



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Visualizando Dados do Bolsa Família em Forma de Grafo

José Welkson de Oliveira Cardoso

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Orientadora
Prof.a Dr.a Maristela Terto de Holanda

Brasília
2017



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Visualizando Dados do Bolsa Família em Forma de Grafo

José Welkson de Oliveira Cardoso

Monografia apresentada como requisito parcial
para conclusão do Curso de Computação — Licenciatura

Prof.a Dr.a Maristela Terto de Holanda (Orientadora)
CIC/UnB

Prof. Dr. Marcio Victorino Prof. Dr. Ruben Cruz
Universidade de Brasília Universidade de Brasília

Prof. Dr. Pedro Antônio Dourado De Rezende
Coordenador do Curso de Computação — Licenciatura

Brasília, 18 de Novembro de 2017

Dedicatória

Dedico a...

Agradecimentos

Agradeço primeiramente a Deus por ter me abençoado este momento de apresentar o Trabalho de Conclusão de Curso (TCC), e que Ele tem me proporcionado muitas pessoas que cruzaram meu caminho desde então, minha noiva Carol que sempre me apoiou desde o começo me dando forças para continuar, a minha família que sempre me apoiou, a minha orientadora professora Maristela Terto de Holanda, e também um agradecimento especial para o professor Gustavo Van Erven, que tem me dado um suporte muito bom no Neo4j. Agradeço à todos colegas que me ajudaram de alguma forma nesse projeto e claro a Universidade de Brasília por ter me proporcionado os momentos alegres e tensos desses anos. Obrigado.

Resumo

O Brasil possui um grande volume de dados abertos, porém, a visualização destes é algo difícil pela sociedade. Especificamente em relação ao Programa Bolsa Família, o Governo Federal disponibiliza dados desde 2011 dentro do portal. Fazendo uso de dados públicos e abertos, o presente projeto de pesquisa analisou a visualização dos mesmos em um banco de dados em grafo, o que possibilita que qualquer leigo ou cidadão entre em contato com o material em questão. O banco de dados foi implementado no Neo4J, gerando, assim, grafos os resultados mensais e anuais de cada pagamento do programa social Bolsa Família.

Palavras-chave: *NEO4J*, Governo Federal, programa social bolsa família, grafo, banco de dados não relacional.

Abstract

Brazil has a large volume of open data, but the visualization of these is difficult for society. Specifically in relation to the Bolsa Família Program, the Federal Government has made available data since 2011 within the portal. Using public and open data, this research project analyzed the visualization of the data in a database in graph, which allows any layman or citizen to get in touch with the material in question. The database was implemented in Neo4J, thus generating graphs the monthly and annual results of each payment of the Bolsa Família social program.

Keywords: NEO4J, Federal Government, social program bolsa família, graph, Non-relational database

Sumário

1	Introdução	1
1.1	Objetivo	2
1.1.1	Objetivos Específicos	2
1.2	Estrutura do Trabalho	2
1.3	Metodologia	2
2	Referencial Teórico	4
2.1	Infraestrutura Nacional de Dados Abertos (INDA)	4
2.2	Portal Brasileiro de Dados Aberto	6
2.3	Programa Bolsa Família	8
2.4	Banco de Dados NoSQL	12
2.4.1	Grafos	16
2.4.2	NEO4J	17
3	Estudo de Caso	19
3.1	Análise	19
3.1.1	Populando os Dados do Programa Bolsa Família no Neo4j	19
3.1.2	Dados Públicos do Programa Bolsa Família	22
3.1.3	Código em Python	25
3.2	Grafos	27
3.2.1	Criação dos nós	27
3.2.2	Criação das arestas (Relationship Types)	30
3.3	Limitações de estudo de caso	40
4	Conclusão	42
4.1	Perspectiva	43
	Referências	44
	Anexo	47

Lista de Figuras

1.1	Diagrama da metodologia para a elaboração do trabalho.	3
2.1	Pirâmide de NoSQL.	14
2.2	Teorema ACID e BASE.	15
3.1	Registro X Tempo.	20
3.2	Fluxograma da população de dados.	21
3.3	Código teste do Create de nós Estados do Brasil.	22
3.4	Códigos de testes do create de relações.	22
3.5	Resultado de um grafo de 999 nós.	23
3.6	Resultados superior de um grafo de 999 nós.	24
3.7	Resultados inferior de um grafo de 999 nós.	24
3.8	O nó (SP).	24
3.9	Resultado de um grafo de 25 nós com nomes.	25
3.10	Parte do <i>script</i> em <i>python</i>	26
3.11	Parte do <i>script</i> em <i>python</i>	26
3.12	Código LOAD CSV (NEO4J).	26
3.13	Grafo com nós de nomes das capitais do País.	27
3.14	Grafo com nós de nomes de algumas capitais (ampliado).	28
3.15	Grafo com nós de UF.	28
3.16	Grafo com nós de UF (ampliado).	29
3.17	Código para criação de nós.	29
3.18	Código de criação.	30
3.19	Código de criação com mais detalhe.	30
3.20	Grafo com arestas com 101 nós - Acre.	31
3.21	Grafo com arestas com 101 nós com zoom - Acre.	32
3.22	Grafo do capital Rio Branco e seus beneficiário com 25 nós.	32
3.23	Grafo do município Olivenca de Alagoas com 25 nós.	33
3.24	Grafo com arestas com 101 nós com ênfase nas relações - Acre.	33
3.25	Grafo com arestas com 101 nós com ênfase nas relações com zoom - Acre.	34

3.26 Grafo com arestas com ênfase nas relações com nós em azul - Acre.	34
3.27 Grafo com arestas com ênfase nas relações com nós em azul com zoom - Acre.	35
3.28 Grafo com arestas com 251 nós - Acre.	35
3.29 Grafo com arestas com 251 nós com zoom - Acre.	36
3.30 Grafo Jan/2016 mais que R\$950.00 , nomes - SP.	37
3.31 Grafo Jan/2016 mais que R\$950.00 , benefício em valor.	37
3.32 Grafo Jan/2016 menor que R\$25.00 , especificado em valor.	38
3.33 Grafo Jan/2016 menor que R\$25.00 , especificado pelo nome.	38
3.34 Resultado do máximo do benefício de todo território.	39
3.35 Código de número máximo de benefício.	39
3.36 Código de número mínimo de benefício.	39
3.37 Resultado do mínimo do benefício de todo território.	39
3.38 Resultado do total arrecadado no Brasil.	40
3.39 Tela de erro de memória.	40
3.40 Tela de erro de timeout.	41

Capítulo 1

Introdução

O Governo Federal possui um conjunto de dados públicos onde é possível encontrar informações sobre os principais gastos e pagamentos do Estado. Estes podem ser encontrados na Infraestrutura Nacional de Dados Abertos (INDA) e em vários outros portais, entre os quais, o Portal da Transparência nos Recursos Públicos Federais, que é de grande importância para a consulta pública.

Os dados públicos são disponibilizados em um formato de armazenamento separado por vírgula, conhecido como *Comma-Separated Values* (CSV). Tais arquivos armazenam informações gerais em determinados campos (nome, local de moradia, valores de recebimento do benefício, entre outros). Os dados são liberados mensalmente, sendo possível a consulta por parte de pessoa física ou jurídica. E ainda, por se tratarem de dados públicos, é direito que qualquer indivíduo possa manipulá-los e analisá-los como quiser.

O Programa Bolsa Família (PBF) é um dos principais projetos do Governo Federal. É um programa que beneficia indivíduos de baixa renda e tem como objetivo combater a pobreza e a desigualdade no Brasil. Em muitos lugares do mundo é reconhecido como referência e, por ser um programa amplo, completo e muito diversificado, detém variabilidade de análises a partir dos seus dados.

Apesar da facilidade de acesso aos dados do PBF, o modo como estão disponibilizados dificulta sua própria análise e entendimento. Neste sentido, indivíduos com pouca experiência em análise de dados teriam certas dificuldades em extrair informações qualitativas de tais fontes. De modo geral, os dados no formato em que são disponibilizados no portal são de difícil manipulação. Assim, a criação de um banco de dados não relacional com base em grafos melhoraria a visualização dos dados e exploraria melhor a riqueza dos mesmos, tornando-os mais acessíveis à população. Esta melhoria se deve ao fato do grafo ser favorável e simples aos olhos de quem vê – leigo, desenvolvedor ou o próprio Governo Federal.

1.1 Objetivo

O presente estudo teve como objetivo apresentar uma análise com os dados do PBF, via otimização de visualização, a partir da criação de um banco de dados não relacional com base em grafos, melhorando, assim, a visualização dos dados, tornando-os mais acessíveis.

1.1.1 Objetivos Específicos

Os seguintes objetivos específicos foram desenvolvidos para a atingir o objetivo geral do documento:

- Modelagem de dados baseado em grafo;
- Definição do processo de ETL (*Extract, Transform, Load*); e
- Visualização de dados como grafo.

1.2 Estrutura do Trabalho

As linhas que se seguem foram estruturadas nos seguintes capítulos:

- Capítulo 2: sobre as infraestruturas de dados abertos fornecidas pelo Governo Federal, para que se obtenha as informações necessárias para a população. Apresenta os conceitos fundamentais e necessários para o desenvolvimento do projeto;
- Capítulo 3: sobre as ferramentas e os materiais utilizados para a realização do trabalho, contrapondo os sucessos e insucessos durante sua trajetória. Abrange a análise feita durante todo o processo do arquivo, para a concepção do banco de dados em si; e
- Capítulo 4: conclusão do estudo, tratando do resultado obtido e gerando as perspectivas futuras com ideias e melhorias a respeito do tema.

1.3 Metodologia

O presente estudo se deu em duas partes, sendo, a primeira, um lado teórico, e a segunda, culminando no desenvolvimento prático.

Neste sentido, na primeira parte fez-se uma pesquisa com os portais que armazenam os arquivos contendo as informações do PBF, além de entender a ETL do Neo4j. Foram utilizados *sites* do próprio programa, além de artigos e fóruns para consulta. A segunda parte foi destinada para a manipulação dos dados extraídos do Portal Transparência nos

Recursos Públicos Federais, sendo necessário uma implementação de programas para a manipulação dos dados antes da inserção no banco de dados.

A Figura 1.1 a seguir, representa um fluxograma da metodologia escolhida para o desenvolvimento da presente pesquisa. A primeira vista fez-se um estudo bibliográfico para a utilização dos portais e do programa NoSQL, e para melhor entender o programa, analisou-se sua ETL. Em seguida, fez-se o download do arquivo necessário para a manipulação dos dados e logo depois a implementação dos programas na linguagem *Python*. Nas etapas finais, foi a inserção no banco e conseqüentemente o desenvolvimento dos grafos e assim chegar na conclusão do trabalho.

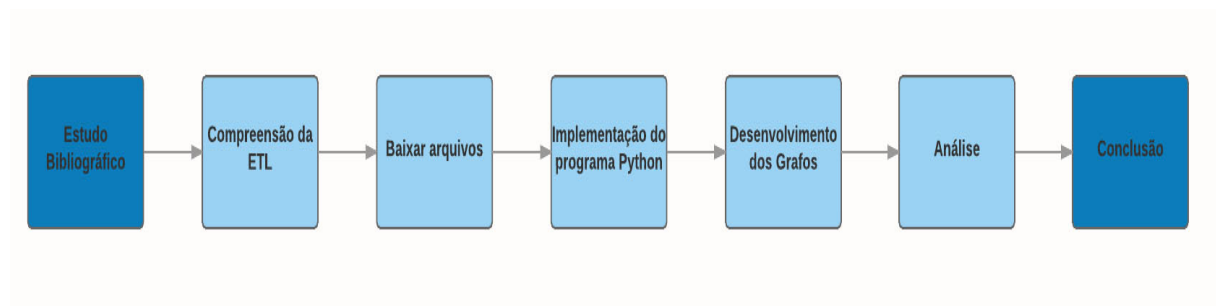


Figura 1.1: Diagrama da metodologia para a elaboração do trabalho.

Capítulo 2

Referencial Teórico

O presente capítulo trata do referencial teórico utilizado no desenvolvimento das linhas que se seguem. Na Seção 2.1 tem-se o estudo da Infraestrutura Nacional de Dados Abertos (INDA), bem como o posicionamento da população em relação aos dados ali reportados. A Seção 2.2 trata do Portal Brasileiro de Dados Abertos, seu objetivo e sua relevância perante a sociedade. Já a Seção 2.3 apresenta um breve relato do Programa Bolsa Família (PBF), bem como dos dados do programa que foram estudados e colocados em análise. Por fim, a Seção 2.4 trata dos bancos de dados não relacionais com base em grafo, além da descrição do teorema CAP (*Consistency, Availability and Partition tolerance*) e uma explicação sobre prevenção a falhas.

2.1 Infraestrutura Nacional de Dados Abertos (INDA)

A INDA é definida por um conjunto de procedimentos e artifícios necessários para atender uma grande variedade de compartilhamento de dados e informações agregados na rede de dados públicos [1].

O portal que se agrega como suporte da INDA é o Portal Brasileiro de Dados Abertos, cujo objetivo é o acesso central de dados públicos governamentais no Brasil. Segundo o site oficial da INDA [1], a visão da infraestrutura é:

"Ser o ponto referencial para a pesquisa, o acesso, o compartilhamento e o uso de dados públicos".

Sua principal missão é padronizar um catálogo governamental com seus dados, além de fazer com que todo usuário tenha, de forma organizada, o acesso aos dados de valor para sociedade. Assim, os objetivos, a política e a gestão relacionada a INDA são [1]:

- Objetivos - Coordenar e orientar, de forma padronizada, a forma de armazenamento, compartilhamento e coleta de informações públicas do governo, além de incentivar

a coleta de valor aos dados públicos e a colaboração na implementação de novos serviços à sociedade.

- Política – Nos poderes de âmbito público, principalmente no Executivo federal, tem-se uma vasta biblioteca de documentos normativos, de planejamento e orientação; e
- Gestão - Realizado pelo Comitê Gestor de órgãos públicos, sociedade civil e Academia.

A comunidade pode catalogar os dados do órgão, promovendo a busca e o acesso aos dados públicos. Logo, cada órgão tem seu papel para a organização dentro do portal de base de dados da INDA, que contém todos os dados liberados a respeito das organizações públicas. Seus dados são liberados pelo *site* dados.gov.br, devidamente catalogados, de forma padronizada para a sociedade.

Para o aumento da quantidade de dados com a ajuda da comunidade aberta, foram criados os *mashups* (mistura). A ideia é que a comunidade em si tenha a liberdade de tomar os dados públicos e misturar de tal modo, aumentando-os ainda mais, observando a criatividade e imaginação de outrem, gerando utilidade pública [2].

Tal idealização se dá basicamente para lograr um cruzamento de dados. A partir do momento em que se cruzam dados abertos para outros fins, novos metadados são criados. Logo, a base para a utilização da comunidade se torna naturalmente maior.

A INDA possui muitas atualizações, criando novos padrões para o aumento da confiança dos dados. As melhorias trazem benefícios em relação à autenticidade das informações geradas. Para dar significado maior aos dados coletados, tal Infraestrutura será efetivamente utilizada com o passar do tempo e com o crescimento da população, onde esta última poderá obter facilmente qualquer tipo de informação por meio de dados.

É importante para a nação se concentrar em obter o máximo de dados possíveis, evitando, na mesma proporção, as perdas durante as coletas. Neste sentido, quanto maior o número de dados, mais abrangentes podem ser as comparações. É possível extrair informações mais concisas a respeito do crescimento econômico e avaliar, por exemplo, se diante da ajuda do Estado, houve melhorias reais para a população.

A participação na INDA se dá gratuitamente, voltada para aqueles que tiverem interesse. Como sugestão da infraestrutura, é preciso implementar uma política de dados abertos no órgão que deseja publicar os seus dados. Conforme o código da INDA, os integrantes do Sistema de Administração dos Recursos de Tecnologia da Informação (SISP) se integram ali automaticamente e, para os demais, é preciso que se preencha um Termo de Adesão.

Segundo o *site* do SISP [3], o sistema que foi fundado em 1994 e atualizado via ditame em 2011, tem como objetivo:

"[...] organizar a operação, controle, supervisão e coordenação dos recursos de informação e informática da administração direta, autárquica e fundacional do Poder Executivo Federal."

Sua estrutura é assim estruturada [3]: órgão central, comissão de coordenação, órgãos setoriais, órgãos seccionais e órgãos correlatos – cada qual com suas características. O órgão central é a secretaria de Tecnologia da Informação (TI) do Ministério Público (MP). A comissão de coordenação é integrada pelos membros dos órgãos setoriais e presidida pelo órgão central. Os órgãos setoriais são representados pelos titulares, pelas unidades de administração dos recursos de TI dos Ministérios e também dos órgãos da Presidência da República (PR). Os órgãos seccionais são representados pelos titulares, das unidades de administração dos recursos de TI das autarquias e fundações. Por fim, os órgãos correlatos são representados pelos seus titulares, das unidades desconcentradas, e formalmente constituídos de administração dos recursos de TI nos órgãos setoriais e seccionais.

Tem-se ainda uma comunidade virtual – provavelmente um dos mais importantes colaboradores para a imersão de conhecimentos e metadados disponibilizados, compostos pelos servidores públicos dos órgãos da SISP e pelos três poderes, englobando as três áreas do serviço público brasileiro.

2.2 Portal Brasileiro de Dados Aberto

O Portal Brasileiro de Dados Abertos constitui-se em um conjunto de dados liberados livremente, para que qualquer pessoa física ou jurídica possa fazer uso como lhe convier, sem patentes, não necessitando de licença para uso e que tenha artifício de controle [4].

O portal, em conjunto com a INDA, se incorpora para a participação total da sociedade. Ele é coordenado pela Secretaria de Tecnologia da Informação (STI) do Ministério do Planejamento, Desenvolvimento e Gestão.

Seu principal objetivo é promover uma solução de agilidade de busca, acesso e cruzamento dos dados do governo brasileiro. Ele foi construído em 2011, sendo regularizado e aprovado pela Lei de Acesso à Informação (LAI) – Lei n. 12.527, de 18 de novembro de 2011 –, delimitando, assim, o acesso às informações cibernéticas voltadas ao governo. A partir daí deu-se um grande salto para a informação pública [5].

De mesmo modo, têm-se outros objetivos, a saber: catalogar, de forma vasta, metadados públicos; obter ferramentas para registrar os catálogos; divulgar os catálogos e aplicações desenvolvidas pela sociedade etc., culminando, assim, mais notoriedade para

tais aplicações, além de promover, de forma diversificada e atraente para a sociedade, o desenvolvimento e a publicação de modelos semânticos.

Os responsáveis pelas referências no portal são as próprias organizações públicas. Neste sentido, na ocorrência de algum problema na informação ou na disponibilização dos dados, quem responde são as referidas organizações. Assim, quando da necessidade de mais informações ou dados para determinada busca, tem-se um espaço no próprio portal indicando com quem resolver a questão. Em suma, manter os dados no ar também é de inteiro compromisso dos próprios órgãos que os colocou [4].

Vale destacar que o portal apenas organiza os metadados e *links* disponibilizados. A hospedagem é de inteira responsabilidade do órgão competente de armazená-los e deixá-los no ar. Para uma criação de catálogos no portal, faz-se importante uma grande massa de dados abertos de coisa pública ou uma grande variedade de áreas determinantes. E ainda, recomenda-se a elaboração de um catálogo de dados abertos na ferramenta denominada *Comprehensive Knowledge Archive Network* (CKAN) [6] – uma solução de gerenciamento de dados totalmente funcional, maduro e *open source*, que torna os dados de descoberta mais apresentáveis.

Então, um conjunto de dados é colocado na própria página, culminando, assim, uma forma de fácil pesquisa. Há uma pergunta corriqueira questionando o que o portal tem a ver com a sociedade, de modo que se têm ali dados que explicitam os direitos básicos do cidadão. E ainda, têm-se informações a respeito de saúde, transporte, segurança pública, educação, gastos governamentais, processo eleitoral etc.

Os portais de dados abertos têm por meio da Lei de Responsabilidade Fiscal – Lei Complementar n. 101, de 04 de maio de 2000 – a transparência na gestão fiscal, e buscam diferenciar e divulgar os gastos na execução pública orçamentária dos Estados, Municípios e do Distrito Federal. Melhor explicitando, tais portais têm por objetivo aumentar o controle das despesas e receitas no governo.

O Portal Brasileiro de Dados Aberto tem por princípio primordial ser o ponto focal de conjuntos de dados e recursos. Assim, cada conglomerado de dados possui uma descrição aberta e diversos caminhos de metadados com a atualização periódica do dado e do órgão responsável. Cada caminho compreende uma fonte de metadados, facilitando, então, uma distribuição de planilha ou até mesmo um método de *webservice*.

Cada recurso imaginado deve ser catalogado de forma reunida de dados separados. Porém, é preferível que os recursos sejam coletados de forma coesa na mesma base de dados, não espalhados e desorganizados futuramente. Assim, tem-se certa facilidade na hora de catalogar, além de ajudar na organização do metadado, sem que perca alguma parte de informação concisa [2].

O portal não deve conter nenhum dado sigiloso, a não ser que o responsável do dado

deixe que tal ação se manifeste. Ademais, tudo é amparado pela LAI [5]. O dado não passa pelo processo de catalogação do portal, tampouco por processo de abertura. Quando um dado sigiloso é colocado à disposição, tal ação deve ser comunicada antecipadamente, para que este seja protegido, seguindo-se corretamente a Lei. Em caso de descumprimento, o órgão ou a entidade responderão pelos termos no art. 34 do ditame em questão.

É sabido que as referidas informações precisam ser publicadas e estar à disposição ao público. Neste sentido, a sociedade tem o direito de saber manusear as informações, bem como analisá-las de modo fácil e intuitivo.

2.3 Programa Bolsa Família

No Brasil, na década de 1950, o ex-presidente do conselho da *Food and Agriculture Organization* (FAO) Josué de Castro, debateu sobre segurança alimentar para ganhar a atenção no País. Desde então, na década anterior, se distribuíam cestas básicas para regiões específicas, de forma indireta, para as áreas carentes. Porém, com algumas denúncias de corrupção e desvio das cestas, além da carência de fiscalização, o então o ex-presidente Fernando Henrique Cardoso implantou os programas sociais no País com a parceria de algumas Organizações Não Governamentais (ONGs), com destaque para a ação da Rede de Proteção Social, onde estavam inclusos todos os programas de distribuição de renda [7].

A concepção do então Programa Bolsa Família (PBF) veio como forma de inspiração do Bolsa Escola, originado por Cristovam Buarque na segunda metade da década de 1980, que mais tarde foi implementado em 2001 pelo Governo Federal [7].

Já no ano de 2002, havia 5 milhões de famílias sendo beneficiadas pela Rede de Proteção Social. O governo se viu na situação de unificar o Bolsa Família e a ampliação de tais programas em um único cadastro, ou seja, fez-se a junção de todos os programas em um único programa, culminando na criação do Ministério de Desenvolvimento Social e Combate à Fome, gerando melhor expectativa de fiscalização e eficácia administrativa. Desde sua unificação, o programa deu certo, ampliando seu atendimento de cinco milhões de famílias para 14 milhões de famílias [7].

Segundo a revista Carta Capital ¹. o objetivo principal do PBF é combater e reduzir a pobreza em um curto espaço de tempo, buscando aliviar os problemas decorrentes da situação que releva a pobreza e, a longo prazo, interromper um ciclo absurdo de pobreza gerado no País.

A ideia para fazer o programa de transferência unificada teve origem em três países condicionados, a saber: Bangladesh, México e o Brasil, com interesse também de três

¹<https://www.cartacapital.com.br/sociedade/entenda-como-funciona-o-bolsa-familia-248.html>

países africanos, quais sejam: Etiópia, Quênia e África do Sul. Já em outras partes do mundo, o país Euroasiático da Turquia, o sudeste asiático (Camboja), o sul asiático (Paquistão) e o Estado de Nova York, nos Estados Unidos da América, teve início o processo idealizado no Brasil. Egito, Indonésia, África do Sul e Gana enviaram seus representantes para conhecerem o programa aqui no Brasil [8].

O PBF é tido como um mecanismo condicional de transferência de recurso e, em geral, tem por objetivo ajudar as famílias mais necessitadas. A renda ali predisposta varia de acordo com a necessidade da família, ou seja, de R\$ 85,00 (oitenta e cinco reais) a R\$ 230,00 (duzentos e trinta reais), com o benefício por família entre R\$ 85,00 (oitenta e cinco reais) a R\$195,00 (cento e noventa e cinco reais) – dados de 2014. Atualmente, o valor varia de R\$ 143,57 (cento e quarenta e três reais e cinquenta e sete centavos) a R\$ 236,47 (duzentos e trinta e seis reais e cinquenta e sete centavos), reajustados desde o segundo semestre do ano de 2016. Dentro do programa social em questão, houve uma alteração com novos ramos de atendimento, com base em cartões alimentação, auxílio-gás, bolsa alimentação e bolsa escola. O governo vem mantendo os benefícios para a melhoria da vida cotidiana do beneficiado. O objetivo da unificação é garantir, com maior eficácia, a agilidade da liberação do dinheiro e redução da burocracia [9].

O objetivo do PBF também é romper um ciclo de ocorrência há gerações em relação à pobreza no País, entre um curto e longo prazo com transição de renda. Tal Programa foi considerado um ensinamento pelo mundo no que tange à erradicação da pobreza, tendo sua nomeação, segundo o jornal *The Economist* [10], como:

“Um esquema anti-pobreza originado na América Latina que está ganhando adeptos mundo afora. ”

O Programa também teve menção no renomado jornal francês *Le Monde* [11], conforme se segue:

“O programa Bolsa Família amplia, sobretudo, o acesso à educação, a qual representa a melhor arma, no Brasil ou em qualquer lugar do mundo, contra a pobreza.”

Por conseguinte, os resultados obtidos com o Programa foram logo aparecendo. Assim, no ano de 2006, mais de 11,1 milhões de famílias foram contempladas e, no País, mais de 45 milhões de indivíduos. Naquele ano, o orçamento total foi de R\$ 8,2 bilhões – 0,4% do Produto Interno Bruto (PIB). Alguns analistas arriscaram em afirmar que foi possível a redução de 27% o índice da miséria no Brasil [12].

E ainda, identificou-se um crescimento real de 71% do salário mínimo daqueles que se encontravam na miséria. Assim, a geração de emprego foi impactada para melhor, gerando 21 milhões de empregos, melhorando a merenda escolar e a transparência da participação

da sociedade, acarretando na recriação do Conselho Nacional de Segurança Alimentar e Nutricional (CONSEA). Com isso, o Brasil conseguiu a redução de extrema pobreza em 75% no ano de 2006 [12]. Com os dados atualizados do corrente ano, identificou-se uma piora nos resultados por ocasião da crise política e econômica, fazendo com que a arrecadação diminuísse, além dos cortes orçamentários [13] [14].

De modo estrutural, o PBF possui uma série de análises para enquadramento e objetivo das famílias no Programa, como, por exemplo, o fato das famílias se enquadrarem no padrão financeiro da renda *per capita* pré-definida, buscando a Prefeitura Municipal, informando os dados no Cadastro Único; quando apta, o benefício é liberado de característica pessoal, para um encadeamento de seleção da Caixa Econômica Federal (CEF), tornando o processo mais imparcial possível, sem que haja interferência política [15].

Recentemente, a cidade americana de Nova York fixou o programa denominado *Opportunity NYC* – um programa inicial que atende apenas cinco mil famílias de baixa renda, seguindo as regras básicas de igual forma brasileira. Segundo o *Opportunity NYC* [16]:

“Nosso maior desafio foi adaptar para a realidade nova-iorquina esses programas de países onde a renda média é menor. Examinamos o Bolsa-Família em relação às contrapartidas de comparecimento às aulas e como o aumento na frequência leva a uma elevação no número de alunos que concluem o ensino médio. ”

As críticas ao PBF são bem relevantes. Como um dos criadores pioneiros, Cristovam Buarque aponta que a retirada da palavra “escola” do programa tirou a ênfase que se tinha do programa para a educação – um princípio básico para o desenvolvimento da economia a longo prazo.

Neste sentido, Cristovam Buarque assevera [17]:

“Colaborou para isso o fato de o Lula ter tirado o nome ‘escola’ do Bolsa Escola. Quando criei esse nome, havia um objetivo: colocar na cabeça da população pobre que a escola era algo tão importante que ela ganharia dinheiro para o filho estudar. O Lula chegou e disse: ‘A pobreza é uma coisa tão preocupante que você vai ganhar um benefício por ser pobre’. Deixou de ser uma contrapartida para a ida do filho à escola. Essa contrapartida não é cobrada com a devida ênfase. A coisa amoleceu quando Lula tirou o programa do ministério da Educação, onde o Fernando Henrique tinha colocado, e levou para o ministério do Desenvolvimento Social. ”

Muitos outros críticos já apontaram que o Programa era basicamente uma bolsa eleitoral, que reunia eleitores cativos, ofertando, de certo modo, dinheiro e benefício ínfimo. O PBF várias vezes foi apelidado de “Bolsa Esmola” ou “Bolsa Miséria”. E é bem longe do que o Banco Nacional pensa, conforme expresso pelo jornal francês *Le Monde* [18]:

"À luz de uma série de investigações no terreno, essa crítica revela ser amplamente infundada. A quantia média recebida por uma família pobre é três ou quatro vezes

mais reduzida do que o salário mínimo (R\$ 180). Portanto, de qualquer maneira, mais vale descolar um emprego, mesmo que este seja pouco qualificado. Longe de serem indolentes, as famílias interessadas trabalham, de fato, muito mais do que as outras."

No entanto, críticos e apoiadores reclamam sobre a participação demasiada da política no cunho social do Programa. Tem-se, portanto, a tendência de modificar o auxílio da política social bem empregada que é adquirida pelo Estado e paga pela população brasileira. Como consequência de tal preocupação, no ano de 2008, o Tribunal de Contas da União (TCU) aprovou a cassação dos mandatos do governador e vice-governador do Estado da Paraíba, acusados de desvios e má distribuição do dinheiro, por meio de cheque como meio de garantia do dinheiro, contendo um bilhete com as seguintes frases [19] [20] :

"Esse é um presente do governador, lembre-se dele. Com os cumprimentos, Cássio Cunha Lima, governador. "

No ano de 2014, o PBF beneficiou 14.145.274 famílias, ou seja, quase 50 milhões de indivíduos. Naquele ano, o benefício era classificado em quatro categorias, quais sejam: 1) básico; 2) variável; 3) variável para jovem; e, 4) superação da extrema pobreza. O valor do benefício básico e da categoria de extrema pobreza era de até R\$ 77,00 (setenta e sete reais), a categoria variável era de R\$ 35,00 (trinta e cinco reais), e a categoria variável para jovem era de R\$ 42,00 (quarenta e dois reais).

O valor mais alto já pago pelo PBF foi em 2012: R\$1.332,00 (um mil e trezentos e trinta e dois reais) para uma família de 19 membros. Tal valor foi vinculado com outro programa social denominado Brasil Carinhoso (já extinto). Atualmente, para receber o teto há uma série de condições já explicitadas, como, por exemplo, o valor máximo de R\$ 336,00 (trezentos e trinta e seis reais) [21]. O governo sabe quem deve receber o Bolsa Família com base nas informações registradas na plataforma denominada Cadastro Único [22].

Como prioridade tem-se as famílias gestantes e sob regras a serem cumpridas, tendo que manter as crianças entre 6 e 17 anos com frequência ativa nas escolas; e as mulheres grávidas e amamentando deve estar em meio ao acompanhamento de saúde.

Em 2016, a regra para a maior renda per capita aumentou de R\$ 77,00 (setenta e sete reais) para R\$ 154 (cento e cinquenta e quatro reais). Para as outras categorias, tudo continuou como antes, porém, com maior fiscalização e pesquisa. Diante do exposto, foi possível observar que a população brasileira está saindo da miséria, logrando uma vida melhor do que há dois anos. No primeiro semestre de 2016, com muitas mudanças e análises por parte do Governo Federal, instigou-se uma melhor fiscalização e distribuição do Programa. Tal fato se deveu por parte da economia do País e a crise

política. Após algumas mudanças políticas, muitas famílias deveriam deixar de receber o benefício ou estavam fora do encaixe da renda *per capita* determinada pelo Programa. E ainda, observou-se que o Cadastro Único estava desatualizado e, de modo, minucioso, trabalharam para reformular novamente o Bolsa Família [15].

2.4 Banco de Dados NoSQL

A presente seção trata da fundamentação teórica sobre o banco de dados NoSQL, com ênfase no modelo com base nos grafos aqui utilizados.

O termo “NoSQL” vem do *Not Only Structured Query Language*, que surgiu em 1998, através de uma solução de banco de dados que não fornecia uma interface SQL. Porém, o sistema em questão tem por base a arquitetura relacional [23].

Na conferência denominada *no:sql(east)*, com Johan Oskarsdon – organizador e criador do Last.fm, juntamente com Carlo Strozzi – grande pesquisador da área do NoSQL [24] – teve início a percepção de que as maiores empresas, como, por exemplo, a Google, na época com investimento no *BigTable*, estavam migrando seus dados de um banco relacional para um banco não relacional. Atualmente, as empresas que utilizam o NoSQL são: *Comcast, Netflix, Pitney Bowes, Disney, RSA, Symantec, HP, AT&T, Impetus, ebay, Ericsson, Adobe, Cisco, MTV*, sem falar do *Facebook e Amazon* [25].

Tal migração se deu por conta do novo conceito que estava se criando, em especial, a permissão constante, rápida e organizada do processamento de dados, gerando, assim, uma era de fluidez nos dados e melhoria de desempenho. Neste sentido, o NoSQL trouxe uma nova proposta para não perder as informações correlacionadas no banco e que tenha fluidez. O sistema, então, é convocado para agregar mais servidores e, se um falhar, há outros poderão suprir um eventual erro.

Sabendo que os bancos relacionais têm capacidade limitada para grandes volumes de dados, começou a se pensar em outros modelos para a obtenção da sustentação da grande massa de dados. Neste ínterim, o NoSQL serviu muito bem para projetistas que queriam um banco de dados consistente e bem embasado, sem que houvessem erros na base de dados. Logo, com uma grande quantidade de dados e fazendo a melhor manipulação possível, conseqüentemente, tem-se alguma melhoria, como, por exemplo, o desempenho e a fluidez dos dados.

O NoSQL por sua vez, se engloba em banco de dados distribuídos que tem por característica a escalabilidade horizontal [26], que resulta no aumento do número de máquinas disponíveis sem prejudicar o banco em si. Por sua vez, com uma grande quantidade de nós, gera-se um armazenamento e processamento com maior desempenho e, por conseguinte, tal escalabilidade possui baixa latência. Tal ação se dá quando da agilidade do

tempo de processamento. O amplo acesso acarreta em um extenso intervalo de leitura e escrita no banco; porém, a não consistência e sua escalabilidade incorreta podem acarretar um *crash*, que se dá devido à falha de funcionamento de um computador ou de parte do sistema operacional – por vezes, um congelamento do programa em execução ou da tela da sua máquina.

Em um banco de dados NoSQL tem-se um conjunto de APIs, cujo papel é fazer com que o banco não perca a estrutura na memória, sendo possível a reversão quando de uma falha. Como exemplo, tem-se a *API MORPHIA*, a *API DataSAX*, a *API DIANA* – muito utilizadas pelo Cassandra DB para aplicações de NoSQL. Estas devem ser simples, a fim de gerar a possibilidade de recuperação de dados de forma consistente e eficiente.

No NoSQL tem-se o teorema CAP, assim detalhado [27]:

- **Consistência:** o sistema continua consistente mesmo após a execução de uma operação, ou seja, é considerado consistente depois de uma atualização de certo dado, e os usuários que tem acesso à informação poderá acessá-lo em tempo real;
- **Disponibilidade:** sistema após implementação; assegurado, ficará ativo durante certo período de tempo; e
- **Tolerância a particionamento:** capacidade de um sistema continuar ativo mesmo depois de um erro na rede.

Um dos fatores do teorema CAP é que apenas duas das três propriedades supramencionadas podem ser atribuídas ao mesmo tempo. Para melhor entender a questão, a seguir, tem-se as opções para melhor otimizar o teorema, de modo que quando do uso, o sistema possa melhorar e evitar falhas durante sua concepção [27]:

- **CP (Consistência e Partição):** apenas a consistência e a partição serão selecionadas, uma vez que para sistemas que necessitam de consistência forte e tolerância a particionamento, é vantajoso que deixe de utilizar a disponibilidade. Exemplos de bancos que se organiza da forma CP: *BigTable*, *HBase*, *MongoDB*.
- **AP (Disponibilidade e Partição):** apenas a disponibilidade e a partição serão selecionadas, de forma que esteja íntegro a todo o momento, pois existem sistemas que não podem ficar *off line*; logo, é preciso abdicar da consistência. Exemplos de bancos para a estrutura: *Cassandra*, *Riak* ou *Amazon Dynamo*; e
- **CA (Consistência e Disponibilidade):** apenas a consistência e a disponibilidade serão selecionadas, pois, mediante consistência forte e alta disponibilidade, o sistema não sabe lidar com a falha de particionamento. Como exemplo dessa estrutura tem-se o *Neo4j*.

A Figura 2.1, a seguir, retrata as principais ferramentas NoSQL utilizadas. Neste ínterim, os NoSQL *Riak* e *Cassandra* têm as características de disponibilidade e tolerância a particionamento, e o *MongoDB*, *Redis* e *HDFS* são relatados como CP. O *Neo4j*, por sua vez, tem as características de Disponibilidade e Consistência – um banco caracterizado como CA, conforme descrito na figura a seguir:

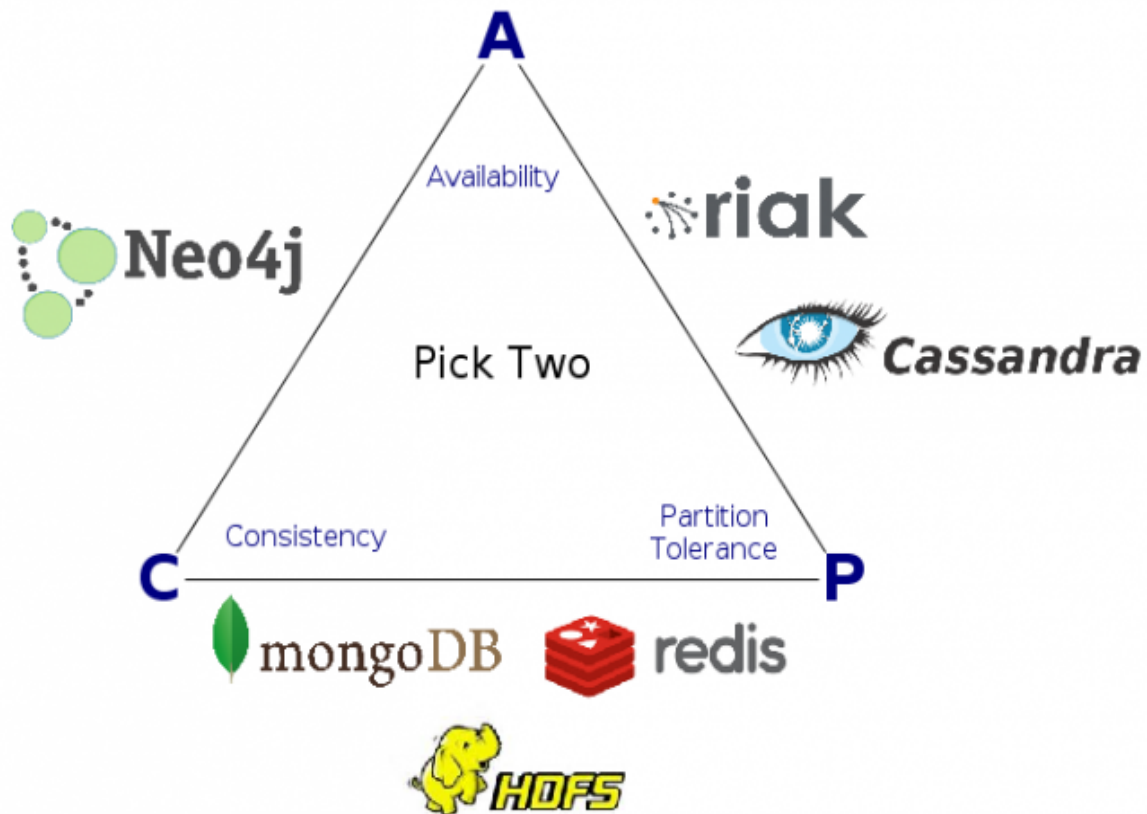


Figura 2.1: Pirâmide de NoSQL.

O NoSQL tem como um dos pontos focais o armazenamento dos dados. Assim, é possível fazer uso do modelo chamado orientado a documentos, chave/valor, família de colunas e orientado a grafo [28]. É importante frisar que nenhum modelo NoSQL é mais importante que o outro. A variância de um NoSQL para outro são suas orientações. As características supramencionadas tem por foco o objetivo de criação de banco de dados, onde cada banco tem suas particularidades, gerando, assim, formas de leitura e escrita mais ágeis [28] [29].

De forma coesa, o orientado a grafo tem pré-requisitos (nós, arestas e atributos). Logo, o modelo orientado a grafo é indicado em cenários de consultas complexas, apresentando alto desempenho e gerando uma melhora nas aplicações de alto grau de relacionamento.

Entre os mais importantes bancos de dados não relacionais orientados a grafos tem-se: *Neo4j*, *GraphDB*, *OrientDB* [29].

Em relação à distribuição de dados, alguns sistemas proporcionam o particionamento e a replicação dos dados, e há outros que deixam tal função para o cliente. No entanto, grande parte das soluções é distribuída, como, por exemplo, o *MongoDB* ou o *CouchDB*, que promovem a replicação das informações contidas no banco.

Para melhor consistência de um banco de dados, tem-se o paradigma denominado BASE (*Basically Available, Soft state, Eventual consistency*) [30], sendo basicamente uma forma de distribuir os dados em diferentes localidades, tornando sempre disponível e não se preocupando com a consistência.

A Figura 2.2, a seguir, apresenta mais claramente as propriedades e onde elas se encaixam no contexto de banco de dados. Neste sentido, têm-se alguns exemplos de bancos, sendo utilizados atualmente dentro do contexto ACID e BASE.

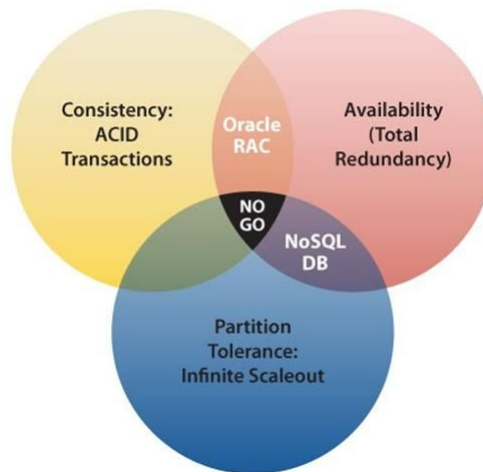


Figura 2.2: Teorema ACID e BASE.

Logo, o BASE proporciona a consistência eventual dos dados, e o ACID proporciona a consistência absoluta dos dados, não gerando flexibilidade. O primeiro paradigma, segundo a explicação de Brito [31]:

“[...] caracteriza por ser basicamente disponível, um sistema não precisa estar necessariamente em estado consistente o tempo todo, ele se tornaria consistente no devido momento. ”

Assim, quando das tecnologias NoSQL, a prevalência dos bancos de dados orientados a grafos (incluindo o Neo4J) utilizam o modelo ACID para assegurar que seus dados estejam protegidos e, coerentemente, armazenados [32].

2.4.1 Grafos

Atualmente, é cada vez maior o volume de dados para mapear e gerar redes mais conectadas e organizadas. Neste sentido, têm-se desafios para melhorar a manipulação de dados, sendo necessário tornar o processo mais compacto e leve, aliado à capacidade de suporte nos *hardwares* e melhor desempenho.

Os grafos, já de forma estruturada, são utilizados para se tornar úteis de diversos modos. No entanto, tem-se uma provocação na área das tecnologias, como, por exemplo, a formação e definição de dados em forma de grafos, bem como a geração de uma interconexão de dados com arestas e vértices, sendo algo significativo nos estudos para outras áreas de conhecimento.

Diante do exposto, faz-se importante o estudo dos grafos para os bancos de dados, por ser uma ferramenta que bem se entrelaça com as inter-relações dos dados. É muito utilizado, por exemplo, em redes sociais (*Facebook*, por exemplo), paginação da *web*, e até mesmo em redes metabólicas, na área da biomedicina (com as relações de mapas genéticos) e também na área de química (com estudos em estruturas químicas) [33].

Tem-se que o grafo orientado, é também denominado quíver ou dígrafo, e que similarmente, é por definição uma relação matemática de conjuntos de objetos denominados vértices, arestas ou mapas [34].

Os grafos são reproduzidos em vértices e arestas, apresentando maior facilidade de criação de uma estrutura modelada, variando o espaço de armazenamento com melhor qualidade de desempenho. Com essa estrutura montada, suas conexões e relacionamentos são concatenadas entre os objetos gerados. Por definição, cada nó dispõe um par de chave/valor, gerando um caminho para melhor navegação nos dados [28].

O Sistema Gerenciador de Banco de Dados (SGBD) em grafos contém um conjunto de manipulação de dados, onde se tem a criação, atualização, consulta e destruição de dados. Normalmente são essas operações otimizadas para desempenho transacional, com controle de integridade e garantia de disponibilidade. As informações são armazenadas na forma de entidades (vértices), relacionamentos (arestas) e propriedades.

Em relação ao desempenho de banco de dados em grafo, destacam-se três aspectos importantes, segundo Kemper [35]:

- Desempenho: bancos consistentes, e seu desempenho se mantém linear, invariavelmente do volume de dados trabalhado;
- Flexibilidade: são seguramente aditivos, quer dizer, é possível a adição de novos nós, novos relacionamentos, sem afetar as aplicações existentes;

- Agilidade: ajustado com padrões iterativo e incrementais utilizados na construção do banco, não requer esquema predefinido dos grafos e oferece maior facilidade para testes.

Os bancos de dados em grafo se destacam na aplicabilidade de interconectividade [36]. Os exemplos das utilidades são:

- Grafo de redes sociais: coleciona informações para melhorar as relações entre os indivíduos.
- Sistema de recomendações: com o sistema inteligente, ele faz com que os usuários sejam sugeridos sobre produtos, a partir da previsão com base em vários tipos de informação do cliente; e
- Bioinformática: uso intenso para mapeamento de informações complexas, como, por exemplo, o gene humano, cadeias enzimáticas e protéicas.

Tal ferramenta auxilia muito bem em questão de análise de dados, com a possibilidade de visualização na forma de desenhos. Logo, se observa que o grafo pode ajudar a entender melhor os dados abertos do PBF e, assim, colocando-se à disposição da comunidade.

2.4.2 NEO4J

Neo Technology é a empresa responsável pela criação do Sistema Gerenciador de Banco de Dados com base em grafo Neo4J. O código é aberto à comunidade para que se possa ser modelado e melhorado de acordo com a necessidade. Atualmente, o Neo4j possui o maior ecossistema de qualquer banco de dados em relação a grafos, com mais de 2.000.000 *downloads*. Sua versão Enterprise suporta alta disponibilidade de *clustering*, fornecendo característica de banco de dados completo, englobando o ACID, e oferece como ressaltado pelo CEO da Neo Technology denominado de Emil Eifrem: "*run-time*, o ambiente de transações em tempo real". Então, OLTP (*Online Transaction Processing* ou Processamento de Transações em Tempo Real) e outros casos de uso de missão crítica [37].

O CEO da Neo Technology acredita que o modelo cognitivo mais poderoso para o desenvolvimento de relações entre tipos de dados é o *whiteboard* [38]. Ou seja, quando um indivíduo logra passar seu conhecimento, sua ideia, em um quadro branco, não há limites para compor desenhos e grafos em um programa. Para ele, o Neo4j imita o modelo gráfico que *whiteboard*. Segundo o CEO da empresa, ele assim declarou sobre o desempenho de consulta: "O desempenho da consulta com conjuntos de dados conectados pode ser literalmente 1000 vezes mais rápido do que bancos de dados tradicionais, uma vez que é um banco de dados gráfico nativo". As redes sociais, com base na gestão de

identidade e acesso, no roteamento, na análise de dependência e de detecção de fraude em aplicativos, adotaram todas DBs grafo devido à sua velocidade e facilidade do uso do banco [39].

O grafo é implementado de forma íntegra no NoSQL denominado Neo4j. Tal banco de dados não relacional tem a total capacidade de criação de grafos e, assim, manipula, de forma íntegra, obtendo resultados por meio de grafos.

O Banco de Dados Não Relacional Neo4J é robusto, apresentando alta funcionalidade em relação a grafos, além de uma eficiência para uma consulta de muitos relacionamentos. Sua versão do banco está liberada para muitos sistemas operacionais, e é alicerçado em armazenamento e processamento nativo. Como todo banco de dados tem uma linguagem, o Neo4j não é diferente; sua linguagem é denominada *Cypher* e tem uma semelhança com SQL.

Como tem base em grafos, sua especialidade é voltada para redes sociais, tendo prioridade em relação à privacidade e melhor algoritmo de recomendações para os usuários. Com uma pequena delimitação do Neo4J, e por ser um banco de dados distribuído, sua versão paga é chamada de versão *Enterprise*. Assim, seu acesso é possível por computadores diferentes, englobando a possibilidade de uma rede de *cluster*. Logo, o Neo4J é um *open source* muito utilizado entre as empresas, gerando, assim, processos de negócio muito visado pelas companhias.

Há uma ferramenta que auxilia o Neo4j, que é o *Neo4j Shell*. Ele ajuda na manipulação de dados por via *prompt* de comando. Tal ação aproxima bastante os desenvolvedores que estão acostumados a usar este tipo de ferramenta. Existem usuários e desenvolvedores que preferem mais o lado gráfico. Outra ferramenta que auxilia muito bem é o *Neoclipse*, possibilitando, assim, a visibilidade gráfica dos nós e os relacionamentos direto do banco de dados construído.

Um quesito muito importante que deve ser observado é a usabilidade desse banco não relacional, uma vez que ele peca em alguns aspectos, como, por exemplo, a multiplataforma. Neste sentido foi desenvolvido o *Neo4J embedded*. O desenvolvedor pode adicionar um arquivo *.jar* dentro do projeto no qual se inclui e, assim, utilizá-lo como uma espécie de biblioteca dentro do próprio projeto, facilitando a multiplataforma.

Para aqueles que não optarem por java, tem a distribuição com a versão *server* chamado de *Neo4J Server*, fazendo chamadas *REST* e permitindo o uso de outras plataformas que não sejam implementadas em java.

Com as aplicações e diversidades aqui apresentadas, o Neo4j se torna uma aplicação de NoSQL altamente viável e rentável de forma íntegra, sendo direta para a maioria das aplicações, e percorrendo uma grande quantidade de melhorias para os desenvolvedores e usuários.

Capítulo 3

Estudo de Caso

O presente capítulo trata do material utilizado para a realização da pesquisa, além do desempenho de máquina e a realização do banco de dados com base em grafo. Na Seção 3.1 tem-se a exposição da análise dos dados públicos dentro do Programa Bolsa Família (PBF) – ponto central da discussão, populando o banco com os dados do referido Programa, e gerando uma análise em base de grafos. A seguir, na Seção 3.2, indica-se a construção propriamente dita dos grafos.

3.1 Análise

O objetivo é apresentar os resultados do estudo de caso, implementando um banco de dados em grafo, fazendo uso da base de dados públicos do PBF. A proposta principal é levar a simplicidade de visualização dos dados do Programa em questão pelo Portal da Transparência do Governo Federal. Neste sentido, faz-se importante a visualização da construção de tal processo, ou seja, os grafos e a geração de dados do Portal Brasileiro de Dados Abertos.

3.1.1 Populando os Dados do Programa Bolsa Família no Neo4j

Para a consolidação do banco de dados com base em grafo, fez-se uso de dois computadores, sendo um do Laboratório de Bioinformática e Dados (LaBiD), disponibilizado pelo Departamento de Ciência da Computação (CIC) da Universidade de Brasília (UnB), e o outro sendo um computador doméstico. Os dois possuíam o sistema operacional *Windows 10 Enterprise*. O computador do laboratório tem o processador Intel(R) Xeon(R) CPU E31220 @3.10 GHz 3.09 GHz, e o computador de uso doméstico tem o processador Intel(R) Core(TM) i5-4200U CPU @1.60 HZ @2.30HZ. O computador do laboratório possui 12GB de RAM, e o computador doméstico possui 4GB de RAM.

Neste sentido, teve início no mês de janeiro de 2016 o acesso ao Portal da Transparência, para, assim, se concretizar a manipulação de dados.

De 100 registros introduzidos na ferramenta Neo4j, o tempo de conclusão da ação foi de 450 milissegundos. O experimento seguinte foi a ação de 500 linhas que rodaram com 1459 milissegundos. Posteriormente, com 100 mil registros, o tempo determinado é com pelo menos de 9 a 12 segundos, levando em consideração a não criação dos relacionamentos entre os nós. A Figura 3.1, a seguir, apresenta a velocidade e a quantidade de registros dos experimentos realizados. Aqui é possível constatar que a partir de 150 mil registros, o tempo de processamento aumenta consideravelmente.

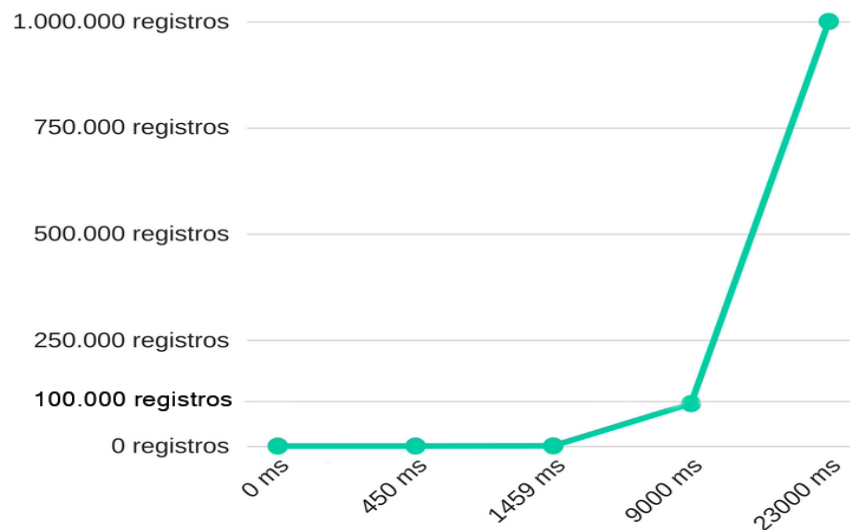


Figura 3.1: Registro X Tempo.

O primeiro grande desafio encontrado na presente pesquisa foi popular o Neo4j com os dados do PBF. Assim, quando se desejou inserir mais que 2000 mil nós, obtiveram-se problemas no Neo4j.

O próprio Neo4j possui uma interface onde é possível fazer a carga dos dados. A primeira tentativa se deu fazendo uso dos recursos do computador do LABID/CIC/UnB. No entanto, esperou-se a finalização da carga durante 72 horas, porém, sem sucesso. Foi possível observar que a inserção dos dados estava muito lenta, acarretando no estouro de memória do computador durante o processo.

Inicialmente, a técnica utilizada de inserção nó a nó dentro da plataforma Neo4j foi muito lenta. Com o erro, não foi possível intermediar os dados do PBF, uma vez que na base do referido Programa, a cada mês, tem-se mais de 7 milhões de registros.

Depois de pesquisas sobre ETL no Neo4j, obteve-se o comando denominado LOAD CSV dentro do *Cypher*. Depois de alguns testes na linguagem, verificou-se, com a referida consulta, a agilidade na inserção dos dados, tendo em vista a simplicidade de exportação de um arquivo .CSV para o Neo4j. Porém, para a carga do grande volume de dados do PBF, foi necessário implementar um programa para a carga dos dados no Neo4j. A linguagem concebida deste foi *Python*. De modo resumido, tal linguagem tem como característica, desde sua concepção, a legibilidade e produtividade, ou seja, foi criada pra produzir código útil e leve [40].

Primeiramente, para a utilização concisa do aproveitamento do programa implementado, fez-se necessário a conversão para a extensão .txt, a extinção da tabulação e a reconversão para o formato CSV, de aceitação do Neo4j, para, então, a execução de outro programa implementado que altere no formato almejado, ou seja, trocando pontuação e inserindo mais uma coluna, quando necessário.

No arquivo original não se tem a separação dos dados por regiões, dificultando a visualização e análise dos dados do PBF. Neste sentido, foi essencial implementar mais uma coluna, culminando na inserção dos dados em cada linha conforme a Unidade da Federação (UF). Para tanto, foi necessário implementar uma nova lógica do Programa.

Para melhor compreender a forma de popular o banco, a Figura 3.2, a seguir, evidencia um fluxograma com maiores detalhes.

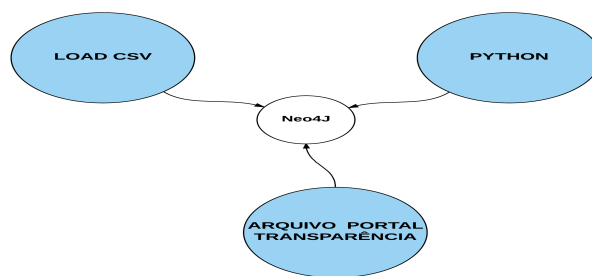


Figura 3.2: Fluxograma da população de dados.

Para teste inicial, o tempo total de execução do programa na máquina do LaBiD/CIC/UnB foi de três minutos para manipular 12 milhões de registros, sendo constatada uma dife-

rença entre a manipulação manual dos registros no Neo4j e sua manipulação completa no programa.

3.1.2 Dados Públicos do Programa Bolsa Família

Com a proposta de fazer uso de todos os dados do PBF, foi considerada a inserção de 2000 mil nós de uma vez. A Figura 3.3, a seguir, retrata um *notepad plus*, para demonstrar a consulta para criação dos nós para inserção no Neo4j. Já na Figura 3.4, a seguir, tem-se as *queries* de exemplo para a criação das relações, cujo objetivo é formar uma variedade, com execução no Neo4j.

```

1 CREATE (estado:Estado {sigla:"sigla", nome_capital:"capital", Regiao:"região"})
2 CREATE (alagoas:nomeEstado {estado:"Alagoas", sigla:"AL", Regiao:"Nordeste"})
3 CREATE (acre:nomeEstado {estado:"Acre", sigla:"AC", nome_capital:"Rio Branco", Regiao:"Norte"})
4 CREATE (amapa:nomeEstado {estado:"Amapá", sigla:"AP", nome_capital:"Macapá", Regiao:"Norte"})
5 CREATE (amazonas:nomeEstado {estado:"Amazonas", sigla:"AM", nome_capital:"Manaus", Regiao:"Norte"})
6 CREATE (bahia:nomeEstado {estado:"Bahia", sigla:"BA", nome_capital:"Salvador", Regiao:"Nordeste"})
7 CREATE (ceara:nomeEstado {estado:"Ceará", sigla:"CE", nome_capital:"Fortaleza", Regiao:"Nordeste"})
8 CREATE (distritoFederal:nomeEstado {estado:"Distrito Federal", sigla:"DF", nome_capital:"Brasília", Regiao:"Centro-Oeste"})
9 CREATE (esperitoSanto:nomeEstado {estado:"Espírito Santo", sigla:"ES", nome_capital:"Vitória", Regiao:"Sudeste"})
10 CREATE (goias:nomeEstado {estado:"Goiás", sigla:"GO", nome_capital:"Goiânia", Regiao:"Centro-Oeste"})
11 CREATE (maranhao:nomeEstado {estado:"Maranhão", sigla:"MA", nome_capital:"São Luis", Regiao:"Nordeste"})
12 CREATE (matoGrosso:nomeEstado {estado:"Mato Grosso", sigla:"MT", nome_capital:"Cuiabá", Regiao:"Centro-Oeste"})
13 CREATE (matoGrossoDoSul:nomeEstado {estado:"Mato Grosso do Sul", sigla:"MS", nome_capital:"Campo Grande", Regiao:"Centro-Oeste"})
14 CREATE (minasGerais:nomeEstado {estado:"Minas Gerais", sigla:"MG", nome_capital:"Belo Horizonte", Regiao:"Sudeste"})
15 CREATE (para:nomeEstado {estado:"Pará", sigla:"PA", nome_capital:"Belém", Regiao:"Norte"})
16 CREATE (paraiba:nomeEstado {estado:"Paraíba", sigla:"PB", nome_capital:"João Pessoa", Regiao:"Nordeste"})
17 CREATE (parana:nomeEstado {estado:"Paraná", sigla:"PR", nome_capital:"Curitiba", Regiao:"Sul"})
18 CREATE (pernambuco:nomeEstado {estado:"Pernambuco", sigla:"PE", nome_capital:"Recife", Regiao:"Nordeste"})
19 CREATE (piaui:nomeEstado {estado:"Piauí", sigla:"PI", nome_capital:"Teresina", Regiao:"Nordeste"})
20 CREATE (rioDeJaneiro:nomeEstado {estado:"Rio de Janeiro", sigla:"RJ", nome_capital:"Rio de Janeiro", Regiao:"Sudeste"})
21 CREATE (rioGrandeDoNorte:nomeEstado {estado:"Rio Grande do Norte", sigla:"RN", nome_capital:"Natal", Regiao:"Nordeste"})
22 CREATE (rioGrandeDoSul:nomeEstado {estado:"Rio Grande do Sul", sigla:"RS", nome_capital:"Porto Alegre", Regiao:"Sul"})
23 CREATE (rondonia:nomeEstado {estado:"R Rondônia", sigla:"RO", nome_capital:"Porto Velho", Regiao:"Norte"})
24 CREATE (roraima:nomeEstado {estado:"Roraima", sigla:"RR", nome_capital:"Boa Vista", Regiao:"Norte"})
25 CREATE (santaCatarina:nomeEstado {estado:"Santa Catarina", sigla:"SC", nome_capital:"Florianópolis", Regiao:"Sul"})
26 CREATE (saoPaulo:nomeEstado {estado:"São Paulo", sigla:"SP", nome_capital:"São Paulo", Regiao:"Sudeste"})
27 CREATE (sergipe:nomeEstado {estado:"Sergipe", sigla:"SE", nome_capital:"Aracaju", Regiao:"Nordeste"})
28 CREATE (tocantins:nomeEstado {estado:"Tocantins", sigla:"TO", nome_capital:"Palmas", Regiao:"Norte"})
29

```

Figura 3.3: Código teste do Create de nós Estados do Brasil.

```

1 --Execução do Código CYPHER, contendo os estados e suas relações com os favorecidos
2
3 --São Paulo
4
5 MATCH (saoPaulo:nomeEstado {estado:'São Paulo'})
6 MATCH (SP:Fv {sigla:'SP'})
7 CREATE (saoPaulo)-[:natural_de]->(SP)
8
9 MATCH (e:estado {estado:'Sao Paulo'})
10 MATCH (n:nome {sigla:'SP'})
11 CREATE (e)-[:relacionado]->(n)
12
13 --Bahia
14
15 MATCH (bahia:nomeEstado {estado:'Bahia'})
16 MATCH (BA:Fv {sigla:'BA'})
17 CREATE (bahia)-[:natural_de]->(BA)
18
19 MATCH (e:estado {estado:'Bahia'})
20 MATCH (n:nome {sigla:'BA'})
21 CREATE (e)-[:natural_de]->(n)
22
23 --Execução, contendo as relações de favorecidos com a região
24 --Só trocar a região e a sigla para ver outras regiões
25
26 --Acre
27
28 MATCH (Norte:rg {sigla:'AC'})
29 MATCH (AC:Fv {sigla:'AC'})
30 CREATE (Norte)-[:natural_de]->(AC)
31
32 MATCH (e:estado_total {sigla:'RN'})
33 MATCH (n:nome {Regiao:'Nordeste'})
34 CREATE (e)-[:natural_de_teste]->(n)

```

Figura 3.4: Códigos de testes do create de relações.

Neste sentido, os grafos foram gerados naturalmente sem restrição de erro de memória e com sua complexidade baixa. O processo elevou, então, para 2000 nós e, a partir daí, a execução ficou mais lenta, mas executando com sucesso. Os nós foram criados e sua execução apresentou o tempo de 17852 ms.

Na Figura 3.5, a seguir, tem-se um grafo com 999 nós, onde no centro há um círculo constando a UF do Acre e, em volta, os beneficiários do PBF.

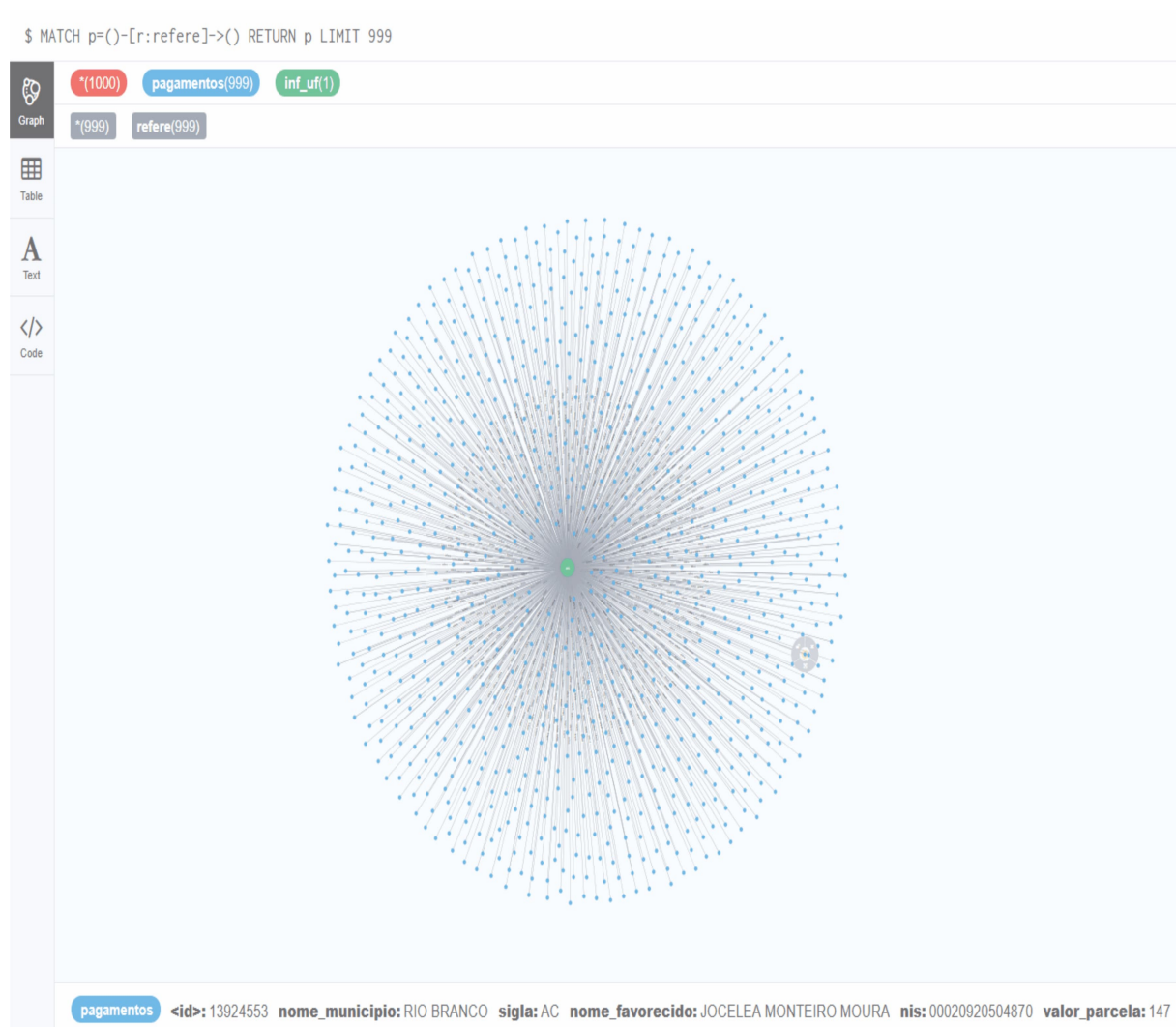


Figura 3.5: Resultado de um grafo de 999 nós.

A seguir, tem-se as representações existentes na Figura 3.5. Primeiramente, na parte superior, encontram-se os nós e as referências. Tem-se ali, de forma simples, a consulta utilizada para evidenciar o resultado em grafo. Já na parte superior da Figura 3.6, a seguir, tem-se a quantidade de nós no total (em vermelho), os nós voltados apenas os beneficiários (em azul) (pagamentos) e o nó restante (em verde) designado para a UF escolhida; Por último, os nós na cor cinza se referem ao número das relações:

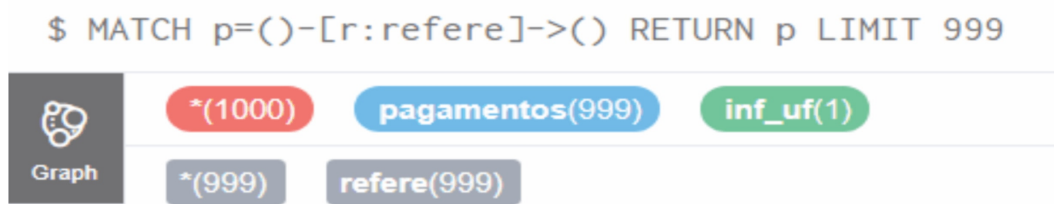


Figura 3.6: Resultados superior de um grafo de 999 nós.

A parte inferior é a característica do nó do grafo, conforme exposto na Figura 3.7, a seguir, onde se tem várias informações apresentando o ID único para cada beneficiário, a saber: nome do Município e a sigla de localização do favorecido, nome completo do usufruidor, seu Número de Identificação Social (NIS) (único, que caracteriza cada beneficiário) e o valor da parcela recebida como benefício mensal.



Figura 3.7: Resultados inferior de um grafo de 999 nós.

Como se tem o nó no centro das informações do grafo, ali pode constar as instâncias, tais como: nome, cidade, Estado, região, País, continente, ou o que se almeja. Na Figura 3.8, a seguir, tem-se explícito o nó em laranja (Estado de São Paulo do Brasil).

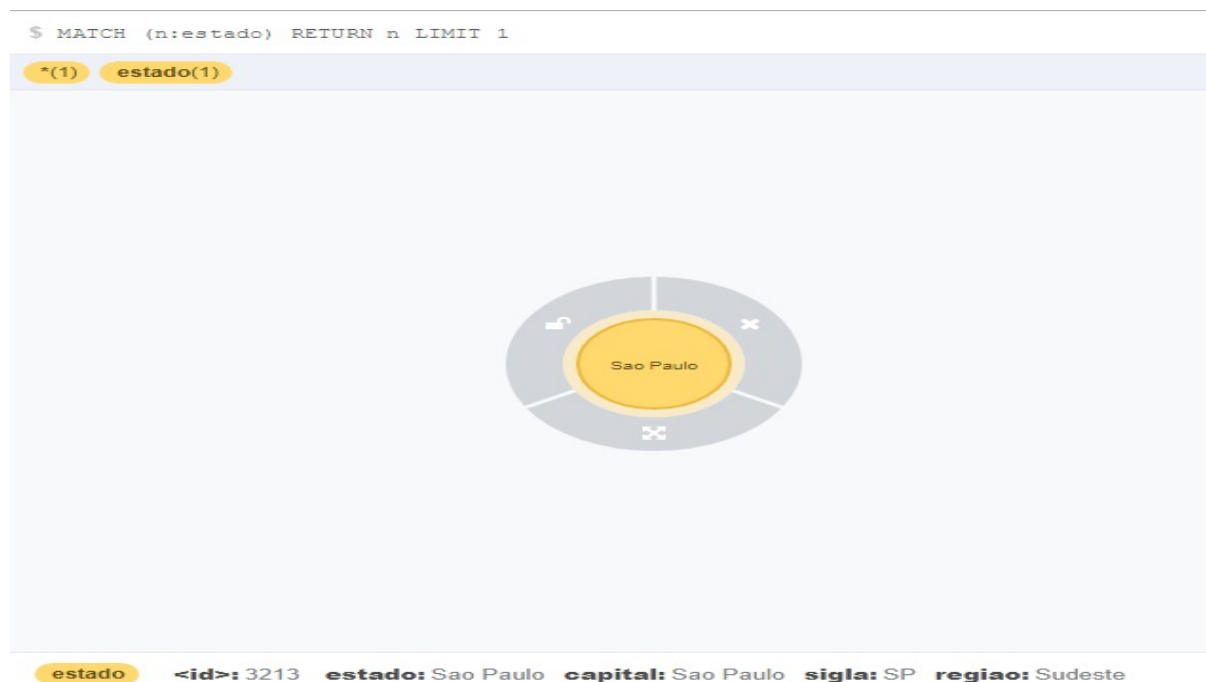


Figura 3.8: O nó (SP).

Para melhor visualização do grafo, em menor quantidade, tem-se para o limite de 25, conforme expresso na Figura 3.9, a seguir, sendo o verde o Estado do Acre, e os nós em azul os beneficiários do PBF.

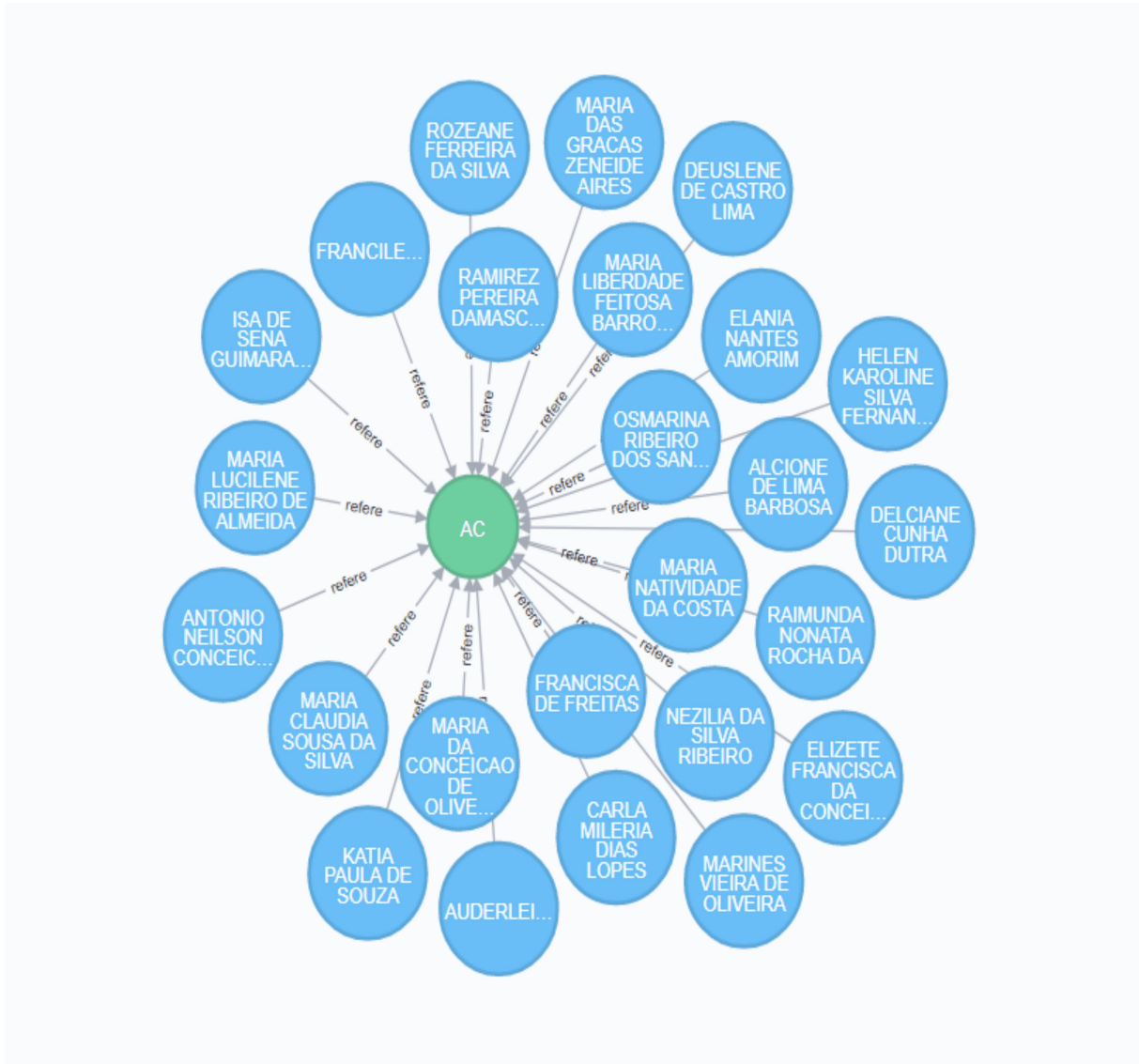


Figura 3.9: Resultado de um grafo de 25 nós com nomes.

3.1.3 Código em Python

Conforme demonstrado na Figura 3.10, tem-se uma parte do código implementado em *Python* – uma forma pensada, quando o primeiro muda a extensão do arquivo e mapeia onde está a tabulação. A íntegra deste encontra-se em Anexo I.

```

with open('pagamentos.csv', "r") as my_input_file:
    for line in my_input_file:
        line = line.split(',', 0)
        text_list.append(" ".join(line))

```

Figura 3.10: Parte do *script* em *python*.

Na Figura 3.11, está outro trecho do programa conversor de extensão .txt para CSV, com suas alterações no arquivo, sendo mapeada cada região, com sua respectiva UF, e trocando pontuações. O restante dos dados encontra-se em Anexo I.

```

with open('paga.txt', 'r') as f:
    # primeira linha
    line = f.readline()
    line = ','.join(['%s'%field.strip() for field in line.split('\t')])+'\n'

```

Figura 3.11: Parte do *script* em *python*.

O arquivo CSV, em conformidade com o formato do Neo4j, foi importado para o banco. Fazendo uso do LOAD CSV do banco de dados não relacional, e de uma pasta pré-definida pelo banco denominada *import*, é preciso apenas alocar o arquivo dentro da pasta em questão e chamar com a função para a importação de dados do banco. A Figura 3.12, se refere ao exemplo da consulta utilizada no momento da importação, com base no LOAD CSV suportado pelo Neo4j. Para melhor visualização do *script*, vide Anexo I.

```

USING PERIODIC COMMIT 1000
LOAD CSV WITH HEADERS FROM 'file:///pagamento_certo.csv' AS pagamentos
CREATE (:inf_uf_ac {sigla: pagamentos.uf, nome_municipio: pagamentos.nome_municipio, nome_favorecido:
pagamentos.nome_favorecido, valor_parcela: ToInt(pagamentos.valor_parcela), regioao: pagamentos.regiao})

```

Figura 3.12: Código LOAD CSV (NEO4J).

Por fim, a inserção da base do mês de janeiro de 2016 do Portal da Transparência possui um tempo consideravelmente rápido. Neste sentido, apenas 13 minutos foi o tempo necessário para importar toda a base com os 14 milhões de registros. Com os dados importados e com o Neo4j rodando sem problemas na máquina, é possível avançar para a próxima fase: a geração da base de grafos.

3.2 Grafos

3.2.1 Criação dos nós

A Figura 3.13, a seguir, apresenta apenas 300 nós dos 12 milhões importados no banco. O Município é uma das chaves dos nós, e possui um ID como modo único de caracterização de cada nó no grafo, facilitando as buscas. O ID se dá automaticamente pelo próprio banco de dados. Na Figura em questão, é possível observar como é convidativo ver os nós e a atenção do que ali se encontra expresso.

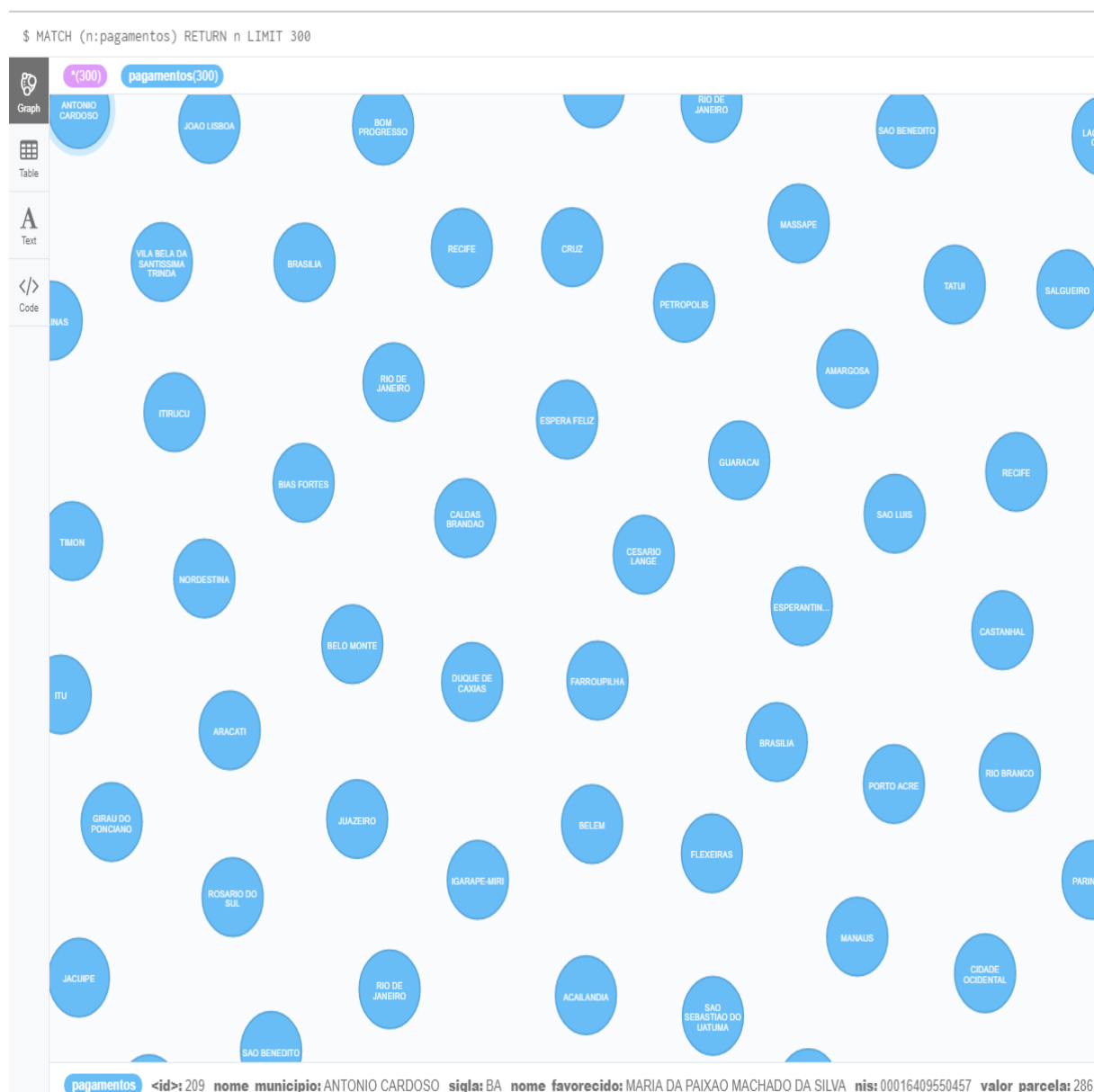


Figura 3.13: Grafo com nós de nomes das capitais do País.

De forma mais ampliada, a Figura 3.14, a seguir, apresenta os nós:



Figura 3.14: Grafo com nós de nomes de algumas capitais (ampliado).

A seguir, na Figura 3.15 destaca todos os 27 Estados aqui retratados:

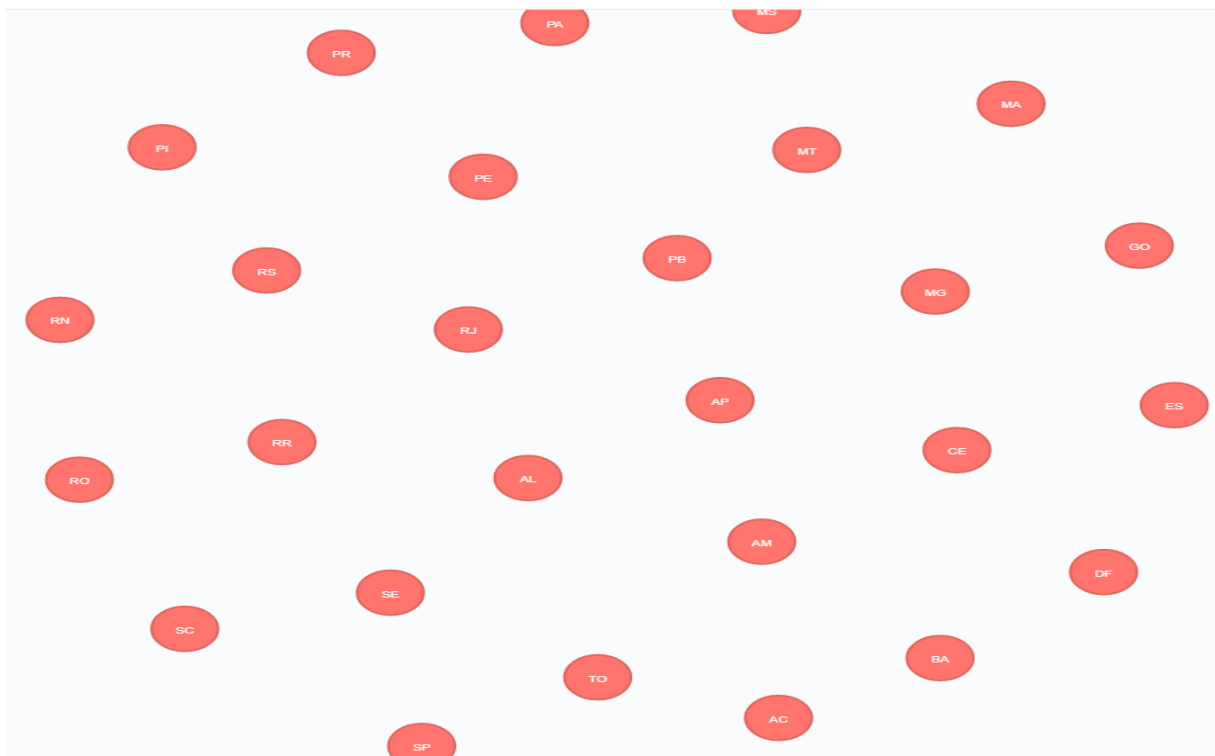


Figura 3.15: Grafo com nós de UF.

Para melhor visualizar, de modo ampliado:

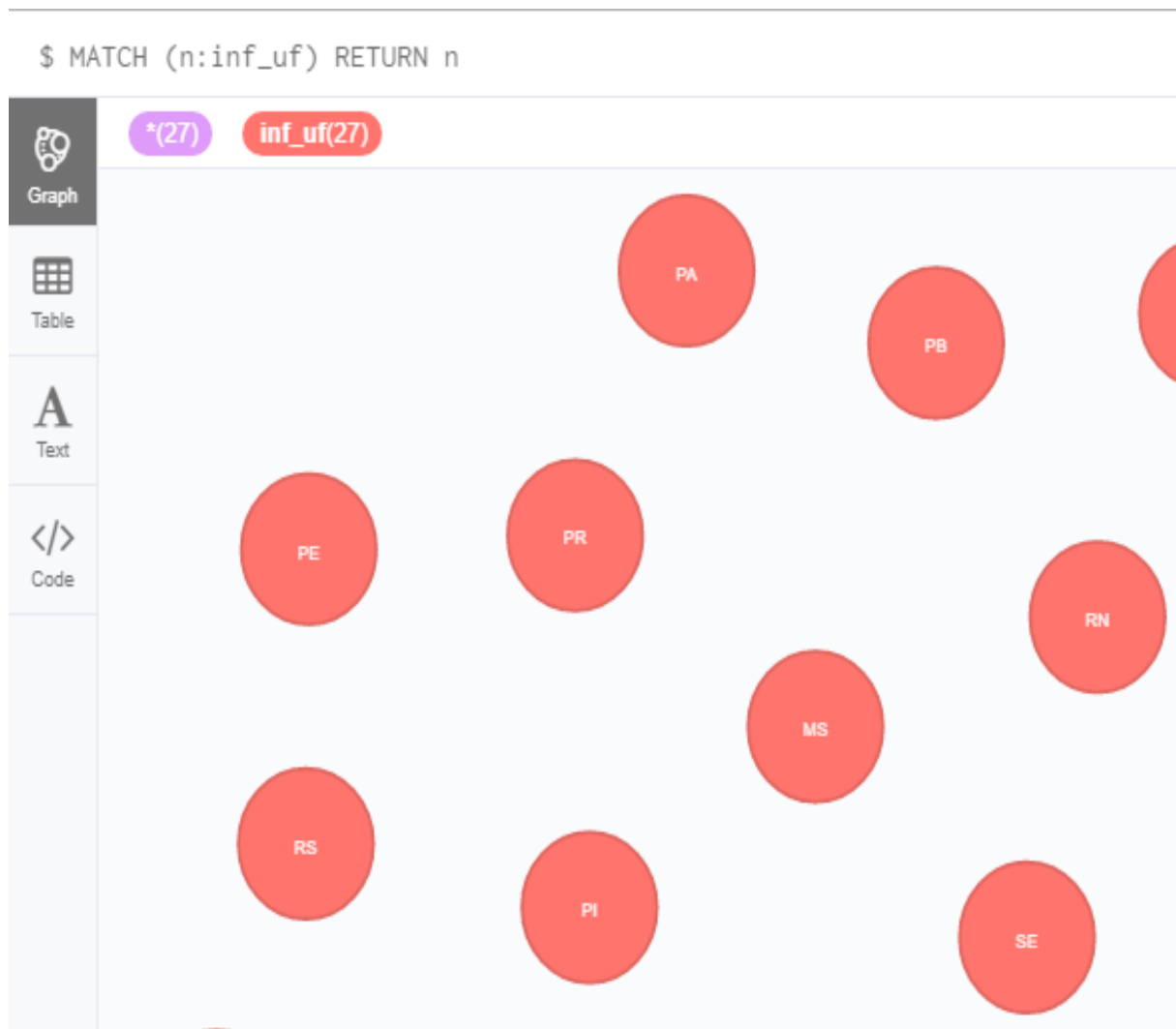


Figura 3.16: Grafo com nós de UF (ampliado).

A criação de nó se dá quando da importação dos dados do PBF para o banco, já com o *script* do *Cypher*. Tal ação é automática e, para a consulta – rápida e intuitiva – é possível fazer uso de cores e organização. Em seguida tem-se um pequeno trecho do código utilizado para importação e criação dos nós, conforme exposto na Figura 3.17, a seguir.

```
1 USING PERIODIC COMMIT 1000
2 LOAD CSV WITH HEADERS FROM 'file:///pagamentos_total.csv' AS
3 pagamentos
4 CREATE (:pagamentos {sigla: pagamentos.uf, nome_municipio:
```

Figura 3.17: Código para criação de nós.

Neste sentido, na linha 3, **pagamentos** é o nome da base que foi importado do PBF. No trecho em questão se retorna apenas 300 nós a livre escolha do banco.

Inicialmente, é possível notar muitas diferenças no que tange à visualização dos dados, como, por exemplo, o desenho dos dados. Portanto, sendo o desenho de grafos melhor identificável e visualizado, logo, se aprimora o modo de visualização destes no PBF.

3.2.2 Criação das arestas (Relationship Types)

A criação das arestas no Neo4j é um processo complexo. A forma mais simples é colocar dois nós, ou seja, um já pré-definido com as arestas (**unidades federativa** e as **regiões** do Brasil), e outro com a importação completa do PBF. Assim, é possível a manipulação dos dados em melhores condições. O código exposto na Figura 3.18, a seguir, se deu para a consulta de uma Região específica de um nó. E, a partir da base completa que se tem a nomeação dos pagamentos do Programa em questão. Com base em tal procedimento, foi possível obter o resultado da relação entre o primeiro e o segundo nó.

```
MATCH (e:inf_uf {regiao: 'Norte'})
MATCH (n:pagamentos {uf: 'AC'})
CREATE (n)-[:refere]->(e)
```

Figura 3.18: Código de criação.

A Figura 3.19 demonstra como pode-se ter maior profundidade em uma consulta no Neo4j.

```
MATCH (e:inf_uf {regiao: 'Norte', sigla: 'AC'})
MATCH (n:pagamentos {uf: 'AC'})
CREATE (n)-[:refere]->(e)
```

Figura 3.19: Código de criação com mais detalhe.

Porém, faz-se importante observar que os Estados que são maiores por sua extensão territorial e com maior densidade populacional consequentemente apresentarão mais beneficiados pelo programa.

Nas arestas em questão, é possível observar a visualização de vários modos. A Figura 3.20, a seguir, trata dos nós com suas relações, ou seja, os nós em azul são os Municípios da região Norte (especificamente, do Estado do Acre), e o nó em vermelho é a UF do Acre. A Figura 3.24, a seguir, apresenta as mesmas informações da anterior, porém, com ênfase nas relações. Elas se encontram em cor lilás, a fim de evidenciar sua participação nesta imagem. A Figura 3.26, a seguir, também detém maior ênfase na parte das relações, porém, com uma ressalva maior no nó central, onde agora se expressa a região (Norte). A Figura 3.28, a seguir, demonstra de modo evidente os nós da cor azul com seus respectivos nomes (dos Municípios) e, no centro, a UF em questão. Tem-se nas Figuras 3.22 e 3.23, é mais um exemplo para que se possa visualizar a variedade que se pode mostrar um grafo. As Figuras 3.21, 3.25, 3.27, 3.29, são para uma melhora na visualização das imagens de ângulo mais afastado, nisso foi aplicado um *zoom* para melhor visibilidade das figuras.

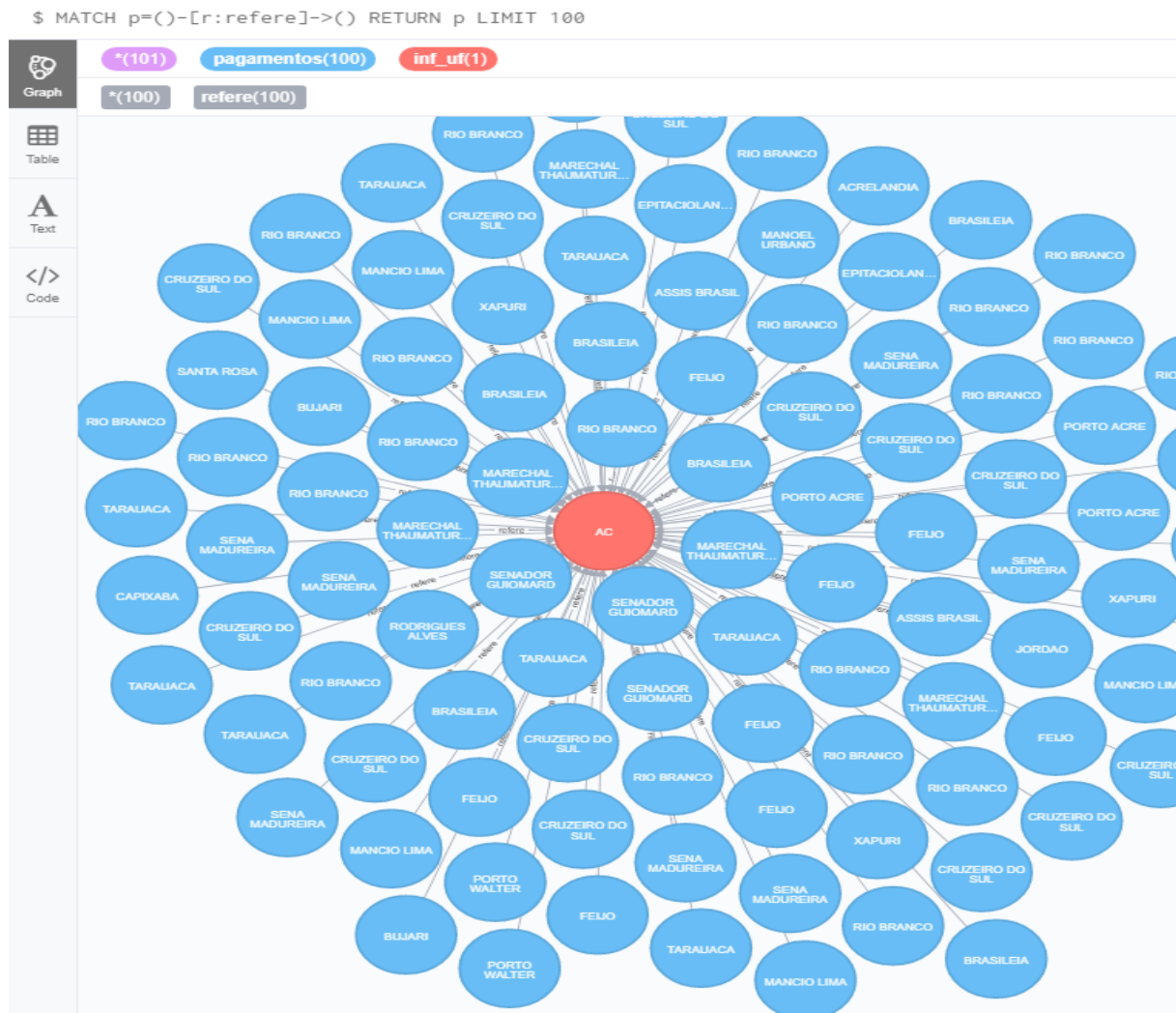


Figura 3.20: Grafo com arestas com 101 nós - Acre.

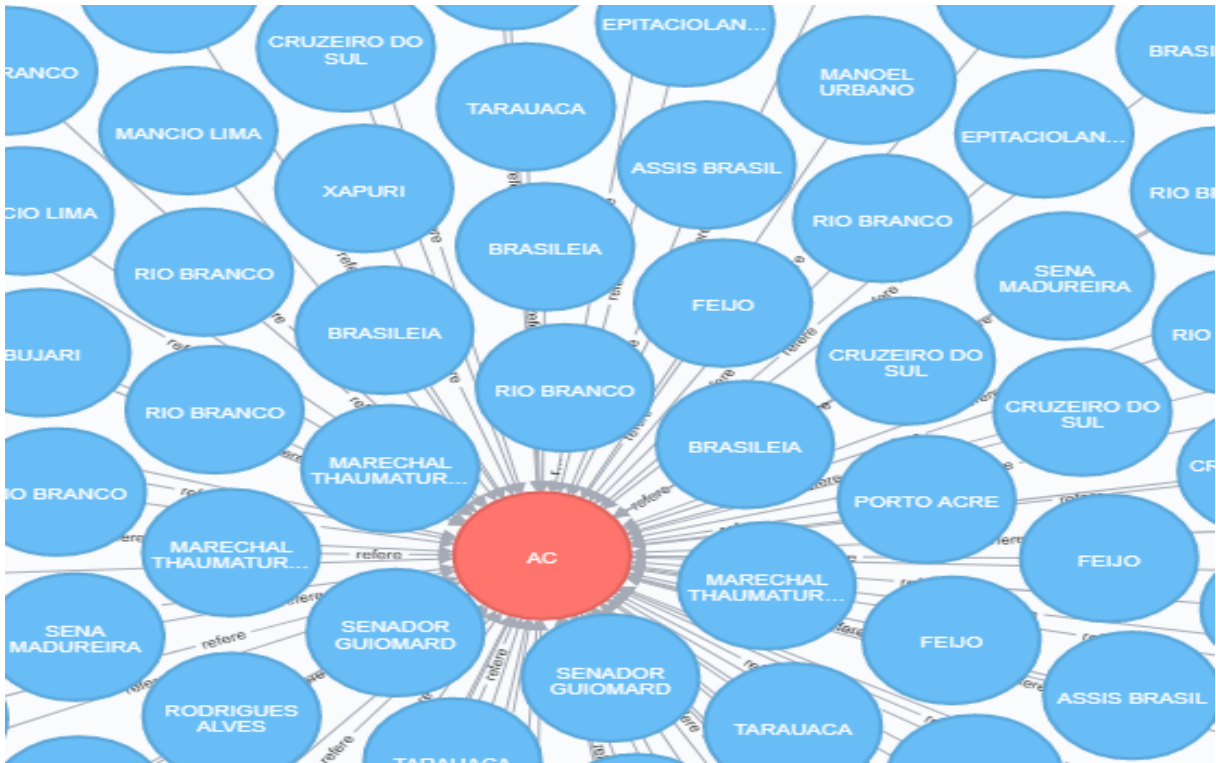


Figura 3.21: Grafo com arestas com 101 nós com zoom - Acre.



Figura 3.22: Grafo do capital Rio Branco e seus beneficiário com 25 nós.

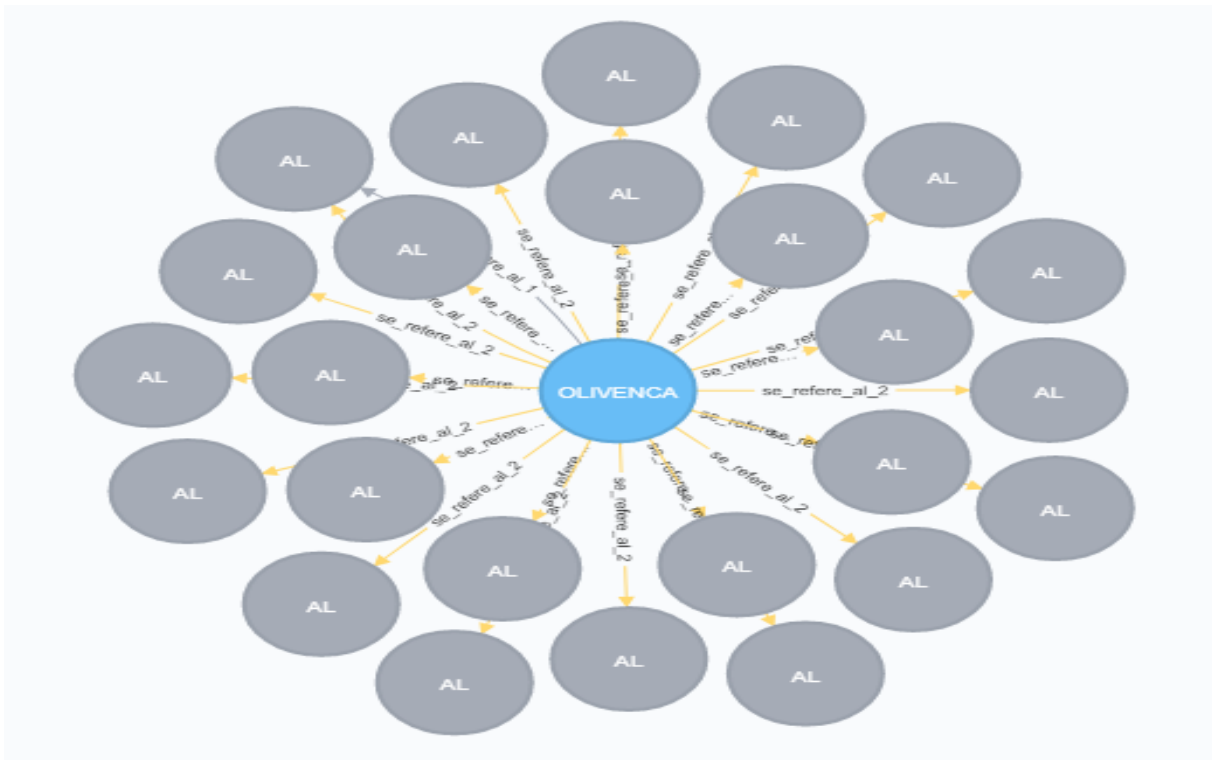


Figura 3.23: Grafo do município Olivença de Alagoas com 25 nós.

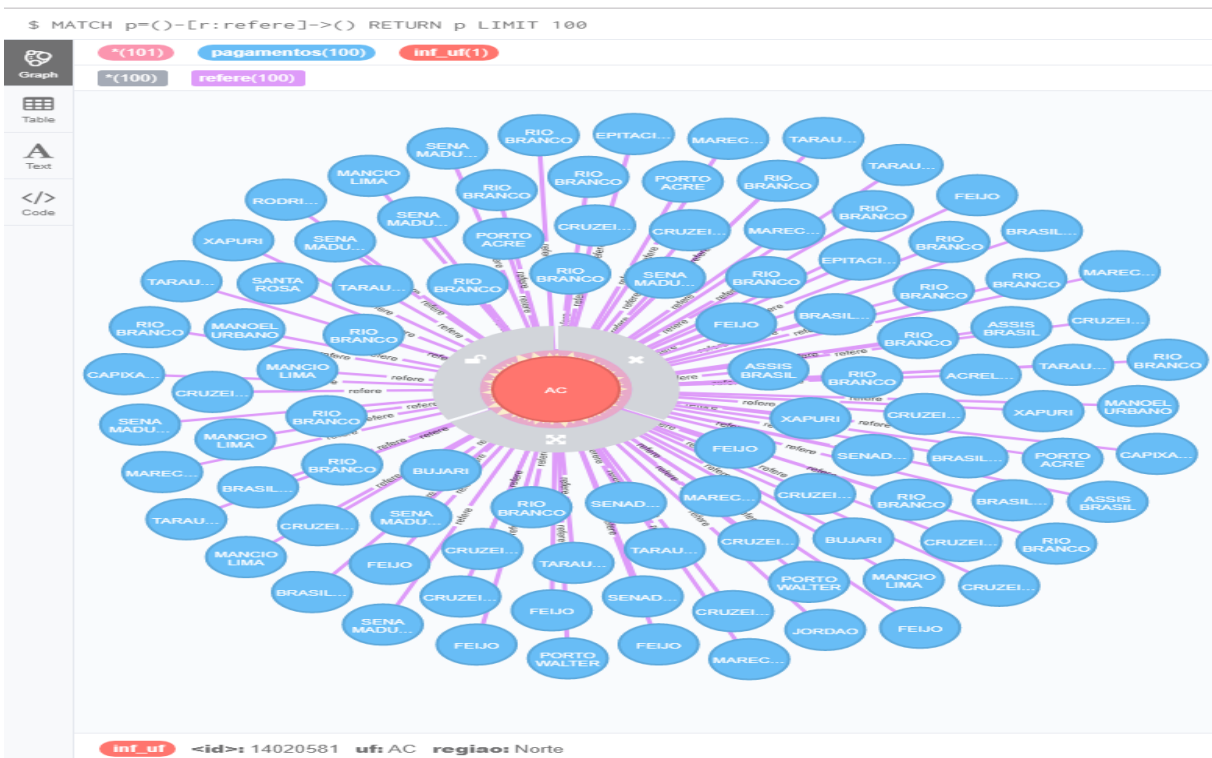


Figura 3.24: Grafo com arestas com 101 nós com ênfase nas relações - Acre.

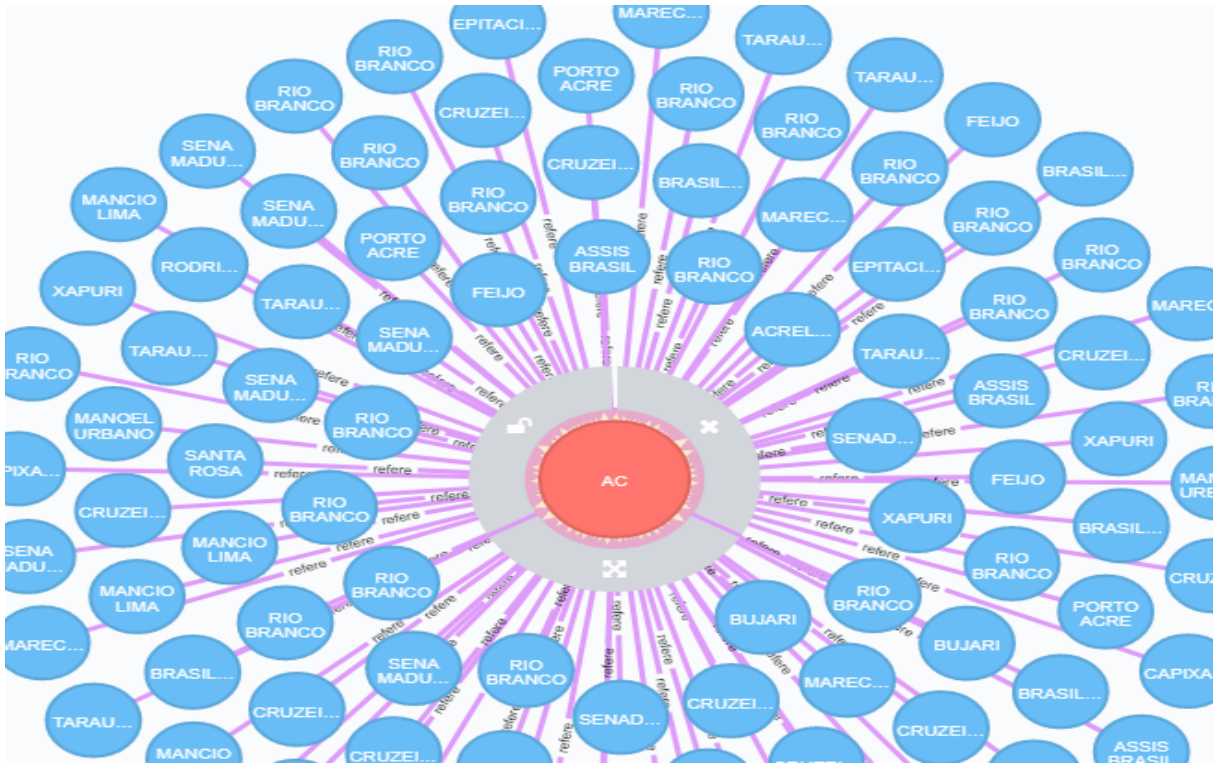


Figura 3.25: Grafo com arestas com 101 nós com ênfase nas relações com zoom - Acre.

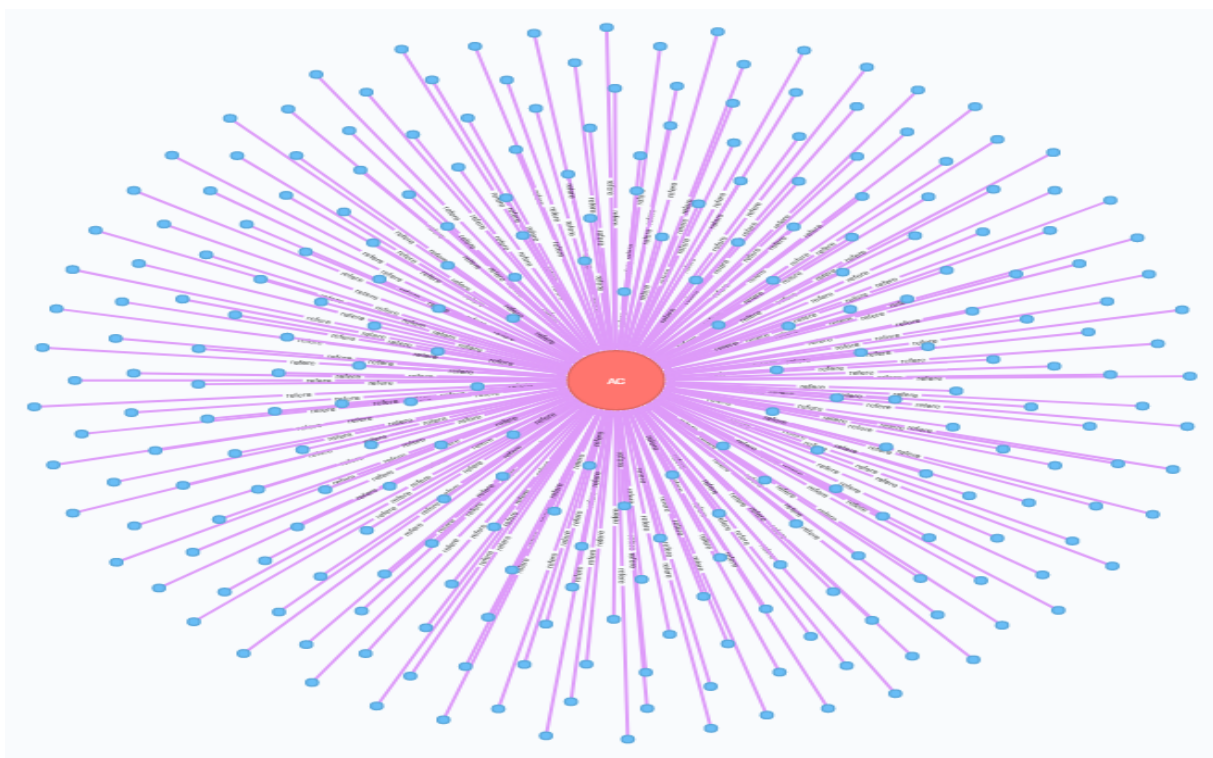


Figura 3.26: Grafo com arestas com ênfase nas relações com nós em azul - Acre.

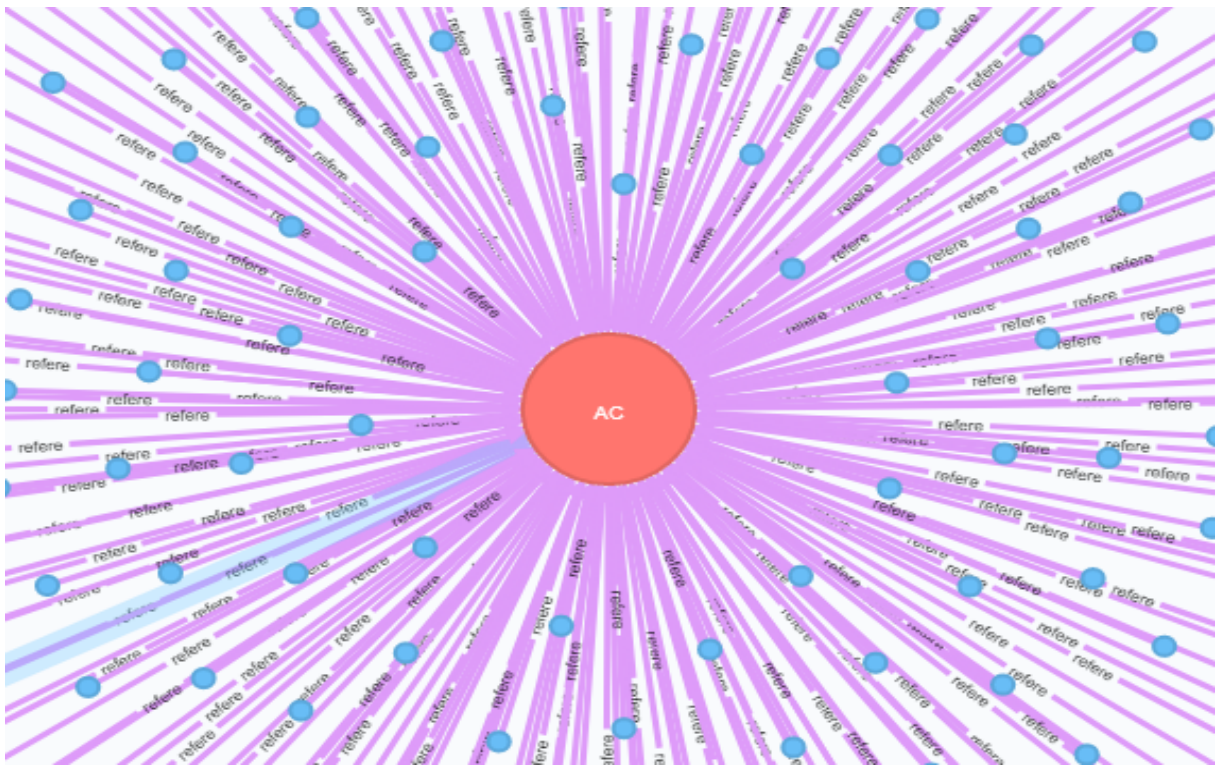


Figura 3.27: Grafo com arestas com ênfase nas relações com nós em azul com zoom - Acre.

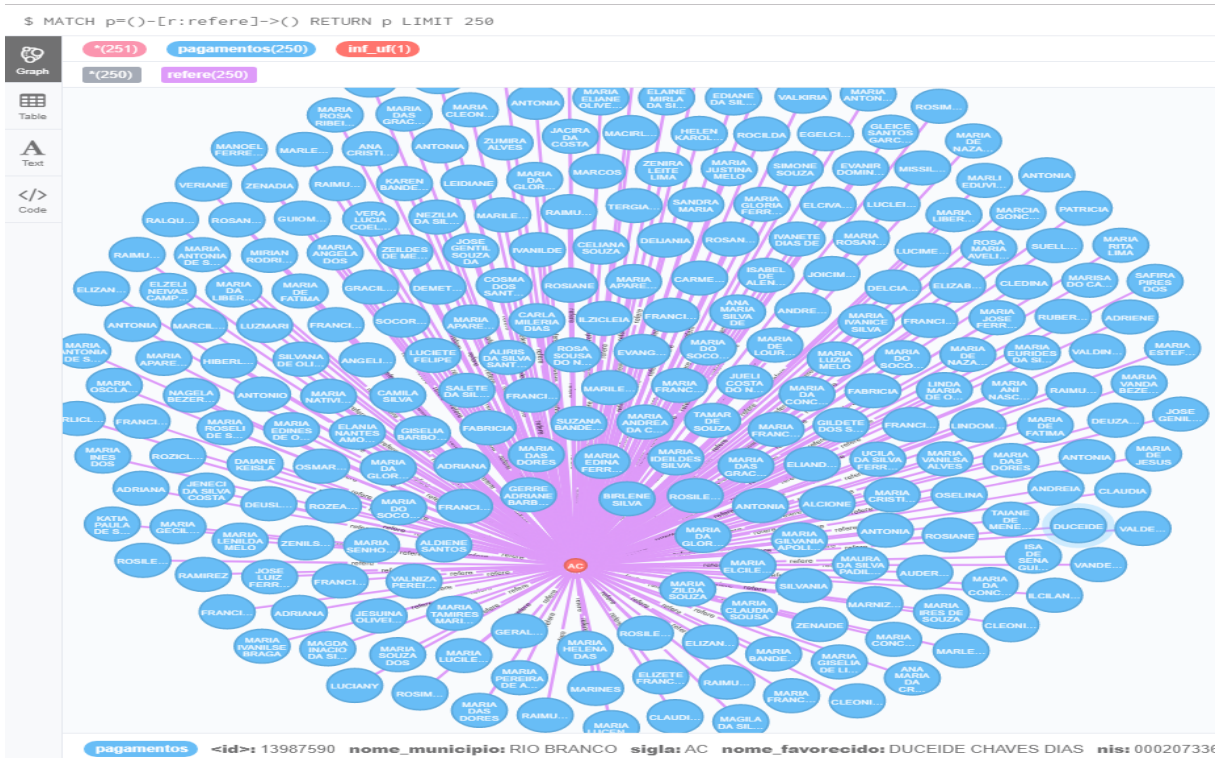


Figura 3.28: Grafo com arestas com 251 nós - Acre.

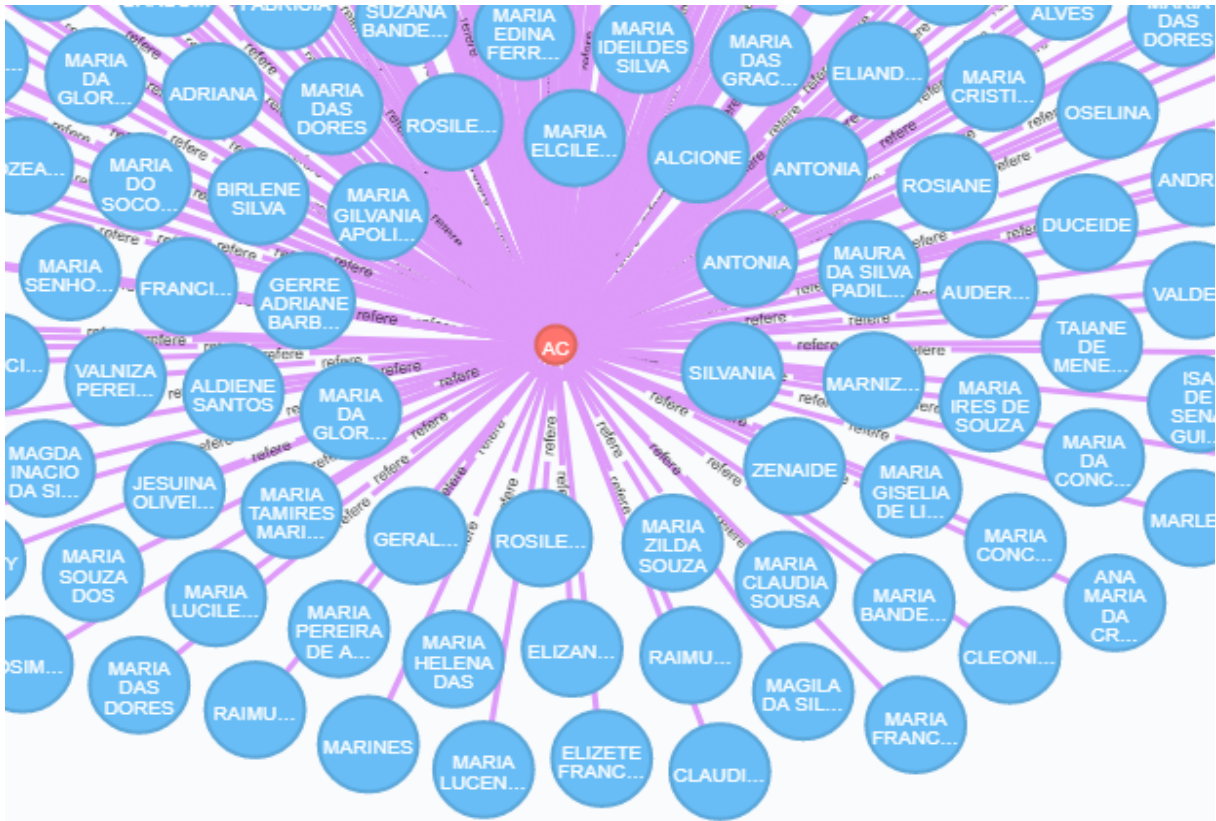


Figura 3.29: Grafo com arestas com 251 nós com zoom - Acre.

Por questões de limitações de *hardware*, de todas as UF's do Brasil em relação à quantidade de dados do PBF, somente é possível visualizar as menores unidades administrativas, a saber: **Acre, Amazonas, Alagoas, Amapá, Distrito Federal, Espírito Santo, Goiás, Maranhão, Paraná, Rio Grande do Norte, Rio Grande do Sul, Rondônia, Roraima, São Paulo, Santa Catarina, Sergipe, Tocantins.**

Para diminuir os problemas de processamento relacionados com o *hardware*, tentou-se aprimorar a consulta com consultas mais restritas e melhor desenvolvidas. Com um aprimoramento da busca, têm-se ainda alguns Estados que não foram inseridos no banco de dados devido ao equipamento apresentar baixa limitação, quais sejam: **Bahia, Ceará, Mato Grosso, Mato Grosso do Sul, Minas Gerais, Pará, Pernambuco, Paraíba, Piauí, Rio de Janeiro.**

Com consultas mais complexas, como, por exemplo: a busca pelos maiores valores de cada Estado; quem foi o indivíduo que ganhou mais no Estado de São Paulo; ou, qual foi o valor ganho no mês de janeiro de 2016, foi possível em qual Estado do Brasil se ganha mais, por exemplo, para valores maiores que R\$ 950,00 (novecentos e cinquenta reais) no Estado de São Paulo, no período do mês de janeiro do ano de 2016, conforme expresso nas Figuras 3.30 e 3.31. A Figura 3.30 demonstra nos nós em azul o nome do favorecido

com uma bolsa maior que R\$ 950,00 (novecentos e cinquenta reais), e o nó em verde é a UF onde se encontram os beneficiados. Já na Figura 3.31, o nó em azul aponta o valor em dinheiro, e o nó em verde é a UF em questão:

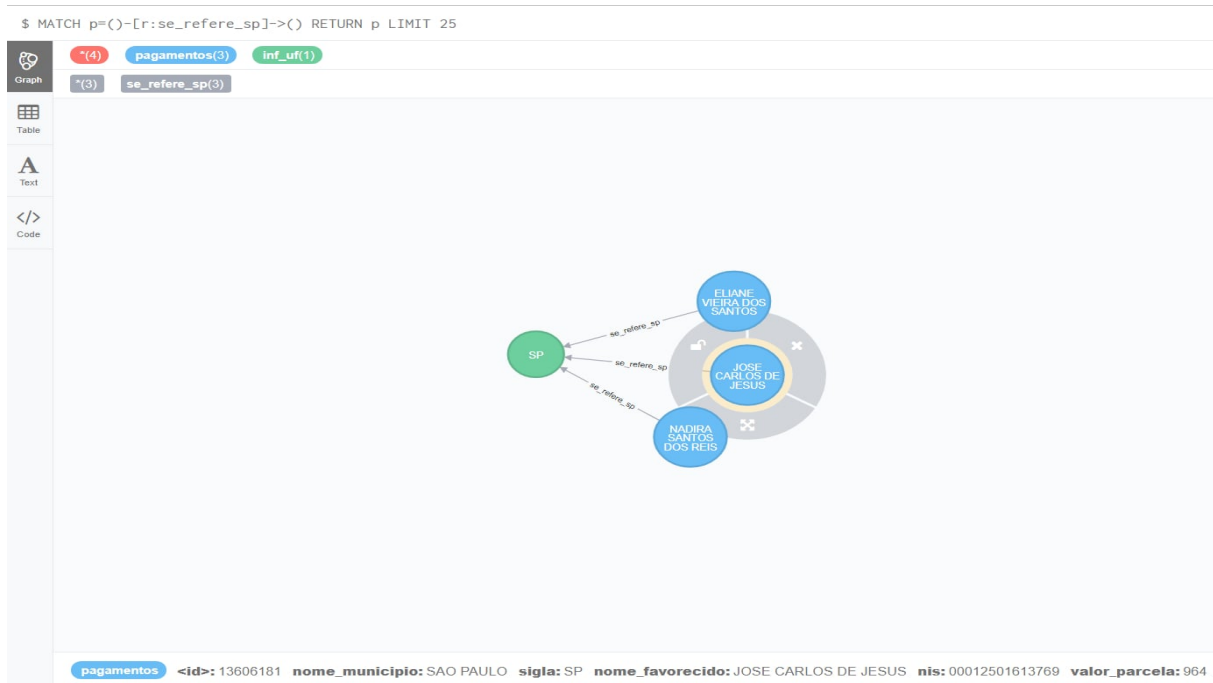


Figura 3.30: Grafo Jan/2016 mais que R\$950.00 , nomes - SP.

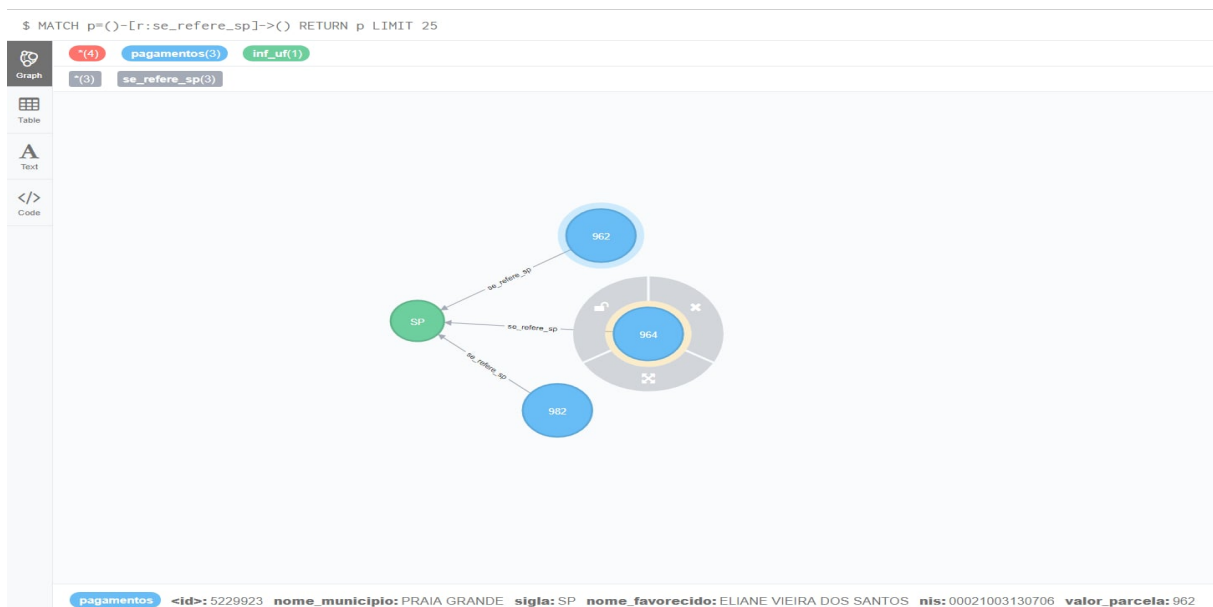


Figura 3.31: Grafo Jan/2016 mais que R\$950.00 , benefício em valor.

Para saber quanto é o mínimo do benefício em reais do Estado de São Paulo, conforme exposto nas Figuras 3.32 e 3.33, a seguir, tem-se na Figura 3.32, no nó em azul, o valor em

dinheiro, e no nó em verde a UF; e na Figura 3.33, o nó em azul é o nome do beneficiário, e o nó em verde, a UF. O valor de R\$ 1,00 (um real) é apresentado no grafo, uma vez que depois de verificar o mesmo NIS dos beneficiados, é possível confirmar que são restos a pagar do mês anterior, ou então, adiantamentos para que o pagamento saia nos valores do teto de cada mês. Nas Figuras em questão tem-se ainda quem são os beneficiados no mês de janeiro de 2016 que supostamente receberam como benefício apenas R\$ 1,00 (um real).

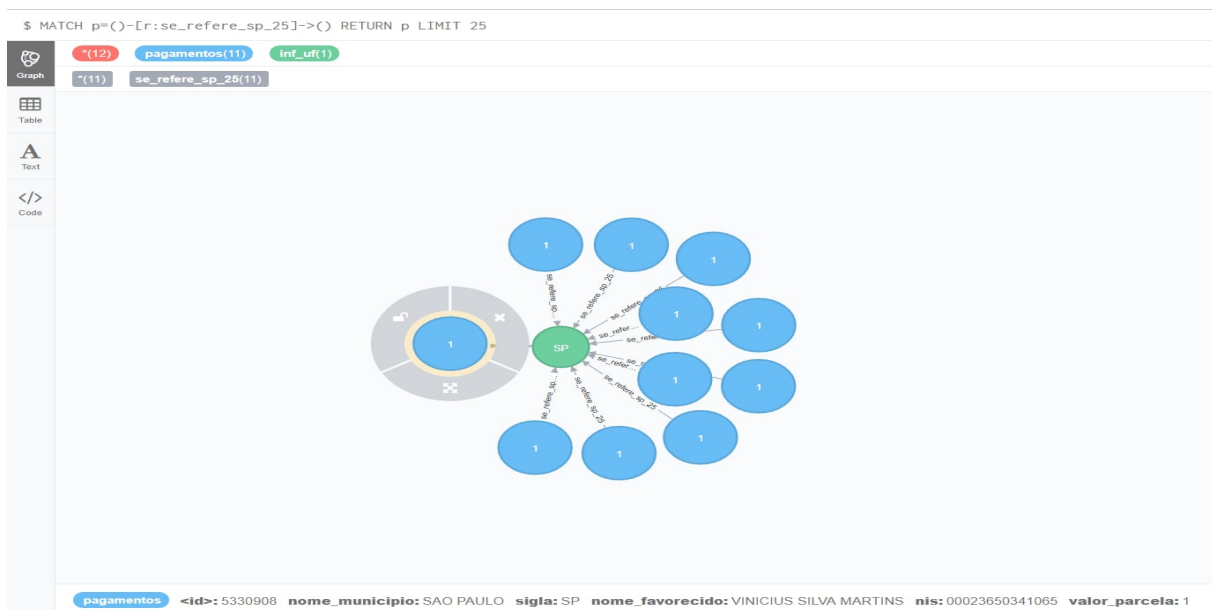


Figura 3.32: Grafo Jan/2016 menor que R\$25.00 , especificado em valor.

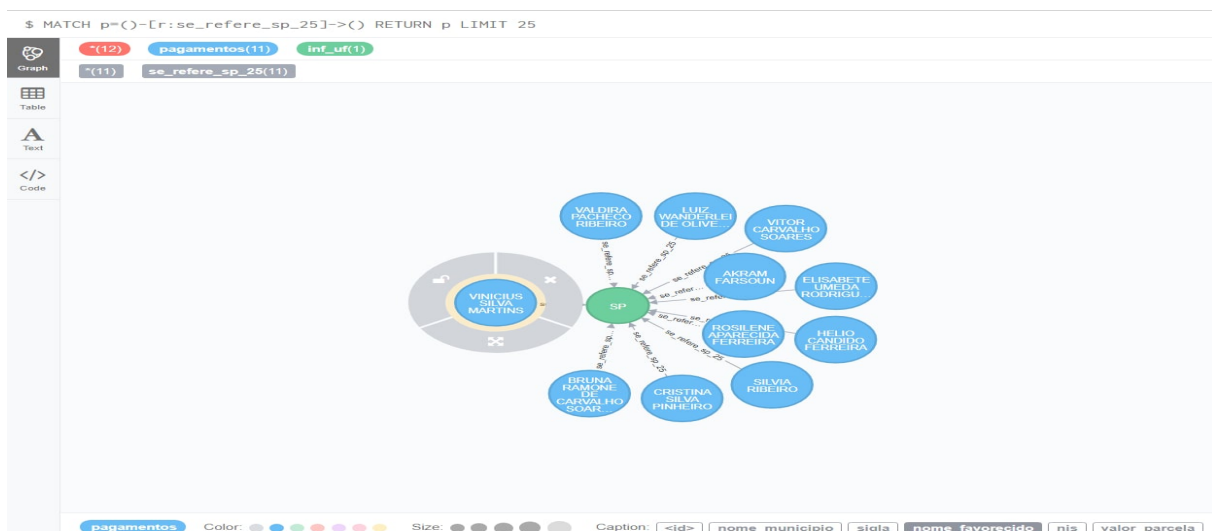


Figura 3.33: Grafo Jan/2016 menor que R\$25.00 , especificado pelo nome.

Quando do uso dos resultados com consultas semelhantes aos relacionais, é possível realizar tal ação pelo Neo4j, liberando uma nova linha de pensamento de delineamento da praticidade do banco. Neste ínterim, na Figura 3.34, por exemplo, tem-se uma tabela onde a primeira coluna se configura no resultado do maior valor da bolsa; a segunda coluna, com o nome do favorecido; a terceira coluna, o Município onde se encontra a favorecida; e, por fim, na última coluna, a sigla da unidade federativa.

```
$ MATCH (n:pagamentos) RETURN MAX(n.valor_parcela), MAX(n.nome_favorecido), MAX(n.nome_municipio), MAX(n.sigla)
```

MAX(n.valor_parcela)	MAX(n.nome_favorecido)	MAX(n.nome_municipio)	MAX(n.sigla)
996	"ZYSLLANE LYDIAN SOARES DA SILVA"	"ZORTEA"	"TO"

Figura 3.34: Resultado do máximo do benefício de todo território.

O *script Cypher* escrito para obter-se a tabela anterior serviu para saber de quem é o maior valor do benefício do Brasil. É possível também visualizar pelo Neo4j, em forma de tabela, o dono do benefício e onde foi agregado o maior benefício daquele mês, conforme exposto na Figura 3.35.

```
MATCH (n:pagamentos) RETURN MAX(n.valor_parcela),  
MAX(n.nome_favorecido), MAX(n.nome_municipio), MAX(n.sigla)
```

Figura 3.35: Código de número máximo de benefício.

Para saber qual a menor contribuição do governo para o programa, e para quem foi destinado este benefício, bastou rodar essa consulta, conforme a Figura 3.36, e resultado na Figura 3.37.

```
MATCH (n:pagamentos) RETURN MIN(n.valor_parcela),  
MIN(n.nome_favorecido), MIN(n.nome_municipio), MIN(n.sigla)
```

Figura 3.36: Código de número mínimo de benefício.

```
$ MATCH (n:pagamentos) RETURN MIN(n.valor_parcela), MIN(n.nome_favorecido), MIN(n.nome_municipio), MIN(n.sigla)
```

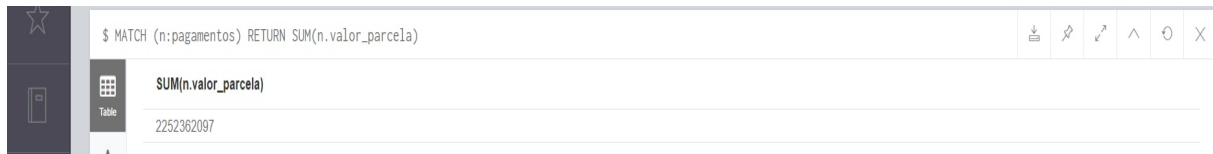
MIN(n.valor_parcela)	MIN(n.nome_favorecido)	MIN(n.nome_municipio)	MIN(n.sigla)
1	"AABI MARIA DA SILVA"	"ABADIA DE GOIAS"	"AC"

Figura 3.37: Resultado do mínimo do benefício de todo território.

Para saber o total que o governo informou durante o mês de Janeiro de 2016, para com o benefício da população, é preciso selecionar o seguinte *script*:

```
MATCH (n:pagamentos) RETURN SUM(n.valor_parcela)
```

No caso em questão, o resultado é de R\$ 2.252.362,097 (dois milhões, duzentos e cinquenta e dois mil, trezentos e sessenta e dois reais e noventa e sete centavos) apenas no mês de janeiro de 2016, conforme evidenciado na Figura 3.38, a seguir.



SUM(n.valor_parcela)
2252362097

Figura 3.38: Resultado do total arrecadado no Brasil.

3.3 Limitações de estudo de caso

Por falta de equipamentos robustos para melhor aprofundamento no estudo em questão, bem como a oferta dos grafos necessários, obteve-se resultado através de um Estado relativamente pequeno e com poucos favorecidos pelo programa social em análise. Logo, para lograr os grafos de uma UF mais robusta (Bahia, Sergipe e São Paulo, por exemplo), onde se tem muitos favorecidos, é preciso maior quantidade de dados.

Diante do exposto, um dos erros elencados foi o **erro de memória**, ou seja, a falta de memória para a execução da faixa de relacionamento que se quis obter. Sobre a questão, a seguir, tem-se o tipo erro identificado quando da geração da massa de dados maior que a capacidade do equipamento. A Figura 3.39, a seguir, trata da falta de memória e a provável modificação na configuração que poderia resolver o problema; porém, tentou-se alterar a configuração, sendo ação inviável para solucionar a falta de memória.

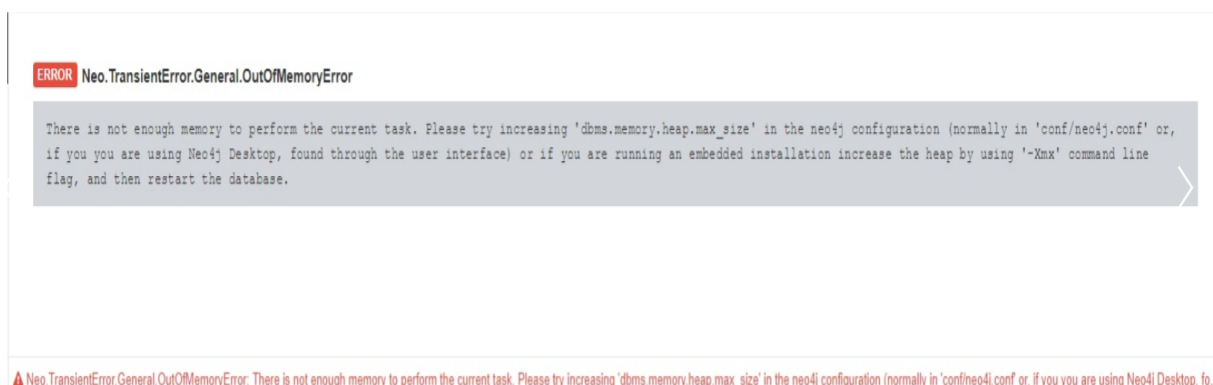


Figura 3.39: Tela de erro de memória.

Enquanto se executa uma inserção mais complexa no Neo4j para atuar no limite da capacidade, e tendo o processamento das relações que almeja, ele exibe uma tela de

timeout, conforme expresso na Figura 3.40, a seguir, afirmando que “o banco saiu do ar e vai reconectar”. Porém, depois de horas de espera, o software trava, forçando sua saída do banco e reiniciando o sistema para a continuidade do trabalho. Logo, perde-se a inclusão de registros já colocados, incluindo as relações já criadas entre os nós. Por último, fez-se a configuração do número de *commit* dos registros dentro do Neo4J, porém, o erro persistiu.

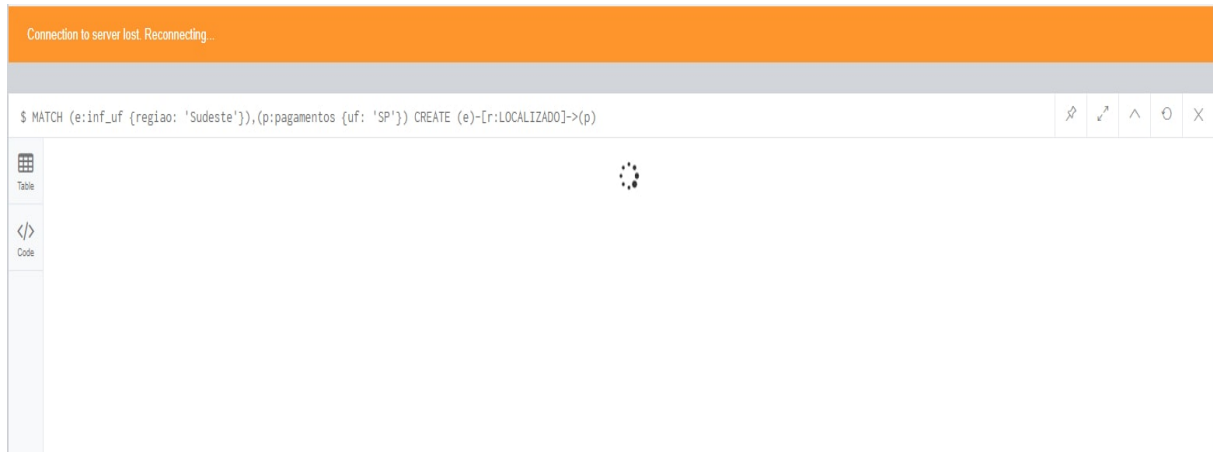


Figura 3.40: Tela de erro de timeout.

Capítulo 4

Conclusão

No presente estudo foi concebida uma análise e testes concernentes em um banco de dados não relacional (NoSQL) denominado Neo4J, com base no programa social Bolsa Família. O objetivo aqui foi colocar à disposição da comunidade melhor visualização dos dados ali enviados para o Portal da Transparência.

Tendo em vista os prós e contras a respeito do banco de dados em questão, ele não se dá em sistema de distribuição gratuita. Neste sentido, para a obtenção de resultados melhores, de forma gratuita, faz-se necessário um equipamento robusto e consideravelmente veloz para a geração de grafos mais complexos, sendo necessário maior investimento para a geração de dados de tal complexidade.

No entanto, é um banco que possui uma dinâmica interessante de se trabalhar. É visualmente bonito e apresenta variabilidade de formas que se pode trabalhar, conforme a criatividade do usuário ou programador.

O resultado final seria mais interessante quando de um equipamento melhor, com a geração de dados de cada mês e à disposição da comunidade. No entanto, com a geração dos grafos apresentados, ficou claro que com o desenvolvimento destes, é possível analisar melhor os gastos e as despesas do governo no que diz respeito ao Programa Bolsa Família (PBF). Posto que, em algumas regiões onde se tem um nível social mais baixo e que, mesmo assim, o benefício é relativamente pouco para aquelas famílias, existem núcleos familiares de muitos filhos, e o benefício é baixo – algo facilmente percebido nos grafos gerados.

O Neo4J em si é muito utilizado no Brasil, ainda que seu potencial possa ser melhorado pela comunidade. Aqui foi preciso recorrer aos fóruns e à documentação de fora da língua portuguesa, mediante várias pesquisas de algo em vídeo ou artigos científicos, a fim de adquirir o conhecimento dos erros relatados durante a criação e o processamento do banco. Além disso, foi preciso pesquisar e entender a ETL (*Extract, Transform, Load*) pelo fato da sua diversificação das formas de se trabalhar dentro Neo4j. Além de a sua linguagem

ser semelhante ao SQL, que muito facilita quando da implementação de algo no banco, tem as particularidades existentes no NoSQL.

4.1 Perspectiva

Como sugestões de melhoria em futuros testes tem-se o que se segue:

- Por meio de licença do Neo4j *Enterprise*, realizar testes com as mesmas bases, porém, agora no banco de dados distribuído, verificando, assim, o tempo de processamento;
- Melhor comparação entre um ano e outro com o banco de dados do PBF.
- Melhores técnicas de consultas, melhoria de escolha das variáveis, estudando-se melhor a ETL do Neo4J, buscando encontrar mecanismos de busca mais eficientes para melhor agilizar a importação das bases do PBF dentro do Neo4j; e
- Trabalhar com APIs, como, por exemplo, API Java; trabalhar em cima de milhares de nós para fazer consultas espaciais, inter-relacionando tais dados com união, inteseção e disjunção de geometrias.

Referências

- [1] H. Augusto, “Inda.” Disponível em: <http://wiki.dados.gov.br/Print.aspx?Page=MainPage>. Acessado em 08 de Agosto de 2016. 4
- [2] “Portal de dados abertos.” Disponível em: <http://dados.gov.br/pagina/faq>. Acessado em 12 de Novembro de 2016. 5, 7
- [3] “Sisp - sistema de administração dos recursos de tecnologia da informação.” Disponível em: <http://www.governoeletronico.gov.br/eixos-de-atuacao/governo/sistema-de-administracao-dos-recursos-de-tecnologia-da-informacao-sisp>. Acessado em 12 de Agosto de 2016. 6
- [4] P. Catia, “Portal brasileiro de dados abertos.” Disponível em: <https://goo.gl/j5LnQV>. Acessado em 08 de Agosto de 2016. 6, 7
- [5] “Lai - lei de acesso a informação.” Disponível em: <http://www.acessoainformacao.gov.br/assuntos/conheca-seu-direito/a-lei-de-acesso-a-informacao>. Acessado em 13 de Novembro de 2016. 6, 8
- [6] “Comprehensive knowledge archive network.” Disponível em: <https://ckan.org/>. Acessado em 1 de Janeiro de 2017. 7
- [7] C. André, “O verdadeiro ‘pai’ do bolsa família: As origens do programa de redistribuição de renda no centro do debate eleitoral.” Disponível em: http://www.huffpostbrasil.com/2014/10/16/o-verdadeiro-pai-do-bolsa-familia-as-origens-do-programa-de-r_a_21666143/. Acessado em 10 de Julho de 2016. 8
- [8] “Modelo do bolsa família foi "exportado" para 52 países.” Disponível em: <http://www.brasil.gov.br/cidadania-e-justica/2016/01/modelo-do-bolsa-familia-foi-exportado-para-52-paises>. Acessado em 15 de Outubro de 2016. 9
- [9] “Reajuste do bolsa família começa a ser pago em junho.” Disponível em: <http://www.brasil.gov.br/economia-e-emprego/2014/05/reajuste-do-bolsa-familia-comeca-a-ser-pago-em-junho>. Acessado em 18 de Setembro de 2016. 9
- [10] “Anti-poverty programmes helping the poorest of the poor.” Disponível em: <https://goo.gl/xtTWGZ>. Acessado em 12 de Agosto de 2016. 9

- [11] S. D. Leonardo de Siqueira Lima, “Por que o bolsa família é bom e deve ser ampliado?.” Disponível em: <http://terraeoeconomico.com.br/por-que-o-bolsa-familia-e-bom-e-deve-ser-ampliado-2>. Acessado em 22 de Outubro de 2016. 9
- [12] P. Lisandra, “Um em cada quatro brasileiros está no bolsa-família.” Disponível em: <http://politica.estadao.com.br/noticias/geral,um-em-cada-quatro-brasileiros-esta-no-bolsa-familia,38787>. Acessado em 27 de Setembro de 2016. 9, 10
- [13] S. Dom, “Modelo do bolsa família foi "exportado" para 52 países.” Disponível em: <https://goo.gl/foiF3M>. Acessado em 15 de Setembro de 2017. 10
- [14] “Governo cancela reajuste do bolsa família de julho - o popular.” Disponível em: <https://goo.gl/akQPD8>. Acessado em 17 de Julho de 2017. 10
- [15] “Bolsa família 2016: Regras.” Disponível em: <http://calendariobolsafamilia2016.com/bolsa-familia-2016-regras/>. Acessado em 20 de Setembro de 2016. 10, 12
- [16] K. LUCADAMO, “Poor results doom anti-poverty project opportunity nyc.” Disponível em: <https://goo.gl/R3bvfvz>. Acessado em 20 de Setembro de 2016. 10
- [17] R. Marques, “Bolsa família.” Disponível em: <http://umavisaoparapolitica.blogspot.com.br/2010/11/bolsa-familia.html>. Acessado em 04 de Agosto de 2016. 10
- [18] “Bolsa família: Jornal francês elogia programa e rebate críticas.” Disponível em: http://liderancadoptbahia.com.br/novo/noticias.php?id_noticia=10493. Acessado em 22 de Outubro de 2016. 10
- [19] D. Abreu, “Tse cassa mandato do governador da paraíba, cássio cunha lima.” Disponível em: <http://g1.globo.com/Noticias/Politica/0,,MUL870319-5601,00-TSE+CASSA+MANDATO+DO+GOVERNADOR+DA+PARAIBA+CASSIO+CUNHA+LIMA.html>. Acessado em 05 de Fevereiro de 2017. 11
- [20] “O passado turvo do líder do psdb no senado.” Disponível em: <http://www.diariodocentrodomundo.com.br/essencial/o-passado-turvo-do-lider-do-psdb-no-senado/>. Acessado em 05 de Fevereiro de 2017. 11
- [21] “Bolsa família 2018.” Disponível em: <http://calendariodobolsafamilia.com.br/bolsa-familia-2018/>. Acessado em 10 de Junho de 2017. 11
- [22] D. Weber, “Em família de 19 pessoas, bolsa vai a rs 1.332.” Disponível em: <https://oglobo.globo.com/brasil/em-familia-de-19-pessoas-bolsa-vai-r-1332-5316065>. Acessado em 18 de Setembro de 2016. 11

- [23] Strozzi.it., “Nosql relational database management system: Home page.” Disponível em: http://www.strozzi.it/cgi-bin/CSA/tw7/I/en_US/nosql/HomePage. Acessado em 04 de Novembro de 2017. 12
- [24] A. Porcelli, “O que é nosql? - java magazine 86,” *Java Magazine - DEVMEDIA*, vol. 1, no. 1, p. 1, 2013. 12
- [25] R. W. Brito, “Neo4j na prática como entrar no mundo dos bancos de dados de grafo,” *Faculdade Farias Brito e Universidade de Fortaleza*, vol. 1, no. 1, pp. 1–6, 2012. 12
- [26] R. C. Aniceto and R. F. Xavier, “Um estudo sobre a utilização do banco de dados nosql cassandra em dados biológicos.” Disponível em: http://bdm.unb.br/bitstream/10483/7927/1/2014_RodrigoCardosoAniceto_ReneFreireXavier.pdf. Acessado em 05 de Dezembro de 2016. 12
- [27] N. L. Seth Gilbert, “Brewer’s conjecture and the feasibility of consistent, available, partition-tolerant web services,” *Newsletter ACM SIGACT News*, vol. 33, no. 1, pp. 51–59, 2002. 13
- [28] G. Renato, “Bancos de dados nosql: uma visão geral.” Disponível em: <https://imasters.com.br/banco-de-dados/bancos-de-dados-nosql-uma-visao-geral/?trace=1519021197&source=single>. Acessado em 04 de Novembro de 2017. 14, 16
- [29] I. Vinicius, “Introdução aos bancos de dados nosql.” Disponível em: <http://www.devmedia.com.br/introducao-aos-bancos-de-dados-nosql/26044>. Acessado em 05 de Janeiro de 2017. 14, 15
- [30] C. A. de Oliveira, “Bancos de dados nosql.” Disponível em: https://www.ibm.com/developerworks/community/blogs/tlcbre/entry/bancos_de_dados_nosql?lang=en. Acessado em 18 de Fevereiro de 2017. 15
- [31] G. S. Leite, “Análise comparativa do teorema cap entre bancos de dados nosql e bancos de dados relacionais.” Disponível em: <http://cdn.ffb.edu.br/sites/default/files/tcc-20102-gleidson-sobreira-leite.pdf>. Acessado em 15 de Fevereiro de 2017. 15
- [32] S. Bryce Merkl, “Graph databases for beginners: Acid vs. base explained.” Disponível em: <https://neo4j.com/blog/acid-vs-base-consistency-models-explained/>. Acessado em 20 de Julho de 2016. 15
- [33] E. S. Prigol, “Redes sociais com banco de dados orientado a grafos,” *1*, vol. 01, no. 1, pp. 24–25, 2016. 16
- [34] P. Feofiloff, “Digrafos simétricos.” Disponível em: https://www.ime.usp.br/~pf/analise_de_algoritmos/aulas/symmetricgraphs.html. Acessado em 05 de Janeiro de 2017. 16
- [35] C. Kemper, *Beginning Neo4j*. Apress, Berkely, CA, USA, 2015. 16

- [36] R. S. Raqueline Pentead, R. M. Diego Hoss, Jaqueline Nande, and C. H. Wal-mir Couto, “Um estudo sobre bancos de dados em grafos nativos.” Disponível em: <http://www.inf.ufpr.br/carmem/pub/erbd2014-artigo.pdf>. Acessado em 05 de Abril de 2017. 17
- [37] A. Agricola, “Sdtimes looks at the new crop of nosql databases – no sql and the not only sql data stores.” Disponível em: <https://neo4j.com/news/springtime-for-graph-databases/>. Acessado em 22 de Fevereiro de 2017. 17
- [38] P. Ivan, “Wayblazer cognitive computing application powered by ibm watson e neo4j.” Disponível em: <https://neo4j.com/blog/wayblazer-watson-neo4j/>. Acessado em 08 de Janeiro de 2017. 17
- [39] A. Handy, “Springtime for databases,” *SD Times Software Development*, vol. 1, no. 1, p. 1, 2013. 18
- [40] “Python: O que é? por que usar?.” Disponível em: <http://pyscience-brasil.wikidot.com/python:python-oq-e-pq>. Acessado em 10 de Agosto de 2016. 21

Anexo I

Anexo

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%Este é o programa conversor implementado em \textit{python}, no qual
%converte .CSV para .TXT.
%
%% PROGRAMA CONVERSOR %%
%
%#!/usr/bin/env python
%# encoding: utf-8
%
%import csv
%
%text_list = []
%
%with open('pagamentos.csv', "r") as my_input_file:
%    for line in my_input_file:
%        line = line.split(',', 0)
%        text_list.append(" ".join(line))
%
%with open('paga.txt', "w") as my_output_file:
%
%    for line in text_list:
%        my_output_file.write(line)
%    print('Arquivo exportado para TXT com sucesso!')
%
%% PROGRAMA CONVERSOR FIM %%
```

```

%
%
%Já este, é o programa conversor implementado em \textit{python}, não qual
%ele tira a tabulação, insere a Região brasileira conforme as Unidades
%Federativa. E por fim converte de volta para .CSV. Também fazendo as
%alterações necessárias, no qual o nome da coluna em questão é colocado apenas
%minúsculas e sem pontuação.
%
%
%% PROGRAMA SEGUNDA PARTE CONVERSOR %%
%
%#!/usr/bin/env python
%# encoding: utf-8
%
%regiao = {'AC': 'Norte', 'AL': 'Nordeste', 'AP': 'Norte', 'AM': 'Norte', 'BA':
%'Nordeste', 'CE': 'Nordeste', 'DF': 'Centro_Oeste', 'ES': 'Sudeste', 'GO':
%'Centro_Oeste', 'MA': 'Nordeste', 'MT': 'Centro_Oeste', 'MS': 'Centro_Oeste', 'MG':
%'Sudeste', 'PA': 'Norte', 'PB': 'Nordeste', 'PR': 'Sul', 'PE': 'Nordeste', 'PI':
%'Nordeste', 'RJ': 'Sudeste', 'RN': 'Nordeste', 'RS': 'Sul', 'RO': 'Norte', 'RR':
%'Norte', 'SC': 'Sul', 'SP': 'Sudeste', 'SE': 'Nordeste', 'TO': 'Norte'}
%
%output_fd = open('pagamento_certo.csv', 'w')
%
%with open('paga.txt', 'r') as f:
%
%    # primeira linha
%    line = f.readline()
%    line = ','.join(['%s'%field.strip() for field in line.split('\t')])+'\n'
%    line = '_'.join(['%s'%field.strip() for field in line.split(' ')])+'\n'
%    line = '-'.join(['%s'%field.strip() for field in line.split('-')])+'\n'
%    line = 'o'.join(['%s'%field.strip() for field in line.split('ó')])+'\n'
%    line = 'i'.join(['%s'%field.strip() for field in line.split('í')])+'\n'
%    line = 'ç'.join(['%s'%field.strip() for field in line.split('ç')])+'\n'
%    line = 'ã'.join(['%s'%field.strip() for field in line.split('ã')])+'\n'
%    line = 'ê'.join(['%s'%field.strip() for field in line.split('ê')])+'\n'
%    line = 'uf'.join(['%s'%field.strip() for field in line.split('UF')])+'\n'
%    line = 'codigo_siafi_municipio'.join(['%s'%field.strip() for field in
%    line.split('Codigo_SIAFI_Municipio')])+'\n'
%    line = 'nome_municipio'.join(['%s'%field.strip() for field in

```

```

%   line.split('Nome_Municipio')))+'\n'
%   line = 'codigo_funcao'.join(['%s'%field.strip() for field in
%   line.split('Codigo_Funcao')])))+'\n'
%   line = 'codigo_subfuncao'.join(['%s'%field.strip() for field in
%   line.split('Codigo_Subfuncao')])))+'\n'
%   line = 'codigo_programa'.join(['%s'%field.strip() for field in
%   line.split('Codigo_Programa')])))+'\n'
%   line = 'codigo_acao'.join(['%s'%field.strip() for field in
%   line.split('Codigo_Acao')])))+'\n'
%   line = 'nis_favorecido'.join(['%s'%field.strip() for field in
%   line.split('NIS_Favorecido')])))+'\n'
%   line = 'nome_favorecido'.join(['%s'%field.strip() for field in
%   line.split('Nome_Favorecido')])))+'\n'
%   line = 'fonte_finalidade'.join(['%s'%field.strip() for field in
%   line.split('Fonte_Finalidade')])))+'\n'
%   line = 'valor_parcela'.join(['%s'%field.strip() for field in
%   line.split('Valor_Parcela')])))+'\n'
%   line = 'mes_competencia'.join(['%s'%field.strip() for field in
%   line.split('Mes_Competencia')]))+',regiao'+'\n'
%   output_fd.write(line)
%
%   # outras linhas
%   for line in f:
%       uf = line.split()[0]
%       line = ', '.join(['%s'%field.strip() for field in
%       line.split('\t')]))+', '+regiao[uf.replace('"', ' ')]+''\n'
%       output_fd.write(line)
%
%print('Arquivo exportado para CSV com sucesso!')
%output_fd.close()
%
%% PROGRAMA SEGUNDA PARTE CONVERTOR FIM %%
%
%
%É mostrado aqui, qual a consulta que se teve no Neo4j de maior
%destaque, essa consulta foi a importação depois de convertido. Com essa
%consulta podemos já criar o nó que vai ser relacionado dentro do Neo4j.
%Sendo assim, já se popula o banco e insere o nó.
%
%
```

```
%  
%% CODIGO DE BUSCA NEO4J %%  
%  
%USING PERIODIC COMMIT 1000  
%LOAD CSV WITH HEADERS FROM 'file:///pagamentos_total.csv' AS  
%pagamentos  
%CREATE (:pagamentos {sigla: pagamentos.uf, nome_municipio:  
%pagamentos.nome_municipio,  
%nis: pagamentos.nis_favorecido, nome_favorecido:  
%pagamentos.nome_favorecido,  
%valor_parcela: ToInt(pagamentos.valor_parcela)})  
%  
%% CODIGO DE BUSCA NEO4J FIM %%  
%
```