# PROJETO FINAL – MERCADO DE TRABALHO


***2023***

## EQUIPE

Anderson Melo

Aska Pereira

Diego Aguiar

Jessica Staudt

Pedro Barrionovo

Rosana Santos

SUMÁRIO

## 1. Objetivos

O objetivo desse projeto tem como compreender o mercado de trabalho no Brasil, abordando o panorama geral do mercado de trabalho formal no Brasil entre os anos de 2012 e 2022, buscando saber as movimentações dos registros com CLT/CNPJ e dos informais, realizando um comparativo do mercado de trabalho formal e informal enquanto explora as características demográficas da população do país. Além disso, busca-se entender como o mercado de trabalho se comportou durante o período de 10 anos, incluindo uma análise breve do impacto da COVID-19 e como se encontra o mercado de trabalho pós-pandemia.

Em relação à população brasileira, este projeto visa abordar as diferenças de gênero, raciais, salariais, educacionais e sobre a inclusão e a participação de pessoas com deficiência no mercado de trabalho. Explora-se como a população está inserida no mercado de trabalho e quem são as pessoas que têm a maior participação no mercado de trabalho formal.



## 2. Sobre os dados

Os dados utilizados neste projeto foram encontrados no sítio eletrônico www.basedosdados.org , onde é possível encontrar diversas coleções de forma gratuita.

Dentre estas coleções, iremos utilizar as seguintes bases:

- Cadastro Geral de Empregados e Desempregados – CAGED: contém os dados de demissão e admissão para aqueles registrados com CLT que ocorreram entre 2007 e vão até a presente data.
- Pesquisa Nacional por Amostra de Domicílios Contínua (PNAD-C) – esta pesquisa é realizada pelo Instituto Brasileiro de Geografia e Estatística (IBGE) e possui dados produzidos continuamente sobre o mercado de trabalho, associadas a características demográficas e educacionais.

- Instituto de Pesquisa Econômico Aplicado – IPEA: traz de forma trimestral os registros encontrados da PNAD-C que vai de março de 2012 até maio de 2023.
- Censo 2010: feito pelo IBGE, traz dados sobre o Brasil na época de sua realização.

## 3. Metodologia

Neste projeto foi utilizada a metodologia KDD (Knowledge Discovery in Databases) que é um processo para encontrar conhecimento em grandes conjuntos de dados. Ele envolve as seguintes etapas: definição do problema, coleta de dados, pré-processamento de dados, mineração de dados, interpretação dos resultados e avaliação dos resultados.

O KDD é um processo iterativo, o que significa que as etapas podem ser repetidas várias vezes até que os resultados sejam satisfatórios.

- **Seleção dos dados e Extração**
  - Busca e acesso aos bancos de dados
  - Verificação do formato dos dados
  - Seleção e Extração dos dados para análise
  - Estratégias de análise

- **Pré-Processamento Big Query**
  - Análise da qualidade dos dados
  - Integridade dos dados: limpeza, correção e remoção de dados inconsistentes

- **Transformação dos dados Big Query e Python - Tratamento**
  - Normalização, Padronização e Tradução dos dados
  - Redução dos dados
  - Visualização de gráficos Heatmap e Dispersão no PySpark

- Carregamento dos dataframes resultantes na GCP e MongoDB

- **Modelagem dos dados**
  - Dashboards do tipo Tático
    - Monitora desempenho mensal, com objetivos de decisões estratégicas por Gerentes
  - Modelagem dos Dashboard no Power BI

- **Interpretação e Avaliação dos padrões**
  - Descoberta de Conhecimento (Conclusões e Insights)
  - Análise SWOT

## 4. Fluxo de Trabalho (Workflow)

As bases de dados do CAGED, PNAD-C e Censo 2010 foram encontradas no www.basedosdados.org e acessadas via Big Query. Já a base de dados do IPEA foi encontrada no site do instituto, na seção de dados do órgão e extraída via download convencional.

Após isto, foi feito o tratamento e transformação das bases maiores via SQL devido a impossibilidade de carregar via script em Python utilizando o Colab. Já a base do Ipea, foi tratada com script em Python utilizando o Colab com as ferramentas Pandas e PySpark, além do metplotlib.

Para dar saída nessas bases, utilizamos uma biblioteca da base dos dados para realizar o download das bases tratados diretamente no Google Drive que depois foi passado para o Google Cloud Plataform (GCP) pelo Cloud Storage utilizando a bucket. Após isso, realizamos a conexão com a BigQuery para podermos utilizar dentro do Power BI que foi utilizado para a construção dos dashboards apresentados neste projeto.

Segue o fluxo de trabalho:


Abaixo, uma rápida descrição das ferramentas utilizadas neste projeto:

**Google Colaboratory**, também conhecido como Colaboratory ou simplesmente Colab, é um serviço gratuito baseado em nuvem que permite aos usuários criar e executar notebooks Jupyter em um navegador da web. Os notebooks são documentos interativos que podem conter texto, código, equações, gráficos e outros conteúdos. Eles são uma ótima maneira de compartilhar ideias e colaborar com outros.

**Pandas** é uma biblioteca de código aberto para análise de dados em Python. É projetado para trabalhar com dados estruturados, como tabelas de planilhas e arquivos CSV. O Pandas fornece uma variedade de recursos para manipular, analisar e visualizar dados.

**PySpark** é uma API Python para Apache Spark. Ele permite que os usuários escrevam e executem aplicativos Spark usando Python. O PySpark é uma ferramenta poderosa para processamento de dados em grande escala.

**MongoDB** é um banco de dados NoSQL orientado a documentos de código aberto, que é usado para armazenar e gerenciar grandes quantidades de dados. Ele é projetado para ser flexível e escalável, e é usado por uma ampla variedade

de empresas, incluindo startups, grandes empresas e organizações governamentais.

**Google Cloud Platform (GCP)** é uma plataforma de computação em nuvem que oferece uma variedade de serviços, incluindo computação, armazenamento, rede, big data, machine learning, inteligência artificial, análise e muito mais. GCP é uma plataforma escalável e confiável que pode ajudar empresas de todos os tamanhos a criar e executar seus aplicativos em nuvem.

**Google Cloud Storage** é um serviço de armazenamento de objetos que permite armazenar e acessar dados de qualquer tamanho no Google Cloud Platform. O Cloud Storage é um serviço altamente escalável e confiável que pode ser usado para armazenar uma variedade de dados, incluindo imagens, vídeos, arquivos de log e dados de backup.

**Google BigQuery** é um data warehouse analítico totalmente gerenciado, sem servidor e baseado em nuvem do Google Cloud Platform que permite armazenar e analisar grandes conjuntos de dados estruturados. Ele pode lidar com até 1 petabyte por dia, e você só paga pelo que armazena e analisa. O BigQuery é ideal para empresas que precisam analisar grandes quantidades de dados para tomar decisões informadas.

**Power BI** é um conjunto de ferramentas de análise de negócios baseado em nuvem da Microsoft que ajuda os usuários a coletar, analisar e visualizar dados de uma variedade de fontes. Ele pode ser usado para criar relatórios, painéis e histórias de dados que podem ser compartilhados com outras pessoas.

# 5. Análise dos Dados

## 5.1. ESCOLHA DOS DADOS

As bases de dados para este trabalho foram escolhidas com o objetivo de enriquecer a análise de dados sobre o tema de "Mercado de Trabalho".

O processo de escolha das bases de dados adotado pelo grupo refletiu a busca por informações relevantes que pudessem proporcionar uma compreensão abrangente sobre as dinâmicas do mercado de trabalho no contexto brasileiro. Além disso, o grupo também procurou bases de dados que

possibilitassem ser feito um recorte temporal e social sobre minorias de representatividade no mercado de Trabalho do Brasil.

Para atender a esses objetivos, foram estabelecidas diretrizes específicas para a escolha das bases de dados:

1. Abrangência Demográfica e Social: A fonte das bases de dados escolhidas foram o Censo (IBGE - Instituto Brasileiro de Geografia e Estatística) e o PNAD (Pesquisa Nacional por Amostra de Domicílio) para fornecer um panorama da população brasileira em relação a critérios cruciais, como Idade, Raça ou Cor, Gênero, Pessoa com Deficiência, Escolaridade e Estado. A inclusão dessas informações em números absolutos e porcentagens de amostragem garantiram uma visão ampla das características demográficas e sociais da população.

2. Dinâmica do Mercado de Trabalho Formal: Para mostrar as movimentações de admissões e demissões dentro do mercado de trabalho formal no Brasil, a fonte das bases de dados usada foi do CAGED (Cadastro Geral de Desempregados e Empregados). Esta fonte permitiu ao grupo analisar tendências, flutuações e padrões de dados em números absolutos para entender a dinâmica do emprego formal no Brasil.

3. Comparação entre Mercado Formal e Informal: A estratégica usada para explorar as diferenças e semelhanças entre esses dois aspectos essenciais da economia teve como fonte o IPEA (Instituto de Pesquisa Econômica Aplicada). Essa análise por amostragem foi valiosa para destacar as características distintas de ambos os setores.

Em resumo, as bases de dados selecionadas foram o resultado de uma avaliação criteriosa das opções de bases de dados disponíveis. Cada base forneceu dados para o grupo produzir insights valiosos sobre o mercado de trabalho no Brasil. As instituições que foram fontes para as bases de dados são amplamente reconhecidas por sua autoridade e rigor na coleta e disseminação de dados.

Portanto, as bases de dados selecionadas ofereceram um arcabouço sólido para a pesquisa, permitindo que o grupo formulasse questões de pesquisa relevantes para conduzir análises sobre as dinâmicas do mercado de trabalho no contexto brasileiro. Além, de possibilitar ser feito um recorte temporal e social sobre minorias de representatividade no mercado de trabalho do Brasil.

## 5.2. MATRIZ SWOT

| FORÇAS | FRAQUEZAS |
|---|---|
| **Diversidade Demográfica:** A população brasileira é diversificada em termos de idade, gênero, raça e habilidades, o que pode ser um recurso valioso para a economia e a força de trabalho.<br>**Reforma Trabalhista:** A reforma trabalhista de 2017 trouxe flexibilidade para as relações de trabalho, o que pode incentivar a criação de empregos formais, permitindo adaptações às necessidades do mercado. | **Desigualdades Sociais:** O país enfrenta desigualdades no mercado de trabalho em relação a pessoas de deficientes, raça e gênero.<br>**Educação:** Com as análises, foi possível identificar que a maioria da população brasileira se enquadra no nível fundamental incompleto e que a maioria da população que está trabalhando no mercado formal tem o ensino médio completo. |
| **OPORTUNIDADES** | **AMEAÇAS** |
| **Capacitação e Educação:** Investir em educação e treinamento pode melhorar as habilidades da população e torná-la mais apta para empregos formais e qualificados.<br>**Análise Comparativa:** Comparar o mercado de trabalho formal e informal, bem como identificar as características demográficas, oferece a oportunidade de identificar disparidades e criar estratégias para reduzir essas diferenças. | **Riscos de Saúde Pública:** Eventos como pandemias (ex: COVID-19) podem impactar tanto o mercado formal quanto o informal, causando perda de empregos.<br>**Insegurança Econômica:** A natureza instável do trabalho informal pode resultar em insegurança financeira, sem garantia de renda estável, benefícios ou proteção contra demissões. |

## 5.3. PERGUNTAS DE NEGÓCIO

**Tendências de Emprego por Setor:** Qual é a distribuição dos empregos por setor na economia brasileira ao longo dos anos? Quais setores estão crescendo ou diminuindo em termos de emprego?

**Taxas de Desemprego:** Como as taxas de desemprego variam ao longo do tempo? Existem padrões sazonais ou tendências de longo prazo?

**Perfil do Emprego:** Idade, gênero, raça, deficiência e nível de educação têm impacto nas oportunidades de emprego?

**Renda por Setor:** Existem disparidades significativas de renda entre os setores?

**Diversidade Étnica:** Como a população brasileira se autodeclara em termos étnicos? Quais são as proporções de diferentes grupos étnicos?

**Níveis de Educação:** Qual é a distribuição dos níveis de educação na população?

**Proporção de Trabalho Formal e Informal:** Qual é a proporção de empregos formais em comparação com empregos informais ao longo do tempo? Essa proporção varia entre diferentes setores da economia?

**Educação e Formalidade:** Existe uma correlação entre o nível de educação e a probabilidade de um emprego ser formal? Os trabalhadores com maior educação têm mais chances de encontrar empregos formais?

**Evolução da Formalidade:** Existem tendências ao longo do tempo que mostram mudanças na formalidade dos empregos? A formalidade está aumentando ou diminuindo?

**Gênero e Formalidade:** Existe uma diferença de gênero na escolha entre empregos formais e informais? As mulheres são mais propensas a trabalhar em empregos informais?

**Taxa de Desemprego durante a Pandemia:** Como a taxa de desemprego variou durante a pandemia?

**Informalidade:** Com a pandemia, a informalidade aumentou ou diminuiu?

## 5.4. POWER BI

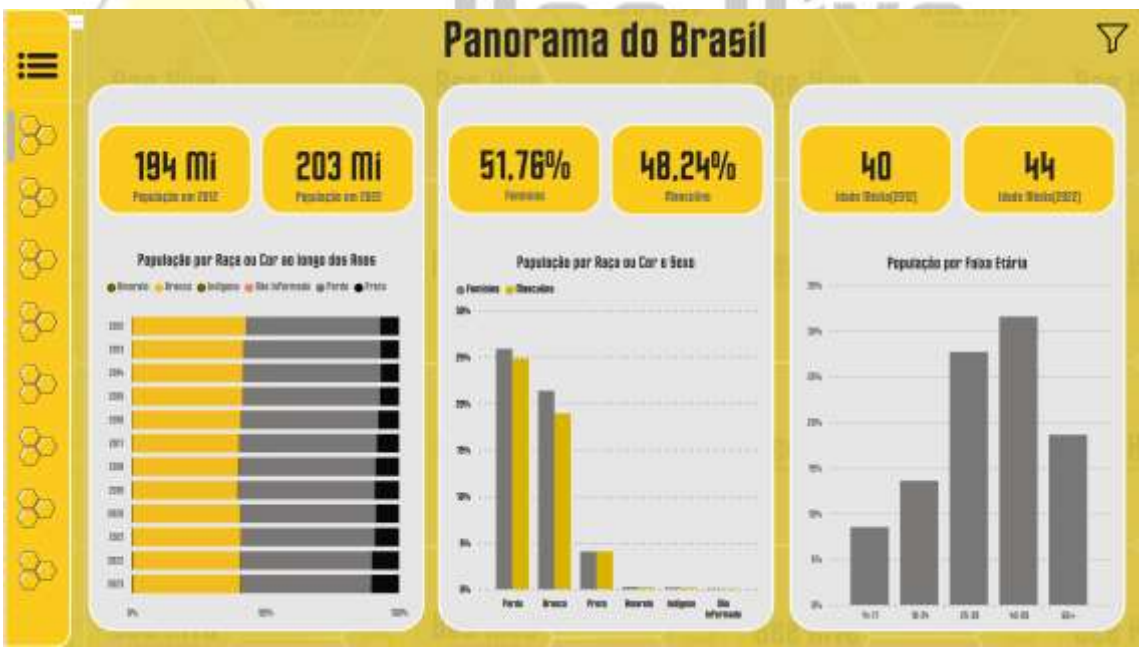
Um dashboard é uma ferramenta que ajuda as empresas a acompanharem o desempenho de suas operações e tomar decisões baseadas em dados. O Power BI é um software que permite às empresas criarem dashboards personalizados para atender às suas necessidades específicas.

Ele oferece uma variedade de recursos que tornam possível criar dashboards atraentes e informativos. Os usuários podem conectar o Power BI a uma variedade de fontes de dados, incluindo bancos de dados, planilhas e arquivos CSV. Também oferece uma variedade de visualizações, incluindo gráficos, tabelas e mapas.

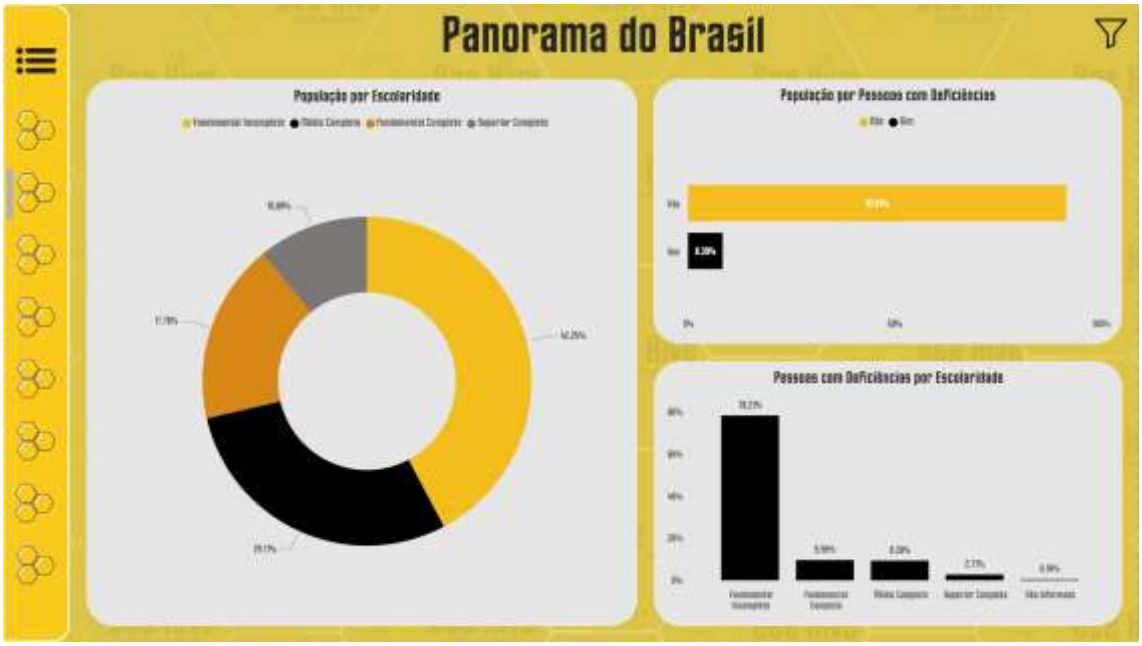
Os dashboards dentro dele podem ser usados para visualizar uma variedade de dados, incluindo vendas, marketing, finanças e operações. Podem ser usados para identificar tendências, padrões e oportunidades, além de serem usados para acompanhar o progresso em relação a metas e objetivos.

Abaixo estão os modelos criados por nossa equipe:

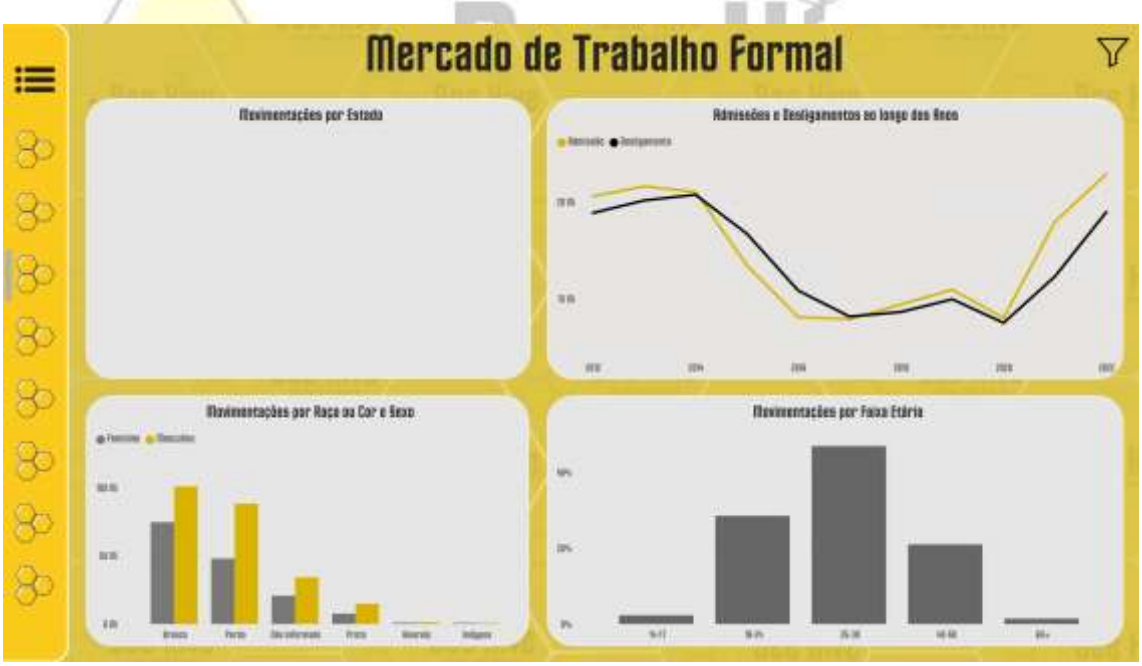
Dashboard Panorama do Brasil 01



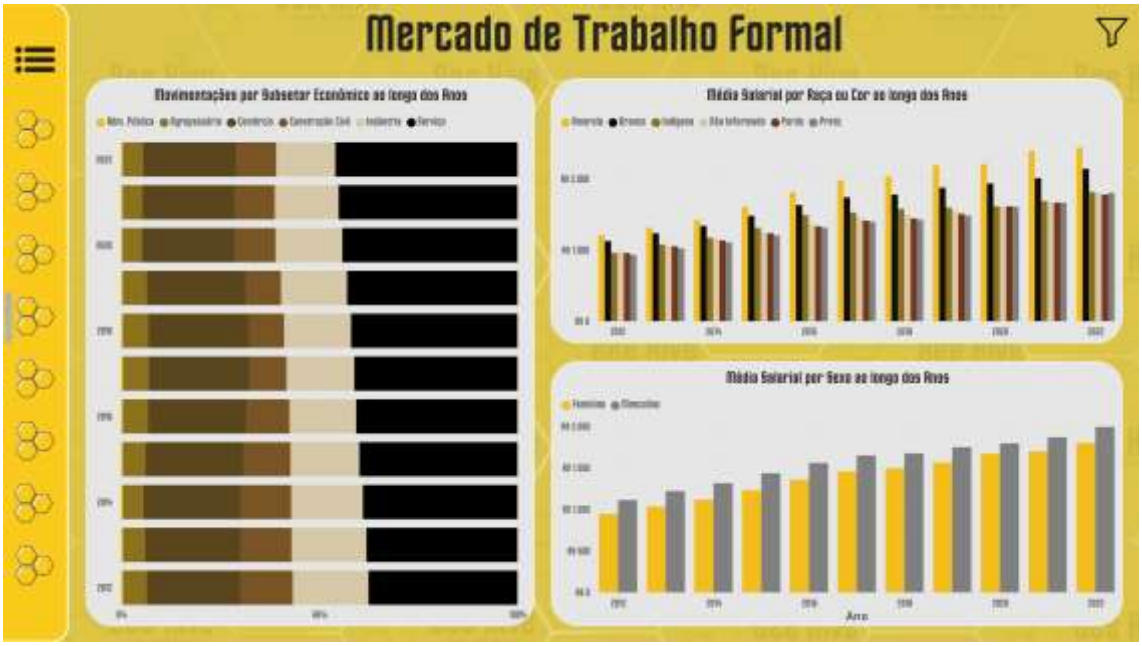
Dashboard Panorama do Brasil 02



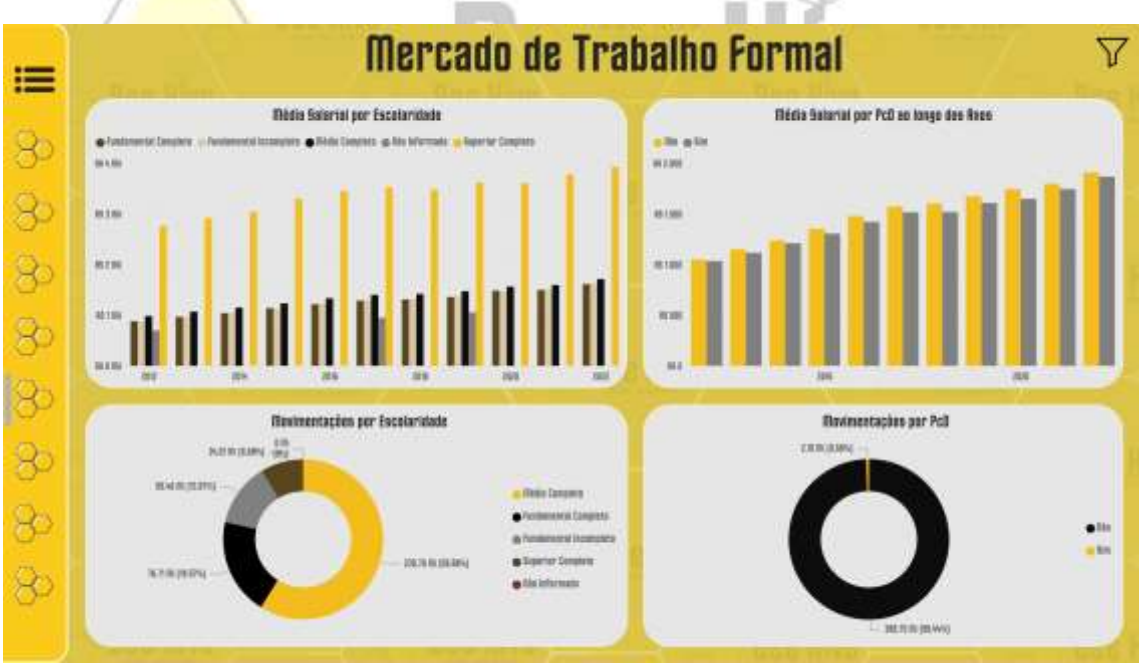

Dashboard Mercado de Trabalho Formal 01



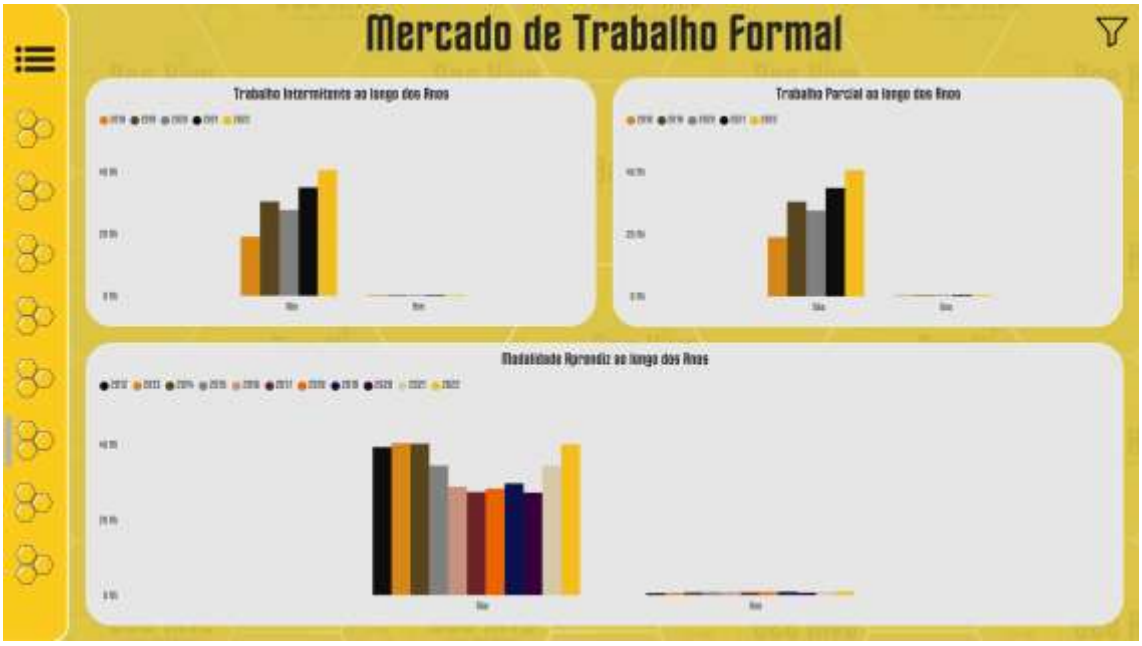
Dashboard Mercado de Trabalho Formal 02



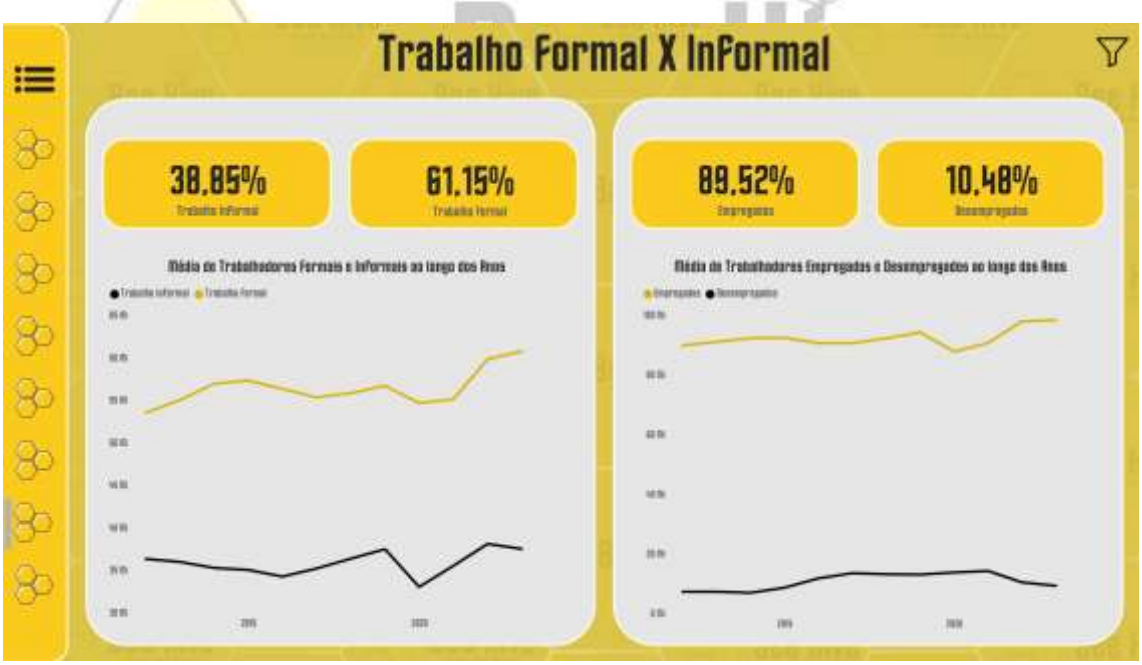

Dashboard Mercado de Trabalho Formal 03



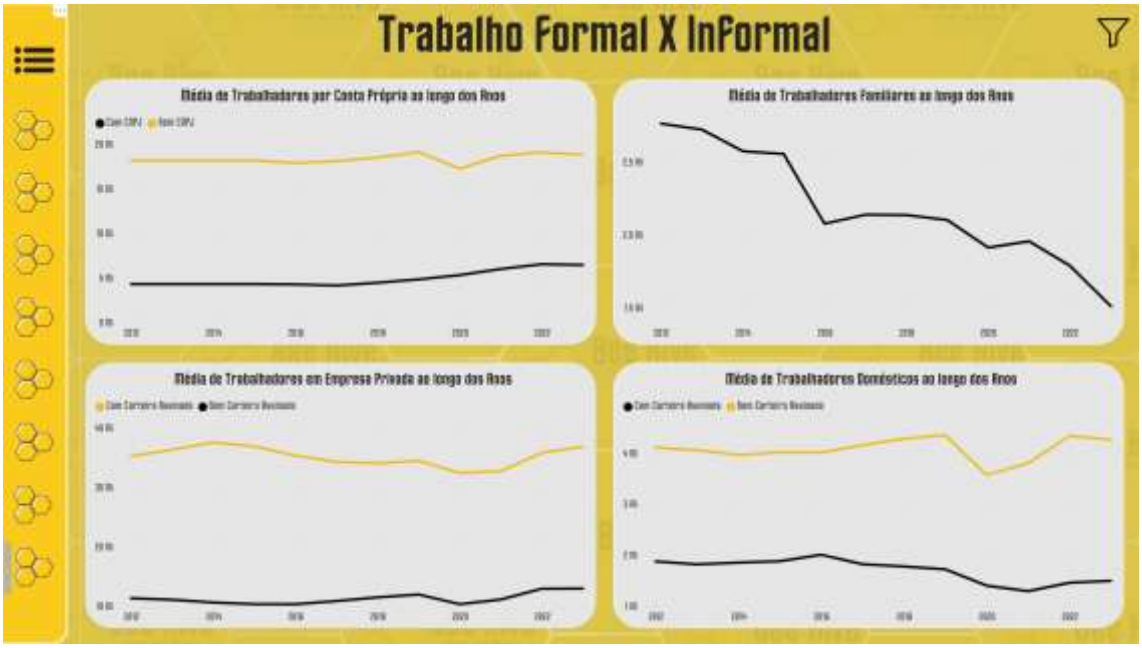
Dashboard Mercado de Trabalho Formal 04




Dashboard Mercado de Trabalho Formal vs. Informal 01



Dashboard Mercado de Trabalho Formal vs. Informal 01




## 6. ESTRUTURA DO CÓDIGO ETL

Segue a estrutura do ETL feito no SQL e COLAB:

- SQL para CAGED:

  Tratamento

  - Pré-análise e primeiros Insights

  - Seleção dos atributos a serem utilizados na base

  - Verificação de nulos e inconsistências dentro dos atributos selecionados

  - Tradução

  - Conversão de tipos

  - Validação

  Extração

  Carregamento

- SQL para PNAD-C:

  Tratamento

  - Pré-análise e primeiros Insights

  - Seleção dos atributos a serem utilizados na base

  - Verificação de nulos e inconsistências dentro dos atributos selecionados

  - Tradução

  - Conversão de tipos

  - Validação

  Extração

  Carregamento

- SQL para Censo 2010:

  Tratamento

  - Pré-análise e primeiros Insights

  - Seleção dos atributos a serem utilizados na base

  - Verificação de nulos e inconsistências dentro dos atributos selecionados

  - Tradução

  - Conversão de tipos

  - Validação

  Extração

  Carregamento

- COLAB IPEA:

  Extração

  Transformação

  - Pré-análise

  - Seleção das colunas a serem analisadas

  - Tradução

  - Verificação

  - Verificação do schema via PySpark

  - Carregamento

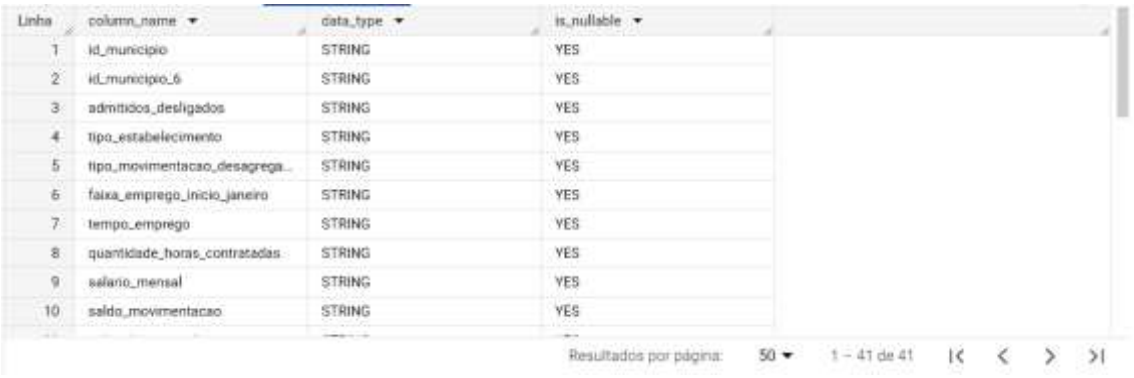
## 7. CÓDIGO ETL

- CAGED, PNAD-C e Censo 2012

CAGED 2012 - 2019

Primeiramente será feita uma consulta para verificar quais são os atributos da tabela e verificar quais são os seus tipos.

```
SELECT
  column_name,
  data_type,
  is_nullable
FROM
                                                `basedosdados-staging.br_me_caged_staging.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'microdados_antigos';
```



| Linha | column_name | data_type | is_nullable |
|---|---|---|---|
| 1 | id_municipio | STRING | YES |
| 2 | id_municipio_6 | STRING | YES |
| 3 | admitidos_desligados | STRING | YES |
| 4 | tipo_estabelecimento | STRING | YES |
| 5 | tipo_movimentacao_desagrega... | STRING | YES |
| 6 | faixa_emprego_inicio_janeiro | STRING | YES |
| 7 | tempo_emprego | STRING | YES |
| 8 | quantidade_horas_contratadas | STRING | YES |
| 9 | salario_mensal | STRING | YES |
| 10 | saldo_movimentacao | STRING | YES |

Resultados por página: 50 1 – 41 de 41

É possível ver que todas as colunas da tabela são do tipo string.

Para a análise foram escolhidos os seguintes atributos:

ano, sigla_uf, admitidos_desligados, salario_mensal, indicador_aprendiz, indicador_trabalho_intermitente, indicador_trabalho_parcial, indicador_portador_deficiencia, grau_instrucao, idade, sexo, raca_cor, subsetor_ibge.

A consulta a seguir irá verificar se o atributo “ano” possui alguma inconsistência como valores nulos, anos que não estão no formato correto (por exemplo, "212" em vez de "2012" ou "abc" em vez de um valor numérico), anos em formatos diferentes (como "2019" e "19" para o mesmo ano) ou anos que não estejam na cobertura temporal informada que seria de 2012 a 2019).

```
SELECT
  ano AS Ano,
  COUNT(*) AS QTD
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY Ano
ORDER BY Ano ASC;
```

| Linha | Ano | QTD |
|---|---|---|
| 1 | 2007 | 27065186 |
| 2 | 2008 | 31866459 |
| 3 | 2009 | 31380169 |
| 4 | 2010 | 36272736 |
| 5 | 2011 | 39559197 |
| 6 | 2012 | 39995837 |
| 7 | 2013 | 41153415 |
| 8 | 2014 | 41169404 |
| 9 | 2015 | 35348975 |
| 10 | 2016 | 29715447 |

Nenhuma inconsistência foi verificada na coluna “ano”.

A consulta feita para o atributo “sigla_uf” tem por objetivo verificar inconsistências como verificar valores duplicados, ou seja, verificar se existem estados com a mesma sigla (mesmo estado escrito de forma diferente ou com diferentes grafias), verificar valores em branco, contabilizando todos os valores nulos que a coluna possui.

```
SELECT
  sigla_uf,
  COUNT(*) AS QTD_SIGLA_UF
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY sigla_uf
ORDER BY sigla_uf ASC;
```

| Linha | sigla_uf | QTD_SIGLA_UF |
|---|---|---|
| 1 | AC | 646943 |
| 2 | AL | 3245834 |
| 3 | AM | 4331791 |
| 4 | AP | 609698 |
| 5 | BA | 16830253 |
| 6 | CE | 10601783 |
| 7 | DF | 7508217 |
| 8 | ES | 9733312 |
| 9 | GO | 15531580 |
| 10 | MA | 4150219 |



Nenhuma inconsistência foi verificada na coluna “sigla_uf”.

O atributo “admitidos_desligados” contempla as formas de admissão e desligamentos registradas. A consulta irá verificar as inconsistências como valores nulos e valores de código diferentes daqueles que estão descritos no dicionário fornecido pela fonte de dados.

```
SELECT
```

```
  admitidos_desligados,
  COUNT(*) AS QTD_ADMITIDOS_DESLIGADOS
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY admitidos_desligados
ORDER BY admitidos_desligados ASC;
```

| Linha | admitidos_desligados | QTD_ADMITIDOS_DE |
|---|---|---|
| 1 | 01 | 224709432 |
| 2 | 02 | 217329740 |

Nenhuma inconsistência foi verificada na coluna “admitidos_desligados”.

O atributo “salario_mensal” mostra os valores de salário associados a cada registro. A consulta irá verificar as inconsistências como valores nulos e valores que não sejam numéricos.

```
SELECT
  salario_mensal,
  COUNT(*) AS QTD_SALARIO_MENSAL
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY salario_mensal
ORDER BY salario_mensal ASC;
```

| Linha | salario_mensal | QTD_SALARIO_MENSAL |
|---|---|---|
| 1 | 0.0 | 3130310 |
| 2 | 1000.0 | 4565575 |
| 3 | 10000.0 | 62898 |
| 4 | 100000.0 | 284 |
| 5 | 100001.0 | 1 |
| 6 | 100002.0 | 1 |
| 7 | 10001.0 | 307 |
| 8 | 100011.0 | 1 |
| 9 | 100012.0 | 1 |
| 10 | 100014.0 | 1 |

Resultados por página: 50 1 – 50 de 69665

Nenhuma inconsistência foi verificada na coluna “salario_mensal”, porém precisará trocar o seu tipo para INT64 em etapas futuras.

O atributo “indicador_aprendiz” mostra se o registro é referente a um aprendiz ou não. Os possíveis valores são “0” ou “1” e qual outro valor será considera uma inconsistência.

```
SELECT
```

```
  indicador_aprendiz,
  COUNT(*) AS QTD_INDICADOR_APRENDIZ
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY indicador_aprendiz
ORDER BY indicador_aprendiz ASC;
```

| Linha | indicador_aprendiz | QTD_INDICADOR_APRENDIZ |
|---|---|---|
| 1 | 0 | 434662466 |
| 2 | 1 | 7376706 |

Nenhuma inconsistência foi verificada na coluna “indicador_aprendiz”.

O atributo “indicador_trabalho_intermitente” mostra se o registro é referente a modalidade de trabalho intermitente. Os possíveis valores são “0” ou “1”.

```
SELECT
  indicador_trabalho_intermitente,
  COUNT(*) AS QTD_INDICADOR_TRABALHO_INTERMITENTE
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY indicador_trabalho_intermitente
ORDER BY indicador_trabalho_intermitente ASC;
```

| Linha | indicador_trabalho_intermitente | QTD_INDICADOR_TR |
|---|---|---|
| 1 | null | 391918615 |
| 2 | 0 | 49836136 |
| 3 | 1 | 284421 |

A acentuada quantidade de valores nulos se deve ao contexto dos dados. A modalidade de trabalho intermitente só foi aplicada a partir do ano de 2018.

O atributo “indicador_trabalho_parcial” mostra se o registro é referente a modalidade de trabalho parcial. Os possíveis valores são “0” ou “1”.

```
SELECT
  indicador_trabalho_parcial,
  COUNT(*) AS QTD_INDICADOR_TRABALHO_PARCIAL
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY indicador_trabalho_parcial
```

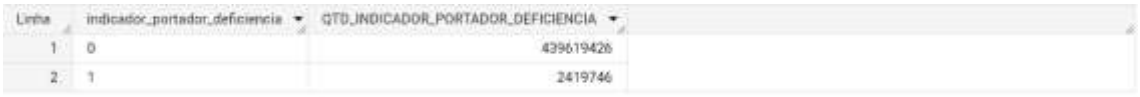
```
ORDER BY indicador_trabalho_parcial ASC;
```

| Linha | indicador_trabalho_parcial | QTD_INDICADOR_TR |
|---|---|---|
| 1 | null | 391918615 |
| 2 | 0 | 49902060 |
| 3 | 1 | 218497 |

Da mesma forma do atributo anterior, a acentuada quantidade de valores nulos se deve ao contexto dos dados. Inclusive essas quantidades são iguais nos dois atributos. A modalidade de trabalho parcial só foi aplicada a partir do ano de 2018.

O atributo “indicador_portador_deficiencia” mostra se o registro é referente a um portador de deficiência ou não. Os possíveis valores são “0” ou “1” e qual outro valor será considera uma inconsistência.

```
SELECT
  indicador_portador_deficiencia,
  COUNT(*) AS QTD_INDICADOR_PORTADOR_DEFICIENCIA
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY indicador_portador_deficiencia
ORDER BY indicador_portador_deficiencia ASC;
```



| Linha | indicador_portador_deficiencia | QTD_INDICADOR_PORTADOR_DEFICIENCIA |
|---|---|---|
| 1 | 0 | 439619426 |
| 2 | 1 | 2419746 |

Nenhuma inconsistência foi verificada na coluna “indicador_portador_deficiencia”.

A coluna “grau_instrucao” tem como possíveis valores “01”, “02”, “03”, “04”, “05”, “06”, “07”, “08” ou “09” conforme as informações trazidas pelo dicionário de dados disponibilizado pela fonte de dados. Quaisquer valores diferentes desses serão considerados como inconsistências.

```
SELECT
  grau_instrucao,
  COUNT(*) AS QTD_GRAU_INSTRUCAO
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
```

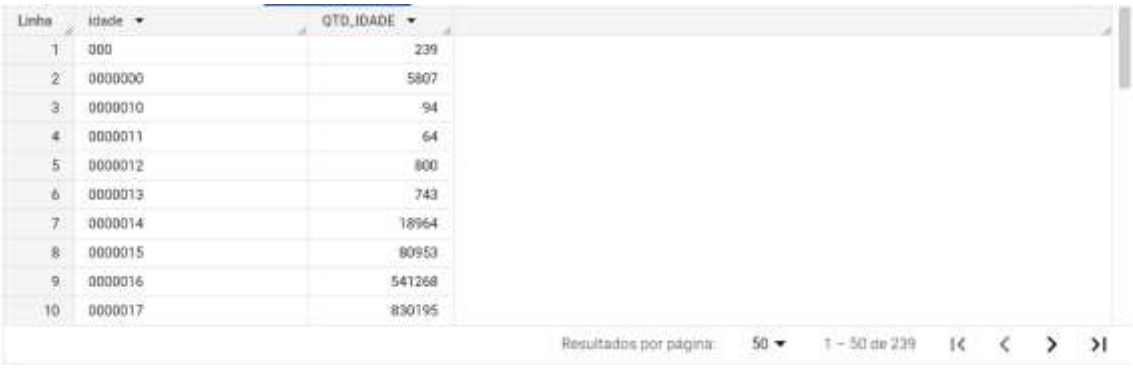
```
GROUP BY grau_instrucao
ORDER BY grau_instrucao ASC;
```

| Linha | grau_instrucao | QTD_GRAU_INSTRUCAO |
|---|---|---|
| 1 | null | 10 |
| 2 | 01 | 2777220 |
| 3 | 02 | 18539440 |
| 4 | 03 | 18459325 |
| 5 | 04 | 38499243 |
| 6 | 05 | 57508845 |
| 7 | 06 | 44881048 |
| 8 | 07 | 212956209 |
| 9 | 08 | 15997836 |
| 10 | 09 | 32420026 |

As inconsistências encontradas foram apenas em relação a valores nulos que serão classificados como “Não Informado” em etapas futuras.

O atributo “idade” mostra os valores das idades das pessoas associadas a cada registro. A consulta irá verificar as inconsistências como valores nulos e valores que não sejam numéricos.

```
SELECT
  idade,
  COUNT(*) AS QTD_IDADE
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY idade
ORDER BY idade ASC;
```



| Linha | idade | QTD_IDADE |
|---|---|---|
| 1 | 000 | 239 |
| 2 | 0000000 | 5807 |
| 3 | 0000010 | 94 |
| 4 | 0000011 | 64 |
| 5 | 0000012 | 800 |
| 6 | 0000013 | 743 |
| 7 | 0000014 | 18964 |
| 8 | 0000015 | 80953 |
| 9 | 0000016 | 541268 |
| 10 | 0000017 | 830195 |

Resultados por página: 50 1 – 50 de 239

As inconsistências encontradas foram apenas em relação a valores com zeros a esquerda, pois a coluna “idade” ela é do tipo string. Para corrigir, a etapa de

tratamento irá transformar a coluna para o tipo INT64 e, com isso, consertará essas inconsistências.

O atributo “sexo” mostra qual o gênero da pessoa referente àquele registro. Os possíveis valores são “0” ou “1” e qual outro valor será considera uma inconsistência.

```
SELECT
  sexo,
  COUNT(*) AS QTD_SEXO
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY sexo
ORDER BY sexo ASC;
```

| Linha | sexo | QTD_SEXO |
|---|---|---|
| 1 | 01 | 278888600 |
| 2 | 02 | 163150572 |

Nenhuma inconsistência foi verificada na coluna “sexo”.

O atributo “raca_cor” tem como possíveis valores “01”, “02”, “03”, “04”, “05”, “06”, “07”, “08”, “09” ou “99” conforme as informações trazidas pelo dicionário de dados disponibilizado pela fonte de dados. Quaisquer valores diferentes desses serão considerados como inconsistências.

```
SELECT
  raca_cor,
  COUNT(*) AS QTD_RACA_COR
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY raca_cor
ORDER BY raca_cor ASC;
```

| Linha | raca_cor | QTD_RACA_COR |
|---|---|---|
| 1 | null | 100630 |
| 2 | 01 | 1070926 |
| 3 | 02 | 229561433 |
| 4 | 04 | 25122255 |
| 5 | 06 | 2668463 |
| 6 | 08 | 148191553 |
| 7 | 09 | 35320777 |
| 8 | 99 | 3135 |

O último atributo a ser investigado é “subsetor_ibge” tem como possíveis valores numéricos entre “01” e “25” conforme as informações trazidas pelo dicionário de dados disponibilizado pela fonte de dados onde são caracterizados pelo seu setor econômico do registro em questão. Quaisquer valores diferentes desses serão considerados como inconsistências.

```
SELECT
  subsetor_ibge,
  COUNT(*) AS QTD_SUBSETOR_IBGE
FROM `basedosdados-staging.br_me_caged_staging.microdados_antigos`
GROUP BY subsetor_ibge
ORDER BY subsetor_ibge ASC;
```



| Linha | subsetor_ibge | QTD_SUBSETOR_IBG |
|---|---|---|
| 1 | null | 1 |
| 2 | 1 | 996508 |
| 3 | 2 | 3823341 |
| 4 | 3 | 6009240 |
| 5 | 4 | 5235557 |
| 6 | 5 | 2157896 |
| 7 | 6 | 2825214 |
| 8 | 7 | 4386689 |
| 9 | 8 | 2682325 |
| 10 | 9 | 3077818 |

Resultados por página: 50 1 – 35 de 35

Algumas inconsistências foram encontradas. Os valores nulos que serão classificados como “Não Informado” em etapas futuras. Alguns valores distintos estão representando o mesmo valor (por exemplo “ 1” e “1”) e será feito um tratamento em etapas futuras para retirar o “espaço” dessa sobressalente desses valores.

O código apresentado a seguir é uma consulta que manipula dados da tabela no dataset da fonte de dados. A consulta possui várias etapas, que envolvem o

tratamento e seleção dos dados de acordo com determinados critérios. Vamos analisar cada parte do código:

1. A primeira parte utiliza a cláusula **WITH** para criar uma tabela temporária chamada "Tabela" com os seguintes campos:
    - Ano (ano de referência dos dados, convertido para INT64)
    - Estado (sigla da unidade federativa, convertido para STRING)
    - Movimentacao (tipo de movimentação desagregada, convertido para STRING)
    - Salario_Mensal (salário mensal, convertido para FLOAT64)
    - Aprendiz (indicador de aprendiz, convertido para STRING)
    - Trabalho_intermitente (indicador de trabalho intermitente, convertido para STRING)
    - Trabalho_Parcial (indicador de trabalho parcial, convertido para STRING)
    - PcD (indicador de portador de deficiência, convertido para STRING)
    - Escolaridade (grau de instrução, convertido para STRING)
    - Idade (idade do trabalhador, convertido para INT64)
    - Sexo (sexo do trabalhador, convertido para STRING)
    - Raca_Cor (raça/cor do trabalhador, convertido para STRING)
    - Subsetor (subsetor IBGE, convertido para INT64)
2. Em seguida, a tabela "basedosdados-staging.br_me_caged_staging.microdados_antigos" é consultada e os dados são filtrados e transformados em tipos apropriados para cada campo, conforme descrito na primeira parte.

3. O campo chamado "Escolaridade" é criado com base nos valores do campo "grau_instrucao". Esse campo é uma transformação dos valores de grau de instrução dos trabalhadores e segue as seguintes regras:
    - Se o valor de "grau_instrucao" for '01', '02', '03' ou '04', o valor de "Escolaridade" será '1' que representará o ensino fundamental incompleto.
    - Se o valor de "grau_instrucao" for '05' ou '06', o valor de "Escolaridade" será '2' que representará o ensino fundamental completo.

- Se o valor de "grau_instrucao" for '07' ou '08', o valor de "Escolaridade" será '3' que representará o ensino médio completo.
- Se o valor de "grau_instrucao" for '09', '10' ou '11', o valor de "Escolaridade" será '4' que representará o ensino superior completo.
- Para qualquer outro valor de "grau_instrucao", o valor de "Escolaridade" será '99' que será interpretado como não informado.

4. O atributo "subsetor" é criado fazendo o tratamento onde serão tirados todos os “espaços” dos valores da coluna.
5. A cláusula **WHERE** é utilizada para filtrar os dados da tabela "Tabela". Os registros são selecionados com base nos seguintes critérios:
    - O campo "Ano" deve estar em uma das opções '2019', '2018', '2017', '2016', '2015', '2014', '2013' ou '2012', pois será feita a análise a partir do ano de 2012.
    - O campo "Idade" não pode ser nenhum dos seguintes valores: '000', '0000000', '0000010', '0000011', '0000012', '0000013', '010', '011', '012', '013', '11', '13', pois a análise será feita a partir dos 14 anos de idade.
6. Por fim, a consulta principal é feita, selecionando os seguintes campos da tabela "Tabela":
    - Ano, Estado, Movimentacao, Salario_Mensal, Aprendiz, Trabalho_Intermitente, Trabalho_Parcial, PcD, Escolaridade, Idade, Sexo, Raca_Cor.
    - O campo "Subsetor_Economico" é criado com base nos valores do campo "subsetor". Ele recebe valores de acordo com algumas condições onde serão agrupados os setores econômicos:
        - Se o valor de "subsetor" for menor ou igual a 13, o valor de "Subsetor_Economico" será '01'.
        - Se o valor de "subsetor" for igual a 14 ou estiver entre 18 e 23, o valor de "Subsetor_Economico" será '02'.
        - Se o valor de "subsetor" for igual a 15, o valor de "Subsetor_Economico" será '03'.
        - Se o valor de "subsetor" for igual a 16 ou igual a 17, o valor de "Subsetor_Economico" será '04'.

- Se o valor de "subsetor" for igual a 24, o valor de "Subsetor_Economico" será '05'.
- Se o valor de "subsetor" for igual a 25, o valor de "Subsetor_Economico" será '06'.
- Para quaisquer outros valores de "subsetor", o valor de "Subsetor_Economico" será '99'.

7. Um último filtro é aplicado na consulta final usando a cláusula **WHERE**. Os registros são selecionados com base nos seguintes critérios:
   - O campo "salario_mensal" deve estar entre 243 e 121500 (inclusive). Foram considerados 0,3 a 150 salários conforme a recomendação dos estudos do IPEA.
   - O campo "idade" deve ser maior ou igual a 14, pois a idade mínima para o trabalho formal é 14 anos como menor aprendiz.

A consulta final resulta em uma tabela com os campos mencionados, contendo dados filtrados e transformados de acordo com os critérios estabelecidos.

```
WITH Tabela AS (
  SELECT
SAFE_CAST(ano AS INT64) Ano,
SAFE_CAST(sigla_uf AS STRING) Estado,
SAFE_CAST(admitidos_desligados AS STRING) Movimentacao,

SAFE_CAST(salario_mensal AS FLOAT64) Salario_Mensal,
SAFE_CAST(indicador_aprendiz AS STRING) Aprendiz,
CASE
   WHEN SAFE_CAST(indicador_trabalho_intermitente AS STRING) = '1' THEN
'1'
   WHEN SAFE_CAST(indicador_trabalho_intermitente AS STRING) = '0' THEN
'0'
   ELSE '3'
END AS Trabalho_intermitente,
CASE
   WHEN SAFE_CAST(indicador_trabalho_parcial AS STRING) = '1' THEN '1'
   WHEN SAFE_CAST(indicador_trabalho_parcial AS STRING) = '0' THEN '0'
```

```
    ELSE '3'
END AS Trabalho_Parcial,
SAFE_CAST(indicador_portador_deficiencia AS STRING) PcD,
CASE
        WHEN SAFE_CAST(grau_instrucao AS STRING) = '01' OR
SAFE_CAST(grau_instrucao AS STRING) = '02'OR
SAFE_CAST(grau_instrucao AS STRING)= '03' OR
SAFE_CAST(grau_instrucao AS STRING) = '04' THEN '1'
        WHEN SAFE_CAST(grau_instrucao AS STRING) = '05' OR
SAFE_CAST(grau_instrucao AS STRING) = '06' THEN '2'
        WHEN SAFE_CAST(grau_instrucao AS STRING) = '07' OR
SAFE_CAST(grau_instrucao AS STRING) = '08' THEN '3'
        WHEN SAFE_CAST(grau_instrucao AS STRING) = '09' OR
SAFE_CAST(grau_instrucao AS STRING) = '10'OR
SAFE_CAST(grau_instrucao AS STRING) = '11' THEN '4'
    ELSE '99'
END AS Escolaridade,
SAFE_CAST(idade AS INT64) Idade,
SAFE_CAST(sexo AS STRING) Sexo,
CASE
    WHEN SAFE_CAST(raca_cor AS STRING) = '01' THEN '01'
    WHEN SAFE_CAST(raca_cor AS STRING) = '02' THEN '02'
    WHEN SAFE_CAST(raca_cor AS STRING) = '04' THEN '04'
    WHEN SAFE_CAST(raca_cor AS STRING) = '06' THEN '06'
    WHEN SAFE_CAST(raca_cor AS STRING) = '08' THEN '08'
    ELSE '99'
END AS Raca_Cor,
SAFE_CAST(REPLACE(subsetor_ibge, ' ', '') AS INT64) AS subsetor

from `basedosdados-staging.br_me_caged_staging.microdados_antigos`

WHERE Ano IN ('2019', '2018', '2017', '2016', '2015', '2014', '2013', '2012')
    AND idade NOT IN ('000', '0000000', '0000010', '0000011', '0000012',
'0000013', '010', '011', '012', '013', '11', '13')
```

```
)

SELECT Ano, Estado, Movimentacao, Salario_Mensal, Aprendiz, Trabalho_Intermitente, Trabalho_Parcial, PcD,
    Escolaridade, Idade, Sexo, Raca_Cor,
    CASE
      WHEN subsetor <= 13 THEN '01'
      WHEN subsetor = 14 OR subsetor >= 18 OR subsetor <= 23 THEN '02'
      WHEN subsetor = 15 THEN '03'
      WHEN subsetor = 16 OR subsetor = 17 THEN '04'
      WHEN subsetor = 24 THEN '05'
      WHEN subsetor = 25 THEN '06'
      ELSE '99'
    END AS Subsetor_Economico

FROM Tabela

WHERE (salario_mensal <= 121500 AND salario_mensal >= 243 AND idade >= 14);
```

| Linha | Ano | Estado | Movimentacao | Salario_Mensal | Aprendiz | Trabalho_Intermitente | Trabalho_Parcial | PcD | Escolaridade | Idade | Sexo | Raca_C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2012 | SC | 02 | 710.0 | 0 | 3 | 3 | 0 | 3 | 20 | 02 | 99 |
| 2 | 2012 | SC | 04 | 1772.0 | 0 | 3 | 3 | 0 | 4 | 30 | 01 | 02 |
| 3 | 2012 | PR | 06 | 759.0 | 0 | 3 | 3 | 0 | 3 | 32 | 02 | 02 |
| 4 | 2012 | PR | 02 | 622.0 | 0 | 3 | 3 | 0 | 1 | 34 | 01 | 02 |
| 5 | 2012 | PR | 08 | 1001.0 | 0 | 3 | 3 | 0 | 3 | 25 | 01 | 08 |
| 6 | 2012 | DF | 06 | 1319.0 | 0 | 3 | 3 | 0 | 3 | 41 | 01 | 99 |
| 7 | 2012 | BA | 11 | 1300.0 | 0 | 3 | 3 | 0 | 3 | 22 | 02 | 08 |
| 8 | 2012 | CE | 02 | 800.0 | 0 | 3 | 3 | 0 | 3 | 24 | 01 | 08 |
| 9 | 2012 | CE | 02 | 698.0 | 0 | 3 | 3 | 0 | 3 | 46 | 01 | 08 |
| 10 | 2012 | MT | 02 | 1041.0 | 0 | 3 | 3 | 0 | 1 | 44 | 01 | 08 |

Resultados por página: 50 1 – 50 de 273666042

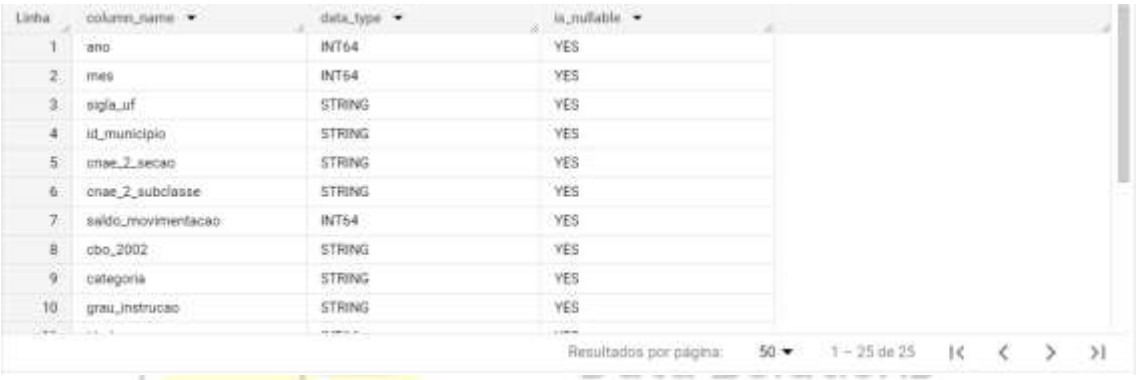
Essa consulta será utilizada em para fazer o download em um arquivo CSV através de uma função em python.

CAGED 2020 – 2023

A diferença desta tabela para aquela que foi tratada anteriormente está em

alguns parâmetros que foram modificados pelo CAGED. Muitas das estratégias adotadas anteriormente serão idênticas a desta.

```
SELECT
  column_name,
  data_type,
  is_nullable
FROM
  `basedosdados.br_me_caged.INFORMATION_SCHEMA.COLUMNS`
WHERE
  table_name = 'microdados_movimentacao';
```



| Linha | column_name | data_type | is_nullable |
|---|---|---|---|
| 1 | ano | INT64 | YES |
| 2 | mes | INT64 | YES |
| 3 | sigla_uf | STRING | YES |
| 4 | id_municipio | STRING | YES |
| 5 | cnae_2_secao | STRING | YES |
| 6 | cnae_2_subclasse | STRING | YES |
| 7 | saldo_movimentacao | INT64 | YES |
| 8 | cbo_2002 | STRING | YES |
| 9 | categoria | STRING | YES |
| 10 | grau_instrucao | STRING | YES |

Resultados por página: 50 — 1 – 25 de 25

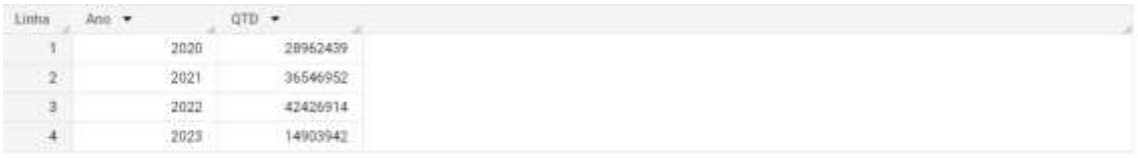
Para a análise foram escolhidos os seguintes atributos:

ano, sigla_uf, tipo_movimentacao, salario_mensal, indicador_aprendiz, indicador_trabalho_intermitente, indicador_trabalho_parcial, tipo_deficiencia, grau_instrucao, idade, sexo, raca_cor, cnae_2_secao.

A consulta a seguir irá verificar se o atributo “ano” possui alguma inconsistência como valores nulos, anos que não estão no formato correto (por exemplo, "202" em vez de "2020" ou "abc" em vez de um valor numérico), anos em formatos diferentes (como "2020" e "20" para o mesmo ano) ou anos que não estejam na cobertura temporal informada que seria de 2020 a 2023).

```
SELECT
  ano AS Ano,
  COUNT(*) AS QTD
FROM ` basedosdados.br_me_caged.microdados_movimentacao`
```

```
GROUP BY Ano
ORDER BY Ano ASC;
```



| Linha | Ano | QTD |
|---|---|---|
| 1 | 2020 | 28962439 |
| 2 | 2021 | 36546952 |
| 3 | 2022 | 42426914 |
| 4 | 2023 | 14903942 |

A consulta abaixo irá verificar se o atributo “sigla_uf” possui alguma inconsistência como por exemplo(“ ”) e seleciona a sigla do estado (sigla_uf) e a contagem de registros (QTD) da tabela. Em seguida, agrupa os resultados por sigla do estado e ordena os resultados pela sigla do estado em ordem ascendente (ASC), ou seja, conta o número de registros em cada estado e exibe os resultados em ordem alfabética.

```
SELECT
  sigla_uf,
  COUNT(*) AS QTD
FROM ` basedosdados.br_me_caged.microdados_movimentacao`

GROUP BY sigla_uf
ORDER BY sigla_uf ASC;
```

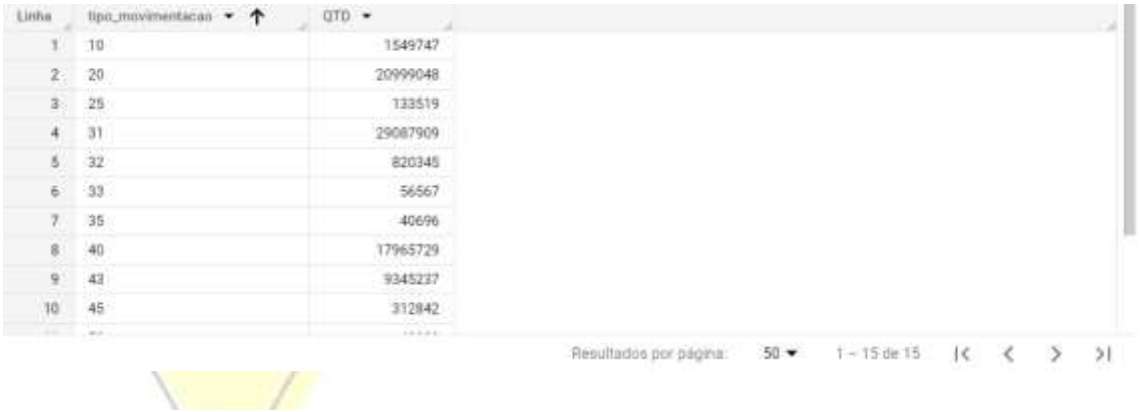
| Linha | sigla_uf | QTD |
|---|---|---|
| 1 | AC | 235072 |
| 2 | AL | 896412 |
| 3 | AM | 1208979 |
| 4 | AP | 183370 |
| 5 | BA | 4464029 |
| 6 | CE | 2866469 |
| 7 | DF | 2103550 |
| 8 | ES | 2504141 |
| 9 | GO | 4502066 |
| 10 | MA | 1244391 |

Resultados por página: 50 1 – 27 de 27

Esta consulta verifica se o atributo “tipo_movimentacao” possui alguma inconsistência como valores nulos, valores numéricos em formatos incorretos e seleciona o tipo de movimentação e a contagem de registros (QTD) da tabela,

agrupa os resultados por tipo de movimentação e ordena os resultados pelo tipo de movimentação em ordem ascendente (ASC). Em outras palavras, a consulta conta o número de registros de cada tipo de movimentação e exibe os resultados em ordem alfabética.

```
SELECT
  tipo_movimentacao,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY tipo_movimentacao
ORDER BY tipo_movimentacao ASC;
```



| Linha | tipo_movimentacao | QTD |
|---|---|---|
| 1 | 10 | 1549747 |
| 2 | 20 | 20999048 |
| 3 | 25 | 133519 |
| 4 | 31 | 29087909 |
| 5 | 32 | 820345 |
| 6 | 33 | 56567 |
| 7 | 35 | 40696 |
| 8 | 40 | 17965729 |
| 9 | 43 | 9345237 |
| 10 | 45 | 312842 |

Resultados por página: 50 1 – 15 de 15

A consulta verifica se o atributo “salario_mensal” possui alguma inconsistência como valores nulos, valores numéricos em formatos incorretos e seleciona a coluna salario_mensal e a contagem de registros (QTD) da tabela em seguida, agrupa os resultados por salário mensal e ordena os resultados pelo salário mensal em ordem ascendente (ASC). Em outras palavras, a consulta conta o número de registros em cada faixa de salário mensal e exibe os resultados em ordem crescente.

```
SELECT
  salario_mensal,
  COUNT(*) AS QTD
```

```
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY salario_mensal
ORDER BY salario_mensal ASC;
```

| Linha | salario_mensal | QTD |
|---|---|---|
| 1 | null | 54927 |
| 2 | 0.0 | 1614945 |
| 3 | 0.01 | 29416 |
| 4 | 0.02 | 259 |
| 5 | 0.03 | 3004 |
| 6 | 0.04 | 277 |
| 7 | 0.05 | 111 |
| 8 | 0.06 | 145 |
| 9 | 0.07 | 65 |
| 10 | 0.08 | 93 |

Resultados por página: 50 1 – 50 de 1048642

Esta consulta verifica se há inconsistências e seleciona a coluna indicador_aprendiz e a contagem de registros (QTD) da tabela, agrupa os resultados por indicador_aprendiz e ordena os resultados por indicador_aprendiz em ordem ascendente (ASC) ou seja exibe os resultados em ordem crescente.

```
SELECT
  indicador_aprendiz,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY indicador_aprendiz
ORDER BY indicador_aprendiz ASC;
```

| Linha | indicador_aprendiz | QTD |
|---|---|---|
| 1 | 0 | 120047659 |
| 2 | 1 | 2792588 |

A consulta verifica se há inconsistências nas colunas como valores nulos, números que não estão formatados corretamente, seleciona a coluna indicador_trabalho_intermitente e a contagem de registros (QTD) da tabela. Em

seguida, agrupa os resultados por indicador_trabalho_intermitente e ordena os resultados por indicador_trabalho_intermitente em ordem ascendente (ASC) e exibe os resultados em ordem crescente.

```
SELECT
  indicador_trabalho_intermitente,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY indicador_trabalho_intermitente
ORDER BY indicador_trabalho_intermitente ASC;
```

| Linha | indicador_trabalho_intermitente | QTD |
|---|---|---|
| 1 | 0 | 121299563 |
| 2 | 1 | 1381725 |
| 3 | 9 | 158959 |

Esta consulta verifica se há algumas inconsistências e visa contar a quantidade de ocorrências para cada valor único da coluna indicador_trabalho_parcial. Os resultados são agrupados de acordo com os valores dessa coluna e ordenados de forma ascendente.

```
SELECT
  indicador_trabalho_parcial,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY indicador_trabalho_parcial
ORDER BY indicador_trabalho_parcial ASC;
```

| Linha | indicador_trabalho_parcial | QTD |
|---|---|---|
| 1 | 0 | 119478900 |
| 2 | 1 | 1313196 |
| 3 | 9 | 2048151 |

Essa consulta foi criada para verificar se há inconsistências e determinar a frequência com que cada tipo de deficiência aparece na coluna "tipo_deficiencia". Os resultados são organizados agrupando as ocorrências por tipo de deficiência e, em seguida, organizados em ordem alfabética crescente.

```
SELECT
  tipo_deficiencia,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY tipo_deficiencia
ORDER BY tipo_deficiencia ASC;
```

| Linha | tipo_deficiencia | QTD |
|---|---|---|
| 1 | 0 | 122209110 |
| 2 | 1 | 270245 |
| 3 | 2 | 113826 |
| 4 | 3 | 118060 |
| 5 | 4 | 66868 |
| 6 | 5 | 25294 |
| 7 | 6 | 36844 |

A finalidade dessa consulta é determinar a quantidade de ocorrências para cada nível de educação registrado na coluna "grau_instrucao". Os resultados são organizados em grupos de acordo com os diversos níveis de educação e, posteriormente, são dispostos em ordem alfabética crescente.

```
SELECT
  grau_instrucao,
```

```
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY grau_instrucao
ORDER BY grau_instrucao ASC;
```

| Linha | grau_instrucao | QTD |
|---|---|---|
| 1 | 1 | 517132 |
| 2 | 10 | 252045 |
| 3 | 11 | 92734 |
| 4 | 2 | 2647311 |
| 5 | 3 | 1996849 |
| 6 | 4 | 5731700 |
| 7 | 5 | 9325422 |
| 8 | 6 | 9345698 |
| 9 | 7 | 75437005 |
| 10 | 8 | 5150161 |

Resultados por página: 50 1 – 13 de 13

O propósito dessa consulta é contabilizar quantas vezes cada idade aparece na coluna "idade". Os resultados são organizados em grupos de acordo com as várias idades presentes e, em seguida, são dispostos em ordem crescente com base nas idades.

```
SELECT
  idade,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY idade
ORDER BY idade ASC;
```

| Linha | idade | QTD |
|---|---|---|
| 1 | null | 8597 |
| 2 | 14 | 58183 |
| 3 | 15 | 179412 |
| 4 | 16 | 631063 |
| 5 | 17 | 1079350 |
| 6 | 18 | 3756771 |
| 7 | 19 | 4542263 |
| 8 | 20 | 4892206 |
| 9 | 21 | 5031166 |
| 10 | 22 | 5109254 |

Resultados por página: 50 1 – 50 de 88

Essa consulta busca determinar a quantidade de ocorrências para cada categoria de sexo registrada na coluna "sexo". Os resultados são organizados em grupos com base nos diferentes valores de sexo e, posteriormente, são arranjados em ordem alfabética crescente.

```
SELECT
  sexo,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY sexo
ORDER BY sexo ASC;
```

| Linha | sexo | QTD |
|---|---|---|
| 1 | 1 | 73762543 |
| 2 | 3 | 49077704 |



O objetivo é contar quantas vezes cada categoria de raça/cor é registrada na coluna "raca_cor". Os resultados são agrupados de acordo com as diferentes categorias de raça/cor e ordenados em ordem alfabética crescente.

```
SELECT
  raca_cor,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY raca_cor
```

```
ORDER BY raca_cor ASC;
```

| Linha | raca_cor | QTD |
|---|---|---|
| 1 | 1 | 44464682 |
| 2 | 2 | 7083513 |
| 3 | 3 | 41130035 |
| 4 | 4 | 586820 |
| 5 | 5 | 244933 |
| 6 | 6 | 29175126 |
| 7 | 9 | 155138 |

Essa consulta SQL busca determinar quantas vezes cada setor econômico da Classificação Nacional de Atividades Econômicas (CNAE) de 2ª seção é mencionado na coluna "cnae_2_secao". Os resultados são agrupados com base nas diferentes categorias da CNAE de 2ª seção e organizados em ordem crescente de cnae_2_secao.

```
SELECT
  cnae_2_secao,
  COUNT(*) AS QTD
FROM `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY cnae_2_secao
ORDER BY cnae_2_secao ASC;
```

| Linha | cnae_2_secao | QTD |
|---|---|---|
| 1 | A | 6466539 |
| 2 | B | 363017 |
| 3 | C | 18707623 |
| 4 | D | 116406 |
| 5 | E | 652492 |
| 6 | F | 12081382 |
| 7 | G | 29032817 |
| 8 | H | 6551762 |
| 9 | I | 7114912 |
| 10 | J | 2675465 |

Resultados por página: 50 1 – 21 de 21

TRATAMENTO CAGED 2020-2023

O código abaixo cria uma tabela temporária fazendo a tipagem para cada coluna. Os dados selecionados são transformados e categorizados de acordo com várias regras. O código organiza os dados por ano, estado, tipo de movimentação, salário mensal, indicadores de aprendiz e trabalho intermitente, entre outros. Além disso, as informações sobre deficiência, escolaridade, idade, sexo, raça/cor e setor econômico são categorizadas em diferentes classes conforme as especificações.

O código também faz algumas transformações nos dados da tabela original. Por exemplo, ele converte as colunas tipo_movimentacao e cnae_2_secao para string e as colunas salario_mensal para float64 e idade para int64. Ele também usa o CASE para converter os valores das colunas indicador_aprendiz, indicador_trabalho_intermitente, indicador_trabalho_parcial, tipo_deficiencia, grau_instrucao, sexo, raca_cor e cnae_2_secao para números mais gerenciáveis. Ele filtra os dados de acordo com várias condições. Selecionar os dados de ano, estado, movimentação, salario mensal, aprendiz, trabalho intermitente, trabalho parcial, Pcd, escolaridade, idade, sexo, raça/cor e subsetor econômico da tabela . Filtra os resultados para incluir somente as linhas em que o salário mensal está entre 350 e 172950, a idade é amiof ou igual a 14, o trabalho intermitente não é igual a 9 e o trabalho parcial não é igual a 9.

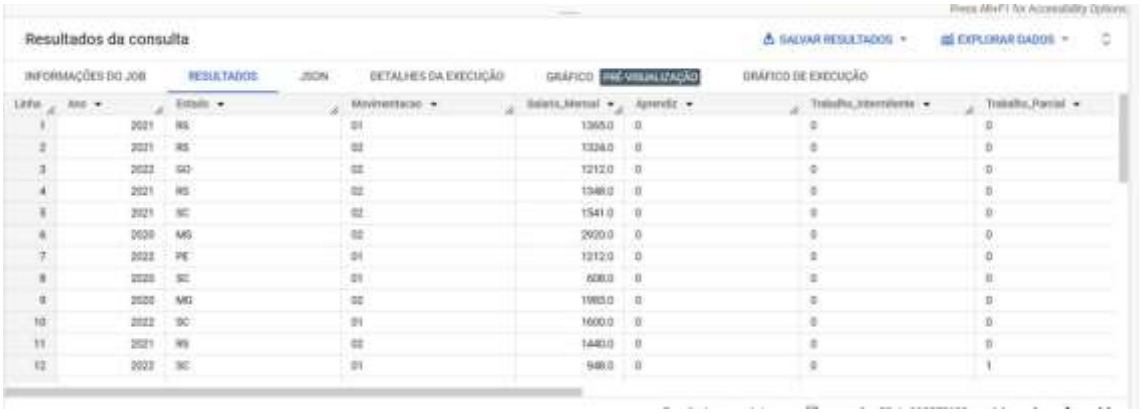
```
WITH Tabela AS (
  SELECT
SAFE_CAST(ano AS INT64) Ano,
SAFE_CAST(sigla_uf AS STRING) Estado,
CASE
      WHEN SAFE_CAST(tipo_movimentacao AS STRING) = '10' OR
SAFE_CAST(tipo_movimentacao AS STRING)= '20' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '25'
```

```
        OR SAFE_CAST(tipo_movimentacao AS STRING) = '35' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '70' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '97' THEN '01'
      WHEN SAFE_CAST(tipo_movimentacao AS STRING) = '31' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '32' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '33'
        OR SAFE_CAST(tipo_movimentacao AS STRING) = '40' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '43' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '45'
        OR SAFE_CAST(tipo_movimentacao AS STRING) = '50' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '60' OR
SAFE_CAST(tipo_movimentacao AS STRING)= '80'
        OR SAFE_CAST(tipo_movimentacao AS STRING) = '90' OR
SAFE_CAST(tipo_movimentacao AS STRING) = '98' THEN '02'
   ELSE '99'
END AS Movimentacao,
SAFE_CAST(salario_mensal AS FLOAT64) Salario_Mensal,
SAFE_CAST(indicador_aprendiz AS STRING) Aprendiz,
SAFE_CAST(indicador_trabalho_intermitente AS STRING) AS
Trabalho_Intermitente,
SAFE_CAST(indicador_trabalho_parcial AS STRING) AS Trabalho_Parcial,
CASE
   WHEN SAFE_CAST(tipo_deficiencia AS STRING) = '0' THEN '0'
   ELSE '1'
END AS PcD,
CASE
      WHEN SAFE_CAST(grau_instrucao AS STRING) = '1' OR
SAFE_CAST(grau_instrucao AS STRING) = '2'OR SAFE_CAST(grau_instrucao
AS STRING)= '3'
    OR SAFE_CAST(grau_instrucao AS STRING) = '4' THEN '1'
      WHEN SAFE_CAST(grau_instrucao AS STRING) = '5' OR
SAFE_CAST(grau_instrucao AS STRING) = '6' THEN '2'
      WHEN SAFE_CAST(grau_instrucao AS STRING) = '7' OR
SAFE_CAST(grau_instrucao AS STRING) = '8' THEN '3'
```

```
        WHEN SAFE_CAST(grau_instrucao AS STRING) = '9' OR
SAFE_CAST(grau_instrucao AS STRING) = '10'OR
SAFE_CAST(grau_instrucao AS STRING) = '11'
     OR SAFE_CAST(grau_instrucao AS STRING) = '80'THEN '4'
   ELSE '99'
END AS Escolaridade,
SAFE_CAST(idade AS INT64) Idade,
CASE
   WHEN SAFE_CAST(sexo AS STRING) = '1' THEN '01'
   ELSE '02'
END AS Sexo,
CASE
   WHEN SAFE_CAST(raca_cor AS STRING) = '5' THEN '01' #Indígena
   WHEN SAFE_CAST(raca_cor AS STRING) = '1' THEN '02' #Branco
   WHEN SAFE_CAST(raca_cor AS STRING) = '2' THEN '04' #Preto
   WHEN SAFE_CAST(raca_cor AS STRING) = '4' THEN '06' #Amarelo
   WHEN SAFE_CAST(raca_cor AS STRING) = '3' THEN '08' #Pardo
   ELSE '99'                                  #Não Informado
END AS Raca_Cor,
CASE
       WHEN SAFE_CAST(cnae_2_secao AS STRING) = 'B' OR cnae_2_secao
= 'C' THEN '01' #Indústria
           WHEN SAFE_CAST(cnae_2_secao AS STRING) = 'F' THEN '03'
#Construção Civil
           WHEN SAFE_CAST(cnae_2_secao AS STRING)= 'G' THEN '04'
#Comércio
        WHEN SAFE_CAST(cnae_2_secao AS STRING) = 'O' THEN '05' #Adm
Pública
           WHEN SAFE_CAST(cnae_2_secao AS STRING) = 'A' THEN '06'
#Agropecuária
       ELSE '02' #Serviço
END AS Subsetor_Economico,

FROM `basedosdados.br_me_caged.microdados_movimentacao`
```

)

```
SELECT Ano, Estado, Movimentacao, Salario_Mensal, Aprendiz, Trabalho_Intermitente, Trabalho_Parcial, PcD,
    Escolaridade, Idade, Sexo, Raca_Cor,Subsetor_Economico,

FROM Tabela

WHERE (Salario_Mensal <= 172950 AND Salario_Mensal >= 350 AND Idade >= 14 AND Trabalho_Intermitente != '9' AND Trabalho_Parcial != '9');
```




A consulta  seleciona o tipo de movimento e a contagem de registros (qtd) da tabela . Em seguida, agrupa os resultados por tipo de movimento e ordena os resultados pelo número de registros (qtd) em ordem decrescente (DESC), ou seja a consulta conta o número de registros de cada tipo de movimento e exibe os resultados em ordem decrescente

```
SELECT
  tipo_movimentacao,
  Count(*) AS qtd
FROM
```

```
    `basedosdados.br_me_caged.microdados_movimentacao`
GROUP BY
    tipo_movimentacao
ORDER BY
    qtd DESC;
Select *
FROM
    `basedosdados.br_me_caged.microdados_movimentacao`
```

CENSO 2010

A consulta sobre o atributo “sigla_uf” tem por objetivo verificar inconsistências como verificar valores duplicados, ou seja, verificar se existem estados com a mesma sigla (mesmo estado escrito de forma diferente ou com diferentes grafias), verificar valores em branco, contabilizando todos os valores nulos que a coluna possui.

```
SELECT
  sigla_uf AS Estado,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY Estado
ORDER BY Estado ASC
```

| Linha | Estado | QTD |
|---|---|---|
| 1 | AC | 93675 |
| 2 | AL | 349966 |
| 3 | AM | 295034 |
| 4 | AP | 78344 |
| 5 | BA | 1550842 |
| 6 | CE | 846164 |
| 7 | DF | 116458 |
| 8 | ES | 400130 |
| 9 | GO | 707043 |
| 10 | MA | 793241 |

Resultados por página: 50 1 – 27 de 27

O atributo “v0601” está relacionado ao sexo da pessoa daquele registro. Segundo o dicionário do banco de dados eles podem assumir os valores “1” ou “2” e a consulta seguir irá verificar se possui valores em branco ou algum valor diferente daqueles determinados no dicionário.

```
SELECT
  v0601 AS Sexo,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY Sexo
ORDER BY Sexo ASC
```

| Linha | Sexo | QTD |
|---|---|---|
| 1 | 1 | 10241757 |
| 2 | 2 | 10393715 |

O atributo “v6036” está relacionado à idade da pessoa daquele registro. A consulta feita irá verificar se possui valores em branco e valores que não sejam numéricos.

```
SELECT
  v6036 AS Idade,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY Idade
ORDER BY Idade ASC
```

| Linha | Idade | QTD |
|---|---|---|
| 1 | 0 | 299588 |
| 2 | 1 | 299046 |
| 3 | 2 | 302767 |
| 4 | 3 | 310493 |
| 5 | 4 | 321844 |
| 6 | 5 | 328203 |
| 7 | 6 | 324535 |
| 8 | 7 | 333044 |
| 9 | 8 | 336714 |
| 10 | 9 | 358472 |

Resultados por página: 50 1 – 50 de 138

O atributo “v0606” está relacionado à raça ou cor da pessoa daquele registro. Segundo o dicionário do banco de dados eles podem assumir os valores “1”, “2”, “3”, “4”, “5” ou “9” e a consulta seguir irá verificar se possui valores em branco ou algum valor diferente daqueles determinados no dicionário.

```
SELECT
  v0606 AS Raca_Cor,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY Raca_Cor
ORDER BY Raca_Cor ASC
```

| Linha | Raca_Cor | QTD |
|---|---|---|
| 1 | 1 | 9704314 |
| 2 | 2 | 1455841 |
| 3 | 3 | 211945 |
| 4 | 4 | 9148854 |
| 5 | 5 | 111834 |
| 6 | 9 | 2684 |

O atributo “v6400” está relacionado à escolaridade da pessoa daquele registro. Segundo o dicionário do banco de dados eles podem assumir os valores “1”, “2”, “3”, “4” ou “5” e a consulta seguir irá verificar se possui valores em branco ou algum valor diferente daqueles determinados no dicionário.

```
SELECT
  v6400 AS Escolaridade,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY Escolaridade
ORDER BY Escolaridade ASC
```

| Linha | Escolaridade | QTD |
|---|---|---|
| 1 | 1 | 12888594 |
| 2 | 2 | 2923750 |
| 3 | 3 | 3604712 |
| 4 | 4 | 1126886 |
| 5 | 5 | 91530 |

Os atributos “v0614”, “v0615” e “v0616”” estão relacionados, respectivamente, à deficiência visual, deficiência auditiva e deficiência física da pessoa daquele registro. Segundo o dicionário do banco de dados eles podem assumir os valores “1”, “2”, “3”, “4” ou “9” e as consultas seguir irá verificar se possui valores em branco ou algum valor diferente daqueles determinados no dicionário.

```
SELECT
  v0614 AS DefVisual,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY DefVisual
ORDER BY DefVisual ASC
```

| Linha | DefVisual | QTD |
|---|---|---|
| 1 | 1 | 49079 |
| 2 | 2 | 679955 |
| 3 | 3 | 3116842 |
| 4 | 4 | 16784146 |
| 5 | 9 | 5450 |

```
SELECT
  v0615 AS DefAuditivo,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY DefAuditivo
ORDER BY DefAuditivo ASC
```

| Linha | DefAuditivo | QTD |
|---|---|---|
| 1 | 1 | 35653 |
| 2 | 2 | 207447 |
| 3 | 3 | 836715 |
| 4 | 4 | 19551910 |
| 5 | 9 | 3747 |

```
SELECT
  v0616 AS DefFisica,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY DefFisica
ORDER BY DefFisica ASC
```

| Linha | DefFisica | QTD |
|---|---|---|
| 1 | 1 | 79368 |
| 2 | 2 | 415707 |
| 3 | 3 | 973439 |
| 4 | 4 | 19162907 |
| 5 | 9 | 4051 |

O atributo “v0617” está relacionado à deficiência intelectual da pessoa daquele registro. Segundo o dicionário do banco de dados eles podem assumir os valores “1”, “2” ou “9” e a consulta seguir irá verificar se possui valores em branco ou algum valor diferente daqueles determinados no dicionário.

```
SELECT
  v0617 AS DefIntelectual,
  COUNT(*) AS QTD
FROM `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`
GROUP BY DefIntelectual
ORDER BY DefIntelectual ASC
```

| Linha | DefIntelectual | QTD |
|---|---|---|
| 1 | 1 | 294078 |
| 2 | 2 | 20337847 |
| 3 | 9 | 3547 |

A consulta a seguir consiste em definir uma tabela temporária para transformar as colunas da tabela original em formatos mais legíveis usando a cláusula "SAFE_CAST" para evitar erros de tipo de dados.

A coluna "Estado" é criada a partir da coluna "sigla_uf" da tabela original e recebe o alias "Estado".

A coluna "Sexo" é criada a partir da coluna "V0601" da tabela original. Se o valor for '1', é considerado "Masculino" (código '01'), caso contrário, é considerado "Feminino" (código '02').

A coluna "Raca_Cor" é criada a partir da coluna "V0606" da tabela original. Os valores numéricos são mapeados para códigos de acordo com a seguinte correspondência para se adequar ao padrão adotado nas outras tabelas: ‘1’ é "Branco" (código '02'), ‘2’ é "Preto" (código '04'), ‘3’ é "Amarelo" (código '06'), ‘4’ é "Pardo" (código '08'), ‘5’ é "Indígena" (código '01') e qualquer outro valor é considerado como "Não Informado" (código '99').

A coluna "Idade" é criada a partir da coluna "V6036" da tabela original. O tipo é convertido para INT64 por se tratar de um atributo numérico.

A coluna "Escolaridade" é criada a partir da coluna "V6400" da tabela original, mantendo o mesmo formato.

A coluna "PcD" é criada a partir das colunas "V0614", "V0615", "V0616" e "V0617" da tabela original. Se alguma das colunas "V0614", "V0615", "V0616" tiver um valor '1', indicando uma dificuldade permanente ou tiver um valor ‘2’

indicando grandes dificuldade ou se a coluna "V0617" tiver valor ‘1’, indicando deficiência mental/intelectual permanente, é considerado como "Sim" (código '1') para portador de deficiência, caso contrário, é considerado como "Não" (código '0') para portador de deficiência.

A consulta final seleciona todas as colunas da tabela temporária e filtra os registros onde a idade (coluna "Idade") está entre 14 e 121 anos. Foi escolhida essa faixa etária, pois a idade mínima para ingressar no mercado de trabalho formal é com 14 anos como aprendiz e a idade máxima já registrada de um brasileiro foi de 121 anos.

```
WITH
  Tabela AS (
  SELECT
    SAFE_CAST(sigla_uf AS STRING) Estado,
    CASE
      WHEN SAFE_CAST(V0601 AS STRING) = '1' THEN '01'#Masculino
    ELSE
    '02'#Feminino
  END
    AS Sexo,
    CASE
      WHEN SAFE_CAST(v0606 AS STRING) = '1' THEN '02' #'Branco'
      WHEN SAFE_CAST(v0606 AS STRING) = '2' THEN '04' #'Preto'
      WHEN SAFE_CAST(v0606 AS STRING) = '3' THEN '06' #'Amarelo'
      WHEN SAFE_CAST(v0606 AS STRING) = '4' THEN '08' #'Pardo'
      WHEN SAFE_CAST(v0606 AS STRING) = '5' THEN '01' #'Indígena'
    ELSE
    '99' #Não Informado
  END
    AS Raca_Cor,
    SAFE_CAST(v6036 AS INT64) Idade,
    SAFE_CAST(v6400 AS STRING) Escolaridade,
    CASE
```

```
      WHEN SAFE_CAST(v0614 AS STRING) = '1'OR SAFE_CAST(v0614 AS STRING) = '2' OR SAFE_CAST(v0615 AS STRING) = '1'OR SAFE_CAST(v0615 AS STRING) = '2' OR SAFE_CAST(v0616 AS STRING) = '1'OR SAFE_CAST(v0616 AS STRING) = '2' OR SAFE_CAST(v0617 AS STRING) = '1' THEN '1'#Sim
    ELSE
    '0' #Não
  END
    AS PcD,
  FROM
    `basedosdados.br_ibge_censo_demografico.microdados_pessoa_2010`)
SELECT
  *
FROM
  Tabela
WHERE
  Idade >= 14
  AND Idade <= 121;
```



| Linha | Estado | Sexo | Raca_Cor | Idade | Escolaridade | PcD |
|---|---|---|---|---|---|---|
| 1 | AC | 02 | 08 | 23 | 1 | 0 |
| 2 | AC | 02 | 08 | 58 | 1 | 0 |
| 3 | AC | 02 | 08 | 20 | 3 | 0 |
| 4 | AC | 02 | 02 | 26 | 3 | 0 |
| 5 | AC | 02 | 08 | 14 | 1 | 0 |
| 6 | AC | 01 | 08 | 26 | 2 | 0 |
| 7 | AC | 02 | 02 | 41 | 3 | 0 |
| 8 | AC | 02 | 08 | 25 | 1 | 0 |
| 9 | AC | 02 | 08 | 21 | 3 | 0 |
| 10 | AC | 01 | 08 | 46 | 3 | 0 |

Resultados por página: 50 1 – 50 de 15892813

Uma tabela temporária é definida para transformar as colunas da tabela original em formatos mais legíveis usando a cláusula "SAFE_CAST" para evitar erros de tipo de dados.

A query a seguir cria uma nova tabela com informações mais legíveis e categorizadas, a partir dos dados da tabela censo2010. Os campos Estado,

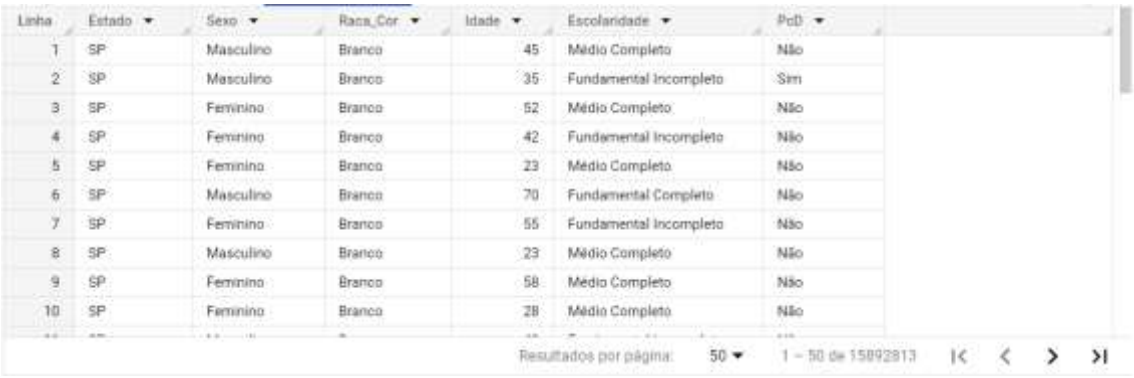
Sexo, Raca_Cor, Idade, Escolaridade e PcD foram convertidos ou categorizados para facilitar a análise e o uso dessas informações.

| **Coluna Original** | **Tipo Original** | **Categorização** |
|---|---|---|
| Estado | STRING | STRING (Sigla dos estados) |
| Sexo | STRING | Masculino / Feminino |
| Raca_Cor | STRING | Branco / Preto / Amarelo / Pardo / Indígena / Não Informado |
| Idade | INT64 | INT64 (idade) |
| Escolaridade | STRING | Fundamental Incompleto / Fundamental Completo / Médio Completo / Superior Completo / Não Informado |
| PcD | STRING | Sim / Não / Não Informado |



```
CREATE TABLE
  `symphone-project.projeto_final.censo2010-final` AS
SELECT
  SAFE_CAST(Estado AS STRING) Estado,
  CASE
    WHEN SAFE_CAST(Sexo AS STRING) = '1' THEN 'Masculino'
  ELSE
  'Feminino'
END
  AS Sexo,
  CASE
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '2' THEN 'Branco'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '4' THEN 'Preto'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '6' THEN 'Amarelo'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '8' THEN 'Pardo'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '1' THEN 'Indígena'
```

```
  ELSE
  'Não Informado'
END
  AS Raca_Cor,
  SAFE_CAST(Idade AS INT64) Idade,
  CASE
    WHEN SAFE_CAST(Escolaridade AS STRING) = '1' THEN 'Fundamental
Incompleto'
    WHEN SAFE_CAST(Escolaridade AS STRING) = '2' THEN 'Fundamental
Completo'
   WHEN SAFE_CAST(Escolaridade AS STRING) = '3' THEN 'Médio Completo'
     WHEN SAFE_CAST(Escolaridade AS STRING) = '4' THEN 'Superior
Completo'
  ELSE
  'Não Informado'
END
  AS Escolaridade,
  CASE
   WHEN SAFE_CAST(PcD AS STRING) = '1' THEN 'Sim'
   WHEN SAFE_CAST(PcD AS STRING) = '0' THEN 'Não'
  ELSE
  'Não Informado'
END
  AS PcD
FROM
  `symphone-project.projeto_final.censo2010`
```



| Linha | Estado | Sexo | Raca_Cor | Idade | Escolaridade | PcD |
|---|---|---|---|---|---|---|
| 1 | SP | Masculino | Branco | 45 | Médio Completo | Não |
| 2 | SP | Masculino | Branco | 35 | Fundamental Incompleto | Sim |
| 3 | SP | Feminino | Branco | 52 | Médio Completo | Não |
| 4 | SP | Feminino | Branco | 42 | Fundamental Incompleto | Não |
| 5 | SP | Feminino | Branco | 23 | Médio Completo | Não |
| 6 | SP | Masculino | Branco | 70 | Fundamental Completo | Não |
| 7 | SP | Feminino | Branco | 55 | Fundamental Incompleto | Não |
| 8 | SP | Masculino | Branco | 23 | Médio Completo | Não |
| 9 | SP | Feminino | Branco | 58 | Médio Completo | Não |
| 10 | SP | Feminino | Branco | 28 | Médio Completo | Não |

Resultados por página: 50 1 – 50 de 15892813

**Unindo Tabelas: CAGED (2012 - 2019) e CAGED (2020 - 2023)**

CAGED (2012 – 2023)

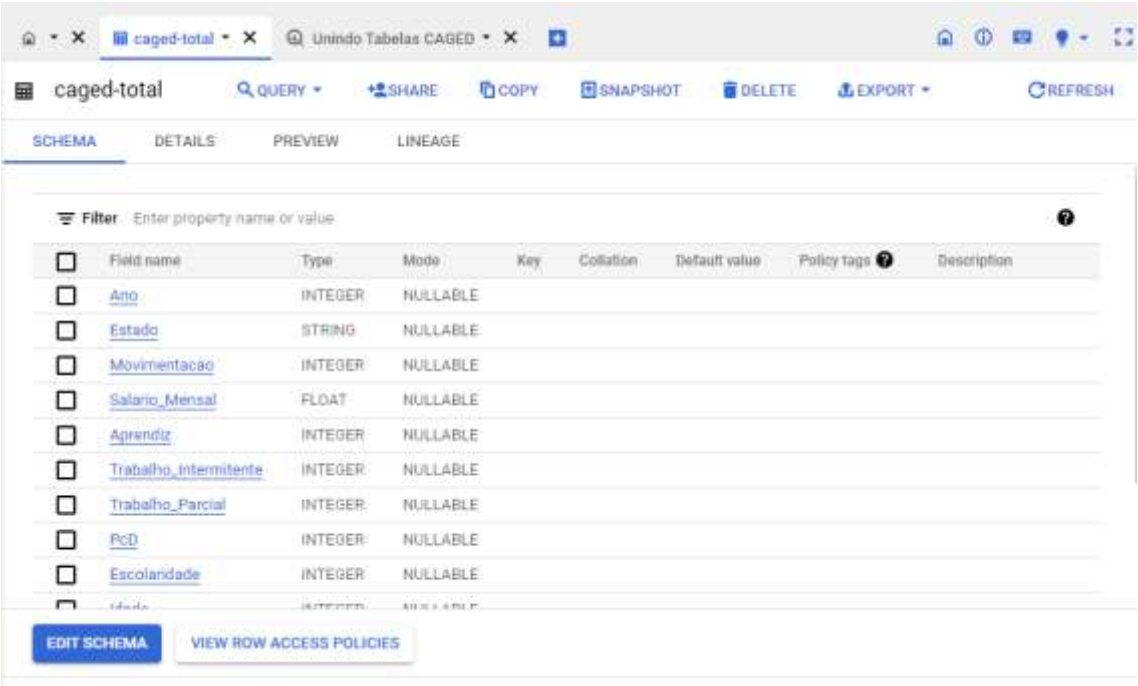
Após as etapas de transformação e mineração de dados realizadas tanto na tabela CAGED (2012-2019) quanto na tabela CAGED (2020 - 2023) iremos realizar a união das duas tabelas, já que o processo foi realizado separadamente devido a grande quantidade de dados contidos antes da mineração.

A instrução **CREATE TABLE** foi usada para criar uma nova tabela chamada caged-total no projeto `projeto_final`. Essa instrução é frequentemente usada para definir a estrutura de uma nova tabela.

A cláusula **UNION ALL** é usada para combinar todas as linhas das duas tabelas, incluindo duplicatas (se houver) e sem remover os registros duplicados. Portanto, o resultado final na tabela caged-total conterá todas as linhas das tabelas caged-2012-2019 e caged-2020-2023. O schema das tabelas foi elaborado e verificado anteriormente a fim de manter compatibilidade de atributos entre as duas tabelas.

```
CREATE TABLE `symphone-project.projeto_final.caged-total` AS
```

```
SELECT * FROM `symphone-project.projeto_final.caged-2012-2019`

UNION ALL

SELECT * FROM  `symphone-project.projeto_final.caged-2020-2023`
```




Realizando consulta para visualizar os dados da tabela criada:

```
SELECT * FROM `symphone-project.projeto_final.caged-total`
```


Query results

SAVE RESULTS | EXPLORE DATA

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART | PREVIEW | EXECUTION GRAPH

| Row | Ano | Estado | Movimentacao | Salario_Mensal | Aprendiz | Trabalho_Intermitent | Trabalh |
|---|---|---|---|---|---|---|---|
| 1 | 2012 | ES | 1 | 311.0 | 1 | 3 | |
| 2 | 2012 | MA | 2 | 622.0 | 0 | 3 | |
| 3 | 2012 | SP | 1 | 622.0 | 1 | 3 | |
| 4 | 2012 | MG | 1 | 365.0 | 1 | 3 | |

Podemos visualizar com essa amostra de dados, que muitos dados estão em forma numérica, o que foi corrigido na parte de “Tradução da tabela”, apresentada abaixo.

**Tradução da Tabela CAGED Total**

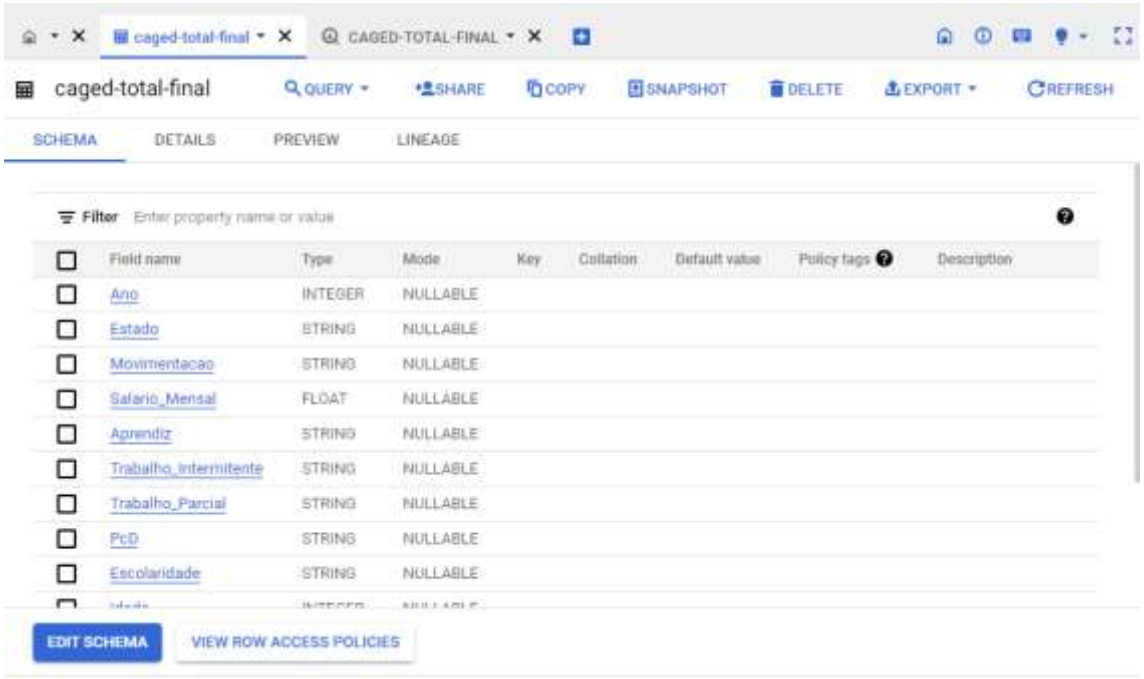
Foi realizada a criação de uma nova tabela nomeada `caged-total-final` no projeto `projeto_final`, dentro do ambiente do BigQuery. A nova tabela é derivada da transformação dos dados da tabela `caged-total`, a transformação envolve a conversão de alguns campos para formatos mais legíveis, como substituir dados numéricos por dados descritivos, como "Admissão" ou "Desligamento", e categorizar informações, como raça, escolaridade, e etc.

- **CREATE TABLE** cria uma nova tabela no `projeto_final`
- **SELECT ... FROM** seleciona as colunas da tabela `caged-total`
- A função **SAFE_CAST** foi usada para converter os valores dessas colunas para tipos de dados específicos (como INT64, FLOAT64 ou STRING), e em seguida, aplica a lógica de transformação usando a cláusula **CASE** para tornar os dados mais compreensíveis. Por exemplo, a coluna **Movimentacao** é convertida para valores como "Admissão", "Desligamento" ou "Não Informado" com base no valor numérico original.
- **END AS**, usado para encerrar a cláusula CASE

```
CREATE TABLE `symphone-project.projeto_final.caged-total-final` AS
SELECT
SAFE_CAST(Ano AS INT64) Ano,
SAFE_CAST(Estado AS STRING) Estado,
CASE
   WHEN SAFE_CAST(Movimentacao AS STRING) = '1' THEN 'Admissão'
   WHEN SAFE_CAST(Movimentacao AS STRING) = '2' THEN 'Desligamento'
   ELSE 'Não Informado'
END AS Movimentacao,
SAFE_CAST(Salario_Mensal AS FLOAT64) Salario_Mensal,
CASE
   WHEN SAFE_CAST(Aprendiz AS STRING) = '1' THEN 'Sim'
   WHEN SAFE_CAST(Aprendiz AS STRING) = '0' THEN 'Não'
   ELSE 'Não Informado'
END AS Aprendiz,
CASE
   WHEN SAFE_CAST(Trabalho_Intermitente AS STRING) = '1' THEN 'Sim'
   WHEN SAFE_CAST(Trabalho_Intermitente AS STRING) = '0' THEN 'Não'
   ELSE 'Não se Aplica'
```

```
END AS Trabalho_Intermitente,
CASE
   WHEN SAFE_CAST(Trabalho_Parcial AS STRING) = '1' THEN 'Sim'
   WHEN SAFE_CAST(Trabalho_Parcial AS STRING) = '0' THEN 'Não'
   ELSE 'Não se Aplica'
END AS Trabalho_Parcial,
CASE
   WHEN SAFE_CAST(PcD AS STRING) = '1' THEN 'Sim'
   WHEN SAFE_CAST(PcD AS STRING) = '0' THEN 'Não'
   ELSE 'Não Informado'
END AS  PcD,
CASE
        WHEN SAFE_CAST(Escolaridade AS STRING) = '1' THEN 'Fundamental Incompleto'
        WHEN SAFE_CAST(Escolaridade AS STRING) = '2' THEN 'Fundamental Completo'
        WHEN SAFE_CAST(Escolaridade AS STRING) = '3' THEN 'Médio Completo'
        WHEN SAFE_CAST(Escolaridade AS STRING) = '4' THEN 'Superior Completo'
        ELSE 'Não Informado'
END AS  Escolaridade,
SAFE_CAST(Idade AS INT64) Idade,
CASE
        WHEN SAFE_CAST(Sexo AS STRING) = '1' THEN 'Masculino'
        ELSE 'Feminino'
```

```
END AS Sexo,
CASE
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '2' THEN 'Branco'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '4' THEN 'Preto'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '6' THEN 'Amarelo'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '8' THEN 'Pardo'
    WHEN SAFE_CAST(Raca_Cor AS STRING) = '1' THEN 'Indígena'
    ELSE 'Não Informado'
END AS Raca_Cor,
CASE
    WHEN SAFE_CAST(Subsetor_Economico AS STRING) = '1' THEN 'Indústria'
    WHEN SAFE_CAST(Subsetor_Economico AS STRING) = '2' THEN 'Serviço'
    WHEN SAFE_CAST(Subsetor_Economico AS STRING) = '3' THEN 'Construção Civil'
    WHEN SAFE_CAST(Subsetor_Economico AS STRING) = '4' THEN 'Comércio'
    WHEN SAFE_CAST(Subsetor_Economico AS STRING) = '5' THEN 'Adm. Pública'
    WHEN SAFE_CAST(Subsetor_Economico AS STRING) = '6' THEN 'Agropecuária'
    ELSE 'Não Informado'
END AS Subsetor_Economico

FROM `symphone-project.projeto_final.caged-total`;
```




Realizando consulta para visualizar os dados da tabela criada:

```
SELECT * FROM `symphone-project.projeto_final.caged-total-final`
```


A nova tabela`caged-total-final` contém os mesmos dados da tabela original `caged-total`, mas com alguns campos transformados para tornar as informações mais claras e significativas. Isso pode facilitar análises e a apresentação desses dados no relatório, parte da visualização.

**Consultas Colunas PNADC**

Estaremos utilizando SQL para interagir com um banco de dados que contém dados da Pesquisa Nacional por Amostra de Domicílios Contínua (PNADC) do Instituto Brasileiro de Geografia e Estatística (IBGE).

Essa consulta vai selecionar todas as colunas da tabela **microdados** na base de dados **basedosdados.br_ibge_pnadc** e retornar no máximo 1000 linhas de resultados.

```
SELECT * FROM `basedosdados.br_ibge_pnadc.microdados` LIMIT 1000 ;
```

| Linha | ano | trimestre | id_uf | sigla_uf | capital | rm_ride | id_upa |
|---|---|---|---|---|---|---|---|
| 1 | 2017 | 3 | 16 | AP | null | 16 | 160000321 |
| 2 | 2017 | 3 | 16 | AP | null | 16 | 160000321 |
| 3 | 2017 | 3 | 16 | AP | null | 16 | 160000321 |
| 4 | 2017 | 3 | 16 | AP | null | 16 | 160000321 |

Resultados por página: 50 1 – 50 de 1000

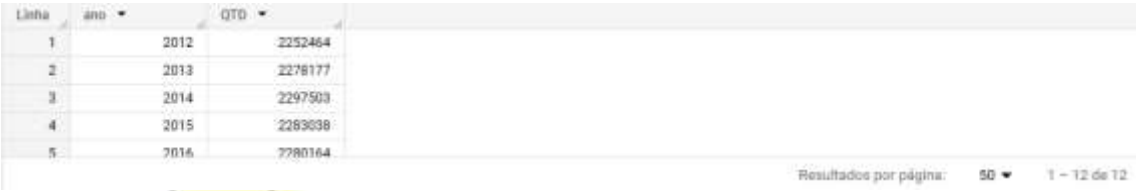
A consulta retorna um conjunto de resultados contendo informações sobre as colunas da tabela 'microdados', incluindo seus nomes, tipos de dados e se elas permitem valores nulos ou não. Isso é útil para entender a estrutura da tabela e como os dados estão organizados nela.

```
SELECT
  column_name,
  data_type,
  is_nullable
FROM
  `basedosdados.br_ibge_pnadc.INFORMATION_SCHEMA.COLUMNS`
```

```
WHERE
  table_name = 'microdados';
```

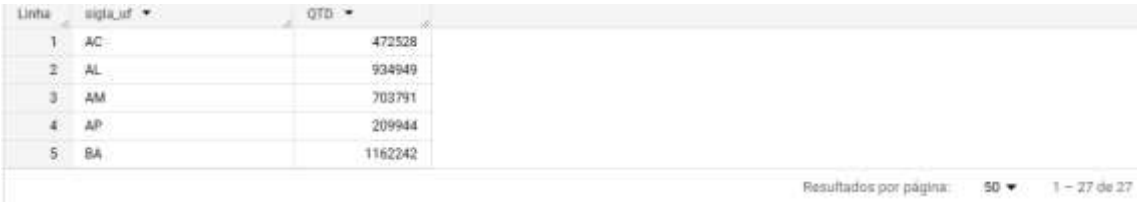
Essa consulta vai fornecer um conjunto de resultados que mostra a quantidade total de registros para cada ano na base de dados da PNADC do IBGE, ordenados de forma ascendente pelos ano.

```
SELECT
  ano,
  COUNT(*) as QTD
FROM `basedosdados.br_ibge_pnadc.microdados`
GROUP BY ano
ORDER BY ano;
```



As próximas consultas seguirão mesmo padrão, irão nos retornar resultados segundo os registros solicitados que constam na base de dados PNADC do IBGE.

```
SELECT
  sigla_uf,
  COUNT(*) as QTD
FROM `basedosdados.br_ibge_pnadc.microdados`
GROUP BY sigla_uf
ORDER BY sigla_uf;
```



| Linha | sigla_uf | QTD |
|---|---|---|
| 1 | AC | 472528 |
| 2 | AL | 934949 |
| 3 | AM | 703791 |
| 4 | AP | 209944 |
| 5 | BA | 1162242 |

Resultados por página: 50 1 – 27 de 27

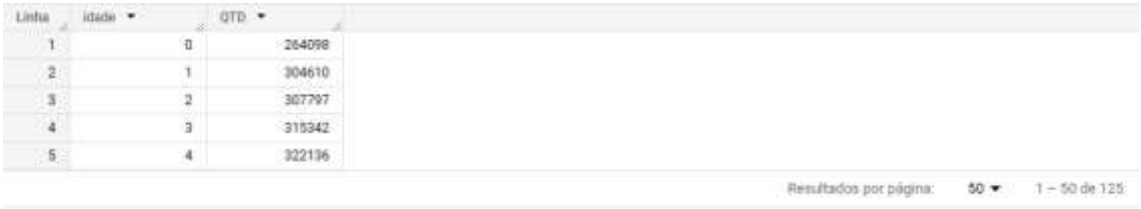
```
SELECT
  V2007 AS sexo,
  COUNT(*) as QTD
FROM `basedosdados.br_ibge_pnadc.microdados`
GROUP BY V2007
ORDER BY V2007;
```

| Linha | sexo | QTD |
|---|---|---|
| 1 | 1 | 11532157 |
| 2 | 2 | 12086570 |

```
SELECT
  V2010 AS raca_cor,
  COUNT(*) as QTD
FROM `basedosdados.br_ibge_pnadc.microdados`
GROUP BY V2010
ORDER BY V2010;
```

| Linha | raca_cor | QTD |
|---|---|---|
| 1 | 1 | 9358723 |
| 2 | 2 | 1793983 |
| 3 | 3 | 95936 |
| 4 | 4 | 12279280 |
| 5 | 5 | 84840 |
| 6 | 9 | 5965 |

```
SELECT
  V2009 AS idade,
  COUNT(*) as QTD
FROM `basedosdados.br_ibge_pnadc.microdados`
GROUP BY V2009
ORDER BY V2009;
```



| Linha | idade | QTD |
|---|---|---|
| 1 | 0 | 264098 |
| 2 | 1 | 304610 |
| 3 | 2 | 307797 |
| 4 | 3 | 315342 |
| 5 | 4 | 322136 |

Resultados por página: 50 1 – 50 de 125

```
SELECT
  VD3004 AS grau_instrucao,
  COUNT(*) as QTD
FROM `basedosdados.br_ibge_pnadc.microdados`
GROUP BY VD3004
ORDER BY VD3004;
```

| Linha | grau_instrucao | QTD |
|---|---|---|
| 1 | *null* | 1513983 |
| 2 | 1 | 2164153 |
| 3 | 2 | 9003762 |
| 4 | 3 | 1905223 |
| 5 | 4 | 1463376 |
| 6 | 5 | 4685435 |

**TRATAMENTO PNADC**

Essa consulta realiza uma série de transformações em uma tabela chamada "Tabela" e, em seguida, seleciona as linhas que correspondem a determinados

critérios de idade. Vou explicar o que cada parte do código faz:

**WITH Tabela AS (...)**: Nesta parte, estamos criando uma tabela temporária chamada "Tabela" por meio de um comando CTE. Isso envolve selecionar várias colunas da tabela **basedosdados.br_ibge_pnadc.microdados** e realizar algumas transformações nos dados. Convertemos as colunas para os tipos de dados corretos, como INT64 e STRING, e também mapeando valores de diferentes colunas para novos valores (como mapear códigos de raça/etnia para categorias específicas). Essa parte do código está essencialmente preparando os dados para a análise subsequente.

**SELECT * FROM Tabela WHERE Idade >= 14 AND Idade <= 121;**: Nesta parte, estamos selecionando todas as colunas da tabela "Tabela" que foi definida anteriormente. Estamos aplicando um filtro usando a cláusula WHERE para selecionar apenas as linhas em que a coluna "Idade" está dentro do intervalo de 14 a 121 anos. Utilizado para excluir registros com idades inválidas ou irrelevantes para a análise.



```
WITH Tabela AS (
    SELECT
      SAFE_CAST(ano AS INT64) Ano,
      SAFE_CAST(sigla_uf AS STRING) Estado,
      CASE
        WHEN SAFE_CAST(V2007 AS STRING) = '1' THEN '01'
        ELSE '02'
      END AS Sexo,
      CASE
        WHEN SAFE_CAST(V2010 AS STRING) = '1' THEN '02' #'Branco'
        WHEN SAFE_CAST(V2010 AS STRING) = '2' THEN '04' #'Preto'
        WHEN SAFE_CAST(V2010 AS STRING) = '3' THEN '06' #'Amarelo'
        WHEN SAFE_CAST(V2010 AS STRING) = '4' THEN '08' #'Pardo'
        WHEN SAFE_CAST(V2010 AS STRING) = '5' THEN '01' #'Indígena'
        ELSE '99'                                  #Não Informado
```

```
        END AS Raca_Cor,
        SAFE_CAST(V2009 AS INT64) AS Idade,

        CASE
         WHEN SAFE_CAST(VD3004 AS STRING) = '1' OR SAFE_CAST(VD3004
AS STRING) = '2' THEN '1'
         WHEN SAFE_CAST(VD3004 AS STRING) = '3' OR SAFE_CAST(VD3004
AS STRING) = '4' THEN '2'
         WHEN SAFE_CAST(VD3004 AS STRING) = '5' OR SAFE_CAST(VD3004
AS STRING) = '6' THEN '3'
          WHEN SAFE_CAST(VD3004 AS STRING) = '7' THEN '4'
          ELSE '99'
        END AS Escolaridade
  FROM `basedosdados.br_ibge_pnadc.microdados`)


SELECT *
FROM Tabela
WHERE Idade >= 14 AND Idade <= 121;
```

| Linha | Ano | Estado | Sexo | Raca_Cor | Idade | Escolaridade |
|---|---|---|---|---|---|---|
| 1 | 2012 | RR | 01 | 08 | 22 | 1 |
| 2 | 2012 | RR | 01 | 08 | 32 | 1 |
| 3 | 2012 | RR | 01 | 02 | 84 | 1 |
| 4 | 2012 | RR | 02 | 08 | 15 | 2 |
| 5 | 2012 | RR | 02 | 08 | 29 | 3 |

Resultados por página: 50 1 – 50 de 18927512

**TRADUÇÃO DA TABELA PNADC**

Estamos criando uma nova tabela chamada **symphone-project.projeto_final.pnad-c-ibge-final** a partir dos resultados de uma consulta realizada na tabela **symphone-project.projeto_final.pnad-c-ibge-novo**. A nova tabela terá colunas transformadas e mapeadas conforme as instruções no código.

• CREATE TABLE symphone-project.projeto_final.pnad-c-ibge-final AS ...: Aqui estamos criando uma nova tabela chamada pnad-c-ibge-final no projeto

projeto_final com o dataset symphone-project. O AS indica que os resultados da consulta seguinte serão usados para preencher essa nova tabela. Essas transformações incluem alterações nos tipos de dados e a aplicação de mapeamentos para melhor representar os dados de raça/etnia, sexo e escolaridade.

• SELECT ...: Nesta parte estamos selecionando várias colunas da tabela symphone-project.projeto_final.pnad-c-ibge-novo e aplicando transformações e mapeamentos em algumas colunas.

A coluna "Ano" é convertida para o tipo INT64.
A coluna "Estado" é mantida como STRING.
A coluna "Sexo" é mapeada de '1' para 'Masculino' e '2' (presumivelmente) para 'Feminino'.
A coluna "Raca_Cor" é mapeada de acordo com códigos para as categorias de raça/etnia, como '2' para 'Branco', '4' para 'Preto' e assim por diante.
A coluna "Idade" é convertida para o tipo INT64.
A coluna "Escolaridade" é mapeada de acordo com códigos para categorias de nível de escolaridade, como '1' para 'Fundamental Incompleto' e assim por diante.

```
CREATE TABLE `symphone-project.projeto_final.pnad-c-ibge-final`
AS
SELECT
    SAFE_CAST(Ano AS INT64) Ano,
    SAFE_CAST(Estado AS STRING) Estado,
    CASE
      WHEN SAFE_CAST(Sexo AS STRING) = '1' THEN 'Masculino'
      ELSE 'Feminino'
    END AS Sexo,
    CASE
      WHEN SAFE_CAST(Raca_Cor AS STRING) = '2' THEN 'Branco'
      WHEN SAFE_CAST(Raca_Cor AS STRING) = '4' THEN 'Preto'
      WHEN SAFE_CAST(Raca_Cor AS STRING) = '6' THEN 'Amarelo'
      WHEN SAFE_CAST(Raca_Cor AS STRING) = '8' THEN 'Pardo'
```

```
        WHEN SAFE_CAST(Raca_Cor AS STRING) = '1' THEN 'Indígena'
        ELSE 'Não Informado'
      END AS Raca_Cor,
      SAFE_CAST(Idade AS INT64) Idade,

      CASE
        WHEN SAFE_CAST(Escolaridade AS STRING) = '1' THEN 'Fundamental Incompleto'
        WHEN SAFE_CAST(Escolaridade AS STRING) = '2' THEN 'Fundamental Completo'
        WHEN SAFE_CAST(Escolaridade AS STRING) = '3' THEN 'Médio Completo'
        WHEN SAFE_CAST(Escolaridade AS STRING) = '4' THEN 'Superior Completo'
        ELSE 'Não Informado'
      END AS Escolaridade
FROM `symphone-project.projeto_final.pnad-c-ibge-novo` ;
```

| Linha | Ano | Estado | Sexo | Raca_Cor | Idade | Escolaridade |
|---|---|---|---|---|---|---|
| 1 | 2014 | SP | Masculino | Branco | 71 | Fundamental Incompleto |
| 2 | 2012 | SP | Masculino | Branco | 49 | Fundamental Incompleto |
| 3 | 2019 | SP | Masculino | Branco | 77 | Fundamental Incompleto |
| 4 | 2016 | SP | Masculino | Pardo | 51 | Fundamental Incompleto |
| 5 | 2023 | SP | Feminino | Pardo | 54 | Fundamental Incompleto |

Resultados por página: 50 1 – 50 de 18927512

- IPEA | Colab - Pandas e PySpark

Link do colab: https://colab.research.google.com/drive/1V1Pk1XoAJkNqbi2C6FpSYHv5Aa8CBhQk#scrollTo=brO1Mgab1l9X

# Projeto - Mercado de Trabalho

Escola: SoulCode Academy

Curso: Bootcamp Analista de Dados - Martech - AD2

Professor(a): Franciane Rodrigues

Analistas:


- Anderson Melo
- Aska Pereira
- Diego Aguiar
- Jéssica Staudt
- Pedro Barrionovo
- Rosana Santos


**Sobre os Dados**

O Instituto de Pesquisa Econômica Aplicada (Ipea) lançou um estudo com indicadores inéditos no Brasil sobre mercado de trabalho e produtividade. Um deles é o Índice de Qualidade do Trabalho (IQT), que analisa dados de experiência da população ocupada do país.

O conjunto de dados a ser utilizado é proveniente de um estudo que analisa os aspectos determinantes do Mercado de Trabalho no Brasil. O mercado de trabalho brasileiro passou por grandes desafios nos últimos anos, mas também apresenta oportunidades e tendências para o futuro. Alguns fatores que influenciam o mercado de trabalho são: a situação econômica, a qualificação profissional, a demanda por novas habilidades, a diversidade e a inclusão, a tecnologia e a inovação.

Nos links a seguir é possível encontrar esses estudos e irá auxiliar na análise de dados do projeto.

Fonte:

1. https://www.ipea.gov.br/cartadeconjuntura/index.php/category/mercado-de-trabalho

## Pergunta de negócio

### Análise de Dados:

1. O quê? Distribuições de trabalho formal e informal, entre outros.
2. Por quê? Entender e identificar baseado nas escolhas da população a preferência de sua escolha.
3. Quem? Identificar quem são as pessoas ou grupos envolvidos no objetivo.
4. Quando? Verificar as motivações associadas ao mercado de trabalho.
5. Onde? No mercado de trabalho.

## Dicionário

### Dicionário IPEA

**Ano_Mes**: Ano e mês do trimestre móvel
**Populacao**: População total
**Ocupacao**: Pessoas de 14 anos ou mais de idade, ocupadas (no mercado de trabalho) na semana de referência
**Desocupacao**: Pessoas de 14 anos ou mais de idade, desocupadas na semana de referência
**Fora_da_forca**: Pessoas de 14 anos ou mais de idade, fora da força de trabalho na semana de referência
**Privado_com_cart**: Empregado no setor privado com carteira de trabalho assinada
**Privado_sem_cart**: Empregado no setor privado sem carteira de trabalho assinada
**Domestico_com_cart**:Trabalhador doméstico com carteira de trabalho assinada
**Domestico_sem_cart**: Trabalhador doméstico sem carteira de trabalho assinada
**Publi_com_cart**: Empregado no setor público com carteira de trabalho assinada
**Publi_sem_cart**: Empregado no setor público sem carteira de trabalho assinada
**Conta_prop_com_cnpj**: Conta-própria com registro no Cadastro Nacional da Pessoa Jurídica (CNPJ)
**Conta_prop_sem_cnpj**: Conta-própria sem registro no Cadastro Nacional da Pessoa Jurídica (CNPJ)
**Trabalhador_familiar**: Trabalhador familiar auxiliar

## Instalações e Importações

As linhas de código **"!pip install gcsfs"** é utilizadas para instalar pacotes adicionais em um projeto. O comando **"!pip install gcsfs"** instala a biblioteca "gcsfs", que possibilita o acesso a sistemas de arquivos distribuídos, como o Google Cloud Storage, facilitando a leitura e gravação de arquivos na nuvem.

## Preparando o ambiente

Configurando o ambiente de desenvolvimento e preparando a infraestrutura necessária para trabalhar com a biblioteca PySpark, que é a interface Python para o Apache Spark.

```
[ ]  1 # Instalando a biblioteca gcsfs
     2 # Biblioteca usada para interagir com o Google Cloud Storage a partir de código Python.
     3 !pip install gcsfs
```

```
[ ]  1 # Instalando o OpenJDK 8 de forma silenciosa
     2 # O OpenJDK é necessário para executar o Apache Spark, que é implementado em Java
     3 !apt-get install openjdk-8-jdk-headless -qq > /dev/null
     4
     5 # Baixando o arquivo do Apache Spark na versão 3.1.1 compatível com Hadoop 3.2
     6 !wget -q http://archive.apache.org/dist/spark/spark-3.1.1/spark-3.1.1-bin-hadoop3.2.tgz
     7
     8 # Descompactando o arquivo do Apache Spark
     9 !tar xf spark-3.1.1-bin-hadoop3.2.tgz
    10
    11 # Instalando a biblioteca findspark de forma silenciosa
    12 # A biblioteca "findspark" é útil para localizar a instalação do Spark no ambiente Python
    13 !pip install -q findspark
```

```
[ ]
# Importando o módulo 'os' para manipulação do ambiente
import os

# Definindo a variável de ambiente 'JAVA_HOME' para o diretório do Java 8
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"

# Definindo a variável de ambiente 'SPARK_HOME' para o diretório do Spark
os.environ["SPARK_HOME"] = "/content/spark-3.1.1-bin-hadoop3.2"
```

```
[ ]
# Importando a biblioteca findspark para localizar a instalação do Spark
import findspark

# Inicializando a configuração do Spark usando o findspark
findspark.init()

# Importando a classe SparkSession para criar uma sessão Spark
from pyspark.sql import SparkSession

# Criando uma sessão Spark local usando todos os núcleos disponíveis
spark = SparkSession.builder.master("local[*]").getOrCreate()

# Importando a função 'regexp_replace' do Spark para manipulação de strings
from pyspark.sql.functions import regexp_replace

# Configurando o Spark para avaliação imediata das consultas (eager evaluation)
spark.conf.set("spark.sql.repl.eagerEval.enabled", True) # Para deixar a visualição das tabelas mais amigável

# Mostrando a sessão Spark configurada
spark
```



## Declarando as Bibliotecas

Algumas bibliotecas são importadas para facilitar a análise de dados. A linha "import pandas as pd" importa a biblioteca Pandas, que fornece estruturas de dados flexíveis e eficientes para manipulação e análise de dados. A linha "import numpy as np" importa a biblioteca NumPy, que é amplamente utilizada para realizar operações numéricas e matemáticas em arrays multidimensionais. A linha "import os" importa a biblioteca os, que fornece funcionalidades relacionadas ao sistema operacional, como manipulação de caminhos de arquivos. A linha "from google.cloud import storage" importa a biblioteca de armazenamento do Google Cloud, permitindo acesso a serviços de armazenamento em nuvem. Por fim, a linha "import seaborn as sn" importa a biblioteca Seaborn, que é uma biblioteca de visualização de dados baseada no Matplotlib, fornecendo recursos adicionais para a criação de gráficos estatísticos e informativos. Essas bibliotecas serão utilizadas ao longo do projeto para manipular, validar, visualizar e analisar os dados.

```
[ ]
# Importando bibliotecas
'''
os: sistema operacional
pandas: para análise de dados
numpy: para cálculos numéricos
google.cloud.storage: para interação com o Google Cloud Storage
matplotlib.pyplot: para visualização de gráficos
seaborn: para visualização estatística de dados

'''
import os
import pandas as pd
import numpy as np
from google.cloud import storage
import matplotlib.pyplot as plt
import seaborn as sn
```

## Extração

O próximo passo é fazer a extração dos dados. Essa etapa vai depender de onde estarão localizados os dados. Neste projeto em específico serão mostradas como obter os dados localizados em um Google Drive, mas também pelo serviço de armazenamento da GCP (Google Cloud Plataform).

## Google Cloud

```
[ ]  1 # CONFIGURANDO DA CHAVE DE SEGURANCA - ACESSO O PROJETO
     2 serviceAccount = '/content/symphone-project-2b0e69e15a32.json' # Chave criada no IAM
     3 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
```

```
[ ]  1 # Configurações Google Cloud Storage - ACESSO AO BUCKET
     2 client = storage.Client()
     3 bucket = client.get_bucket('mercado-de-trabalho-projeto')
     4 bucket.blob('ipea.xls')
     5 path = 'gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Bruto/ipea.xls'
```

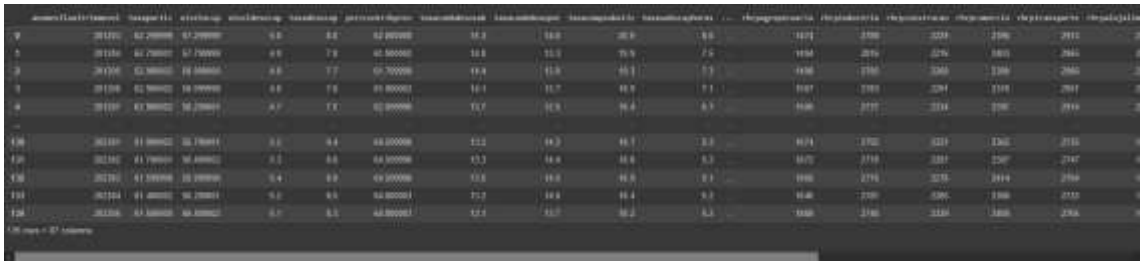
```
[ ]  1 # Abertura da base de dados e cópia de segurança (bucket)
     2 # Ler o arquivo XLS, utilizando pandas
     3 path = "gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Bruto/ipea.xls"  # Substitua pelo caminho real do arquivo XLS
     4 df = pd.read_excel(path, sheet_name='TrimestreNovel')
```

## Pré-Análise

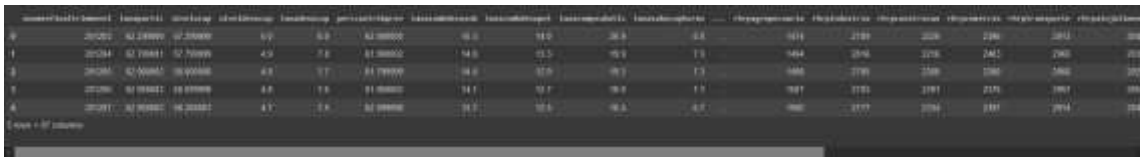
A pré-análise é a primeira etapa do processo ETL (Extract, Transform, Load). Ela é responsável por avaliar os dados brutos, identificar possíveis problemas e corrigi-los antes da transformação. A pré-análise pode incluir as seguintes tarefas:

- Verificar a integridade dos dados: verificar se os dados estão completos, consistentes e sem erros.
- Normalizar os dados: converter os dados para um formato padrão, facilitando a sua transformação e análise.
- Descontaminação dos dados: remover os dados inválidos ou irrelevantes.
- Agrupar os dados: agrupar os dados semelhantes para facilitar a sua análise.
- Resolver os conflitos de dados: identificar e resolver os conflitos de dados, como valores duplicados ou inconsistentes.

```
[ ]  1 # Visualização do DF
     2 df
```



```
[ ]  1 # Função head(): exibe as primeiras linhas do DataFrame
     2
     3 df.head()
```



```
[ ]  1 # Visualização do df de forma aleatória
     2
     3 df.sample(5)
```

```
[ ]  1 # Verificar o tipo de dado em cada coluna
     2 df.dtypes
```

```
anomesfinaltrimmovel      int64
taxapartic              float64
nivelocup               float64
niveldesocup            float64
taxadesocup             float64
                          ...
rhrpalojaliment           int64
rhrpinfcomfinimobadm      int64
rhrpadminpublica          int64
rhrpoutroservico          int64
rhrpservicodomestico      int64
Length: 87, dtype: object
```

```
[ ]  1 # Informações detalhadas do dataframe
     2 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 135 entries, 0 to 134
Data columns (total 87 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   anomesfinaltrimmovel    135 non-null    int64
 1   taxapartic              135 non-null    float64
 2   nivelocup               135 non-null    float64
 3   niveldesocup            135 non-null    float64
 4   taxadesocup             135 non-null    float64
 5   perccontribprev         135 non-null    float64
 6   taxacombdesosub         133 non-null    float64
 7   taxacombdesopot         135 non-null    float64
 8   taxacompsubutlz         133 non-null    float64
 9   taxasubocuphoras        133 non-null    float64
 10  percdesalento           135 non-null    float64
 11  populacao               135 non-null    int64
 12  pop14mais               135 non-null    int64
 13  popnaforca              135 non-null    int64
 14  popocup                 135 non-null    int64
 15  popdesocup              135 non-null    int64
 16  popforadaforca          135 non-null    int64
 17  forcaampliada           135 non-null    int64
 18  forcapotencial          135 non-null    int64
 19  desalentado             135 non-null    int64
 20  contribuinteprev        135 non-null    int64
 21  subocuphoras            133 non-null    float64
 22  empregado               135 non-null    int64
 23  empregpriv              135 non-null    int64
 24  empregprivcomcart       135 non-null    int64
 25  empregprivsemcart       135 non-null    int64
 26  domestico               135 non-null    int64
 27  domesticocomcart        135 non-null    int64
 28  domesticosemcart        135 non-null    int64
 29  empregpubl              135 non-null    int64
 30  empregpublcomcart       135 non-null    int64
 31  empregpublsemcart       135 non-null    int64
 32  estatutmilitar          135 non-null    int64
 33  empregador              135 non-null    int64
 34  empregadorcomcnpj       90 non-null     float64
 35  empregadorsemcnpj       90 non-null     float64
 36  contapropria            135 non-null    int64
 37  contapropriacomcnpj     90 non-null     float64
 38  contapropriasemcnpj     90 non-null     float64
 39  trabfamauxiliar         135 non-null    int64
 40  agropecuaria            135 non-null    int64
 41  industria               135 non-null    int64
 42  construcao              135 non-null    int64
 43  comercio                135 non-null    int64
 44  transporte              135 non-null    int64
 45  alojaliment             135 non-null    int64
 46  infcomfinimobadm        135 non-null    int64
 47  adminpublica            135 non-null    int64
 48  outroservico            135 non-null    int64
 49  servicodomestico        135 non-null    int64
 50  massahabnominaltodos    135 non-null    int64
 51  massaefetnominaltodos   135 non-null    int64
 52  rendhabnominaltodos     135 non-null    int64
```



## Backup

Será feito um backup do dataframe original para caso seja feita uma alteração errônea ou quaisquer outros problemas que se possa ocorrer na manipulação do dataframe e, assim, não precisará fazer novamente a extração dos dados no Google Drive ou na GCP.

```
[ ]  1 # Backup local do dataframe
     2 dfback1 = df.copy()
```

## Transformação

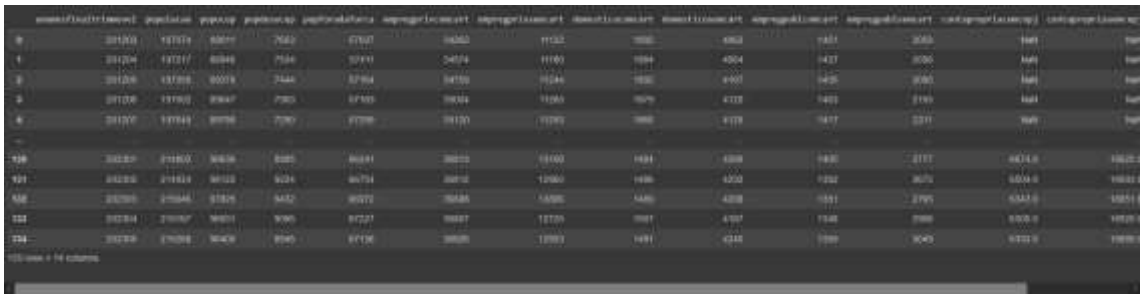
Na etapa de transformação, ocorrem várias etapas fundamentais para preparar os dados de forma adequada antes de serem carregados no destino final. Alguns passos possíveis nessa etapa:

- **Padronização:** Durante a padronização, os dados são ajustados para seguir um formato consistente. Isso pode envolver a normalização de valores, como converter datas em um formato específico, unificar nomenclaturas, aplicar regras de formatação para números, letras maiúsculas/minúsculas, entre outros. A padronização facilita a análise e comparação dos dados posteriormente.
- **Limpeza:** A etapa de limpeza é crucial para remover erros, dados incompletos ou inconsistentes. É comum encontrar dados ausentes, outliers, duplicatas ou registros corrompidos. Durante a limpeza, são aplicadas técnicas como preenchimento de valores faltantes, remoção de duplicatas, correção de erros tipográficos e a identificação de outliers para tratamento adequado.
- **Transformação de tipos de dados:** Como mencionado, diferentes sistemas de origem podem usar tipos de dados diferentes. Durante a etapa de transformação, é necessário converter os tipos de dados para um formato comum. Isso garante que os dados sejam consistentes e possam ser processados corretamente no destino.
- **Normalização:** A normalização é o processo de reorganizar e estruturar os dados para eliminar redundâncias e inconsistências. Essa técnica é comumente usada para reduzir a duplicação de dados e melhorar a eficiência de armazenamento. A normalização envolve a decomposição de dados em várias tabelas relacionadas, seguindo as regras da forma normal.
- **Validação e controle de qualidade:** Durante a transformação, é importante garantir que os dados atendam a determinados critérios de qualidade. Isso pode envolver a validação de valores em relação a regras de negócio, a identificação de valores inconsistentes ou a detecção de dados incompletos. A implementação de regras e restrições durante a transformação pode ajudar a garantir a integridade dos dados.

## Colunas a serem analisadas

```
[ ]
# Criando uma lista com as colunas que serão utilizadas para a análise

colunas = ['anomesfinaltrimmovel',
'populacao',
'popocup',
'popdesocup',
'popforadaforca',
'empregprivcomcart',
'empregprivsemcart',
'domesticocomcart',
'domesticosemcart',
'empregpublcomcart',
'empregpublsemcart',
'contapropriacomcnpj',
'contapropriasemcnpj',
'trabfamauxiliar']

# Ler o arquivo Excel e selecionar apenas as colunas de interesse
# utilizando pandas
df= pd.read_excel(path, sheet_name='TrimestreMovel', usecols=colunas)
```

```
[ ]  1 # Visualizando o novo DataFrame
     2 df
```



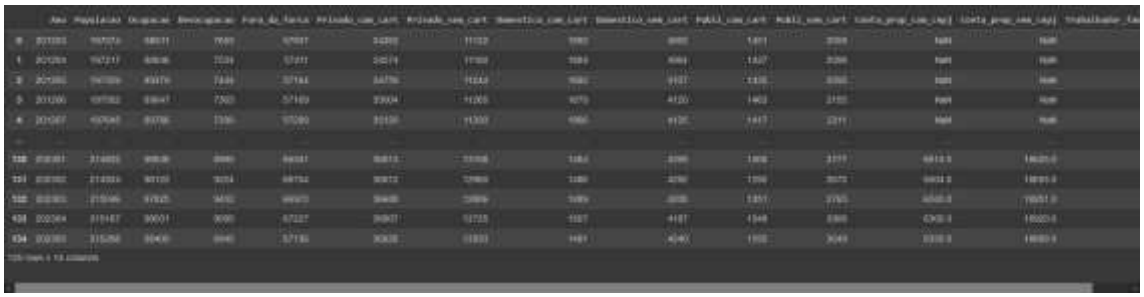
## Backup

```
[ ]  1 # Backup local do dataframe
     2 dfback2 = df.copy()
```

### Tradução

Primeiramente serão traduzidos ou renomeados os atributos do dataframe com o objetivo de melhorar o entendimento sobre os mesmos sem precisar constantemente recorrer ao dicionário de dados.

```
[ ]  1 # Renomeando colunas
     2
     3 df.rename(columns={
     4                    'anomesfinaltrimmovel':'Ano',
     5                    'populacao': 'Populacao' ,
     6                    'popocup':'Ocupacao',
     7                    'popdesocup': 'Desocupacao',
     8                    'popforadaforca': 'Fora_da_forca',
     9                    'empregprivcomcart':'Privado_com_cart',
    10                    'empregprivsemcart': 'Privado_sem_cart', # pode ter fomais e informais
    11                    'domesticocomcart': 'Domestico_com_cart',
    12                    'domesticosemcart': 'Domestico_sem_cart', # pode ter formais e informais
    13                    'empregpublcomcart':'Publi_com_cart',
    14                    'empregpublsemcart':'Publi_sem_cart',
    15                    'contapropriacomcnpj':'Conta_prop_com_cnpj',
    16                    'contapropriasemcnpj':'Conta_prop_sem_cnpj',
    17                    'trabfamauxiliar':'Trabalhador_familiar'},inplace=True)
    18
    19
    20 df
```



## Verificação

Nos passos a seguir serão feitas verificações para encontrar inconsistência nos dados. Geralmente, são feitas as seguintes tratativas: identificar e tratar valores nulos,interpolação,converter os dados para os tipos apropriados, identificar e corrigir inconsistências lógicas.

```
[ ]  1 # Verificando as informações dos atributos do dataframe
     2 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 135 entries, 0 to 134
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Ano                   135 non-null    int64  
 1   Populacao             135 non-null    int64  
 2   Ocupacao              135 non-null    int64  
 3   Desocupacao           135 non-null    int64  
 4   Fora_da_forca         135 non-null    int64  
 5   Privado_com_cart      135 non-null    int64  
 6   Privado_sem_cart      135 non-null    int64  
 7   Domestico_com_cart    135 non-null    int64  
 8   Domestico_sem_cart    135 non-null    int64  
 9   Publi_com_cart        135 non-null    int64  
 10  Publi_sem_cart        135 non-null    int64  
 11  Conta_prop_com_cnpj   90 non-null     float64
 12  Conta_prop_sem_cnpj   90 non-null     float64
 13  Trabalhador_familiar  135 non-null    int64  
dtypes: float64(2), int64(12)
memory usage: 14.9 KB
```

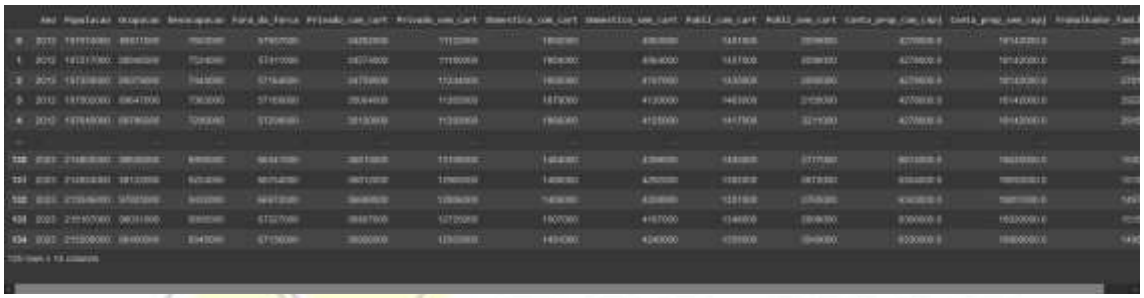
```
[ ]  1 # Verificando se há valores nulos na tabela
     2 df.isnull().sum()
```

```
Ano                     0
Populacao               0
Ocupacao                0
Desocupacao             0
Fora_da_forca           0
Privado_com_cart        0
Privado_sem_cart        0
Domestico_com_cart      0
Domestico_sem_cart      0
Publi_com_cart          0
Publi_sem_cart          0
Conta_prop_com_cnpj    45
Conta_prop_sem_cnpj    45
Trabalhador_familiar    0
dtype: int64
```

# Backup

```
[ ]  1 # Backup local do dataframe
     2 dfback3 = df.copy()
```

```
[ ]  1 # Interpolação de dados faltantes em colunas
     2
     3 '''Realizando a interpolação das colunas com valores nulos para preencher valores
     4 ausentes ou faltantes em uma série temporal ou conjunto de dados contínuos, usando
     5 valores existentes para estimar os valores ausentes.'''
     6
     7 df['Conta_prop_com_cnpj'] = df['Conta_prop_com_cnpj'].interpolate(method='linear', limit_direction='both')
     8 df['Conta_prop_sem_cnpj'] = df['Conta_prop_sem_cnpj'].interpolate(method='linear', limit_direction='both')
     9
    10 df
```





```
[ ]  1 # Checando os Nulos
     2 df.isnull().sum()
```

```
Ano                    0
Populacao              0
Ocupacao               0
Desocupacao            0
Fora_da_forca          0
Privado_com_cart       0
Privado_sem_cart       0
Domestico_com_cart     0
Domestico_sem_cart     0
Publi_com_cart         0
Publi_sem_cart         0
Conta_prop_com_cnpj    0
Conta_prop_sem_cnpj    0
Trabalhador_familiar   0
dtype: int64
```

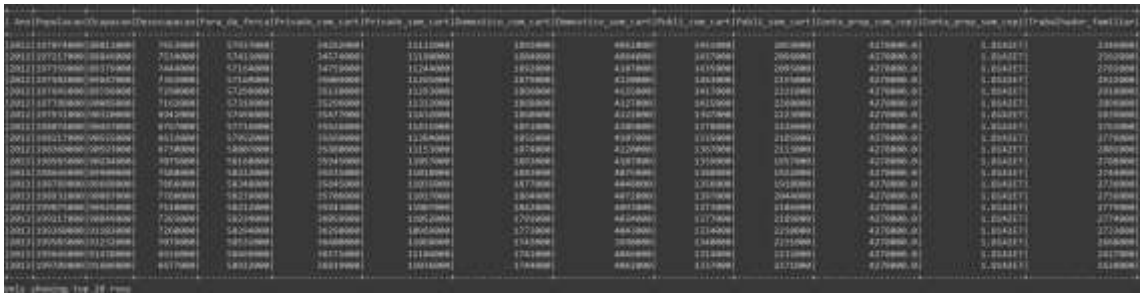
```
[ ]   1 # Verificiando a tipagem das colunas
      2 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 135 entries, 0 to 134
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Ano                   135 non-null    int64  
 1   Populacao             135 non-null    int64  
 2   Ocupacao              135 non-null    int64  
 3   Desocupacao           135 non-null    int64  
 4   Fora_da_forca         135 non-null    int64  
 5   Privado_com_cart      135 non-null    int64  
 6   Privado_sem_cart      135 non-null    int64  
 7   Domestico_com_cart    135 non-null    int64  
 8   Domestico_sem_cart    135 non-null    int64  
 9   Publi_com_cart        135 non-null    int64  
 10  Publi_sem_cart        135 non-null    int64  
 11  Conta_prop_com_cnpj   135 non-null    float64
 12  Conta_prop_sem_cnpj   135 non-null    float64
 13  Trabalhador_familiar  135 non-null    int64  
dtypes: float64(2), int64(12)
memory usage: 14.9 KB
```

```
[ ]
# Criação do schema com pyspark

from pyspark.sql.types import *
from pyspark.sql.types import StructType,StructField,StringType,IntegerType,DoubleType


schema = StructType([ \
    StructField('Ano',IntegerType(),True), \
    StructField('Populacao', IntegerType(), True), \
    StructField('Ocupacao',IntegerType(),True), \
    StructField('Desocupacao',IntegerType(),True), \
    StructField('Fora_da_forca', IntegerType(), True), \
    StructField('Privado_com_cart',IntegerType(),True), \
    StructField('Privado_sem_cart',IntegerType(),True), \
    StructField('Domestico_com_cart', IntegerType(), True), \
    StructField('Domestico_sem_cart',IntegerType(),True), \
    StructField('Publi_com_cart',IntegerType(),True), \
    StructField('Publi_sem_cart', IntegerType(), True), \
    StructField('Conta_prop_com_cnpj',DoubleType(),True), \
    StructField('Conta_prop_sem_cnpj',DoubleType(),True), \
    StructField('Trabalhador_familiar', IntegerType(), True), \
  ])

# Criar um novo DataFrame com o novo schema

df_py = spark.createDataFrame(df, schema=schema)

# Exibir o schema do novo DataFrame

df_py.printSchema()

```

```
root
 |-- Ano: integer (nullable = true)
 |-- Populacao: integer (nullable = true)
 |-- Ocupacao: integer (nullable = true)
 |-- Desocupacao: integer (nullable = true)
 |-- Fora_da_forca: integer (nullable = true)
 |-- Privado_com_cart: integer (nullable = true)
 |-- Privado_sem_cart: integer (nullable = true)
 |-- Domestico_com_cart: integer (nullable = true)
 |-- Domestico_sem_cart: integer (nullable = true)
 |-- Publi_com_cart: integer (nullable = true)
 |-- Publi_sem_cart: integer (nullable = true)
 |-- Conta_prop_com_cnpj: double (nullable = true)
 |-- Conta_prop_sem_cnpj: double (nullable = true)
 |-- Trabalhador_familiar: integer (nullable = true)
```

```
[ ]
# Mostrar o df_py

df_py.show()
```



## Carregamento da Base de Dados Tratada

## Google Cloud

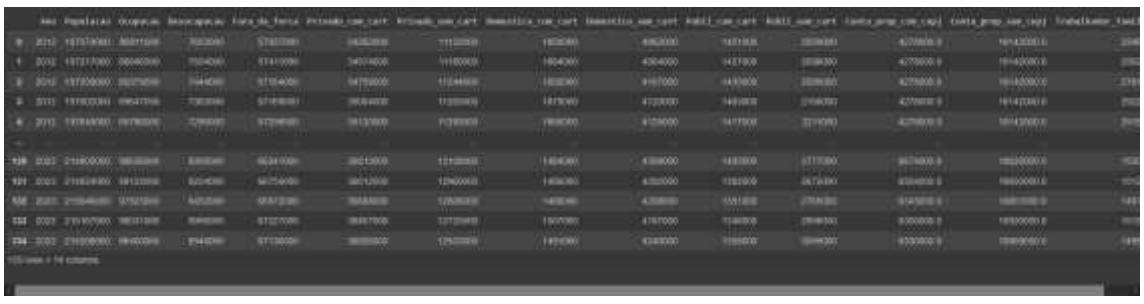
```
[ ]  1 # Exportação da tabela tratada para a bucket
     2 df.to_csv('gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Tratado/ipea_tratado.xls',index=False)
     3
     4 # Exportando csv apenas para fins de consulta no Bigquery, pois o Bigquery não lê arquivo xls
     5 df.to_csv('gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Tratado/ipea_tratado.csv',index=False)
     6
```

## Análise

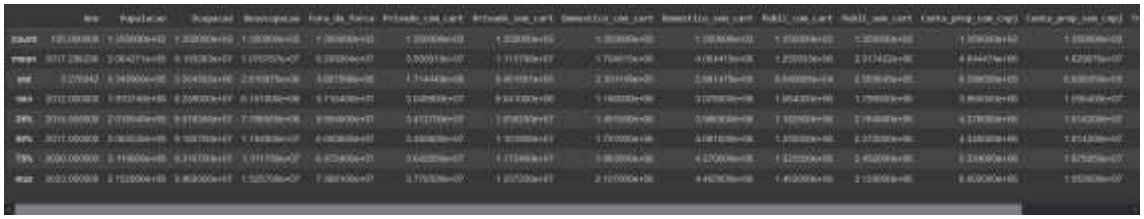


```
[ ]  1 # Criando um backup do Dataframe original tratado
     2 dfback4 = df
```

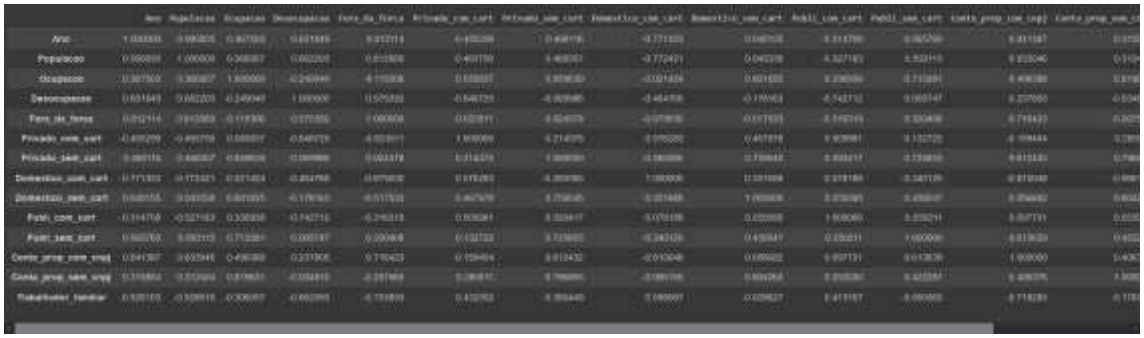
```
1 # Mostrando o backup
2 dfback4
```



```
[ ]  1 # O método describe retorna um resumo de medidas estatística descritivas
     2 # de todos os atributos.
     3 df.describe()
```



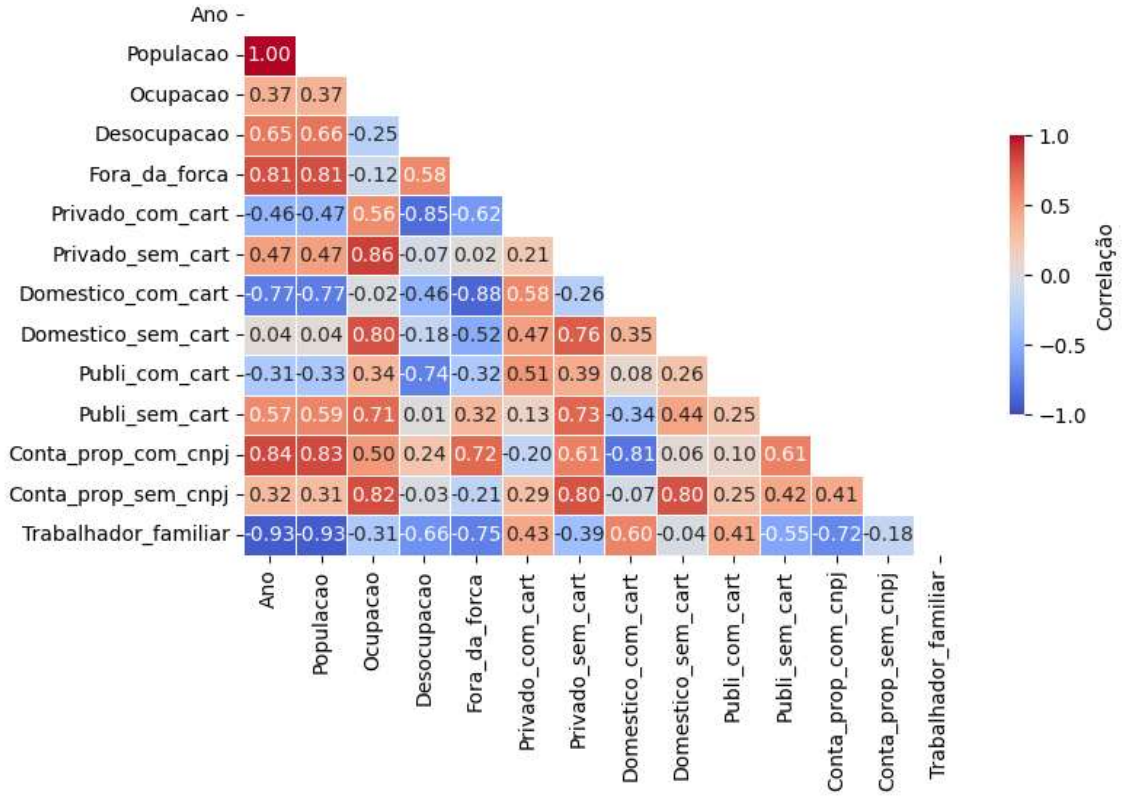
```
[ ]  1 # Calcular a matriz de correlação entre as colunas numéricas
     2 df.corr()
```




## Correlação de Pearson

Serão calculadas as correlações de Pearson de todas as variáveis e será utilizado como estratégia de análise colocar os dados em um mapa de calor para visualizar de uma maneira mais eficaz quais serão correlações relevantes.

```
[ ]
# O método corr() é responsável por calcular todas as correlações existentes
# no dataframe e utilizando o heatmap() da biblioteca seaborn, é possível
# o mapa de calor baseado nessas informações
import seaborn as sns

correlation = df.corr(method='pearson')
mascara = np.triu(np.ones_like(correlation, dtype=bool))
plt.figure(figsize = ((8, 5)))
plot = sns.heatmap(correlation,
                   mask=mascara,
                   annot = True,
                   fmt=".2f", vmax=1, center=0, vmin=-1,
                   cbar=True, cmap='coolwarm',
                   linewidths=.5,
                   cbar_kws={"shrink": .5, 'label': 'Correlação', 'orientation': 'vertical'})

plt.show()
```

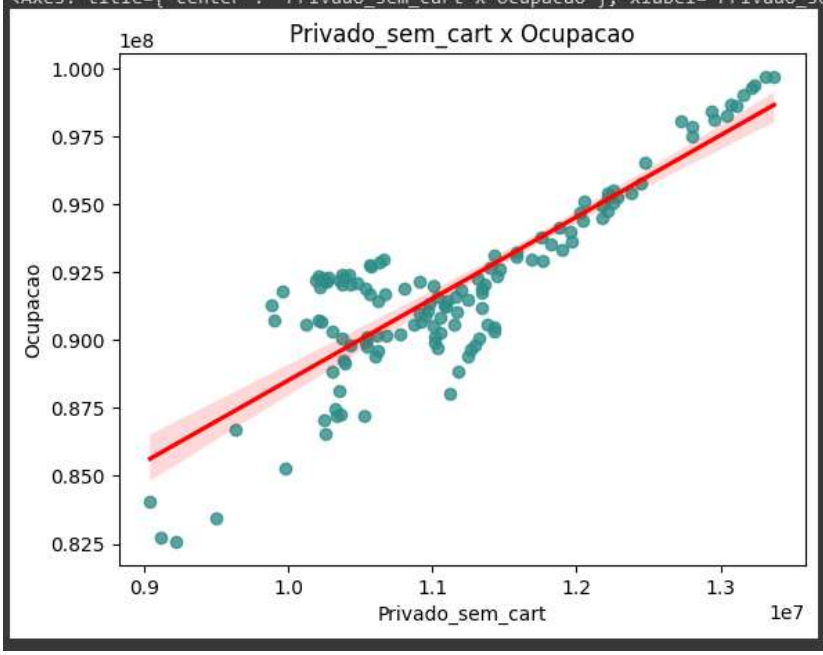


Depois de verificar as correlações, foram escolhidos os seguintes atributos para avaliar em gráficos de dispersão:

- Privado sem carteira assinada x Ocupação
- Doméstico com carteira assinada x Ocupação
- Doméstico sem carteira assinada x Ocupação
- ADM. público sem carteira assinada x Ocupação
- ADM. público com carteira assinada x Ocupação
- Trabalhador familiar x Ocupação
- Trabalhador por conta própria com cnpj x Ocupação
- Trabalhador por conta própria sem cnpj x Ocupação

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador do setor privado sem carteira X Ocupação"
     2 plt.title('Privado_sem_cart x Ocupacao')
     3 sn.regplot(df, x='Privado_sem_cart', y='Ocupacao',color='#2F8E89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Privado_sem_cart x Ocupacao'}, xlabel='Privado_sem_cart', ylabel='Ocupacao'>
```

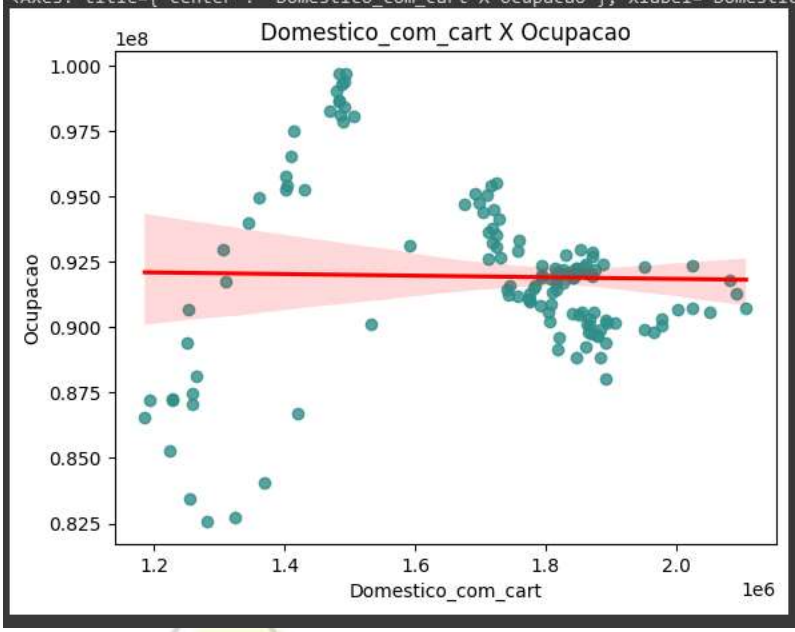



população ocup vs emprego privado sem carteira (Correlação: +0,86):

*Existe uma forte correlação positiva entre a população ocupada e o número de empregados no setor privado sem carteira de trabalho assinada. Isso sugere que, à medida que a população ocupada aumenta, o número de empregados informais no setor privado também tende a aumentar significativamente.

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador domestico com carteira x Ocupação"
     2
     3 plt.title('Domestico_com_cart X Ocupacao')
     4 sn.regplot(df, x='Domestico_com_cart', y='Ocupacao',color='#2F8E89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Domestico_com_cart X Ocupacao'}, xlabel='Domestico_com_cart', ylabel='Ocupacao'>
```

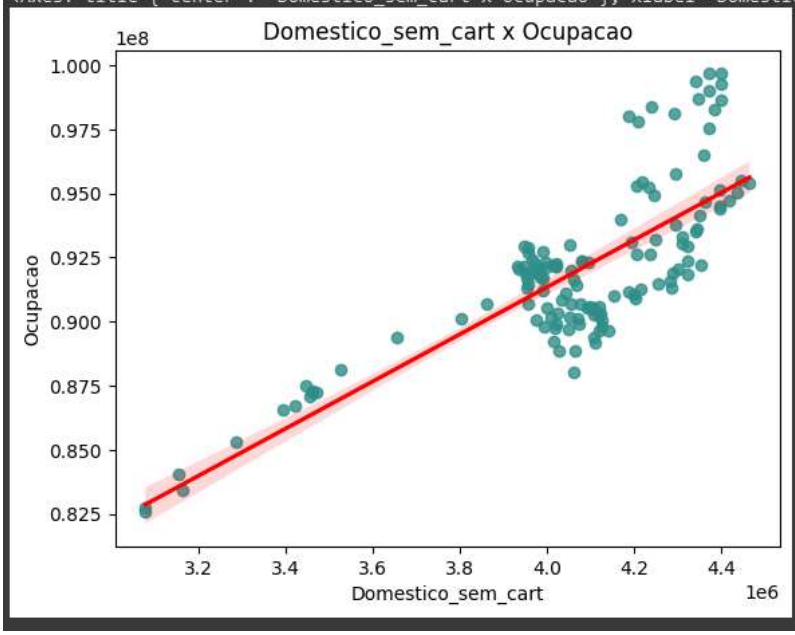



população ocup vs domestico com carteira (Correlação: -0,02):

*Não há uma correlação significativa entre a população ocupada e o número de trabalhadores domésticos com carteira de trabalho assinada.

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador domestico sem carteira x Ocupação"
     2 plt.title('Domestico_sem_cart x Ocupacao')
     3 sn.regplot(df, x='Domestico_sem_cart', y='Ocupacao',color='#2F8E89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Domestico_sem_cart x Ocupacao'}, xlabel='Domestico_sem_cart', ylabel='Ocupacao'>
```

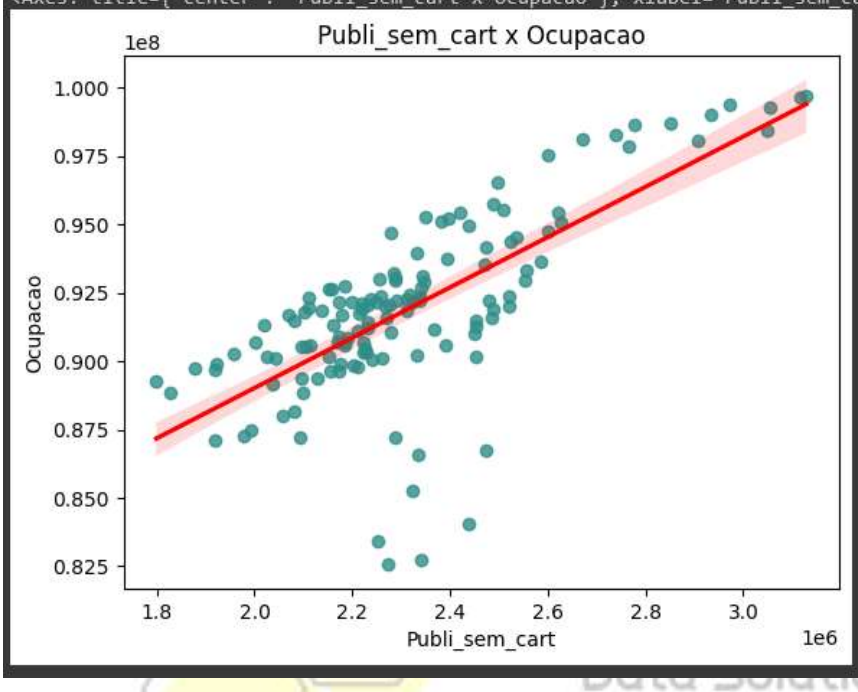


população ocup vs domestico sem carteira (Correlação: +0,80):

*Existe uma forte correlação positiva entre a população ocupada e o número de trabalhadores domésticos sem carteira de trabalho assinada. Isso sugere que, à medida que a população ocupada aumenta, o número de trabalhadores domésticos informais também tende a aumentar consideravelmente.

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador do setor público sem carteira x Ocupação"
     2 plt.title('Publi_sem_cart x Ocupacao')
     3 sn.regplot(df, x='Publi_sem_cart', y='Ocupacao',color='#2F8E89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Publi_sem_cart x Ocupacao'}, xlabel='Publi_sem_cart', ylabel='Ocupacao'>
```

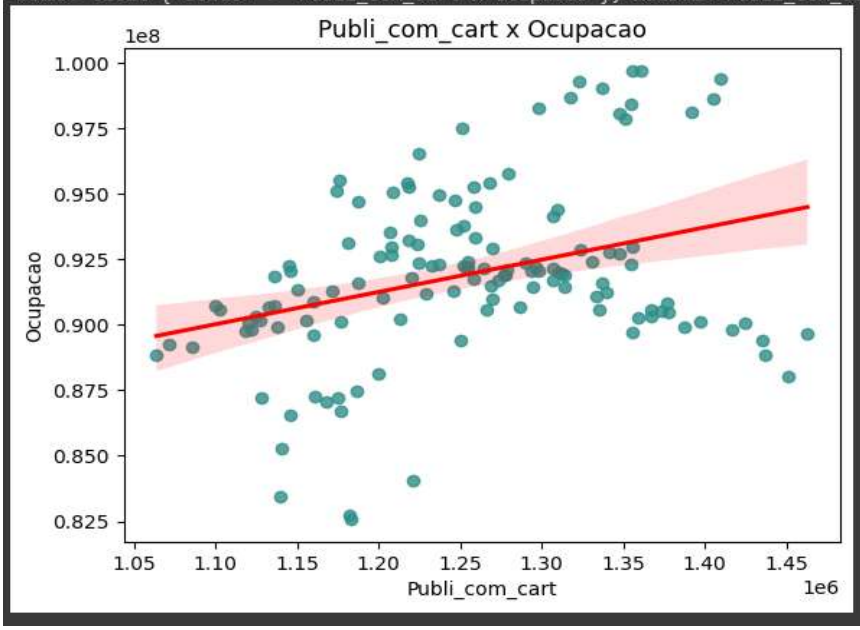



população ocup vs emprego publico sem carteira (Correlação: +0,34):

*Há uma correlação moderada positiva entre a população ocupada e o número de empregados no setor público sem carteira de trabalho assinada. Isso indica que, à medida que a população ocupada aumenta, o número de empregados no setor público também pode aumentar.

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador do setor público com carteira x Ocupação"
     2 plt.title('Publi_com_cart x Ocupacao')
     3 sn.regplot(df, x='Publi_com_cart', y='Ocupacao',color='#2F8E89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Publi_com_cart x Ocupacao'}, xlabel='Publi_com_cart', ylabel='Ocupacao'>
```

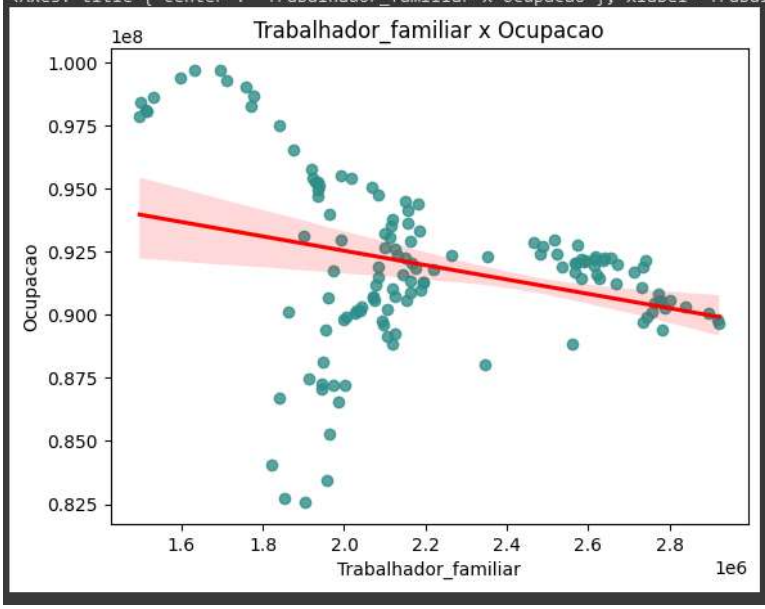



população ocup vs emprego publico com carteira (Correlação: +0,71):

*Existe uma correlação substancial entre a população ocupada e o número de empregados no setor público com carteira de trabalho assinada. À medida que a população ocupada cresce, o número de empregados no setor público também aumenta consideravelmente.

```
1 # Gráfico de regressão linear sobre Trabalhador familiar x Ocupação
2 # Trabalhador familiar auxiliar: pessoas que trabalham auxiliando familiares sem receber remuneração
3 plt.title('Trabalhador_familiar x Ocupacao')
4 sn.regplot(df, x='Trabalhador_familiar', y='Ocupacao',color='#2F8E89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Trabalhador_familiar x Ocupacao'}, xlabel='Trabalhador_familiar', ylabel='Ocupacao'>
```

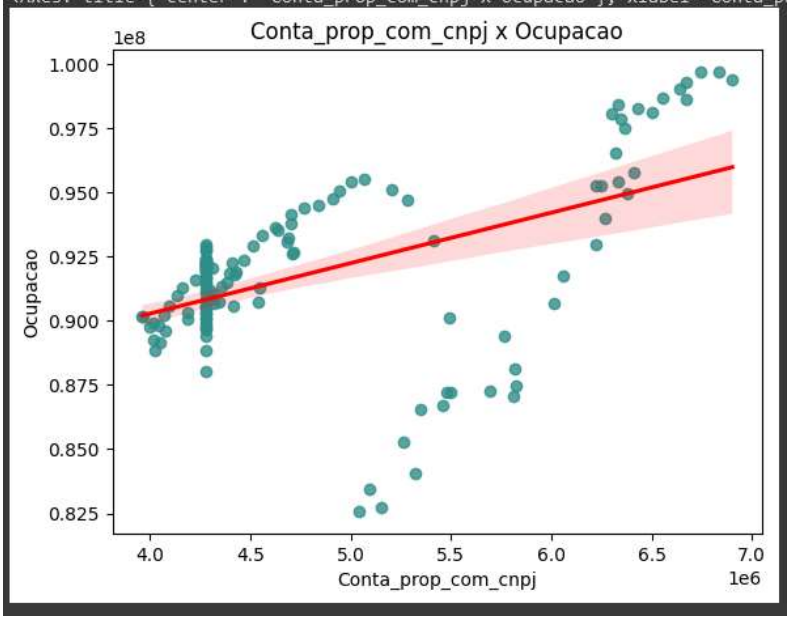



população ocup vs trabalhador familiar auxiliar (Correlação: -0,31):

*Existe uma correlação moderada negativa entre a população ocupada e o número de trabalhadores familiares auxiliares. Isso sugere que, à medida que a população ocupada aumenta, o número de trabalhadores familiares auxiliares tende a diminuir.

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador que trabalha por conta própria (com cnpj) x Ocupação"
     2 plt.title('Conta_prop_com_cnpj x Ocupacao')
     3 sn.regplot(df, x='Conta_prop_com_cnpj', y='Ocupacao',color='#2FBE89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Conta_prop_com_cnpj x Ocupacao'}, xlabel='Conta_prop_com_cnpj', ylabel='Ocupacao'>
```

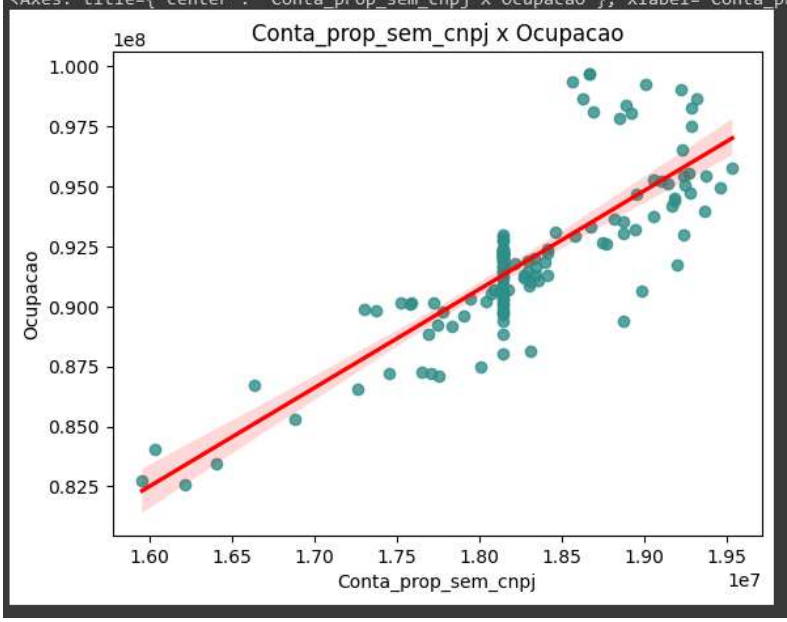



população ocup vs conta propria com cnpj (Correlação: +0,50):

*Há uma correlação moderada positiva entre a população ocupada e o número de pessoas que possuem conta própria com registro no CNPJ. Isso indica que à medida que a população ocupada aumenta, o número de pessoas que possuem uma atividade econômica formal registrada tende a aumentar.

```
[ ]  1 # Gráfico de regressão linear sobre "trabalhador que trabalha por conta própria (sem cnpj) x Ocupação"
     2 plt.title('Conta_prop_sem_cnpj x Ocupacao')
     3 sn.regplot(df, x='Conta_prop_sem_cnpj', y='Ocupacao',color='#2FBE89', line_kws={'color': 'red'})
```

```
<Axes: title={'center': 'Conta_prop_sem_cnpj x Ocupacao'}, xlabel='Conta_prop_sem_cnpj', ylabel='Ocupacao'>
```



população ocup vs conta propria sem cnpj (Correlação: +0,82):

*Existe uma forte correlação positiva entre a população ocupada e o número de pessoas que possuem conta própria sem registro no CNPJ. Isso sugere que, à medida que a população ocupada aumenta, o número de trabalhadores por conta própria informais também aumenta significativamente.

## Conclusão

O mercado de trabalho é uma área complexa e em constante evolução, onde diferentes formas de emprego coexistem e interagem. A distinção entre funcionários com carteira assinada, trabalhadores informais e outras modalidades de emprego cria uma paisagem diversificada e muitas vezes desafiadoras. Aqui estão algumas conclusões a considerar em relação a essas diferentes categorias de trabalhadores:

Os funcionários com carteira assinada geralmente desfrutam de benefícios e proteções legais mais robustos, como FGTS, aposentadoria,décimo terceiro salário, férias remuneradas e benefícios previdenciários. A relação empregador-empregado é formalizada por meio de contratos de trabalho, que delineiam direitos, deveres e responsabilidades de ambas as partes. No entanto, essa forma de emprego também pode ter suas limitações, como menor flexibilidade em termos de horário e maior burocracia. Trabalhadores informais:

Os trabalhadores informais muitas vezes não possuem contratos formais e podem estar envolvidos em atividades não regulamentadas, como trabalho temporário, subemprego ou autônomo. Embora a informalidade possa proporcionar uma maior flexibilidade e independência, os trabalhadores informais enfrentam desafios, como a falta de proteções sociais e direitos trabalhistas. A informalidade pode dificultar o acesso a crédito, planejamento financeiro e benefícios de longo prazo, como aposentadoria.

Além das categorias tradicionais, o mercado de trabalho tem visto o crescimento de outras modalidades, como trabalhadores autônomos, contratos e trabalho temporário.

Essas formas de emprego oferecem flexibilidade e oportunidades de diversificação de habilidades, mas também podem carecer de estabilidade financeira e benefícios associados ao emprego formal. O mercado de trabalho brasileiro historicamente apresenta altos níveis de informalidade, onde muitos trabalhadores atuavam sem registro em carteira ou em condições menos protegidas legalmente. Esforços têm sido feitos para reduzir essa informalidade, como a implementação de políticas para incentivar a formalização e melhorar os direitos trabalhistas


# MongoDB

Guardando os dados da base de dados em arquivos no MongoDb

```
[ ]  1 # Coloque as chaves GCP e MongoDB
     2 !pip install gcsfs
     3 !python -m pip install pymongo
```

```
[ ]  1 # Abertura de bibliotecas de manipulação e análise
     2 import pandas as pd
     3 import numpy as np
     4
     5 # Abertura de bibliotecas de conectores
     6 import os
     7 from google.cloud import storage
     8 from pymongo import MongoClient
```

## Google Cloud

Documentação: https://cloud.google.com/docs/authentication?hl=pt-br

```
[ ]  1 # CONFIGURANDO DA CHAVE DE SEGURANCA - ACESSO O PROJETO
     2 serviceAccount = '/content/symphone-project-2b6e69a15a32.json' # chave de acesso ao projeto na gcp
     3 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
```

```
[ ]  1 # Configurações Google Cloud Storage - ACESSO AO BUCKET
     2 client = storage.Client()
     3 bucket = client.get_bucket('mercado-de-trabalho-projeto')
     4 bucket.blob('ipea.xls')
     5 path = 'gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Bruto/ipea.xls'
```

```
[ ]  1 # Abertura da base de dados e cópia de segurança (bucket)
     2 path = 'gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Bruto/ipea.xls'
     3 df = pd.read_excel(path, sheet_name='TrimestreMovel')
     4 dfback = df.copy()
```

## MongoDB

```
[ ]  1 # Escolhendo a base de dados e coleção
     2 db = client['pandasmongo']
     3 collection = db['brutos_ipea']
```

```
[ ]  1 # Contagem dos documentos
     2 doc_count = collection.count_documents({})
     3 print(doc_count)
```

```
[ ]  1 # Abertura da base de dados e cópia de segurança (bucket)
     2 # Arquivo xls
     3 path = 'gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Bruto/ipea.xls'
     4 df = pd.read_excel(path, sheet_name='TrimestreMovel')
     5 dfback = df.copy()
```

```
[ ]  1 # Conversão para colocar no MongoDB
     2 # esse código saí de tabela e transforma em dicionario no Mongodb
     3 df_dict = df.to_dict("records")
     4 collection.insert_many(df_dict)
```

```
[ ]  1 # Checagem de valores no MongoDB
     2 collection.count_documents({})
```

```
[ ]  1 # Checagem da coleção do MongoDB
     2 for x in collection.find():
     3   print(x)
```

## Tratamento



```
[ ]  1 # gsutil da base de dados tratada diretamente da bucket
     2
     3 path = 'gs://mercado-de-trabalho-projeto/Base de Dados/IPEA/Tratado/ipea_tratado.csv'
     4 df = pd.read_csv(path,
     5                  encoding='ISO-8859-1',
     6                  )
```

## Carregamento


```
[ ]  1 # Carregamento da base de dados tratada no MongoDB
     2 # criando a pasta dos dados tratados no mongo
     3 db2 = client['pandasmongo']
     4 collection2 = db2['tratados_ipea']
     5 collection2.count_documents({})
```

```
[ ]  1 # Conversão de dados para MongoDB
     2 # transformando de tabela para dicionario mongo
     3 df_dict = df.to_dict("records")
     4 collection2.insert_many(df_dict)
```

```
[ ]  1 # Contagem de dados: verificação
     2 collection2.count_documents({})
```

```
[ ]  1 # Checagem da coleção
     2 for x in collection2.find():
     3   print(x)
```

## 8. REFERÊNCIAS


1 – Link para a base do CAGED “microdados antigos”:

https://basedosdados.org/dataset/562b56a3-0b01-4735-a049-eeac5681f056?table=95106d6f-e36e-4fed-b8e9-99c41cd99ecf

2 – Link para a base do CAGED “microdados de movimentações”:

https://basedosdados.org/dataset/562b56a3-0b01-4735-a049-eeac5681f056?table=2245875f-d1ef-490d-be29-4f8fb2191335

3 – Link para a base do PNAD-C “microdados”:

https://basedosdados.org/dataset/9fa532fb-5681-4903-b99d-01dc45fd527a?table=a04fc85d-908a-4393-b51d-1bd517a40210

4 – Link para a base do censo IBGE “microdados_domicilio_2010”:

https://basedosdados.org/dataset/b8e8bd62-4eb9-42f9-9ffa-b5cca093f58e?table=06165a90-ac0b-4811-9cfa-45bb2e7d47fa

5 – Link para a base do IPEA “230811_cc_60_pnadc_trim_movel_e_mensalizadas”:

https://www.ipea.gov.br/cartadeconjuntura/index.php/category/mercado-de-trabalho/