, A.; Kifer, D.; Gehrke, J.; Venkitasubramaniam, M. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD) 1*, 1 (2007), 3–es.

[38] McSherry, F. D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data* (2009), pp. 19–30.

[39] Melnik, S.; Gubarev, A.; Long, J. J.; Romer, G.; Shivakumar, S.; Tolton, M.; Vassilakis, T. Dremel: interactive analysis of web-scale datasets. *Proceedings of the VLDB Endowment 3*, 1-2 (2010), 330–339.

[40] Miloslavskaya, N.; Tolstoy, A. Big data, fast data and data lake concepts. *Procedia Computer Science 88* (2016), 300–305.

[41] Naldi, M.; D'Acquisto, G. Differential privacy: an estimation theory-based method for choosing epsilon. *arXiv preprint arXiv:1510.00917* (2015).

[42] Nergiz, M. E.; Atzori, M.; Clifton, C. Hiding the presence of individuals from shared databases. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data* (2007), pp. 665–676.

[43] Nissim, K.; Raskhodnikova, S.; Smith, A. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing* (2007), pp. 75–84.

[44] Proserpio, D.; Goldberg, S.; McSherry, F. Calibrating data to sensitivity in private data analysis: A platform for differentially-private analysis of weighted datasets. *Proceedings of the VLDB Endowment 7*, 8 (2014), 637–648.

[45] Shoaran, M.; Thomo, A.; Weber, J. Differential privacy in practice. In *Workshop on Secure Data Management* (2012), Springer, pp. 14–24.

[46] Slee, M.; Agarwal, A.; Kwiatkowski, M. Thrift: Scalable cross-language services implementation. *Facebook white paper 5*, 8 (2007), 127.

[47] sobre Proteção de Dado (GDPR), R. G. GDPR regulamento geral sobre proteção de dado (gdpr), 2021.

[48] Sweeney, L. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10*, 05 (2002), 557–570.

[49] Task, C.; Clifton, C. A guide to differential privacy theory in social network analysis. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (2012), IEEE, pp. 411–417.

[50] Warner, S. L. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association 60*, 309 (1965), 63–69.

[51] Wood, A.; Altman, M.; Bembenek, A.; Bun, M.; Gaboardi, M.; Honaker, J.; Nissim, K.; O'Brien, D. R.; Steinke, T.; Vadhan, S. Differential privacy: A primer for a non-technical audience. *Vand. J. Ent. & Tech. L. 21* (2018), 209.

APÊNDICE A – Resultados com outros valores de ϵ

		Erro Relativo com $\epsilon = 0{,}05$		
Query	Δf	RR	Laplace	Gaussiano
Q1	3	0	364,3389112	1324,648502
Q2	0	0	0	0
Q3	0,0161411213	0,143100148	0,005336968348	0,03125544599
Q4	0,0161411213	0,1769881284	0,007537903219	0,04170150827
Q5	0,0161411213	0,2529014308	0,4137441938	1,09825823
Q6	0,0161411213	0,4080804843	0,3123805311	0,6003658363
Q7	0,04211905169	0,07814228988	0,06297571395	1,192551523
Q8	1,065945978	2,446414803	923,7475619	19116,0135
Q9	0,0161411213	0,2904131317	0,006011616578	0,03860210856
Q10	0,0161411213	1,348686964	0,03983032639	0,2877545281

Tabela A.1: Tabela comparativa do Erro Relativo dos mecanismos para todas as consultas. Considerando um $\epsilon = 0{,}05$ para o Laplace e o Gaussiano. Um $\delta = 1e-9$ para o Gaussiano. Também é exibido a sensibilidade global Δf de cada consulta.

		Erro Relativo com $\epsilon = 0{,}1$		
Query	Δf	RR	Laplace	Gaussiano
Q1	3	6,333333333	183,6209667	636,8192553
Q2	0	0	0	0
Q3	0,0161411213	0,1940198852	0,00286088591	0,01213263069
Q4	0,0161411213	0,08460123767	0,007098343303	0,007604537791
Q5	0,0161411213	0,3760248621	0,3192654905	0,4680088825
Q6	0,0161411213	0,8147341629	0,3956066103	0,5834158059
Q7	0,04211905169	0,1658678901	0,01022585196	0,2833380813
Q8	1,065945978	3,790032763	228,69693	4744,909146
Q9	0,0161411213	0,1351611386	0,003015609307	0,01711077233
Q10	0,0161411213	0,7638239935	0,02631772985	0,1359009705

Tabela A.2: Tabela comparativa do Erro Relativo dos mecanismos para todas as consultas. Considerando um $\epsilon = 0{,}1$ para o Laplace e o Gaussiano. Um $\delta = 1e-9$ para o Gaussiano. Também é exibido a sensibilidade global Δf de cada consulta.

		Erro Relativo com $\epsilon = 0{,}25$		
Query	Δf	RR	Laplace	Gaussiano
Q1	3	4,333333333	43,73068008	214,7434623
Q2	0	0	0	0
Q3	0,0161411213	0,2348727923	0,001511636795	0,005881116711
Q4	0,0161411213	0,09705967086	0,001716549349	0,006363528036
Q5	0,0161411213	0,255469225	0,2497964887	0,30724688
Q6	0,0161411213	0,5297383615	0,3409110908	0,3110391149
Q7	0,04211905169	0,2726795776	0,003478917157	0,04720431475
Q8	1,065945978	6,944586103	35,13593972	772,8360535
Q9	0,0161411213	0,2106630062	0,001185467688	0,00669898595
Q10	0,0161411213	0,6763542144	0,008564807974	0,04695896177

Tabela A.3: Tabela comparativa do Erro Relativo dos mecanismos para todas as consultas. Considerando um $\epsilon = 0{,}25$ para o Laplace e o Gaussiano. Um $\delta = 1e-9$ para o Gaussiano. Também é exibido a sensibilidade global Δf de cada consulta.

		Erro Relativo com $\epsilon = 0{,}5$		
Query	Δf	RR	Laplace	Gaussiano
Q1	3	4,666666667	6,017790145	80,88903957
Q2	0	0	0	0
Q3	0,0161411213	0,1191201204	0,0009115468378	0,001898522473
Q4	0,0161411213	0,1486102173	0,001189466381	0,004964443583
Q5	0,0161411213	0,05778057469	0,2969340286	0,2356830991
Q6	0,0161411213	0,3399012383	0,3447292799	0,3573262128
Q7	0,04211905169	0,1228566516	0,001853353042	0,01442300054
Q8	1,065945978	3,896859053	9,222948202	188,012573
Q9	0,0161411213	0,1196212047	0,0005362281849	0,00228678692
Q10	0,0161411213	0,367759611	0,003803042931	0,03102629708

Tabela A.4: Tabela comparativa do Erro Relativo dos mecanismos para todas as consultas. Considerando um $\epsilon = 0{,}5$ para o Laplace e o Gaussiano. Um $\delta = 1e-9$ para o Gaussiano. Também é exibido a sensibilidade global Δf de cada consulta.

		Erro Relativo com $\epsilon = 1$		
Query	Δf	RR	Laplace	Gaussiano
Q1	3	0	2,279557783	23,71093037
Q2	0	0	0	0
Q3	0,0161411213	0,08237377805	0,0002177564521	0,0008264234789
Q4	0,0161411213	0,06088914868	0,0004134109016	0,001808195261
Q5	0,0161411213	0,2165848819	0,3702275217	0,336283357
Q6	0,0161411213	0,1089874293	0,33683408	0,3426710368
Q7	0,04211905169	0,1581532528	0,0007882569649	0,004167854525
Q8	1,065945978	2,84518171	2,291632092	47,68007172
Q9	0,0161411213	0,2050924634	0,0002699469925	0,001019940835
Q10	0,0161411213	0,2700635155	0,002515928654	0,005754629389

Tabela A.5: Tabela comparativa do Erro Relativo dos mecanismos para todas as consultas. Considerando um $\epsilon = 1$ para o Laplace e o Gaussiano. Um $\delta = 1e-9$ para o Gaussiano. Também é exibido a sensibilidade global Δf de cada consulta.