PROJETO: SQL PARA ANÁLISE DE DADOS DO NOVO BOLSA FAMÍLIA

GBC043 - Sistemas de Banco de Dados

Prof(a). Maria Camila Nardini Barioni

Participantes

Ana Alice Cordeiro de Souza - 12211BCC028

Anna Karolyna Pereira Santos - 12221BCC046

Ester Camilly Simplício de Freitas - 12211BCC036

Uberlândia, 30 de Agosto de 2025

ÍNDICE

Nome do Usuário Usado para Criação do Banco de Dados: esterzolas

1. ESPECIFICAÇÃO DO PROBLEMA

O problema abordado neste projeto é a análise socioeconômica e da distribuição de recursos de um dos principais programas de transferência de renda do Governo Federal, o Novo Bolsa Família. O objetivo é estruturar os dados públicos do programa em um banco de dados relacional desenvolvido para, em seguida, extrair informações relevantes que possam auxiliar na tomada de decisão e na compreensão do impacto do programa no Brasil.

O conjunto de dados escolhido para este trabalho consiste nos microdados públicos dos pagamentos do Novo Bolsa Família, referentes ao primeiro semestre de 2024. Estes dados são disponibilizados mensalmente e detalham cada parcela paga a um beneficiário em um determinado município.

A estrutura original dos dados, conforme extraída dos arquivos CSV, pode ser considerada uma única relação bruta contendo 9 atributos principais:

1. MÊS COMPETÊNCIA: O mês em que a folha de pagamentos foi processada e os valores foram disponibilizados.
2. MÊS REFERÊNCIA: O mês ao qual o benefício efetivamente se refere.
3. UF: A sigla da Unidade Federativa onde o beneficiário reside.
4. CÓDIGO MUNICÍPIO SIAFI: Um código numérico único que identifica cada município brasileiro no Sistema Integrado de Administração Financeira (SIAFI).
5. NOME MUNICÍPIO: O nome do município do beneficiário.
6. CPF FAVORECIDO: O Cadastro de Pessoa Física (CPF) do responsável familiar que recebe o benefício, com parte dos dígitos mascarados.
7. NIS FAVORECIDO: O Número de Identificação Social (NIS) do favorecido, que atua como identificador único da pessoa.
8. NOME FAVORECIDO: O nome completo do responsável familiar.
9. VALOR PARCELA: O valor, em Reais (R$), da parcela paga.

Uma característica importante e analiticamente rica do conjunto de dados, observada durante a exploração inicial, é a presença de pagamentos retroativos. Foi verificado que os arquivos de competência do ano de 2024 contêm registros cujo MÊS REFERÊNCIA pertence ao ano de 2023. Isso indica o pagamento de benefícios referentes a períodos anteriores, seja por

acertos cadastrais, decisões administrativas ou judiciais. Essa distinção entre o mês em que o pagamento é processado e o mês a que o benefício se refere é crucial para uma análise temporal precisa do programa.

Referência:

- Fonte: Portal da Transparência do Governo Federal do Brasil.
- Link para o Conjunto de Dados: Novo Bolsa Família

2. MODELO RELACIONAL

Nesta seção é identificado o conjunto de relações que especificam o banco de dados relacional implementado, a partir da análise do conjunto de dados do Novo Bolsa Família. O processo envolveu a definição das dependências funcionais, a normalização detalhada até a Terceira Forma Normal (3FN) e a descrição do modelo final.

2.1. DEPENDÊNCIAS FUNCIONAIS E PROCESSO DE NORMALIZAÇÃO

O ponto de partida foi considerar todos os atributos dos arquivos CSV como uma única relação. A partir da análise semântica dos dados, foram identificadas as seguintes Dependências Funcionais (DFs) válidas:

1. **{NIS FAVORECIDO} → {NOME FAVORECIDO, CPF FAVORECIDO}**
    - **Justificativa:** O Número de Identificação Social (NIS) é um identificador único para cada beneficiário, determinando funcionalmente seu nome e CPF.
2. **{CÓDIGO MUNICÍPIO SIAFI} → {NOME MUNICÍPIO, UF}**
    - **Justificativa:** O código do SIAFI é um identificador único para cada município, determinando seu nome e sua Unidade Federativa (UF).

Com a Chave Primária da relação original sendo {NIS FAVORECIDO, MÊS REFERÊNCIA},foi aplicado o seguinte processo de normalização:

- **Primeira Forma Normal (1FN):** A relação original já se encontrava na 1FN, pois todos os seus atributos eram atômicos.

- **Segunda Forma Normal (2FN):** Foram identificadas dependências parciais, uma vez que NOME FAVORECIDO e CPF FAVORECIDO dependiam apenas de parte da chave primária (NIS FAVORECIDO). Para resolver esta violação, a relação foi decomposta, criando-se a tabela **pessoa** para armazenar os dados dos beneficiários.
- **Terceira Forma Normal (3FN):** Na tabela remanescente, identificou-se uma dependência transitiva: NOME MUNICÍPIO e UF dependiam de CÓDIGO MUNICÍPIO SIAFI, que não era uma chave. Para resolver esta violação, a relação foi novamente decomposta, criando-se a tabela **municipios**.

Este processo resultou na seguinte estrutura final em 3FN.

2.2. ESTRUTURA E DESCRIÇÃO DAS TABELAS DO MODELO RELACIONAL FINAL

O modelo relacional resultante do processo de normalização é composto por três tabelas principais, projetadas para garantir a integridade dos dados e eliminar redundâncias. A seguir, é apresentada a estrutura e a descrição de cada uma delas.

Tabela municipios

- **Estrutura:**
  - codigo_siafi (INTEGER, Chave Primária): Identificador numérico único para cada município.
  - nome (VARCHAR(100)): Nome do município.
  - uf (CHAR(2)): Sigla da Unidade Federativa.
- **Descrição:** Esta tabela funciona como um cadastro centralizado de todos os municípios brasileiros presentes no conjunto de dados. Armazenar os municípios de forma única evita a repetição de seus nomes e UFs a cada um dos milhões de pagamentos, economizando espaço e garantindo consistência. Se o nome de um município precisasse ser corrigido, a alteração seria feita em um único registro.

Tabela pessoa

- **Estrutura:**

- nis (BIGINT, Chave Primária): Número de Identificação Social (NIS), que serve como identificador único do beneficiário.
- nome (VARCHAR(150)): Nome completo do beneficiário.
- cpf (VARCHAR(14)): CPF do beneficiário (mascarado).

- **Descrição:** Esta tabela armazena os dados cadastrais de cada beneficiário (responsável familiar) de forma única. O objetivo é centralizar as informações de cada pessoa, evitando que seu nome e CPF sejam repetidos milhões de vezes na tabela de pagamentos.

Tabela pagamentos

- **Estrutura:**
  2.1. id_pagamento (SERIAL, Chave Primária): Identificador numérico único e automático para cada registro de pagamento.
  2.2. mes_competencia (DATE): O mês em que a folha de pagamentos foi processada.
  2.3. mes_referencia (DATE): O mês ao qual o benefício se refere.
  2.4. municipio_siafi (INTEGER, Chave Estrangeira): Referencia municipios(codigo_siafi).
  2.5. pessoa_nis (BIGINT, Chave Estrangeira): Referencia pessoa(nis).
  2.6. valor_parcela (DECIMAL(10, 2)): O valor exato da parcela paga.
- **Descrição:** Esta é a tabela principal e mais volumosa, registrando cada evento de pagamento. Ela atua como uma tabela de fatos, conectando as dimensões municipios e pessoa através de chaves estrangeiras. Cada linha representa uma única parcela paga a uma pessoa em um determinado município e mês. Foi introduzida uma chave primária substituta (id_pagamento) do tipo SERIAL, uma prática comum de modelagem que simplifica a identificação única de cada transação. Esta abordagem garante um identificador único para cada transação sem depender de uma chave composta, simplificando as operações de junção e referência.

3. CRIAÇÃO DO BANCO DE DADOS

Esta seção descreve os comandos SQL da Linguagem de Definição de Dados (DDL - Data Definition Language) utilizados para criar a estrutura do banco de dados relacional. Os

comandos a seguir implementam o modelo normalizado definido na **seção 2.2 deste documento**, estabelecendo as tabelas, atributos, tipos de dados e as restrições de integridade necessárias para armazenar os dados do programa Novo Bolsa Família de forma consistente e eficiente.

Comandos CREATE TABLE

Os seguintes comandos foram executados para criar as tabelas municipios, pessoa e pagamentos, que compõem o esquema final do banco de dados.

```sql
-- Criação da tabela para armazenar os dados únicos dos municípios
CREATE TABLE municipios (
    codigo_siafi INTEGER PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    uf CHAR(2) NOT NULL
);

-- Criação da tabela para armazenar os dados únicos dos beneficiários
CREATE TABLE pessoa (
    nis BIGINT PRIMARY KEY,
    nome VARCHAR(150) NOT NULL,
    cpf VARCHAR(14)
);

-- Criação da tabela de fatos para registrar cada pagamento individual
CREATE TABLE pagamentos (
    id_pagamento SERIAL PRIMARY KEY,
    mes_competencia DATE NOT NULL,
    mes_referencia DATE NOT NULL,
    municipio_siafi INTEGER NOT NULL,
    pessoa_nis BIGINT NOT NULL,
    valor_parcela DECIMAL(10, 2) NOT NULL,

    -- Definição da chave estrangeira para a tabela 'pessoa'
    CONSTRAINT fk_pessoa
        FOREIGN KEY(pessoa_nis)
        REFERENCES pessoa(nis)
        ON DELETE RESTRICT ON UPDATE CASCADE,

    -- Definição da chave estrangeira para a tabela 'municipios'
    CONSTRAINT fk_municipio
        FOREIGN KEY(municipio_siafi)
        REFERENCES municipios(codigo_siafi)
        ON DELETE RESTRICT ON UPDATE CASCADE);
```

```
-- Popula as tabelas finais a partir dos dados brutos
INSERT INTO municipios (codigo_siafi, nome, uf) SELECT DISTINCT
CAST(codigo_municipio_siafi AS INTEGER), nome_municipio, uf FROM
staging_pagamentos ON CONFLICT (codigo_siafi) DO NOTHING;

INSERT INTO pessoa (nis, nome, cpf) SELECT DISTINCT CAST(nis_favorecido AS
BIGINT), nome_favorecido, cpf_favorecido FROM staging_pagamentos WHERE
nis_favorecido IS NOT NULL AND nis_favorecido <> '' ON CONFLICT (nis) DO
NOTHING;

INSERT INTO pagamentos (mes_competencia, mes_referencia, municipio_siafi,
pessoa_nis, valor_parcela) SELECT TO_DATE(mes_competencia, 'YYYY-MM-DD'),
TO_DATE(mes_referencia, 'YYYY-MM-DD'), CAST(codigo_municipio_siafi AS INTEGER),
CAST(nis_favorecido AS BIGINT), CAST(REPLACE(valor_parcela, ',', '.') AS
DECIMAL(10, 2)) FROM staging_pagamentos WHERE nis_favorecido IS NOT NULL AND
nis_favorecido <> '';

-- Limpa a tabela de passagem que não é mais necessária
DROP TABLE staging_pagamentos;
```

Justificativa das Ações de Chave Estrangeira (ON DELETE / ON UPDATE)

Conforme solicitado na especificação do projeto (OBS 2), a escolha das ações para as chaves estrangeiras na tabela pagamentos foi fundamental para garantir a integridade referencial do banco de dados. As justificativas são as seguintes:

- ON DELETE RESTRICT: Esta opção foi escolhida para ambas as chaves estrangeiras (fk_pessoa e fk_municipio). A regra RESTRICT impede que um registro em uma tabela pai (como um registro em pessoa ou municipios) seja deletado se houver algum registro filho (na tabela pagamentos) que o referenciam. Esta é a política mais segura para este conjunto de dados, pois garante a integridade histórica. Deletar um beneficiário ou um município deixaria registros de pagamento "órfãos", tornando impossível saber quem recebeu o dinheiro ou onde, o que invalidaria qualquer análise financeira ou social.
- ON UPDATE CASCADE: Esta opção também foi escolhida para ambas as chaves estrangeiras. A regra CASCADE garante que, se o valor da chave primária em uma tabela pai for alterado, essa alteração será automaticamente propagada para todos os registros correspondentes na tabela pagamentos. Embora a alteração de um NIS ou de

um código SIAFI seja um evento raro, esta política garante que, caso ocorra, o relacionamento entre as tabelas permaneça consistente sem a necessidade de atualizações manuais, prevenindo a perda de integridade dos dados.

Observação sobre a Ordem de Povoamento das Tabelas

De acordo com a especificação do projeto (OBS 1), a ordem em que as tabelas são populadas é crítica devido às dependências criadas pelas chaves estrangeiras. A tabela pagamentos depende diretamente das tabelas municipios e pessoa. Portanto, o script de carregamento de dados foi projetado para respeitar esta dependência, seguindo a ordem correta:

1. Primeiro, são inseridos todos os registros únicos nas tabelas municipios e pessoa.
2. Somente após as tabelas pai estarem completamente povoadas, são inseridos os registros na tabela pagamentos.

Tentar inserir um registro em pagamentos para um pessoa_nis ou municipio_siafi que ainda não existe em suas respectivas tabelas resultaria em um erro de violação de chave estrangeira.

4. ESPECIFICAÇÃO DE CONSULTAS EM SQL

Esta seção é dedicada à apresentação das consultas em SQL desenvolvidas para a análise do conjunto de dados do Novo Bolsa Família. O objetivo é demonstrar a aplicação prática do banco de dados estruturado para extrair informações relevantes que podem auxiliar gestores na compreensão do panorama do programa e na tomada de decisões estratégicas.
Seguindo a estrutura definida no projeto, esta seção será subdividida na apresentação das consultas e, posteriormente, na implementação de um gatilho e um procedimento armazenado.

Para cada uma das doze consultas propostas, a apresentação seguirá um formato padrão que inclui:

- **Título:** Uma descrição clara do objetivo da consulta.
- **Comando SQL:** O script exato utilizado para extrair os dados.
- **Resultado:** O print da tabela gerada pela execução da consulta no

- PostgreSQL.
- **Gráfico:** Uma representação visual dos dados resultantes para facilitar a análise e a interpretação.
- **Comando SQL para banco amostral**

Cabe ressaltar que os resultados e gráficos exibidos neste relatório foram obtidos a partir da execução das consultas sobre o banco de dados completo, refletindo a análise sobre a totalidade dos dados. Adicionalmente, com o objetivo de facilitar o processo de correção e validação do projeto, foram disponibilizados scripts das consultas para o banco de dados amostral disponibilizado no servidor da FACOM.

4.1. CONSULTAS EM SQL

4.1.1. PRIMEIRA CONSULTA

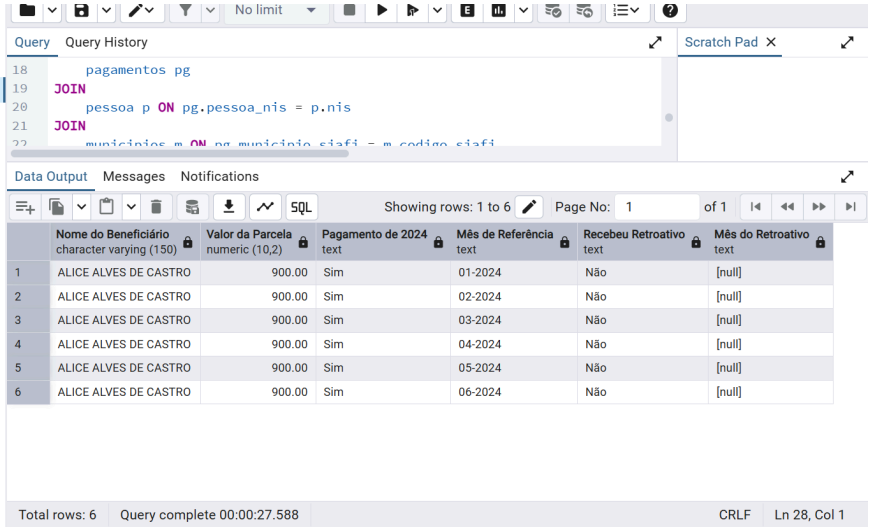
Título: Acompanhamento de Beneficiário: Histórico de Pagamentos em 2024 e Verificação de Retroativos para uma Pessoa de Uberlândia.

Comando SQL:

```
SELECT
    p.nome AS "Nome do Beneficiário",
    pg.valor_parcela AS "Valor da Parcela",
    CASE
        WHEN EXTRACT(YEAR FROM pg.mes_referencia) = 2024 THEN 'Sim'
        ELSE 'Não'
    END AS "Pagamento de 2024",
    TO_CHAR(pg.mes_referencia, 'MM-YYYY') AS "Mês de Referência",
    CASE
        WHEN EXTRACT(YEAR FROM pg.mes_referencia) = 2023 AND
EXTRACT(YEAR FROM pg.mes_competencia) = 2024 THEN 'Sim'
        ELSE 'Não'
    END AS "Recebeu Retroativo",
    CASE        WHEN EXTRACT(YEAR FROM pg.mes_referencia) = 2023 AND
EXTRACT(YEAR FROM pg.mes_competencia) = 2024 THEN
TO_CHAR(pg.mes_referencia, 'MM-YYYY')
        ELSE NULL
    END AS "Mês do Retroativo"
FROM
    pagamentos pg
JOIN
    pessoa p ON pg.pessoa_nis = p.nis
JOIN
    municipios m ON pg.municipio_siafi = m.codigo_siafi
```

```sql
WHERE
    p.nis = 16142837799 AND m.codigo_siafi = 5403
-- A linha abaixo garante a ordenação cronológica correta pela data original
ORDER BY
    pg.mes_referencia ASC;
```

Print do resultado da consulta:



```
18      pagamentos pg
19  JOIN
20      pessoa p ON pg.pessoa_nis = p.nis
21  JOIN
```

Data Output | Messages | Notifications — Showing rows: 1 to 6 — Page No: 1 of 1

	Nome do Beneficiário character varying (150)	Valor da Parcela numeric (10,2)	Pagamento de 2024 text	Mês de Referência text	Recebeu Retroativo text	Mês do Retroativo text
1	ALICE ALVES DE CASTRO	900.00	Sim	01-2024	Não	[null]
2	ALICE ALVES DE CASTRO	900.00	Sim	02-2024	Não	[null]
3	ALICE ALVES DE CASTRO	900.00	Sim	03-2024	Não	[null]
4	ALICE ALVES DE CASTRO	900.00	Sim	04-2024	Não	[null]
5	ALICE ALVES DE CASTRO	900.00	Sim	05-2024	Não	[null]
6	ALICE ALVES DE CASTRO	900.00	Sim	06-2024	Não	[null]

Total rows: 6 | Query complete 00:00:27.588 | CRLF | Ln 28, Col 1

Visualização do Resultado:

	Nome do Beneficiário	Valor da Parcela		Recebeu Retroativo
2	ALICE ALVES DE CASTRO	900.00	01-2024	Não
3	ALICE ALVES DE CASTRO	900.00	02-2024	Não
4	ALICE ALVES DE CASTRO	900.00	03-2024	Não
5	ALICE ALVES DE CASTRO	900.00	04-2024	Não
6	ALICE ALVES DE CASTRO	900.00	05-2024	Não
7	ALICE ALVES DE CASTRO	900.00	06-2024	Não

Nesse tipo de consulta, que retorna um acompanhamento de determinado beneficiário, é mais interessante o tipo tabela para verificar se o beneficiário recebeu ou não sua parcela em dado mês e se vem recebendo pagamento retroativo.

Comando em SQL para banco amostral:

```sql
SELECT
p.nome AS "Nome do Beneficiário",
pg.valor_parcela AS "Valor da Parcela",
   CASE
          WHEN EXTRACT(YEAR FROM pg.mes_referencia) = 2024 THEN 'Sim'
```

```
            ELSE 'Não'
            END AS "Pagamento de 2024",
            TO_CHAR(pg.mes_referencia, 'MM-YYYY') AS "Mês de
Referência",
            CASE
            WHEN EXTRACT(YEAR FROM pg.mes_referencia) = 2023 AND
EXTRACT(YEAR FROM
pg.mes_competencia) = 2024 THEN 'Sim'
            ELSE 'Não'
            END AS "Recebeu Retroativo",
            CASE
            WHEN EXTRACT(YEAR FROM pg.mes_referencia) = 2023 AND
EXTRACT(YEAR FROM
pg.mes_competencia) = 2024 THEN TO_CHAR(pg.mes_referencia, 'MM-YYYY')
            ELSE NULL
            END AS "Mês do Retroativo"
FROM
            projeto_sbd.pagamentos pg
JOIN
            projeto_sbd.pessoa p ON pg.pessoa_nis = p.nis
JOIN
            projeto_sbd.municipios m ON pg.municipio_siafi =
m.codigo_siafi
WHERE
            p.nis = 12980709125 AND m.codigo_siafi = 5403
-- A linha abaixo garante a ordenação cronológica correta pela data
original
ORDER BY
    pg.mes_referencia ASC;
```

4.1.2. SEGUNDA CONSULTA

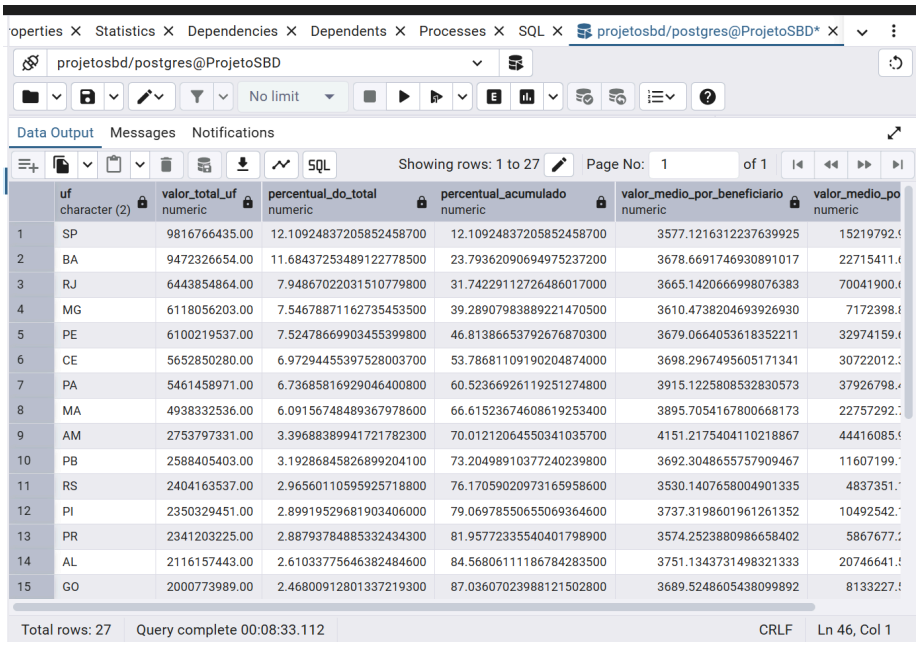
Título: Análise de Desigualdade na Distribuição de Recursos: Concentração Percentual e Acumulada do Bolsa Família nos Estados Brasileiros em 2024.

Comando SQL:

```
WITH StatsPorUF AS (
    -- Calcular as métricas base para cada estado (UF)
    SELECT
        m.uf,
        SUM(p.valor_parcela) AS valor_total_uf,
        COUNT(DISTINCT p.pessoa_nis) AS qtd_beneficiarios_uf,
        COUNT(DISTINCT m.codigo_siafi) AS qtd_municipios_uf
    FROM
        pagamentos AS p
```

```
    JOIN
        municipios AS m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        EXTRACT(YEAR FROM p.mes_referencia) = 2024
    GROUP BY
        m.uf
),
CalculosGerais AS (
    -- Calcular os percentuais e as médias
    SELECT
        uf,
        valor_total_uf,
        qtd_beneficiarios_uf,
        qtd_municipios_uf,
        -- Percentual que cada estado representa do total geral
        (valor_total_uf / SUM(valor_total_uf) OVER ()) * 100 AS
percentual_do_total,
        -- Valor médio por beneficiário no estado
        valor_total_uf / qtd_beneficiarios_uf AS
valor_medio_por_beneficiario,
        -- Valor médio por município no estado
        valor_total_uf / qtd_municipios_uf AS
valor_medio_por_municipio
    FROM
        StatsPorUF
)
-- Calcular o percentual acumulado (Princípio de Pareto)
SELECT
    uf,
    valor_total_uf,
    percentual_do_total,
    -- Soma o percentual da linha atual com a soma de todos os
percentuais das linhas anteriores
    SUM(percentual_do_total) OVER (ORDER BY percentual_do_total DESC)
AS percentual_acumulado,
    valor_medio_por_beneficiario,
    valor_medio_por_municipio
FROM
    CalculosGerais
ORDER BY
    percentual_do_total DESC;
```

Print do resultado da consulta:



operties × Statistics × Dependencies × Dependents × Processes × SQL × projetosbd/postgres@ProjetoSBD* ×

projetosbd/postgres@ProjetoSBD

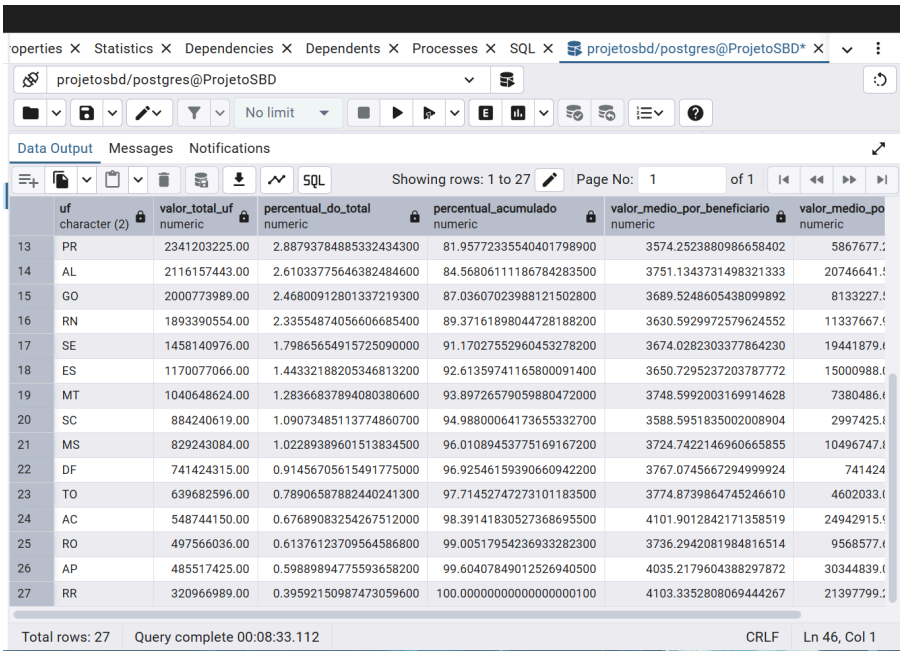
No limit

Data Output | Messages | Notifications

Showing rows: 1 to 27 | Page No: 1 of 1

	uf character (2)	valor_total_uf numeric	percentual_do_total numeric	percentual_acumulado numeric	valor_medio_por_beneficiario numeric	valor_medio_po numeric
1	SP	9816766435.00	12.109248372058524587 00	12.109248372058524587 00	3577.1216312237639925	15219792.(
2	BA	9472326654.00	11.684372534891227785 00	23.793620906949752372 00	3678.6691746930891017	22715411.(
3	RJ	6443854864.00	7.948670220315107798 00	31.742291127264860170 00	3665.1420666998076383	70041900.(
4	MG	6118056203.00	7.546788711627354535 00	39.289079838892214705 00	3610.4738204693926930	7172398.(
5	PE	6100219537.00	7.524786690345539980 0	46.813866537926787030 00	3679.0664053618352211	32974159.(
6	CE	5652850280.00	6.972944553975280037 00	53.786811091902048740 00	3698.2967495605171341	30722012.(
7	PA	5461458971.00	6.736858169290464008 00	60.523669261192512748 00	3915.1225808532830573	37926798.(
8	MA	4938332536.00	6.091567484893679786 00	66.615236746086192534 00	3895.7054167800668173	22757292.(
9	AM	2753797331.00	3.396883899417217823 00	70.012120645503410357 00	4151.2175404110218867	44416085.(
10	PB	2588405403.00	3.192868458269920410 0	73.204989103772402398 00	3692.3048655757909467	11607199.(
11	RS	2404163537.00	2.965601105959257188 00	76.170590209731659586 00	3530.1407658004901335	4837351.(
12	PI	2350329451.00	2.899195296819034060 00	79.069785506550693646 00	3737.3198601961261352	10492542.(
13	PR	2341203225.00	2.887937848855324343 00	81.957723355404017989 00	3574.2523880986658402	5867677.(
14	AL	2116157443.00	2.610337756463824846 00	84.568061111867842835 00	3751.1343731498321333	20746641.(
15	GO	2000773989.00	2.468009128013372193 00	87.036070239881215028 00	3689.5248605438099892	8133227.(

Total rows: 27 | Query complete 00:08:33.112 | CRLF | Ln 46, Col 1



operties × Statistics × Dependencies × Dependents × Processes × SQL × projetosbd/postgres@ProjetoSBD* ×

projetosbd/postgres@ProjetoSBD

No limit

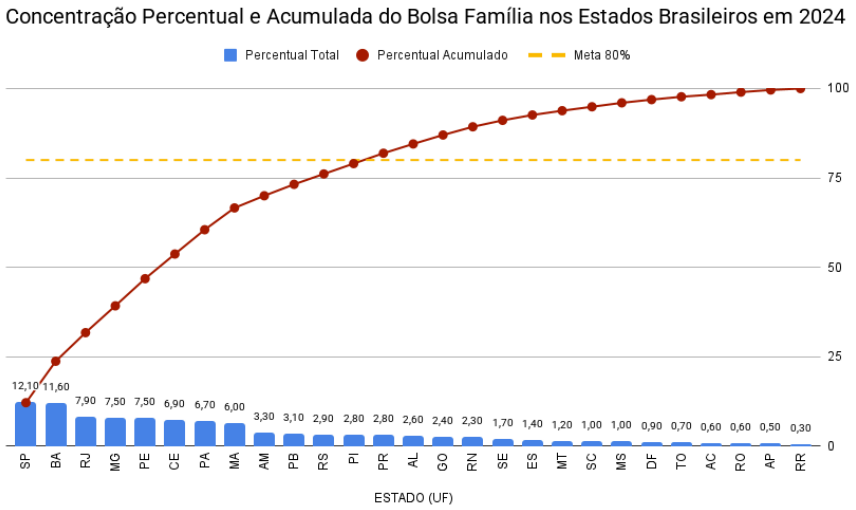
Data Output | Messages | Notifications

Showing rows: 1 to 27 | Page No: 1 of 1

	uf character (2)	valor_total_uf numeric	percentual_do_total numeric	percentual_acumulado numeric	valor_medio_por_beneficiario numeric	valor_medio_po numeric
13	PR	2341203225.00	2.887937848853324343 00	81.957723355404017989 00	3574.2523880986658402	5867677.(
14	AL	2116157443.00	2.610337756463824846 00	84.568061111867842835 00	3751.1343731498321333	20746641.(
15	GO	2000773989.00	2.468009128013372193 00	87.036070239881215028 00	3689.5248605438099892	8133227.(
16	RN	1893390554.00	2.335548740566066854 00	89.371618980447281882 00	3630.5929972579624552	11337667.(
17	SE	1458140976.00	1.798656549157250900 00	91.170275529604532782 00	3674.0282033777864230	19441879.(
18	ES	1170077066.00	1.443321882053468132 00	92.613597411658000914 00	3650.7295237203787772	15000988.(
19	MT	1040648624.00	1.283668378940803806 00	93.897265790598804720 00	3748.5992003169914628	7380486.(
20	SC	884240619.00	1.090734851137748607 00	94.988000641736553327 00	3588.5951835002008904	2997425.(
21	MS	829243084.00	1.022893896015138345 00	96.010894537751691672 00	3724.7422146960665855	10496747.(
22	DF	741424315.00	0.914567056154917750 00	96.925461593906609422 00	3767.0745667294999924	741424
23	TO	639682596.00	0.789065878824402413 00	97.714527472731011835 00	3774.8739864745246610	4602033.(
24	AC	548744150.00	0.676890832542675120 00	98.391418305273686955 00	4101.9012842171358519	24942915.(
25	RO	497566036.00	0.613761237095645868 00	99.005179542369332823 00	3736.2942081984816514	9568577.(
26	AP	485517425.00	0.598898947755936582 00	99.604078490125269405 00	4035.2179604388297872	30344839.(
27	RR	320966989.00	0.395921509874730596 00	100.000000000000000100	4103.3352808069444267	21397799.(

Total rows: 27 | Query complete 00:08:33.112 | CRLF | Ln 46, Col 1

Gráfico:



Comando em SQL para banco amostral:

```
WITH StatsPorUF AS (
    SELECT
        m.uf,
        SUM(p.valor_parcela) AS valor_total_uf,
        COUNT(DISTINCT p.pessoa_nis) AS qtd_beneficiarios_uf,
        COUNT(DISTINCT m.codigo_siafi) AS qtd_municipios_uf
    FROM
        projeto_sbd.pagamentos AS p
    JOIN
        projeto_sbd.municipios AS m ON p.municipio_siafi =
m.codigo_siafi
    WHERE
        EXTRACT(YEAR FROM p.mes_referencia) = 2024
    GROUP BY
        m.uf
),
CalculosGerais AS (
    SELECT
        uf,
        valor_total_uf,
        qtd_beneficiarios_uf,
        qtd_municipios_uf,
        -- Percentual que cada estado representa do total geral
        (valor_total_uf / SUM(valor_total_uf) OVER ()) * 100 AS
percentual_do_total,
        -- Valor médio por beneficiário no estado
        valor_total_uf / qtd_beneficiarios_uf AS
valor_medio_por_beneficiario,
```

```
        -- Valor médio por município no estado
        valor_total_uf / qtd_municipios_uf AS
valor_medio_por_municipio
    FROM
        StatsPorUF
)
SELECT
    uf,
    valor_total_uf,
    percentual_do_total,
    -- Soma o percentual da linha atual com a soma de todos os
percentuais das linhas anteriores
    SUM(percentual_do_total) OVER (ORDER BY percentual_do_total DESC)
AS percentual_acumulado,
    valor_medio_por_beneficiario,
    valor_medio_por_municipio
FROM
    CalculosGerais
ORDER BY
    percentual_do_total DESC;
```

4.1.3. TERCEIRA CONSULTA

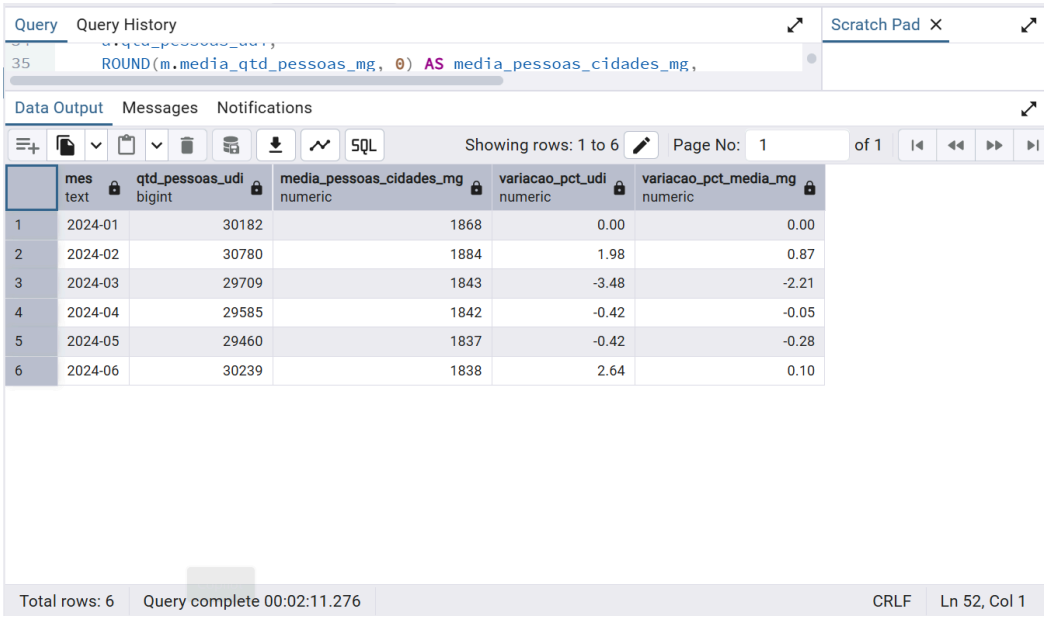
Título: Análise Comparativa da Dinâmica do Bolsa Família: Variação Percentual Mensal de Beneficiários em Uberlândia versus a Média das Cidades de Minas Gerais (2024).

Comando SQL:

```
WITH UdiData AS (
    -- Calcular os totais mensais apenas para Uberlândia
    SELECT
        TO_CHAR(mes_referencia, 'YYYY-MM') AS mes,
        SUM(valor_parcela) AS valor_total_udi,
        COUNT(DISTINCT pessoa_nis) AS qtd_pessoas_udi
    FROM pagamentos p
    JOIN municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE m.nome = 'UBERLANDIA' AND m.uf = 'MG' AND EXTRACT(YEAR FROM
p.mes_referencia) = 2024
    GROUP BY mes
),
MGData AS (
    -- Calcular a média mensal para as cidades de MG
    SELECT
        mes, -- <-- ESTA É A LINHA QUE FOI CORRIGIDA
        AVG(valor_total_por_cidade) AS media_valor_total_mg,
        AVG(qtd_pessoas_por_cidade) AS media_qtd_pessoas_mg
```

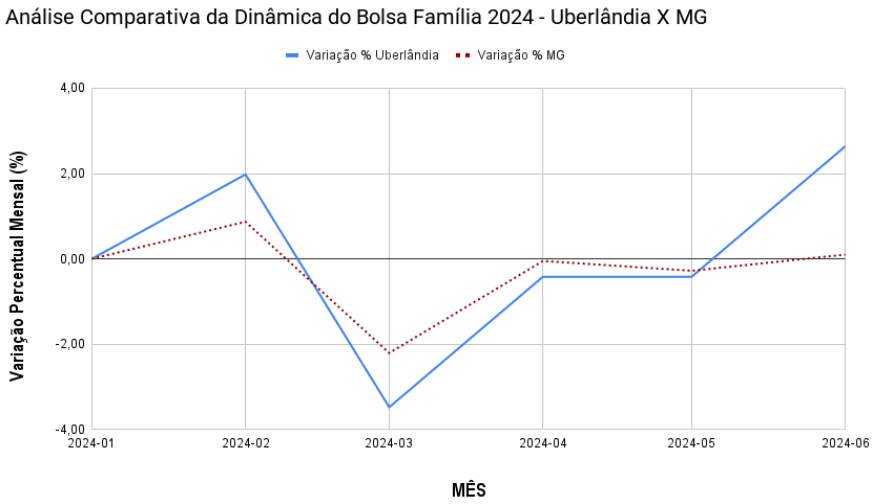
```
    FROM (
        SELECT
            p.municipio_siafi,
            TO_CHAR(p.mes_referencia, 'YYYY-MM') AS mes,
            SUM(p.valor_parcela) as valor_total_por_cidade,
            COUNT(DISTINCT p.pessoa_nis) as qtd_pessoas_por_cidade
        FROM pagamentos p
        JOIN municipios m ON p.municipio_siafi = m.codigo_siafi
        WHERE m.uf = 'MG' AND EXTRACT(YEAR FROM p.mes_referencia) =
2024
        GROUP BY p.municipio_siafi, mes
    ) AS stats_cidades_mg
    GROUP BY mes
)
-- juntar os dados de Uberlândia e da média de MG para comparar
SELECT
    u.mes,
    u.qtd_pessoas_udi,
    ROUND(m.media_qtd_pessoas_mg, 0) AS media_pessoas_cidades_mg,
    -- Cálculo da variação percentual de Uberlândia em relação ao mês
anterior
    ROUND(
        ( (u.qtd_pessoas_udi - LAG(u.qtd_pessoas_udi, 1,
u.qtd_pessoas_udi) OVER (ORDER BY u.mes)) * 100.0 )
        / LAG(u.qtd_pessoas_udi, 1, u.qtd_pessoas_udi) OVER (ORDER BY
u.mes), 2
    ) AS variacao_pct_udi,
    -- Cálculo da variação percentual da média das cidades de MG em
relação ao mês anterior
    ROUND(
        ( (m.media_qtd_pessoas_mg - LAG(m.media_qtd_pessoas_mg, 1,
m.media_qtd_pessoas_mg) OVER (ORDER BY u.mes)) * 100.0 )
        / LAG(m.media_qtd_pessoas_mg, 1, m.media_qtd_pessoas_mg) OVER
(ORDER BY u.mes), 2
    ) AS variacao_pct_media_mg
FROM
    UdiData u
JOIN
    MGData m ON u.mes = m.mes
ORDER BY
    u.mes;
```

Print do resultado da consulta:



	mes text	qtd_pessoas_udi bigint	media_pessoas_cidades_mg numeric	variacao_pct_udi numeric	variacao_pct_media_mg numeric
1	2024-01	30182	1868	0.00	0.00
2	2024-02	30780	1884	1.98	0.87
3	2024-03	29709	1843	-3.48	-2.21
4	2024-04	29585	1842	-0.42	-0.05
5	2024-05	29460	1837	-0.42	-0.28
6	2024-06	30239	1838	2.64	0.10

Gráfico:



Comando em SQL para banco amostral:

```
WITH UdiData AS (
    SELECT
        TO_CHAR(mes_referencia, 'YYYY-MM') AS mes,
        SUM(valor_parcela) AS valor_total_udi,
        COUNT(DISTINCT pessoa_nis) AS qtd_pessoas_udi
    FROM projeto_sbd.pagamentos p
    JOIN projeto_sbd.municipios m ON p.municipio_siafi =
m.codigo_siafi
    WHERE m.nome = 'UBERLANDIA' AND m.uf = 'MG' AND EXTRACT(YEAR FROM
```

```
p.mes_referencia) = 2024
    GROUP BY mes
),
MGData AS (
    SELECT
        mes, -- <-- ESTA É A LINHA QUE FOI CORRIGIDA
        AVG(valor_total_por_cidade) AS media_valor_total_mg,
        AVG(qtd_pessoas_por_cidade) AS media_qtd_pessoas_mg
    FROM (
        SELECT
            p.municipio_siafi,
            TO_CHAR(p.mes_referencia, 'YYYY-MM') AS mes,
            SUM(p.valor_parcela) as valor_total_por_cidade,
            COUNT(DISTINCT p.pessoa_nis) as qtd_pessoas_por_cidade
        FROM projeto_sbd.pagamentos p
        JOIN projeto_sbd.municipios m ON p.municipio_siafi =
m.codigo_siafi
        WHERE m.uf = 'MG' AND EXTRACT(YEAR FROM p.mes_referencia) =
2024
        GROUP BY p.municipio_siafi, mes
    ) AS stats_cidades_mg
    GROUP BY mes
)
SELECT
    u.mes,
    u.qtd_pessoas_udi,
    ROUND(m.media_qtd_pessoas_mg, 0) AS media_pessoas_cidades_mg,
    ROUND(
        ( (u.qtd_pessoas_udi - LAG(u.qtd_pessoas_udi, 1,
u.qtd_pessoas_udi) OVER (ORDER BY u.mes)) * 100.0 )
        / LAG(u.qtd_pessoas_udi, 1, u.qtd_pessoas_udi) OVER (ORDER BY
u.mes), 2
    ) AS variacao_pct_udi,
    ROUND(
        ( (m.media_qtd_pessoas_mg - LAG(m.media_qtd_pessoas_mg, 1,
m.media_qtd_pessoas_mg) OVER (ORDER BY u.mes)) * 100.0 )
        / LAG(m.media_qtd_pessoas_mg, 1, m.media_qtd_pessoas_mg) OVER
(ORDER BY u.mes), 2
    ) AS variacao_pct_media_mg
FROM
    UdiData u
JOIN
    MGData m ON u.mes = m.mes
ORDER BY
    u.mes;
```

4.1.4. QUARTA CONSULTA

Título: Análise Percentual da Dependência Contínua do Bolsa Família em Uberlândia (1º Semestre de 2024).

Comando SQL:

```
WITH BeneficiariosContinuos AS (
    -- Identificar o NIS de todos que receberam por 6 meses em
Uberlândia
    SELECT
        p.pessoa_nis
    FROM
        pagamentos p
    JOIN
        municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        m.nome = 'UBERLANDIA' AND m.uf = 'MG'
        AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
    GROUP BY
        p.pessoa_nis
    HAVING
        COUNT(DISTINCT DATE_TRUNC('month', p.mes_referencia)) = 6
),
TotalBeneficiarios AS (
    -- Contar o total de beneficiários únicos em Uberlândia em 2024
    SELECT
        COUNT(DISTINCT p.pessoa_nis) as total
    FROM
        pagamentos p
    JOIN
        municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        m.nome = 'UBERLANDIA' AND m.uf = 'MG'
        AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
)
-- Apresentar os resultados finais e calcular a porcentagem
SELECT
    (SELECT total FROM TotalBeneficiarios) AS
total_beneficiarios_uberlandia,
    (SELECT COUNT(*) FROM BeneficiariosContinuos) AS
beneficiarios_dependentes_6_meses,
    -- Cálculo da porcentagem
    ROUND(
        ( (SELECT COUNT(*) FROM BeneficiariosContinuos)::DECIMAL /
(SELECT total FROM TotalBeneficiarios)::DECIMAL ) * 100
    , 2) AS porcentagem_dependencia_continua;
```

Print do resultado da consulta:

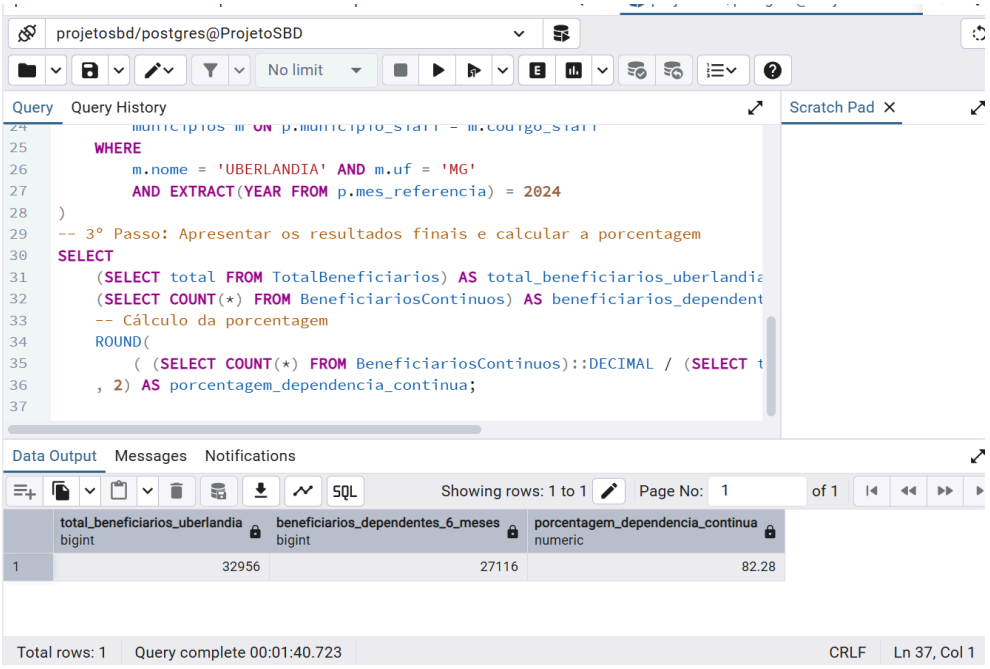
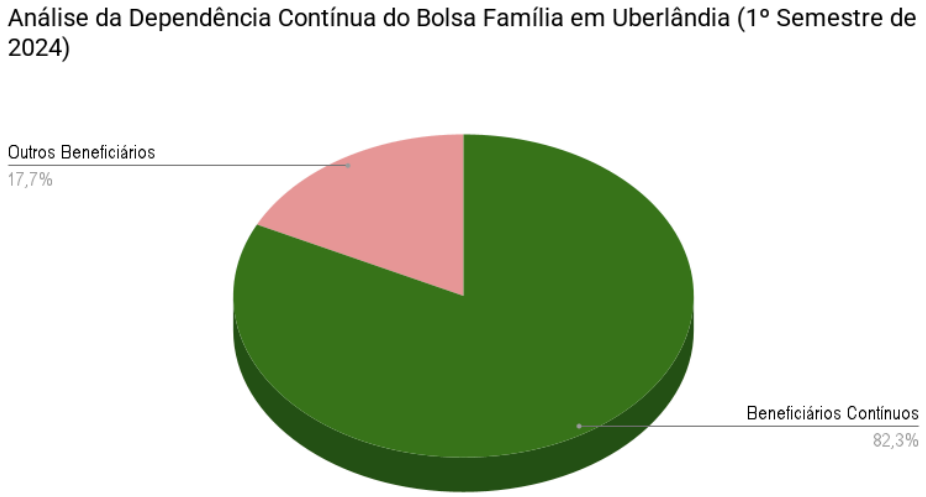


Gráfico:



Comando em SQL para banco amostral:

```
WITH BeneficiariosContinuos AS (
    SELECT
        p.pessoa_nis
    FROM
        projeto_sbd.pagamentos p
    JOIN
        projeto_sbd.municipios m ON p.municipio_siafi =
```

```
m.codigo_siafi
    WHERE
        m.nome = 'UBERLANDIA' AND m.uf = 'MG'
        AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
    GROUP BY
        p.pessoa_nis
    HAVING
        COUNT(DISTINCT DATE_TRUNC('month', p.mes_referencia)) = 6
),
TotalBeneficiarios AS (
    SELECT
        COUNT(DISTINCT p.pessoa_nis) as total
    FROM
        projeto_sbd.pagamentos p
    JOIN
        projeto_sbd.municipios m ON p.municipio_siafi =
m.codigo_siafi
    WHERE
        m.nome = 'UBERLANDIA' AND m.uf = 'MG'
        AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
)
SELECT
    (SELECT total FROM TotalBeneficiarios) AS
total_beneficiarios_uberlandia,
    (SELECT COUNT(*) FROM BeneficiariosContinuos) AS
beneficiarios_dependentes_6_meses,
    -- Cálculo da porcentagem
    ROUND(
        ( (SELECT COUNT(*) FROM BeneficiariosContinuos)::DECIMAL /
(SELECT total FROM TotalBeneficiarios)::DECIMAL ) * 100
    , 2) AS porcentagem_dependencia_continua;
```

4.1.5. QUINTA CONSULTA

Título: Auditoria de Dados: Pagamentos Individuais com Valor Superior a 200% da Média Municipal em 2024.

Comando SQL:

```
WITH MediaPorMunicipio AS (
    SELECT
        municipio_siafi,
        AVG(valor_parcela) AS valor_medio_municipal
    FROM pagamentos
    WHERE EXTRACT(YEAR FROM mes_referencia) = 2024
    GROUP BY municipio_siafi
)
```

```sql
SELECT
    pe.nome AS nome_pessoa,
    m.nome AS municipio,
    m.uf,
    p.valor_parcela,
    mpm.valor_medio_municipal
FROM
    pagamentos p
JOIN
    pessoa pe ON p.pessoa_nis = pe.nis
JOIN
    municipios m ON p.municipio_siafi = m.codigo_siafi
JOIN
    MediaPorMunicipio mpm ON p.municipio_siafi = mpm.municipio_siafi
WHERE
    p.valor_parcela > (mpm.valor_medio_municipal * 2)
    AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
ORDER BY
    p.valor_parcela DESC;
```

Print do resultado da consulta:

	nome_pessoa character varying (150)	municipio character varying (100)	uf character (2)	valor_parcela numeric (10,2)	valor_medio_municipal numeric
1	MARLUCIA RODRIGUES DE SOUSA OLIVEIRA	ITAQUAQUECETUBA	SP	9300.00	684.0532488731879644
2	CRISTIANO SANTOS COSTA	LAGARTO	SE	6924.00	624.2944016945261178
3	ALINE VILLAS BOAS MANOEL SALVADOR	BANDEIRANTES	PR	4790.00	664.8860540290925883
4	ALINE VILLAS BOAS MANOEL SALVADOR	BANDEIRANTES	PR	4790.00	664.8860540290925883
5	ALINE VILLAS BOAS MANOEL SALVADOR	BANDEIRANTES	PR	4790.00	664.8860540290925883
6	JESSICA KAROLAYNE DE LIMA BARBOSA	ITACOATIARA	AM	4664.00	734.0551082978933834
7	JESSICA KAROLAYNE DE LIMA BARBOSA	ITACOATIARA	AM	4664.00	734.0551082978933834
8	RIVAIL VALENTE LEITAO	ALENQUER	PA	4630.00	707.0345152842763705
9	CASSIANE SOARES LEMOS	ALENQUER	PA	4288.00	707.0345152842763705
10	FRANCISCO PEDRO PAIVA MENEZES	ALENQUER	PA	4246.00	707.0345152842763705
11	FRANCISCO PEDRO PAIVA MENEZES	ALENQUER	PA	4246.00	707.0345152842763705
12	RIVALDO SILVA LEITAO	ALENQUER	PA	4238.00	707.0345152842763705
13	RIVALDO SILVA LEITAO	ALENQUER	PA	4238.00	707.0345152842763705
14	LENITA CORREIA DE MENEZES	AUTAZES	AM	4056.00	693.5223625847973561
15	LENITA CORREIA DE MENEZES	AUTAZES	AM	4056.00	693.5223625847973561
16	LENITA CORREIA DE MENEZES	AUTAZES	AM	4056.00	693.5223625847973561
17	LENITA CORREIA DE MENEZES	AUTAZES	AM	4056.00	693.5223625847973561
18	LENITA CORREIA DE MENEZES	AUTAZES	AM	4056.00	693.5223625847973561
19	LENITA CORREIA DE MENEZES	AUTAZES	AM	4056.00	693.5223625847973561
20	CASSIANE SOARES LEMOS	ALENQUER	PA	3996.00	707.0345152842763705
21	ALVINA LEMES	CORONEL VIVIDA	PR	3970.00	647.2692861255037421
22	MATHEUS VIEIRA DA SILVA MOREIRA	ITACOATIARA	AM	3938.00	734.0551082978933834
23	MATHEUS VIEIRA DA SILVA MOREIRA	ITACOATIARA	AM	3938.00	734.0551082978933834
24	EVERTON DA SILVA DE MELO	ITACOATIARA	AM	3938.00	734.0551082978933834
25	EVERTON DA SILVA DE MELO	ITACOATIARA	AM	3938.00	734.0551082978933834
26	EVERTON DA SILVA DE MELO	ITACOATIARA	AM	3938.00	734.0551082978933834
27	EVERTON DA SILVA DE MELO	ITACOATIARA	AM	3938.00	734.0551082978933834
28	EVERTON DA SILVA DE MELO	ITACOATIARA	AM	3938.00	734.0551082978933834
29	RIVALDO SILVA LEITAO	ALENQUER	PA	3896.00	707.0345152842763705
30	CASSIANE SOARES LEMOS	ALENQUER	PA	3804.00	707.0345152842763705
31	PAULO OLIVEIRA DE ARAUJO	ALENQUER	PA	3796.00	707.0345152842763705
32	BERENICE REDZE	RIO VERDE	GO	3774.00	645.2071129957714929
33	BERENICE REDZE	RIO VERDE	GO	3774.00	645.2071129957714929
34	ERINALDO SOARES DA SILVA	ALENQUER	PA	3736.00	707.0345152842763705

Gráfico: Essa consulta retorna beneficiários individuais, portanto, o uso de gráficos para ilustrar o resultado não é adequado.

Comando em SQL para banco amostral:

```
WITH MediaPorMunicipio AS (
    -- 1º Passo: Calcular o valor médio do benefício para cada
município
    SELECT
        municipio_siafi,
        AVG(valor_parcela) AS valor_medio_municipal
    FROM projeto_sbd.pagamentos
    WHERE EXTRACT(YEAR FROM mes_referencia) = 2024
    GROUP BY municipio_siafi
)
-- 2º Passo: Juntar os pagamentos com a média de seu município e
encontrar os outliers
SELECT
    pe.nome AS nome_pessoa,
    m.nome AS municipio,
    m.uf,
    p.valor_parcela,
    mpm.valor_medio_municipal
FROM
    projeto_sbd.pagamentos p
JOIN
    projeto_sbd.pessoa pe ON p.pessoa_nis = pe.nis
JOIN
    projeto_sbd.municipios m ON p.municipio_siafi = m.codigo_siafi
JOIN
    MediaPorMunicipio mpm ON p.municipio_siafi = mpm.municipio_siafi
WHERE
    p.valor_parcela > (mpm.valor_medio_municipal * 2) -- Condição:
valor é mais que 200% acima da
--média
    AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
ORDER BY
    p.valor_parcela DESC;
```

4.1.6. SEXTA CONSULTA

Título: Municípios com Investimento Total Superior a R$ 50 Milhões.

Comando SQL:

```
 SELECT
     m.nome AS nome_municipio,
```

```
    m.uf,
    SUM(p.valor_parcela) AS valor_total_pago,
    COUNT(DISTINCT p.pessoa_nis) AS quantidade_beneficiarios
FROM
    pagamentos AS p
JOIN
    municipios AS m ON p.municipio_siafi = m.codigo_siafi
GROUP BY
    m.codigo_siafi, m.nome, m.uf
HAVING
    SUM(p.valor_parcela) > 50000000
ORDER BY
    valor_total_pago DESC;
```

Print do resultado da consulta:

	nome_municipio character varying (100)	uf character (2)	valor_total_pago numeric	quantidade_beneficiarios bigint
1	SAO PAULO	SP	2690895005.00	768647
2	RIO DE JANEIRO	RJ	2144399529.00	585454
3	FORTALEZA	CE	1271649328.00	350882
4	SALVADOR	BA	1108658837.00	316425
5	MANAUS	AM	1089622204.00	279216
6	BRASILIA	DF	742524567.00	196818
7	BELEM	PA	694293950.00	191555
8	RECIFE	PE	552425583.00	154445
9	BELO HORIZONTE	MG	502713953.00	140026
10	SAO LUIS	MA	488790966.00	132063
11	NOVA IGUACU	RJ	482618805.00	130129
12	DUQUE DE CAXIAS	RJ	464827187.00	125221
13	MACEIO	AL	429969548.00	116447
14	GUARULHOS	SP	420907370.00	114912
15	TERESINA	PI	395091508.00	108329
16	JABOATAO DOS GUARARAP...	PE	364136458.00	103463
17	JOAO PESSOA	PB	323438543.00	89243

Gráfico: Dado o alto volume de resultados (260 municípios), um gráfico de barras se tornaria ilegível e visualmente poluído. Portanto, a visualização em formato de tabela ordenada é a mais eficaz para esta análise.

Comando em SQL para banco amostral:

```
-- Consulta de Diagnóstico: Ver os 10 municípios com maior
investimento total
SELECT
    m.nome AS nome_municipio,
    m.uf,
    SUM(p.valor_parcela) AS valor_total_pago
FROM
```

```
    projeto_sbd.pagamentos AS p
JOIN
    projeto_sbd.municipios AS m ON p.municipio_siafi = m.codigo_siafi
GROUP BY
    m.codigo_siafi, m.nome, m.uf
-- A linha abaixo foi comentada para podermos ver os valores
-- HAVING SUM(p.valor_parcela) > 50000000
ORDER BY
    valor_total_pago DESC
LIMIT 10;
```

4.1.7. SÉTIMA CONSULTA

Título: Concentração de Recursos em Minas Gerais: Comparativo de Verbas entre a Capital (Belo Horizonte) e o Interior em 2024.

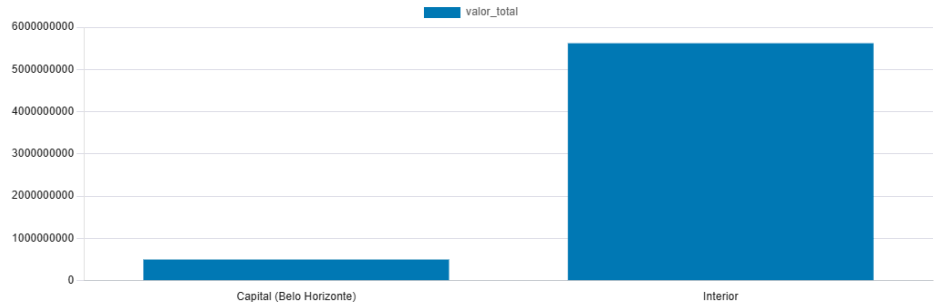
Comando SQL:

```
SELECT
    CASE
        WHEN m.nome = 'BELO HORIZONTE' THEN 'Capital (Belo Horizonte)'
        ELSE 'Interior'
    END AS localizacao,
    SUM(p.valor_parcela) AS valor_total,
    COUNT(DISTINCT p.pessoa_nis) AS total_beneficiarios
FROM
    pagamentos AS p
JOIN
    municipios AS m ON p.municipio_siafi = m.codigo_siafi
WHERE
    m.uf = 'MG' AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
GROUP BY
    localizacao;
```

Print do resultado da consulta:

	localizacao text	valor_total numeric	total_beneficiarios bigint
1	Capital (Belo Horizonte)	498637176.00	140026
2	Interior	5619419027.00	1557467

Gráfico:



Comando em SQL para banco amostral:

```
SELECT
    CASE
        WHEN m.nome = 'BELO HORIZONTE' THEN 'Capital (Belo Horizonte)'
        ELSE 'Interior'
    END AS localizacao,
    SUM(p.valor_parcela) AS valor_total,
    COUNT(DISTINCT p.pessoa_nis) AS total_beneficiarios
FROM
    projeto_sbd.pagamentos AS p
JOIN
    projeto_sbd.municipios AS m ON p.municipio_siafi = m.codigo_siafi
WHERE
    m.uf = 'MG' AND EXTRACT(YEAR FROM p.mes_referencia) = 2024
GROUP BY
    localizacao;
```

4.1.8. OITAVA CONSULTA

Título: Variação no Número de Beneficiários por UF entre Janeiro e Junho de 2024.

Comando SQL:

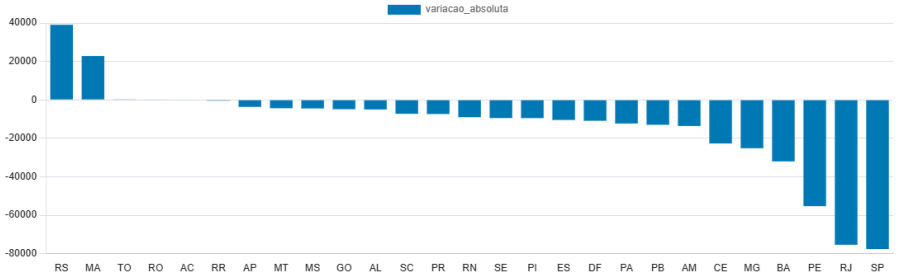
```
SELECT
    m.uf,
    COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-01-01' THEN
p.pessoa_nis END) AS beneficiarios_jan,
    COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-06-01' THEN
p.pessoa_nis END) AS beneficiarios_jun,
    (COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-06-01' THEN
p.pessoa_nis END) -
     COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-01-01' THEN
p.pessoa_nis END)) AS variacao_absoluta
FROM
```

```
    pagamentos p
JOIN
    municipios m ON p.municipio_siafi = m.codigo_siafi
WHERE
    p.mes_referencia IN ('2024-01-01', '2024-06-01') -- Assumindo
formato AAAA-MM-DD
GROUP BY
    m.uf
ORDER BY
    variacao_absoluta DESC;
```

Print do resultado da consulta:

	uf character (2)	beneficiarios_jan bigint	beneficiarios_jun bigint	variacao_absoluta bigint
1	RS	616595	655641	39046
2	MA	1180542	1203327	22785
3	TO	156545	156850	305
4	RO	122835	123011	176
5	AC	128991	128754	-237
6	RR	72067	71659	-408
7	AP	117618	113969	-3649
8	MT	257847	253496	-4351
9	MS	209306	204848	-4458
10	GO	507680	502804	-4876
11	AL	535089	530093	-4996
12	SC	227852	220601	-7251
13	PR	606919	599556	-7363
14	RN	503010	493987	-9023
15	SE	383506	374047	-9459
16	PI	603045	593556	-9489
17	ES	303845	293372	-10473
18	DF	192784	181903	-10881
19	PA	1333551	1321237	-12314
20	PB	672836	659864	-12972
21	AM	644144	630547	-13597
22	CE	1467867	1445175	-22692
23	MG	1593315	1568144	-25171
24	BA	2463794	2431770	-32024
25	PE	1605289	1549978	-55311
26	RJ	1712338	1636959	-75379
27	SP	2559796	2482188	-77608

Gráfico:



Comando em SQL para banco amostral:

```
SELECT
    m.uf,
    COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-01-01' THEN
p.pessoa_nis END) AS beneficiarios_jan,
    COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-06-01' THEN
p.pessoa_nis END) AS beneficiarios_jun,
    (COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-06-01' THEN
p.pessoa_nis END) -
     COUNT(DISTINCT CASE WHEN p.mes_referencia = '2024-01-01' THEN
p.pessoa_nis END)) AS variacao_absoluta
FROM
    projeto_sbd.pagamentos p
JOIN
    projeto_sbd.municipios m ON p.municipio_siafi = m.codigo_siafi
WHERE
    p.mes_referencia IN ('2024-01-01', '2024-06-01') -- Assumindo
formato AAAA-MM-DD
GROUP BY
    m.uf
ORDER BY
    variacao_absoluta DESC;
```

4.1.9. NONA CONSULTA

Título: Mediana de Beneficiários Únicos por Município para Cada Estado

Comando SQL:

```
-- Título: Mediana de Beneficiários Únicos por Município para Cada
Estado
WITH BeneficiariosPorMunicipio AS (
    SELECT
        m.uf,
```

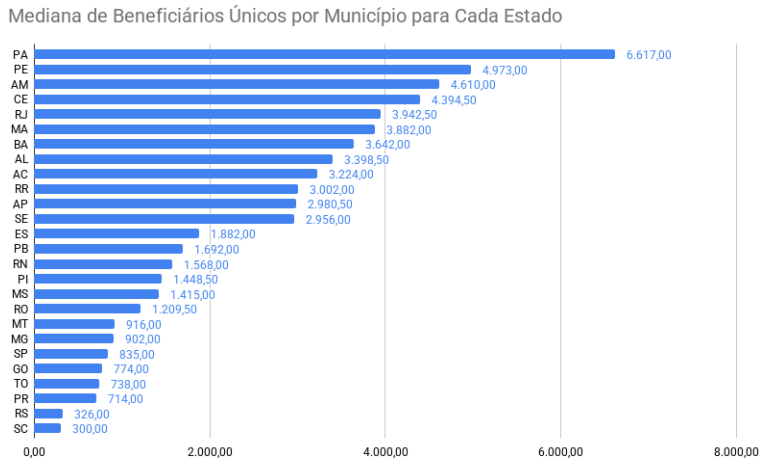
```
            COUNT(DISTINCT p.pessoa_nis) AS total_beneficiarios
        FROM
            pagamentos p
        JOIN
            municipios m ON p.municipio_siafi = m.codigo_siafi
        WHERE
            m.uf != 'DF' -- Excluindo o Distrito Federal
        GROUP BY
            m.uf, m.nome
    )
    SELECT
        uf,
        -- PERCENTILE_CONT(0.5) calcula a mediana (percentil 50)
        PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY total_beneficiarios)
    AS mediana_beneficiarios_por_municipio
    FROM
        BeneficiariosPorMunicipio
    GROUP BY
        uf
    ORDER BY
        mediana_beneficiarios_por_municipio DESC;
```

Print do resultado da consulta:

	uf character (2)	mediana_beneficiarios_por_municipio double precision
1	PA	6617
2	PE	4973
3	AM	4610
4	CE	4394.5
5	RJ	3942.5
6	MA	3882
7	BA	3642
8	AL	3398.5
9	AC	3224
10	RR	3002
11	AP	2980.5
12	SE	2956
13	ES	1882
14	PB	1692
15	RN	1568
16	PI	1448.5
17	MS	1415
18	RO	1209.5
19	MT	916
20	MG	902
21	SP	835
22	GO	774
23	TO	738
24	PR	714
25	RS	326
26	SC	300

Total rows: 26 | Query complete 00:09:54.990

Gráfico:



Comando em SQL para banco amostral:

```
WITH BeneficiariosPorMunicipio AS (
    SELECT
        m.uf,
        COUNT(DISTINCT p.pessoa_nis) AS total_beneficiarios
    FROM
        projeto_sbd.pagamentos p
    JOIN
        projeto_sbd.municipios m ON p.municipio_siafi =
m.codigo_siafi
    WHERE
        m.uf != 'DF' -- Excluindo o Distrito Federal
    GROUP BY
        m.uf, m.nome
)
SELECT
    uf,
    -- PERCENTILE_CONT(0.5) calcula a mediana (percentil 50)
    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY total_beneficiarios)
AS mediana_beneficiarios_por_municipio
FROM
    BeneficiariosPorMunicipio
GROUP BY
    uf
ORDER BY
    mediana_beneficiarios_por_municipio DESC;
```

4.1.10. DÉCIMA CONSULTA

Título: Comparativo de Perfis de Vulnerabilidade: Top 10 Municípios por Valor Médio de Parcela (Nacional vs. Foco Demográfico)

Comando SQL:

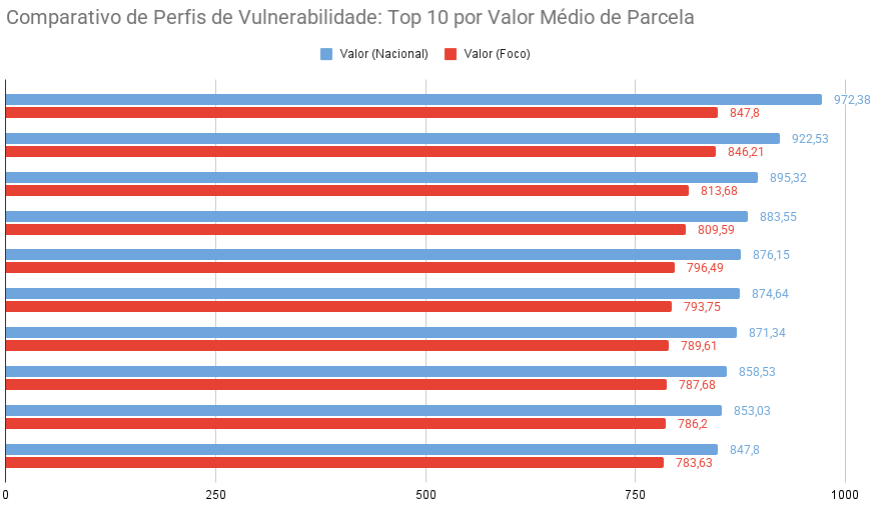
```
WITH FocoDemografico AS (
    SELECT
        'Foco Demográfico' AS categoria,
        m.nome AS municipio,
        m.uf,
        AVG(p.valor_parcela) AS valor_medio_parcela,
        RANK() OVER (ORDER BY AVG(p.valor_parcela) DESC) AS ranking
    FROM pagamentos p
    JOIN municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        m.uf IN ('BA', 'PI', 'MA', 'PA', 'SE')
        AND TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
    GROUP BY m.nome, m.uf
    LIMIT 10
),
Nacional AS (
    -- CTE para o Top 10 nacional
    SELECT
        'Nacional' AS categoria,
        m.nome AS municipio,
        m.uf,
        AVG(p.valor_parcela) AS valor_medio_parcela,
        RANK() OVER (ORDER BY AVG(p.valor_parcela) DESC) AS ranking
    FROM pagamentos p
    JOIN municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
    GROUP BY m.nome, m.uf
    LIMIT 10
)
-- Une os dois resultados em uma única tabela para comparação
SELECT categoria, ranking, municipio, uf, valor_medio_parcela
FROM FocoDemografico
UNION ALL
SELECT categoria, ranking, municipio, uf, valor_medio_parcela
FROM Nacional
ORDER BY ranking, categoria;
```

Print do resultado da consulta:

	categoria text	ranking bigint	municipio character varying (100)	uf character (2)	valor_medio_parcela numeric
1	Foco Demográfico	1	JACAREACANGA	PA	847.8089563019140484
2	Nacional	1	UIRAMUTA	RR	972.3503905627291567
3	Foco Demográfico	2	PORTEL	PA	846.1876229267008409
4	Nacional	2	CAMPINAPOLIS	MT	922.6132056246178928
5	Foco Demográfico	3	BAGRE	PA	813.7430134062562529
6	Nacional	3	JORDAO	AC	895.4593328335832084
7	Foco Demográfico	4	MELGACO	PA	810.3010908168442415
8	Nacional	4	SANTA ROSA DO PURUS	AC	884.0302094522019334
9	Foco Demográfico	5	ANAJAS	PA	796.9627400855145578
10	Nacional	5	SANTO ANTONIO DO ICA	AM	876.2012951760508764
11	Foco Demográfico	6	JURUTI	PA	793.8393904146984119
12	Nacional	6	NORMANDIA	RR	874.8263037024589705
13	Foco Demográfico	7	AFUA	PA	789.6379077918576436
14	Nacional	7	JUTAI	AM	871.2910895182964648
15	Foco Demográfico	8	CHAVES	PA	787.7749856197871729
16	Nacional	8	BENJAMIN CONSTANT	AM	858.5770889288618387
17	Foco Demográfico	9	BELAGUA	MA	787.3021340028244155
18	Nacional	9	SAO GABRIEL DA CACHOEIRA	AM	853.0372021156474713
19	Foco Demográfico	10	GURUPA	PA	783.9183120839444883
20	Nacional	10	JACAREACANGA	PA	847.8089563019140484

Total rows: 20 | Query complete 00:03:08.232

Gráfico:



Comando em SQL para banco amostral:

```sql
WITH FocoDemografico AS (
    -- CTE para o Top 10 dentro do filtro demográfico
    SELECT
        'Foco Demográfico' AS categoria,
        m.nome AS municipio,
        m.uf,
        AVG(p.valor_parcela) AS valor_medio_parcela,
        RANK() OVER (ORDER BY AVG(p.valor_parcela) DESC) AS ranking
    FROM projeto_sbd.pagamentos p
    JOIN projeto_sbd.municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        m.uf IN ('BA', 'PI', 'MA', 'PA', 'SE')
        AND TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
    GROUP BY m.nome, m.uf
    LIMIT 10
),
Nacional AS (
    -- CTE para o Top 10 nacional
    SELECT
        'Nacional' AS categoria,
        m.nome AS municipio,
        m.uf,
        AVG(p.valor_parcela) AS valor_medio_parcela,
        RANK() OVER (ORDER BY AVG(p.valor_parcela) DESC) AS ranking
    FROM projeto_sbd.pagamentos p
    JOIN projeto_sbd.municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
    GROUP BY m.nome, m.uf
    LIMIT 10
)
-- Une os dois resultados em uma única tabela para comparação
SELECT categoria, ranking, municipio, uf, valor_medio_parcela
FROM FocoDemografico
UNION ALL
SELECT categoria, ranking, municipio, uf, valor_medio_parcela
FROM Nacional
ORDER BY ranking, categoria;
```

4.1.11. DÉCIMA PRIMEIRA CONSULTA

Título: Comparativo do Perfil de Pagamentos: Distribuição de Parcelas de Baixo e Alto Valor entre Grupos de Estados

Comando SQL:

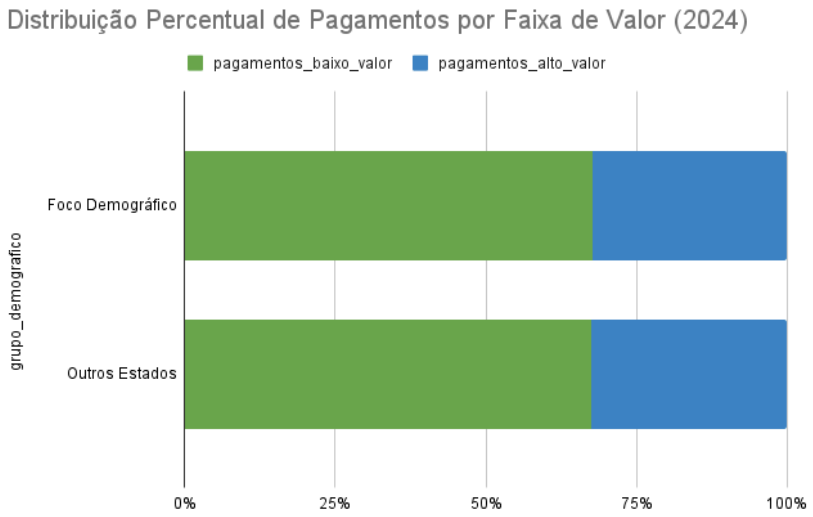
```
WITH PagamentosCategorizados AS (
    SELECT
        p.valor_parcela,
        CASE
            WHEN m.uf IN ('BA', 'PI', 'MA', 'PA', 'SE') THEN 'Foco
Demográfico'
            ELSE 'Outros Estados'
        END AS grupo_demografico
    FROM
        pagamentos p
    JOIN
        municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
)
SELECT
    grupo_demografico,
    COUNT(*) AS total_de_pagamentos,
    COUNT(*) FILTER (WHERE valor_parcela <= 700) AS
pagamentos_baixo_valor,
    COUNT(*) FILTER (WHERE valor_parcela > 700) AS
pagamentos_alto_valor,
    -- Calculando o percentual para uma análise mais clara
    (COUNT(*) FILTER (WHERE valor_parcela > 700)::numeric / COUNT(*)) *
100 AS percentual_alto_valor
FROM
    PagamentosCategorizados
GROUP BY
    grupo_demografico;
```

Print do resultado da consulta:

	grupo_demografico text	total_de_pagamentos bigint	pagamentos_baixo_valor bigint	pagamentos_alto_valor bigint	percentual_alto_valor numeric
1	Foco Demográfico	35661201	24151671	11509530	32.27465614520385895000
2	Outros Estados	87784801	59296853	28487948	32.45202777186907332600

Total rows: 2 Query complete 00:01:56.916

Gráfico:



Comando em SQL para banco amostral:

```
WITH PagamentosCategorizados AS (
    SELECT
        p.valor_parcela,
        CASE
            WHEN m.uf IN ('BA', 'PI', 'MA', 'PA', 'SE') THEN 'Foco
Demográfico'
            ELSE 'Outros Estados'
        END AS grupo_demografico
    FROM
        projeto_sbd.pagamentos p
    JOIN
        projeto_sbd.municipios m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
)
SELECT
    grupo_demografico,
    COUNT(*) AS total_de_pagamentos,
    COUNT(*) FILTER (WHERE valor_parcela <= 700) AS
pagamentos_baixo_valor,
    COUNT(*) FILTER (WHERE valor_parcela > 700) AS
pagamentos_alto_valor,
    -- Calculando o percentual para uma análise mais clara
    (COUNT(*) FILTER (WHERE valor_parcela > 700)::numeric / COUNT(*)) *
100 AS percentual_alto_valor
FROM
    PagamentosCategorizados
```

```
GROUP BY
    grupo_demografico;
```

4.1.12. DÉCIMA SEGUNDA CONSULTA

Título: Perfil Estatístico do Valor das Parcelas por Estado (Mínimo, Máximo, Média e Desvio Padrão)

Comando SQL:

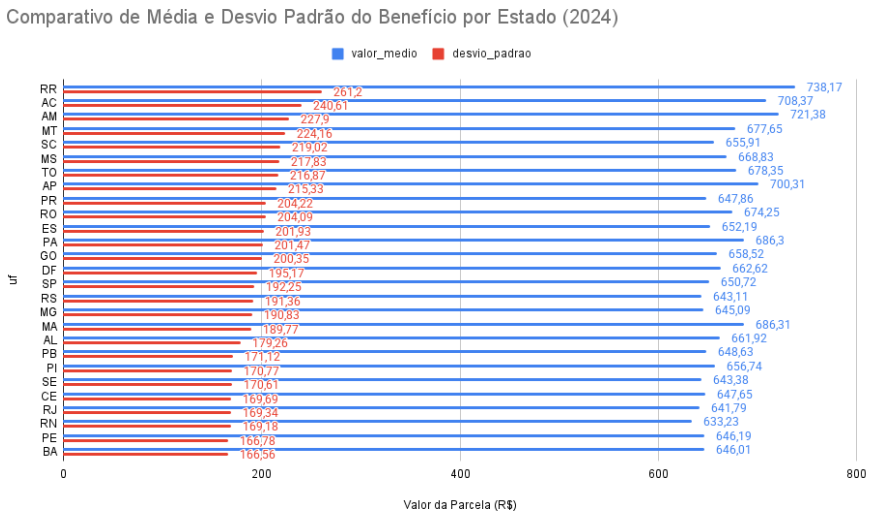
```
SELECT
    m.uf,
    MIN(p.valor_parcela) AS valor_minimo,
    MAX(p.valor_parcela) AS valor_maximo,
    AVG(p.valor_parcela) AS valor_medio,
    STDDEV(p.valor_parcela) AS desvio_padrao
FROM
    pagamentos p
JOIN
    municipios m ON p.municipio_siafi = m.codigo_siafi
WHERE
    TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
GROUP BY
    m.uf
ORDER BY
    desvio_padrao DESC;
```

Print do resultado da consulta:

	uf character (2)	valor_minimo numeric	valor_maximo numeric	valor_medio numeric	desvio_padrao numeric
1	RR	25.00	2988.00	738.1724764438965716	261.198259494617
2	AC	25.00	3030.00	708.3713932377726371	240.613065788436
3	AM	25.00	4664.00	721.3778889001530618	227.904159119878
4	MT	25.00	3514.00	677.6454671960293889	224.159532927083
5	SC	25.00	3000.00	655.9064615909562947	219.018917429252
6	MS	25.00	2880.00	668.8296444224344715	217.829795001021
7	TO	25.00	2688.00	678.3527742488287306	216.868548225450
8	AP	25.00	2788.00	700.3133266790328955	215.327161001183
9	PR	25.00	4790.00	647.8585098555326216	204.219072851216
10	RO	25.00	2696.00	674.2452653260339314	204.092066936592
11	ES	25.00	3406.00	652.1885146576681727	201.929490949690
12	PA	25.00	4630.00	686.3046875157079063	201.469377423842
13	GO	25.00	3774.00	658.5208395525765358	200.345958604909
14	DF	25.00	3046.00	662.6166646111388161	195.168479302135
15	SP	25.00	9300.00	650.7183582575817585	192.253721978206
16	RS	25.00	2944.00	643.1138763780599666	191.355698501873
17	MG	25.00	3664.00	645.0927275900082191	190.832029389031
18	MA	25.00	3362.00	686.3102739066366793	189.768436834965
19	AL	25.00	3230.00	661.9218211505197076	179.255964289826
20	PB	25.00	3046.00	648.6314723933759739	171.120297941083
21	PI	25.00	2986.00	656.7413293654951764	170.768760327030
22	SE	25.00	6924.00	643.3822644866147069	170.605563037384
23	CE	25.00	3046.00	647.6470849724482052	169.692322326861
24	RJ	25.00	3322.00	641.7866355978102273	169.343749699457
25	RN	25.00	2864.00	633.2282810378387056	169.180423603202
26	PE	25.00	2966.00	646.1941579370919788	166.776745234252
27	BA	25.00	3380.00	646.0107215531329928	166.563208031024

Total rows: 27 | Query complete 00:04:10.451

Gráfico:



Comando em SQL para banco amostral:

```
SELECT
```

```
    m.uf,
    MIN(p.valor_parcela) AS valor_minimo,
    MAX(p.valor_parcela) AS valor_maximo,
    AVG(p.valor_parcela) AS valor_medio,
    STDDEV(p.valor_parcela) AS desvio_padrao
FROM
    projeto_sbd.pagamentos p
JOIN
    projeto_sbd.municipios m ON p.municipio_siafi = m.codigo_siafi
WHERE
    TO_CHAR(p.mes_referencia, 'YYYY') = '2024'
GROUP BY
    m.uf
ORDER BY
    desvio_padrao DESC;
```

4.2. PROCEDIMENTO ARMAZENADO E GATILHO

Nesta etapa do projeto são apresentados o procedimento armazenado e o gatilho implementados no banco de dados. Essas funcionalidades foram desenvolvidas para complementar as consultas SQL da seção anterior, que já contou com scripts de aplicação no banco de dados amostral. O objetivo é automatizar tarefas, reforçar regras de negócio e garantir maior consistência na manipulação dos dados. A inclusão desses recursos também evidencia a flexibilidade do sistema em atender às demandas de análise e **controle do programa Novo Bolsa Família.**

4.2.1. PROCEDIMENTO ARMAZENADO

Título: Raio-X de um Município

Ideia: Criar uma função que recebe o código SIAFI e o ano como parâmetros e retorna um resumo completo do impacto do Bolsa Família naquele município, incluindo valor total pago, número de beneficiários únicos, valor médio do benefício e os meses de início e fim de registros.

Comando SQL:

```
CREATE TYPE tipo_raiox_municipio AS (
    nome_municipio VARCHAR(100), -- MUDANÇA AQUI: de TEXT para
VARCHAR(100)
    uf CHAR(2),
    valor_total_pago NUMERIC,
```

```
    beneficiarios_unicos BIGINT,
    valor_medio_beneficio NUMERIC,
    primeiro_mes_registro DATE,
    ultimo_mes_registro DATE
);
-- Passo 2: Recriar a função (o corpo da função continua o mesmo)
CREATE OR REPLACE FUNCTION raiox_municipio_por_ano(p_codigo_siafi
INTEGER, p_ano INTEGER)
RETURNS SETOF tipo_raiox_municipio AS $$
BEGIN
    RETURN QUERY
    SELECT
        m.nome AS nome_municipio,
        m.uf,
        SUM(p.valor_parcela) AS valor_total_pago,
        COUNT(DISTINCT p.pessoa_nis) AS beneficiarios_unicos,
        AVG(p.valor_parcela) AS valor_medio_beneficio,
        MIN(p.mes_referencia) AS primeiro_mes_registro,
        MAX(p.mes_referencia) AS ultimo_mes_registro
    FROM
        pagamentos AS p
    JOIN
        municipios AS m ON p.municipio_siafi = m.codigo_siafi
    WHERE
        p.municipio_siafi = p_codigo_siafi
        AND EXTRACT(YEAR FROM p.mes_referencia) = p_ano -- <-- FILTRO
DINÂMICO
    GROUP BY
        m.nome, m.uf;
END;
$$ LANGUAGE plpgsql;
```

Teste de aplicação:

```
-- Exemplo: Raio-X de Uberlândia em 2024 (supondo código 5403)
SELECT * FROM raiox_municipio_por_ano(5403, 2024);
```

Justificativa (Utilidade e Complexidade): Este procedimento facilita o trabalho de gestores públicos, pois com um único comando é possível obter um diagnóstico detalhado de um município específico. A utilidade está em centralizar cálculos complexos (SUM, COUNT, AVG, MIN, MAX) em uma função reutilizável. A complexidade vem da criação de um tipo customizado (CREATE

TYPE) e do retorno estruturado (SETOF), que organiza os resultados em formato de relatório.

Comando SQL para banco amostral:

```
CREATE TYPE tipo_raiox_municipio AS (
    nome_municipio VARCHAR(100), -- MUDANÇA AQUI: de TEXT para
VARCHAR(100)
    uf CHAR(2),
    valor_total_pago NUMERIC,
    beneficiarios_unicos BIGINT,
    valor_medio_beneficio NUMERIC,
    primeiro_mes_registro DATE,
    ultimo_mes_registro DATE
);
-- Passo 2: Recriar a função (o corpo da função continua o mesmo)
CREATE OR REPLACE FUNCTION raiox_municipio_por_ano(p_codigo_siafi
INTEGER, p_ano INTEGER)
RETURNS SETOF tipo_raiox_municipio AS $$
BEGIN
    RETURN QUERY
    SELECT
        m.nome AS nome_municipio,
        m.uf,
        SUM(p.valor_parcela) AS valor_total_pago,
        COUNT(DISTINCT p.pessoa_nis) AS beneficiarios_unicos,
        AVG(p.valor_parcela) AS valor_medio_beneficio,
        MIN(p.mes_referencia) AS primeiro_mes_registro,
        MAX(p.mes_referencia) AS ultimo_mes_registro
    FROM
        projeto_sbd.pagamentos AS p
    JOIN
        projeto_sbd.municipios AS m ON p.municipio_siafi =
m.codigo_siafi
    WHERE
        p.municipio_siafi = p_codigo_siafi
        AND EXTRACT(YEAR FROM p.mes_referencia) = p_ano -- <-- FILTRO
DINÂMICO
    GROUP BY
        m.nome, m.uf;
END;
$$ LANGUAGE plpgsql;
```

4.2.2. GATILHO

Título: Contagem Automática de beneficiários

Ideia: Adicionar uma coluna qtd_beneficiarios na tabela municipios e criar um gatilho que, a cada novo pagamento inserido, verifique se aquele beneficiário está recebendo pela primeira vez no município. Se for o caso, o contador de beneficiários do município é incrementado automaticamente.

Comando em SQL:

```
ALTER TABLE municipios ADD COLUMN qtd_beneficiarios INTEGER DEFAULT 0;

CREATE OR REPLACE FUNCTION atualizar_contagem_beneficiarios()
RETURNS TRIGGER AS $$
DECLARE
    v_municipio_siafi INTEGER;
BEGIN
    -- Descobre o município do primeiro pagamento da nova pessoa
    SELECT municipio_siafi INTO v_municipio_siafi
    FROM pagamentos
    WHERE pessoa_nis = NEW.nis
    LIMIT 1;

    -- Se encontrou um município, atualiza a contagem
    IF v_municipio_siafi IS NOT NULL THEN
        UPDATE municipios
        SET qtd_beneficiarios = qtd_beneficiarios + 1
        WHERE codigo_siafi = v_municipio_siafi;
    END IF;

    RETURN NULL; -- O retorno é ignorado em triggers AFTER
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_atualiza_contagem_beneficiarios
AFTER INSERT ON pessoa
FOR EACH ROW
EXECUTE FUNCTION atualizar_contagem_beneficiarios();
```

Teste de aplicação:

```
-- 1. Verificar o estado inicial do contador de beneficiários do
município
SELECT nome, qtd_beneficiarios
FROM municipios
```

```
WHERE codigo_siafi = 5403;

-- 2. Inserir uma nova pessoa
INSERT INTO pessoa (nis, nome, cpf)
VALUES (99999999999, 'BENEFICIARIO TESTE', '999.999.999-99');

-- 3. Inserir um pagamento para essa nova pessoa (aciona o gatilho)
INSERT INTO pagamentos (mes_competencia, mes_referencia,
municipio_siafi, pessoa_nis, valor_parcela)
VALUES ('2024-06-01', '2024-06-01', 5403, 99999999999, 700.00);

-- 4. Verificar se o contador foi incrementado
SELECT nome, qtd_beneficiarios
FROM municipios
WHERE codigo_siafi = 5403;
```

Justificativa (Utilidade e Complexidade): A utilidade do gatilho está em **otimizar a análise de dados**: em vez de executar consultas agregadas pesadas (como COUNT) toda vez que for necessário saber quantos beneficiários há em um município, o valor já fica atualizado em tempo real dentro da tabela municipios. Isso melhora a performance em consultas analíticas e relatórios. A complexidade está no fato de que o gatilho é do tipo **AFTER INSERT**, atua em uma tabela diferente daquela onde a operação ocorre (um `INSERT` em `pessoa` resulta em uma busca em `pagamentos` e um `UPDATE` em `municipios`), e precisa garantir consistência ao verificar se o pagamento inserido é realmente o primeiro daquele beneficiário.

Comando SQL para banco amostral:

```
ALTER TABLE projeto_sbd.municipios ADD COLUMN qtd_beneficiarios INTEGER
DEFAULT 0;

CREATE OR REPLACE FUNCTION atualizar_contagem_beneficiarios()
RETURNS TRIGGER AS $$
DECLARE
    v_municipio_siafi INTEGER;
BEGIN
    -- Descobre o município do primeiro pagamento da nova pessoa
    SELECT municipio_siafi INTO v_municipio_siafi
    FROM projeto_sbd.pagamentos
    WHERE pessoa_nis = NEW.nis
```

```
    LIMIT 1;

    -- Se encontrou um município, atualiza a contagem
    IF v_municipio_siafi IS NOT NULL THEN
        UPDATE projeto_sbd.municipios
        SET qtd_beneficiarios = qtd_beneficiarios + 1
        WHERE codigo_siafi = v_municipio_siafi;
    END IF;

    RETURN NULL; -- O retorno é ignorado em triggers AFTER
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_atualiza_contagem_beneficiarios
AFTER INSERT ON projeto_sbd.pessoa
FOR EACH ROW
EXECUTE FUNCTION atualizar_contagem_beneficiarios();
```

4.2.3. Demonstração da Utilidade do Gatilho na Otimização da Nona Consulta

```
-- Título: Mediana de Beneficiários Únicos por Município (Otimizada
com Gatilho)
SELECT
    uf,
    -- A função de mediana agora opera sobre a coluna pré-calculada
    PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY qtd_beneficiarios) AS
mediana_beneficiarios_por_municipio
FROM
    municipios
WHERE
    uf != 'DF' -- Excluindo o Distrito Federal
GROUP BY
    uf
ORDER BY
    mediana_beneficiarios_por_municipio DESC;
```

Nesta seção podemos ver melhor o valor prático do gatilho (trigger) criado no projeto, usando a Nona Consulta ("Mediana de Beneficiários") como exemplo.

- **Antes (Sem o Gatilho):** A consulta era lenta e pesada, pois precisava contar milhões de registros de pagamentos toda vez que era executada para descobrir a quantidade de beneficiários em cada município.
- **Depois (Com o Gatilho):** A consulta se torna muito mais rápida e eficiente. Em vez de recontar tudo, ela simplesmente lê o número de beneficiários de uma coluna que o gatilho mantém atualizada automaticamente.

Em resumo, a seção demonstra que o gatilho, apesar de adicionar uma pequena carga de trabalho no momento de inserir novos dados, oferece um grande ganho de performance para as análises, provando sua **utilidade e complexidade**.