Relatório Índice das Cidades Empreendedoras 2023

César Freitas Vítor Borges Arnaldo

setembro 07, 2022

Resumo

A

Sumário

1 Introducao

Para a construção do Índice das Cidades Empreendedoras (ICE) de 2023, a manipulação e tratamento dos dados, bem como o cálculo dos indicadores, foram feitos em Python, R e SQL. Optou por utilizar diferentes linguagens de programação a fim de otimizar o trabalho com a base de dados. Os scripts realizados estão disponíveis tanto no Apêndice deste relatório quanto em um repositório criado pelos bolsistas, disponível em link.

2 Determinantes

2.1 Determinante Ambiente Regulatório

- Clique aqui para ver o script do Determinante Ambiente Regulatório.

2.1.1 Subdeterminante Tempo de Processos

- Clique aqui para ver o script do Subdeterminante Tempo de Processos.

2.1.1.1 Indicador Tempo de Viabilidade de Localização

	Informações
Fonte	https://estatistica.redesim.gov.br/tempos-abertura
Período	01/2021 ~ 12/2021
Cidades faltantes	São José do Rio Preto (SP), Maringá (PR), Jundiaí (SP), Anápolis (GO)
Variável de Interesse	QTDE. HH VIABILIDADE END
Efeito	Negativo
Problemas com a coleta	Não

Para o cálculo do indicador foi realizado a partir da média simples das variável de interesse, `QTDE. HH VIABILIDADE END`.

$$Tempo\ de\ Viabilidade\ de\ Localização = \frac{\sum_{12} Qtnd.\ Horas\ Viabilidade}{12} \tag{1}$$

Em relação à importação dos dados no site da REDESIM, para nenhum dos 12 meses analisados houve problemas quanto aos arquivos importados e a qualidade dos dados coletados.

2.1.1.2 Indicador Tempo de Registro, Cadastro e Viabilidade de Nome

	Informações
Fonte	https://estatistica.redesim.gov.br/tempos-abertura
Período	01/2021 ~ 12/2021
Cidades faltantes	São José do Rio Preto (SP), Maringá (PR), Jundiaí (SP), Anápolis (GO)
Variável de Interesse	QTDE. HH LIBERAÇÃO DBE
Efeito	Negativo

	Informações
Problemas com a coleta	Não

Assim como o anterior, o cálculo do indicador foi realizado a partir da média simples das variável de interesse, `QTDE. HH LIBERAÇÃO DBE`.

$$Tempo\ de\ Registro,\ Cadastro\ e\ Viabilidade\ de\ Nome = \frac{\sum_{12} Qtnd.\ Horas\ Liberação}{12} \tag{2}$$

Como a base de dados é a mesma do indicador anterior, os mesmo comentários são válidos quanto a qualidade e coleta dos dados.

2.1.1.3 Indicador Taxa de Congestionamento em Tribunais

	Informações
Fonte	https://paineis.cnj.jus.br/QvAJAXZfc/opendoc.htm?document=qvw_l%2FPainelCNJ.qvw&host=QVS%40neodimio03&anonymous=true&sheet=shPDPrincipal
Período	01/2021 ~ 12/2021
Cidades faltantes	-
Efeito	Negativo
Problemas com a coleta	Não

Para a coleta dos dados, ao acessar o painel de Produtividade Mensal da CNJ, foi selecionado o tópico de *Gráficos Customizados*. Os parâmetros de pesquisa selecionados foram:

- Justiça: Justiça Estadual
- Campos Agrupados: Sede Município
- Tipo de variável: Novos, Pendentes e Baixados.

Com a taxa líquida de congestionamento sendo definida como:

$$Tx.\ líquida\ de\ congestionamento = \frac{1 - baixados}{novos + pendentes} \tag{3}$$

2.1.2 Subdeterminante Tributação

- Clique aqui para ver o script do Subdeterminante Tributação.

2.1.2.1 Indicador Alíquota Interna do ICMS

	Informações
Fonte	https://siconfi.tesouro.gov.br/siconfi/index.jsf
Período	01/2021 ~ 12/2021 (Sinconfi), 2018 (PIB)
Cidades faltantes	-

	Informações
Efeito	Negativo
Variável de interesse	ICMS (Siconfi), PIB Municipal (IBGE)
Problemas com a coleta	Não

Foi utilizada a tabela Receitas Orçamentárias (Anexo I-C). Restringindo o montante das receitas filtrando-as para as 'Receitas Brutas Realizadas'. Com o indicador sendo a razão entre esse montante e o PIB Municipal[1] de 2018, o mais recente que foi disponibilizado. Além disso, pela natureza especial da cidade de Brasília (DF), ao invés de usar os dados municípais, tivemos que utilizar os por estado.

$$Alíquota\ ICMS = \frac{Receitas\ Brutas\ Realizadas_{ICMS}}{PIB\ Municipal} \tag{4}$$

2.1.2.2 Indicador Alíquota Interna do IPTU

	Informações
Fonte	https://siconfi.tesouro.gov.br/siconfi/index.jsf
Período	01/2021 ~ 12/2021 (Sinconfi), 2018 (PIB)
Cidades faltantes	-
Efeito	Negativo
Variável de interesse	IPTU (Siconfi), PIB Municipal (IBGE)
Problemas com a coleta	Não

Processo de coleta e tramento análogo ao Indicador Alíquota Interna do ICMS.

$$Alíquota\ IPTU = \frac{Receitas\ Brutas\ Realizadas_{IPTU}}{PIB\ Municipal} \tag{5}$$

2.1.2.3 Indicador Alíquota Interna do ISS

	Informações
Fonte	https://siconfi.tesouro.gov.br/siconfi/index.jsf
Período	01/2021 ~ 12/2021 (Sinconfi), 2018 (PIB)
Cidades faltantes	Carapicuíba (SP)
Efeito	Negativo
Variável de interesse	ISS (Siconfi), PIB Municipal (IBGE)
Problemas com a coleta	Não

Processo de coleta e tramento análogo ao Indicador Alíquota Interna do ICMS.

$$Alíquota\ ISS = \frac{Receitas\ Brutas\ Realizadas_{ISS}}{PIB\ Municipal} \tag{6}$$

[1] Utilizamos o *data lake* público 'Base dos dados', disponível em https://basedosdados.org, para importar esse dado.

2.1.2.4 Indicador Qualidade da Gestão Fiscal

	Informações
Fonte	https://www.firjan.com.br/ifgf/
Período	2021
Cidades faltantes	-
Efeito	Positvo
Variável de interesse	IFGF
Problemas com a coleta	Não

2.1.3 Subdeterminante Complexidade Burocrática

- Clique aqui para ver o script do Subdeterminante Complexidade Burocrática.

2.1.3.1 Indicador Simplicidade Tributária

2.1.3.2 Indicador CNDs Municipais

2.1.3.3 Indicador Atualização de Zoneamento

2.2 Determinante Infraestrutura

2.2.1 Subdeterminante Transporte Interurbano

2.2.1.1 Indicador Conectividade via Rodovias

2.2.1.2 Indicador Número de Decolagens por Ano

2.2.1.3 Indicador Distância ao Porto Mais Próximo

2.2.2 Subdeterminante Condições Urbanas

2.2.2.1 Indicador Acesso à Internet Rápida

2.2.2.2 Indicador Preço Médio do m²

2.2.2.3 Indicador Custo da Energia Elétrica

2.2.2.4 Indicador Taxa de Homicídios

2.3 Determinante Mercado

2.3.1 Subdeterminante Desenvolvimento Econômico

2.3.1.1 Indicador Índice de Desenvolvimento Humano

2.3.1.2 Indicador Crescimento Médio Real do PIB

2.3.1.3 Indicador Número de Empresas Exportadoras com Sede na Cidade

2.3.2 Subdeterminante Clientes Potenciais

2.3.2.1 Indicador PIB per capita

2.3.2.2 Indicador Proporção entre Grandes/Médias e Médias/Pequenas Empresas

2.3.2.3 Indicador Compras Públicas

2.4 Determinante Acesso a Capital

- Clique aqui para ver o script do Determinante Acesso a Capital.

2.4.1 Subdeterminante Capital Disponível

- Clique aqui para ver o script do Subdeterminante Capital Disponível.

2.4.1.1 Indicador Operações de Crédito por Município

2.4.1.2 Indicador Proporção Relativa de Capital de Risco

2.4.1.3 Indicador Capital Poupado per capita

2.5 Determinante Inovação

- Clique aqui para ver o script do Determinante Inovação.

2.5.1 Subdeterminante Inputs

- Clique aqui para ver o script do Subdeterminante Inputs.

2.5.1.1 Indicador Proporção de Mestres e Doutores em C&T

2.5.1.2 Indicador Proporção de Funcionários em C&T

2.5.1.3 Indicador Média de Investimentos do BNDES e da FINEP

2.5.1.4 Indicador Infraestrutura Tecnológica

2.5.1.5 Indicador Contratos de Concessão

2.5.2 Subdeterminante Outputs

- Clique aqui para ver o script do Subdeterminante Outputs.

2.5.2.1 Indicador Patentes

2.5.2.2 Indicador Tamanho da Indústria Inovadora

2.5.2.3 Indicador Tamanho da Economia Criativa

2.5.2.4 Indicador Tamanho das Empresas TIC

2.6 Determinante Capital Humano

2.6.1 Subdeterminante Acesso e Qualidade da Mão de Obra Básica

2.6.1.1 Indicador Nota do Ideb

2.6.1.2 Indicador Proporção de Adultos com Pelo Menos o Ensino Médio Completo

2.6.1.3 Indicador Taxa Líquida de Matrícula no Ensino Médio

2.6.1.4 Indicador Nota Média no Enem

2.6.1.5 Indicador Proporção de Matriculados no Ensino Técnico e Profissionalizante

2.6.2 Subdeterminante Acesso e Qualidade da Mão de Obra Qualificada

2.6.2.1 Indicador Proporção de Adultos com Pelo Menos o Ensino Superior Completo

2.6.2.2 Indicador Proporção de Alunos Concluintes em Cursos de Alta Qualidade

2.6.2.3 Indicador Custo Médio de Salários de Dirigentes

2.7 Determinante Cultura

2.7.1 Subdeterminante Iniciativa

2.7.1.1 Indicador Pesquisas Empreendedora

2.7.1.2 Indicador Pesquisas Empreendedorismo

2.7.1.3 Indicador Pesquisas MEI

2.7.2 Subdeterminante Instituições

2.7.2.1 Indicador Pesquisas SEBRAE

2.7.2.2 Indicador Pesquisas Franquia

2.7.2.3 Indicador Pesquisas Simples Nacional

2.7.2.4 Indicador Pesquisas SENAC

3 Análise de Componentes Principais

4 Correlação com o ICE2022

5 Efeito Parcial da Cultura

6 Ranking

Apêndice

Scripts

```
import pandas as pd
import numpy as np
from funcs import *
from functools import reduce
import basedosdados as bd
```

Amostra

```
# Leitura da base
df = pd.read_excel('POP2021_20220711.xls', 'Municípios')
df = df.head(-32)
df.columns = df.iloc[0]
df = df.drop(0)
df.index = range(len(df))
for i in range(len(df)):
    if type(df.iloc[i]['POPULAÇÃO ESTIMADA']) == str:
        pop = int(df.iloc[i]['POPULAÇÃO ESTIMADA'].split('(')[0].replace('.',''))
        df.at[i, 'POPULAÇÃO ESTIMADA'] = pop
top100 = df.sort_values(by=['POPULAÇÃO ESTIMADA'], ascending=False).head(101)

# Tratando a base
am = pd.read_excel('Amostra.xlsx')
for i in range(len(am)):
    name = am['Município'][i].split('-')[0].strip()
    am.at[i,'Município'] = name

def Union(lst1, lst2):
    final_list = list(set(lst1) | set(lst2))
    return final_list
```

```
final_sample = Union(am['Município'],top100['NOME DO MUNICÍPIO'])

for i in am['Município']:
    if i not in list(top100['NOME DO MUNICÍPIO']):
        print(i)

# Criando o arquivo
top100.to_csv('100-municipios.csv', index=False)
```

Determinante Ambiente Regulatório

- Para retornar ao relatório do Determinante Ambiente Regulatório clique aqui.

```
# Criando o ambiente que criará o determinante
ambiente = {}
```

Subdeterminante Tempo de Processos

- Para retornar ao relatório do Subdeterminante Tempo de Processos clique aqui.

```
## Indicando o Subdeterminante
subdet = 'Tempo de Processos'

############ Indicador Indicador Tempo de Viabilidade de Localização ############

# Looping para ler os 12 meses da variável de tempo de abertura da REDESIM
for i in list(range(1,13)):
    globals()[f"indicador_{i}"] = pd.read_excel(f'DETERMINANTE AMBIENTE REGULATÓRIO/REDESIM/tempos-ab
                                                header=1, usecols="I,R,AA,AB")
    pdList = []
    pdList.extend(value for name, value in locals().items() if name.startswith("indicador"))
    indicador = pd.concat(pdList, axis = 0)

# Tratando a base
indicador['Município'] = indicador['MUNICÍPIO'].str.upper().str.normalize('NFKD').str.encode('ascii',
indicador = database.merge(indicador, how='left', on=['Município', 'UF'])
indicador['Tempo de Viabilidade de Localização'] = indicador['QTDE.  HH VIABILIDADE END']

########## Indicador Tempo de Registro, Cadastro e Viabilidade de Nome ##########

indicador['Tempo de Registro, Cadastro e Viabilidade de Nome'] = indicador['QTDE. HH. LIBERAÇÃO DBE']
indicador = indicador.groupby(['Município','UF']).mean()
indicador = indicador.fillna(0)
```

```
del indicador['QTDE.  HH VIABILIDADE END']
del indicador['QTDE. HH. LIBERAÇÃO DBE']

################ Indicador Taxa de Congestionamento em Tribunais #################

var = ['novos','baixados','pendentes']
for i in list(range(0,3)):
    globals()[f"indicador_pro_{i}"] = pd.read_excel(f'DETERMINANTE AMBIENTE REGULATÓRIO/CNJ/{var[i]}_
    pd_List = []
    pd_List.extend(value for name, value in locals().items() if name.startswith("indicador_pro_"))
    indicador_pro = pd.concat(pd_List, axis = 0)

indicador_pro['Município'] = indicador_pro['Tribunal município'].str.upper().str.normalize('NFKD').st
indicador_pro = database.merge(indicador_pro, how='left',on='Município')
indicador_pro = indicador_pro.pivot_table(index='Município', columns='Tipo variável', values='Indicad
indicador_pro['Taxa de Congestionamento em Tribunais'] = (1-(indicador_pro['BAIXADOS']/(indicador_pro

del indicador_pro['BAIXADOS']
del indicador_pro['NOVOS']
del indicador_pro['PENDENTES']

subdet_tempo = indicador.reset_index().merge(indicador_pro, how='inner', on='Município').set_index(['
subdet_tempo['Cod.IBGE'] = subdet_tempo['Cod.IBGE'].astype(str).str[:7]
amostra['Cod.IBGE'] = amostra['Cod.IBGE'].astype(str)
subdet_tempo = subdet_tempo.merge(amostra, how='right', on='Cod.IBGE')
interesse = ['NOME DO MUNICÍPIO','UF','Tempo de Viabilidade de Localização',
             'Tempo de Registro, Cadastro e Viabilidade de Nome','Taxa de Congestionamento em Tribuna
subdet_tempo = subdet_tempo[interesse].rename(columns={'NOME DO MUNICÍPIO':'Município'})
subdet_tempo = subdet_tempo.set_index(['Município','UF'])

subdet_tempo['Tempo de Viabilidade de Localização'] = negative(subdet_tempo['Tempo de Viabilidade de
subdet_tempo['Tempo de Registro, Cadastro e Viabilidade de Nome'] = negative(subdet_tempo['Tempo de R
subdet_tempo['Taxa de Congestionamento em Tribunais'] = negative(subdet_tempo['Taxa de Congestionamen

# Tratamentos de para os missing e outliers
missing_data(subdet_tempo)
extreme_values(subdet_tempo)

# Criando o subdeterminante
create_subindex(subdet_tempo, subdet)
ambiente[subdet] = subdet_tempo
```

Subdeterminante Tributação

- Para retornar ao relatório do Subdeterminante Tributação clique aqui.

```
# Indicando o Subdeterminante
subdet = 'Tributação'

## Tratamento inicial com os dados do SINCONFI
sinconfi_mun = pd.read_csv("DETERMINANTE AMBIENTE REGULATÓRIO/Sinconfi/finbra_mun.csv",
                           encoding='ISO-8859-1', sep=';', decimal=',')
sinconfi_uf = pd.read_csv("DETERMINANTE AMBIENTE REGULATÓRIO/Sinconfi/finbra_uf.csv",
                          encoding='ISO-8859-1', sep=';', decimal=',')
base = '`basedosdados.br_ibge_pib.municipio`'
project_id = 'double-balm-306418'
var = ('id_municipio, pib')
database = database.reset_index()
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = f'SELECT {var} FROM {base} WHERE ano = 2019 AND id_municipio IN {cod_ibge}'
pib_mun = bd.read_sql(query=query,billing_project_id=project_id)

# Função para calcular os indicadores de alíquota
def sinconfi(df1,df2,pib,imposto,var):
    df_mun = df1[df1['Conta'] == var]
    df_mun = df_mun[df_mun['Coluna'] == 'Receitas Brutas Realizadas']
    df_mun['Cod.IBGE'] = df_mun['Cod.IBGE'].astype(np.int64)
    df_mun = database.merge(df_mun, how='left', on = ['Cod.IBGE','UF'])
    df_mun = df_mun[['Município','UF','Valor']]
    df_mun = df_mun[(df_mun['Município'] != 'BRASILIA')]

    df_uf = df2[df2['Conta'] == var]
    df_uf = df_uf[df_uf['UF'] == 'DF']
    df_uf = df_uf[df_uf['Coluna'] == 'Receitas Brutas Realizadas']
    df_uf['Município'] = ['BRASILIA']
    df_uf = df_uf[['Município','UF','Valor']]

    pib = pib.rename(columns={'id_municipio':'Cod.IBGE'}).astype(np.int64)
    pib = database.merge(pib, how='left', on=['Cod.IBGE']).set_index(['Município','UF'])
    df = df_mun.append(df_uf).merge(pib, how='left',on=['Município','UF']).reset_index(drop=True)
    df[f'Alíquota Interna do {imposto}'] = df['Valor']/df['pib']

    globals()[f'df_{imposto}'] = df.drop(['Valor','pib','Cod.IBGE'], axis=1)

######################## Indicador Alíquota Interna do ICMS ########################
sinconfi(sinconfi_mun,sinconfi_uf,pib_mun,imposto='ICMS',var='1.1.1.8.02.0.0 - Impostos sobre a Produ

######################## Indicador Alíquota Interna do IPTU ########################
```

```
sinconfi(sinconfi_mun,sinconfi_uf,pib_mun,imposto='IPTU',var='1.1.1.8.01.1.0 - Imposto sobre a Propri

###################### Indicador Alíquota Interna do ISS ######################
sinconfi(sinconfi_mun,sinconfi_uf,pib_mun,imposto='ISS',var='1.1.1.8.02.3.0 - Imposto sobre Serviços
df_ISS = df_ISS.fillna(0)

##################### Indicador Qualidade de Gestão Fiscal #####################
df_firjan = pd.read_excel("DETERMINANTE AMBIENTE REGULATÓRIO/Firjan/Firjan - Evolucao por Indicador 2
df_firjan['Município'] = df_firjan['Município'].str.upper().str.normalize('NFKD').str.encode('ascii',
df_firjan = database.merge(df_firjan, how='left', on = ['Município','UF']).fillna(0)
df_firjan = df_firjan.set_index(['Município','UF'])
df_firjan = df_firjan['IFGF 2020'].to_frame()
df_firjan = df_firjan.replace(to_replace=r'nd',value=0,regex=True)
df_firjan = df_firjan.rename(columns={'IFGF 2020':'Qualidade de Gestão Fiscal'})

dfs = [df_ICMS,df_IPTU,df_ISS,df_firjan]

subdet_tri = reduce(lambda left,right: pd.merge(left, right, on=['Município','UF'],
                                                how='outer'), dfs)
subdet_tri = subdet_tri.merge(database, how='right',on=['Município','UF'])
subdet_tri['Cod.IBGE'] = subdet_tri['Cod.IBGE'].astype(str)
subdet_tri = subdet_tri.merge(amostra, how='right', on='Cod.IBGE')
interesse=['NOME DO MUNICÍPIO','UF_x','Alíquota Interna do ICMS','Alíquota Interna do IPTU',
           'Alíquota Interna do ISS','Qualidade de Gestão Fiscal']
subdet_tri = subdet_tri[interesse]
subdet_tri = subdet_tri.rename(columns={'UF_x':'UF','NOME DO MUNICÍPIO':'Município'})
subdet_tri = subdet_tri.set_index(['Município','UF'])

# Tratamento para os indicadores com impacto negativo no índice
subdet_tri.iloc[:,0] = negative(subdet_tri.iloc[:,0])
subdet_tri.iloc[:,1] = negative(subdet_tri.iloc[:,1])
subdet_tri.iloc[:,2] = negative(subdet_tri.iloc[:,2])

# Tratamentos de para os missing e outliers
missing_data(subdet_tri)
extreme_values(subdet_tri)

# Criando o subdeterminante
create_subindex(subdet_tri, subdet)
ambiente[subdet] = subdet_tri
```

Subdeterminante Complexidade Burocrática

- Para retornar ao relatório do Subdeterminante Complexidade Burocrática clique aqui.

```
# Indicando o Subdeterminante
subdet = 'Complexidade Burocrática'

##################### Indicador de Simplicidade Tributária ######################
### Sinconfi: selecionando tributos
tributos = ['1.1.1.2.01.0.0 - Imposto sobre a Propriedade Territorial Rural',
            '1.1.1.3.00.0.0 - Impostos sobre a Renda e Proventos de Qualquer Natureza',
            '1.1.1.3.03.1.0 - Imposto sobre a Renda - Retido na Fonte - Trabalho',
            '1.1.1.3.03.2.0 - Imposto sobre a Renda - Retido na Fonte - Capital',
            '1.1.1.3.03.3.0 - Imposto sobre a Renda - Retido na Fonte - Remessa ao Exterior',
            '1.1.1.3.03.4.0 - Imposto sobre a Renda - Retido na Fonte - Outros Rendimentos',
            '1.1.1.8.01.1.0 - Imposto sobre a Propriedade Predial e Territorial Urbana',
            '1.1.1.8.01.4.0 - Imposto sobre Transmissão ¿Inter Vivos¿ de Bens Imóveis e de Direitos R
            '1.1.1.8.02.1.0 - Imposto sobre Operações Relativas à Circulação de Mercadorias e sobre P
            '1.1.1.8.02.3.0 - Imposto sobre Serviços de Qualquer Natureza',
            '1.1.1.8.02.4.0 - Adicional ISS - Fundo Municipal de Combate à Pobreza',
            '1.1.1.8.02.5.0 - Imposto sobre Vendas a Varejo de Combustíveis Líquidos e Gasosos (IVVC)
            '1.1.2.1.00.0.0 - Taxas pelo Exercício do Poder de Polícia',
            '1.1.2.1.01.0.0 - Taxas de Inspeção, Controle e Fiscalização',
            '1.1.2.1.02.0.0 - Taxas de Fiscalização das Telecomunicações',
            '1.1.2.1.03.0.0 - Taxa de Controle e Fiscalização de Produtos Químicos',
            '1.1.2.1.04.0.0 - Taxa de Controle e Fiscalização Ambiental',
            '1.1.2.1.05.0.0 - Taxa de Controle e Fiscalização da Pesca e Aquicultura',
            '1.1.2.2.01.0.0 - Taxas pela Prestação de Serviços',
            '1.1.2.2.02.0.0 - Emolumentos e Custas Judiciais',
            '1.1.3.8.00.0.0 - Contribuição de Melhoria - Específica de Estados, DF e Municípios',
            '1.2.1.0.01.0.0 - Contribuição para o Financiamento da Seguridade Social - COFINS',
            '1.2.1.0.02.0.0 - Contribuição Social sobre o Lucro Líquido - CSLL',
            '1.2.1.0.03.0.0 - Contribuições para o Regime Geral de Previdência Social - RGPS',
            '1.2.1.0.04.1.0 - Contribuição Patronal de Servidor Ativo Civil para o RPPS',
            '1.2.1.0.04.2.0 - Contribuição do Servidor Ativo Civil para o RPPS',
            '1.2.1.0.04.3.0 - Contribuição do Servidor Inativo para o RPPS',
            '1.2.1.0.04.4.0 - Contribuição do Pensionista para o RPPS',
            '1.2.1.0.04.5.0 - Contribuição Patronal para o RPPS Oriunda de Sentenças Judiciais',
            '1.2.1.0.04.6.0 - Contribuição do Servidor Ativo ao RPPS Oriunda de Sentenças Judiciais',
            '1.2.1.0.04.7.0 - Contribuição do Servidor Inativo ao RPPS Oriunda de Sentenças Judiciais
            '1.2.1.0.04.8.0 - Contribuição do Pensionista ao RPPS Oriunda de Sentenças Judiciais',
            '1.2.1.0.06.1.0 - Contribuição para os Fundos de Assistência Médica - Policiais Militares
            '1.2.1.0.06.2.0 - Contribuição para os Fundos de Assistência Médica dos Bombeiros Militar
            '1.2.1.0.06.3.0 - Contribuição para os Fundos de Assistência Médica dos Servidores Civis'
            '1.2.1.0.06.9.0 - Contribuição para os Fundos de Assistência Médica de Outros Beneficiári
            '1.2.1.0.09.0.0 - Contribuição para os Programas de Integração Social e de Formação do Pa
            '1.2.1.0.10.0.0 - Cota-Parte da Contribuição Sindical',
```

```
            '1.2.1.0.11.0.0 - Contribuições Referentes ao Fundo de Garantia do Tempo de Serviço - FGT
            '1.2.1.0.12.0.0 - Contribuição Social do Salário-Educação',
            '1.2.1.0.99.0.0 - Outras Contribuições Sociais',
            '1.2.1.8.01.1.0 - Contribuição Previdenciária para Amortização do Déficit Atuarial',
            '1.2.1.8.01.2.0 - Contribuição Patronal dos Servidores Civis Inativos',
            '1.2.1.8.01.3.0 - Contribuição Patronal dos Pensionistas Civis',
            '1.2.1.8.02.2.0 - Contribuição do Militar Ativo',
            '1.2.1.8.02.3.0 - Contribuição do Militar Inativo',
            '1.2.2.8.00.0.0 - Contribuições Econômicas Específicas de EST/DF/MUN',
            '1.2.3.0.00.0.0 - Contribuições para Entidades Privadas de Serviço Social e de Formação P
            '1.2.4.0.00.0.0 - Contribuição para o Custeio do Serviço de Iluminação Pública',
            '1.1.1.0.00.0.0 - Impostos',
            '1.1.2.0.00.0.0 - Taxas',
            '1.2.0.0.00.0.0 - Contribuições']

iv = ['1.1.1.2.01.0.0 - Imposto sobre a Propriedade Territorial Rural',
      '1.1.1.3.03.0.0 - Imposto sobre a Renda - Retido na Fonte',
      '1.1.1.8.01.1.0 - Imposto sobre a Propriedade Predial e Territorial Urbana',
      '1.1.1.8.01.4.0 - Imposto sobre Transmissão ¿Inter Vivos¿ de Bens Imóveis e de Direitos Reais s
      'TOTAL DAS RECEITAS (III) = (I + II)']

# Função para criar o IHH com base nos tributos
def sinconfi_ihh(df1,df2):
    df_mun = df1.query('Conta in @tributos')
    df_mun['Cod.IBGE'] = df_mun['Cod.IBGE'].astype(np.int64)
    df_mun = database.merge(df_mun, how='left', on = ['Cod.IBGE','UF'])
    df_mun = df_mun[['Município','UF','Conta','Valor']]

    df_uf = df2.query('Conta in @tributos')
    df_uf = df_uf[df_uf['UF'] == 'DF']
    df_uf['Município'] = ['BRASILIA'] * len(df_uf)
    df_uf = df_uf[['Município','UF','Conta','Valor']]

    df_ihh = df_mun.append(df_uf).reset_index(drop=True)
    df_ihh = df_ihh.pivot_table(index=['Município','UF'], columns='Conta', values='Valor',
                                aggfunc=np.sum,fill_value=0)

    df_ihh['Total I + T + C'] = df_ihh['1.1.1.0.00.0.0 - Impostos'] + df_ihh['1.1.2.0.00.0.0 - Taxas'
    del df_ihh['1.1.1.0.00.0.0 - Impostos']
    del df_ihh['1.1.2.0.00.0.0 - Taxas']
    del df_ihh['1.2.0.0.00.0.0 - Contribuições']
    df_ihh = df_ihh.apply(lambda x: x/df_ihh['Total I + T + C'])
    df_ihh = df_ihh.apply(np.square)
```

```
    del df_ihh['Total I + T + C']
    df_ihh['IHH'] = df_ihh.sum(axis=1)

    globals()['df_ihh'] = df_ihh['IHH'].to_frame()

sinconfi_ihh(sinconfi_mun, sinconfi_uf)

# Função para criar o iv com base nos tributos e receitas
def sinconfi_iv(df1,df2):
    df1 = df1.query('Conta in @iv')
    df1 = df1[df1['Coluna'] == 'Receitas Brutas Realizadas']
    df1['Cod.IBGE'] = df1['Cod.IBGE'].astype(np.int64)
    df1 = database.merge(df1, how='left', on = ['Cod.IBGE','UF'])
    df1 = df1[['Município','UF','Conta','Valor']].dropna()

    df2 = df2.query('Conta in @iv')
    df2 = df2[df2['Coluna'] == 'Receitas Brutas Realizadas']
    df2 = df2[df2['UF'] == 'DF']
    df2['Município'] = ['BRASILIA'] * len(df2)
    df2 = df2[['Município','UF','Conta','Valor']]

    df3 = df1.append(df2).reset_index(drop=True)
    df3 = df3.pivot_table(index=['Município','UF'], columns='Conta', values='Valor',
                          fill_value=0, aggfunc=np.sum)
    df3['Total Impostos'] = df3['1.1.1.2.01.0.0 - Imposto sobre a Propriedade Territorial Rural'] + d
    del df3['1.1.1.2.01.0.0 - Imposto sobre a Propriedade Territorial Rural']
    del df3['1.1.1.3.03.0.0 - Imposto sobre a Renda - Retido na Fonte']
    del df3['1.1.1.8.01.1.0 - Imposto sobre a Propriedade Predial e Territorial Urbana']
    del df3['1.1.1.8.01.4.0 - Imposto sobre Transmissão ¿Inter Vivos¿ de Bens Imóveis e de Direitos R
    df3['Total_receitas'] = df3['TOTAL DAS RECEITAS (III) = (I + II)']
    df3['ind_v'] = df3['Total Impostos']/df3['Total_receitas']

    globals()['df_iv'] = df3['ind_v'].to_frame()

sinconfi_iv(sinconfi_mun, sinconfi_uf)

ind_simpli_tri = df_ihh.merge(df_iv, how='left', on=['Município','UF'])

ind_simpli_tri['Simplicidade Tributária'] = ind_simpli_tri['IHH']*ind_simpli_tri['ind_v']

ind_simpli_tri = ind_simpli_tri.merge(database, how='right',on=['Município','UF'])
ind_simpli_tri['Cod.IBGE'] = ind_simpli_tri['Cod.IBGE'].astype(str)
subdet_complexidade = ind_simpli_tri.merge(amostra, how='right', on='Cod.IBGE')
```

```
interesse=['NOME DO MUNICÍPIO','UF_x','Cod.IBGE','Simplicidade Tributária']
subdet_complexidade = subdet_complexidade[interesse]
subdet_complexidade = subdet_complexidade.rename(columns={'UF_x':'UF',
                                                          'NOME DO MUNICÍPIO':'Município'})

########################## Indicador CNDs Municipais ############################
df_cnd = pd.read_excel('DETERMINANTE AMBIENTE REGULATÓRIO/cnds_municipais.xlsx',
                       usecols='A:C')
subdet_complexidade = subdet_complexidade.merge(df_cnd, how='right', on=['Município','UF'])

###################### Indicador Atualização de Zoneamento ######################
df_zoneamento = pd.read_excel('Arquivos ICE - 23/Ind_Originais_ICE_2022.xlsx', header=5,
                        usecols="B:C,O")

df_zoneamento = df_zoneamento.rename(columns={'2018.1':'Atualização de Zoneamento',
                                              'Ano':'Município'})
df_zoneamento['Atualização de Zoneamento'] = df_zoneamento['Atualização de Zoneamento'] + 1
df_zoneamento['Atualização de Zoneamento'] = np.where(df_zoneamento['Atualização de Zoneamento']==1,

subdet_complexidade = subdet_complexidade.merge(df_zoneamento, how='right', on=['Município','UF'])
subdet_complexidade = subdet_complexidade.set_index(['Município','UF'])
del subdet_complexidade['Cod.IBGE']

# Tratamento para os indicadores com impacto negativo no índice
subdet_complexidade.iloc[:,2] = negative(subdet_complexidade.iloc[:,2])

# Tratamentos de para os missing e outliers
missing_data(subdet_complexidade)
extreme_values(subdet_complexidade)

# Criando o subdeterminante
create_subindex(subdet_complexidade, subdet)
ambiente[subdet] = subdet_complexidade

################# Criando o determinante de Ambiente Regulatório #################
ambiente = pd.concat(ambiente, axis=1)
create_detindex(ambiente, 'Ambiente Regulatório')

ambiente.to_csv('DETERMINANTES/det-AMBIENTE REGULATÓRIO.csv')
```

Determinante Infraestrutura

Determinante Mercado

Determinante Acesso a Capital

- Para retornar ao relatório do Determinante Acesso a Capital clique aqui.

```
# Criando o ambiente que criará o determinante
acesso_capital = {}
```

Subdeterminante Capital Disponível

- Para retornar ao relatório do Subdeterminante Capital Disponível clique aqui.

```
# Indicando o subdeterminante
subdet = 'Capital Disponível'

### 2.5.1. e 2.5.3.

base = '`basedosdados.br_ibge_pib.municipio`'
project_id = 'double-balm-306418'
var = ('id_municipio, pib')
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = f'SELECT {var} FROM {base} WHERE ano = 2019 AND id_municipio IN {cod_ibge}'
pib_mun = bd.read_sql(query=query,billing_project_id=project_id)
pib_mun = pib_mun.rename(columns={'id_municipio':'Cod.IBGE'})

################## Indicador Operações de Crédito por Município ##################
df_bcb = pd.read_excel('DETERMINANTE ACESSO A CAPITAL/202112_ESTBAN.xlsx',
                       header=2, usecols="B,D,V,AQ,AT")
df_bcb = df_bcb.rename(columns={'MUNICIPIO':'Município'})
df_bcb = database.merge(df_bcb, how='left',on=['Município','UF'])
df_bcb = df_bcb.groupby(by=['Município','UF','Cod.IBGE','pop_est']).sum()
df_bcb = df_bcb.reset_index().merge(pib_mun, how='left', on='Cod.IBGE').set_index(['Município','UF'])
df_bcb['Operações de Crédito por Município'] = df_bcb['VERBETE_160_OPERACOES_DE_CREDITO']/df_bcb['pib

###################### Indicador Capital Poupado per capita ######################
df_bcb['420+432'] = df_bcb['VERBETE_420_DEPOSITOS_DE_POUPANCA'] + df_bcb['VERBETE_432_DEPOSITOS_A_PRA
df_bcb['Capital Poupado per capita'] = df_bcb['420+432']/df_bcb['pop_est'].astype(np.int64)

subdet_capital_disp = df_bcb.iloc[:,[0,6,8]]
subdet_capital_disp = subdet_capital_disp.merge(amostra, how='right', on='Cod.IBGE').rename(columns={
interesse=['Município','UF','Cod.IBGE','Operações de Crédito por Município',
           'Capital Poupado per capita']
subdet_capital_disp = subdet_capital_disp[interesse]
```

```
################ Indicador Proporção Relativa de Capital de Risco ################
df_crunchbase = pd.read_excel('DETERMINANTE ACESSO A CAPITAL/crunchbase_2021.xlsx', usecols="A:C").fi
df_crunchbase['Município'] = df_crunchbase['Município'].str.upper().str.normalize('NFKD').str.encode(
df_crunchbase = database.merge(df_crunchbase, how='left', on=['Município','UF']).merge(pib_mun, how='
df_crunchbase['Proporção Relativa de Capital de Risco'] = (df_crunchbase['Total funding amount']*(5.3

# Organizando a ordem os indicadores para calcular o subdeterminante/determinante
subdet_capital_disp = subdet_capital_disp.merge(df_crunchbase, how='right', on='Cod.IBGE')
interesse=['Município_x','UF_x','Operações de Crédito por Município',
           'Proporção Relativa de Capital de Risco','Capital Poupado per capita']
subdet_capital_disp = subdet_capital_disp[interesse]
subdet_capital_disp = subdet_capital_disp.rename(columns={'Município_x':'Município',
                                                          'UF_x':'UF'})
subdet_capital_disp = subdet_capital_disp.set_index(['Município','UF'])

# Tratamentos de para os missing e outliers
missing_data(subdet_capital_disp)
extreme_values(subdet_capital_disp)

# Criando o subdeterminante
create_subindex(subdet_capital_disp, subdet)
acesso_capital[subdet] = subdet_capital_disp

#################### Criando o Determinante Acesso a Capital ####################
acesso_capital = pd.concat(acesso_capital, axis=1)
create_detindex(acesso_capital, 'Acesso a Capital')

acesso_capital.to_csv('DETERMINANTES/det-ACESSO A CAPITAL.csv')
```

Determinante Inovação

- Para retornar ao relatório do Determinante Inovação clique aqui.

```
# Criando o ambiente que criará o determinante
inovacao = {}
```

6.0.0.1 Subdeterminante Inputs

- Para retornar ao relatório do Subdeterminante Inputs clique aqui.

```
# Indicando o subdeterminante
subdet = 'Inputs'

########## 2.6.1.1. Indicador Proporção de Mestres e Doutores em C&T ###########
```

```
variaveis = ('COUNT(quantidade_vinculos_ativos), id_municipio')
base = '`basedosdados.br_me_rais.microdados_estabelecimentos`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = f"SELECT {variaveis} FROM {base} WHERE ano = 2020 AND quantidade_vinculos_ativos > 0 GROUP BY

## Importando o data lake
df_rais = bd.read_sql(query=query, billing_project_id=project_id)
df_rais = df_rais.rename(columns={'id_municipio':'Cod.IBGE'}).set_index('Cod.IBGE')
df_rais = database.merge(df_rais,how='left',on='Cod.IBGE')
df_rais['mil_emp'] = df_rais['f0_']/1000
##
df_capes = pd.read_excel('DETERMINANTE INOVAÇÃO/br-capes-colsucup-discentes-2020-2021-11-10.xlsx', us
df_capes['Município'] = df_capes['NM_MUNICIPIO_PROGRAMA_IES'].str.normalize('NFKD').str.encode('ascii
df_capes = database.merge(df_capes, how='left',on='Município')

areas = ['astronomia / física', 'biotecnologia', 'ciência da computação',
         'ciência de alimentos', 'ciências agrárias I', 'ciências ambientais',
         'ciências biológicas I', 'ciências biológicas II', 'ciências biológicas III',
         'engenharias I', 'engenharias II', 'engenharias III', 'engenharias IV',
         'farmácia', 'geociências', 'matemática / probabilidade', 'estatística',
         'materiais e química']
areas = [x.upper() for x in areas]

df_capes = df_capes[df_capes['NM_SITUACAO_DISCENTE'] == 'TITULADO']
df_capes = df_capes.query('NM_AREA_AVALIACAO in @areas')
df_capes = df_capes.groupby(['Município','Cod.IBGE']).count()
df_capes = df_rais.merge(df_capes, how='left', on='Cod.IBGE').fillna(0)
df_capes['Proporção de Mestres e Doutores em C&T'] = df_capes['NM_AREA_AVALIACAO']/df_capes['mil_emp'

interesse = ['Cod.IBGE','Proporção de Mestres e Doutores em C&T']
subdet_input = df_capes[interesse]

############## 2.6.1.2. Indicador Proporção de Funcionários em C&T ###############
cbo_2002 = tuple(['201105','201110','201115','201205','201210','201215','201220',
                  '201225','202105','202110','202115','202120','203005','203010',
                  '203015','203020','203025','203105','203110','203115','203120',
                  '203125','203205','203210','203215','203220','203225','203230',
                  '203305','203310','203315','203320','203405','203410','203415',
                  '203420','203505','203510','203515','203520','203525','204105',
                  '211105','211110','211115','211120','211205','211210','211215',
                  '212205','212210','212215','212305','212310','212315','212320',
                  '212405','212410','212415','212420','212425','212430','213105',
```

```
'213110','213115','213120','213125','213130','213135','213140',
'213145','213150','213155','213160','213165','213170','213175',
'213205','213210','213215','213305','213310','213315','213405',
'213410','213415','213420','213425','213430','213435','213440',
'214005','214010','214105','214110','214115','214120','214125',
'214130','214205','214210','214215','214220','214225','214230',
'214235','214240','214245','214250','214255','214260','214265',
'214270','214280','214305','214310','214315','214320','214325',
'214330','214335','214340','214345','214350','214360','214365',
'214370','214405','214410','214415','214420','214425','214430',
'214435','214505','214510','214515','214520','214525','214530',
'214535','214605','214610','214615','214705','214710','214715',
'214720','214725','214730','214735','214740','214745','214750',
'214805','214810','214905','214910','214915','214920','214925',
'214930','214935','214940','214945','215105','215110','215115',
'215120','215125','215130','215135','215140','215145','215150',
'215205','215210','215215','215220','215305','215310','215315',
'300105','300110','300305','301105','301110','301115','301205',
'311105','311110','311115','311205','311305','311405','311410',
'311505','311510','311515','311520','311605','311610','311615',
'311620','311625','311705','311710','311715','311720','311725',
'312105','312205','312210','312305','312310','312315','312320',
'313105','313110','313115','313120','313125','313130','313205',
'313210','313215','313220','313305','313310','313315','313320',
'313405','313410','313415','313505','314105','314110','314115',
'314120','314125','314205','314210','314305','314310','314315',
'314405','314410','314610','314615','314620','314625','314705',
'314710','314715','314720','314725','314730','314805','314810',
'314815','314825','314830','314835','314840','314845','316105',
'316110','316115','316120','316305','316310','316315','316320',
'316325','316330','316335','316340','317105','317110','317115',
'317120','317205','317210','318005','318010','318015','318105',
'318110','318115','318120','318205','318210','318215','318305',
'318310','318405','318410','318415','318420','318425','318430',
'318505','318510','318605','318610','318705','318710','318805',
'318810','318815','319105','319110','319205','391105','391110',
'391115','391120','391125','391130','391135','391140','391145',
'391205','391210','391215','391220','391225','391230','395105',
'395110','720105','720110','720115','720120','720125','720130',
'720135','720140','720145','720150','720155','720160','720205',
'720210','720215','720220','721105','721110','721115','721205',
'721210','721215','721220','721225','721305','721310','721315',
'721320','721325','721405','721410','721415','721420','721425',
```

```
                '721430','722105','722110','722115','722205','722210','722215',
                '722220','722225','722230','722235','722305','722310','722315',
                '722320','722325','722330','722405','722410','722415','723105',
                '723110','723115','723120','723125','723205','723210','723215',
                '723220','723225','723230','723235','723240','723305','723310',
                '723315','723320','723325','723330','724105','724110','724115',
                '724120','724125','724130','724135','724205','724210','724215',
                '724220','724225','724230','724305','724310','724315','724320',
                '724325','724405','724410','724415','724420','724425','724430',
                '724435','724440','724505','724510','724515','724605','724610',
                '725005','725010','725015','725020','725025','725105','725205',
                '725210','725215','725220','725225','725305','725310','725315',
                '725320','725405','725410','725415','725420','725505','725510',
                '725605','725610','725705','730105','731105','731110','731115',
                '731120','731125','731130','731135','731140','731145','731150',
                '731155','731160','731165','731170','731175','731180','731205',
                '731305','731310','731315','731320','731325','731330','732105',
                '732110','732115','732120','732125','732130','732135','732140'])

variaveis = ('id_municipio, count(*)')
base = '`basedosdados.br_me_rais.microdados_vinculos`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query_1 = (f'SELECT {variaveis} AS n_cet FROM {base} WHERE ano = 2020 AND cbo_2002 IN'
           f' {cbo_2002} AND id_municipio IN {cod_ibge} GROUP BY id_municipio')
query_2 = (f'SELECT {variaveis} AS n_trab FROM {base} WHERE ano = 2020 AND id_municipio'
           f' IN {cod_ibge} GROUP BY id_municipio')
## Importando o data lake
df_rais_2_1 = bd.read_sql(query=query_1, billing_project_id=project_id)
df_rais_2_2 = bd.read_sql(query=query_2, billing_project_id=project_id)
df_rais_2 = df_rais_2_1.merge(df_rais_2_2, how='left',on='id_municipio')
df_rais_2['Proporção de Funcionários em C&T'] = df_rais_2['n_cet']/df_rais_2['n_trab']

subdet_input = subdet_input.merge(df_rais_2, right_on='id_municipio', left_on='Cod.IBGE')
interesse = ['Cod.IBGE','Proporção de Mestres e Doutores em C&T','Proporção de Funcionários em C&T']
subdet_input = subdet_input[interesse]

######## 2.6.1.3. Indicador Média de Investimentos do BNDES e da FINEP #########
## Importando dados: BNDES
df_bndes = pd.read_excel('DETERMINANTE INOVAÇÃO/naoautomaticas.xlsx',
                         usecols='D:F,I', header=4)
df_bndes = df_bndes.rename({'Município - código':'Cod.IBGE'},axis=1).astype(str)
df_bndes = df_bndes.merge(database, how='right', on='Cod.IBGE')
```

```
df_bndes.iloc[:,3:4] = df_bndes.iloc[:,3:4].apply(pd.to_numeric)
df_bndes = df_bndes.groupby(['Município','UF','Cod.IBGE']).sum()

## Importando dados: FINEP
df_finep = pd.read_excel('DETERMINANTE INOVAÇÃO/19_08_2022_Contratacao.xls',
                         usecols='E,K:M', header=5).drop([4], axis=0)
df_finep['Data Assinatura'] = pd.to_datetime(df_finep['Data Assinatura'], format='%Y-%m-%d')
df_finep = df_finep[(df_finep['Data Assinatura'] >= '2021-01-01 00:00:00')
                    & (df_finep['Data Assinatura'] <= '2021-12-31 00:00:00')]

df_finep['Município'] = df_finep['Município'].str.upper().str.normalize('NFKD').str.encode('ascii', e
df_finep = df_finep.groupby(['Município','UF']).sum()
df_finep = df_finep.merge(database, how='right',on=['Município','UF']).fillna(0)

df_finep_bndes = df_finep.merge(df_bndes, how='left',on=['Município','UF']).fillna(0)
df_finep_bndes = df_finep_bndes.merge(df_rais_2_1, left_on='Cod.IBGE', right_on='id_municipio')
df_finep_bndes['Média de Investimentos do BNDES e FINEP'] = (df_finep_bndes['Valor Finep'] + df_finep

subdet_input = subdet_input.merge(df_finep_bndes, how='right', on='Cod.IBGE')
interesse=['Cod.IBGE','Proporção de Mestres e Doutores em C&T','Proporção de Funcionários em C&T',
           'Média de Investimentos do BNDES e FINEP']
subdet_input=subdet_input[interesse]

################## 2.6.1.4. Indicador Infraestrutura Tecnológica #################
df_inpi_contrato = pd.read_excel('DETERMINANTE INOVAÇÃO/5 - Depósitos de Marcas por Cidade.xls',
                                 usecols='A,B,U,V', header=7).dropna()

df_inpi_contrato = df_inpi_contrato.rename(columns={df_inpi_contrato.columns[0]:'Cod.IBGE',
                                                    df_inpi_contrato.columns[1]:'Município'})

df_inpi_contrato['2018+2019'] = df_inpi_contrato.iloc[:,2] + df_inpi_contrato.iloc[:,3]
df_inpi_contrato['Cod.IBGE'] = df_inpi_contrato['Cod.IBGE'].astype(str)
df_inpi_contrato = df_inpi_contrato.merge(database, how='right', on='Cod.IBGE')
df_inpi_contrato = df_inpi_contrato.merge(df_rais, how='right', on='Cod.IBGE')
df_inpi_contrato['Contratos de Concessão'] = (df_inpi_contrato['2018+2019'])/df_inpi_contrato['mil_em

subdet_input = subdet_input.merge(df_inpi_contrato, how='right', on='Cod.IBGE')
subdet_input = subdet_input.merge(amostra, how='left', on=['Cod.IBGE'])
interesse=['Município','UF','Proporção de Mestres e Doutores em C&T',
           'Proporção de Funcionários em C&T','Média de Investimentos do BNDES e FINEP',
           'Contratos de Concessão']
subdet_input = subdet_input[interesse].set_index(['Município','UF'])
```

```
# Tratamentos de para os missing e outliers
missing_data(subdet_input)
extreme_values(subdet_input)

# Criando o subdeterminante
create_subindex(subdet_input, subdet)
inovacao[subdet] = subdet_input
```

6.0.0.2 Subdeterminante Outputs

- Para retornar ao relatório do Subdeterminante Outputs clique aqui.

```
# Indicando o subdeterminante
subdet = 'Output'

############################## Indicador Patentes ##############################
letras = ['a','b','c']
tipo = ['PI','MU','CA']
for i in list(range(0,3)):
    globals()[f'df_inpi_patente_{i}'] = pd.read_excel(f'DETERMINANTE INOVAÇÃO/5{letras[i]} - Depósito
                                    usecols='A,B,U,V', header=7).dropna().assign(tipo=tipo[i])
    pdList = []
    pdList.extend(value for name, value in locals().items() if name.startswith('df_inpi_patente_'))
    indicador_patente = pd.concat(pdList, axis=0)

indicador_patente = indicador_patente.rename(columns={indicador_patente.columns[0]:'Cod.IBGE',
                                                      indicador_patente.columns[1]:'Município'})

indicador_patente['2018+2019'] = indicador_patente.iloc[:,2] + indicador_patente.iloc[:,3]
indicador_patente['Cod.IBGE'] = indicador_patente['Cod.IBGE'].astype(str)
indicador_patente = indicador_patente.pivot(index='Cod.IBGE', columns='tipo',values='2018+2019').fill
cols = indicador_patente.columns[: indicador_patente.shape[0]]
indicador_patente['CA+MU+PI'] = indicador_patente[cols].sum(axis=1)
indicador_patente = indicador_patente.merge(database, how='right', on='Cod.IBGE')
indicador_patente = indicador_patente.merge(df_rais, how='right', on='Cod.IBGE')
indicador_patente['Patentes'] = (indicador_patente['CA+MU+PI'])/df_inpi_contrato['mil_emp']

subdet_output = indicador_patente[['Cod.IBGE','Patentes']]
subdet_output = subdet_output.merge(amostra, how='right', on='Cod.IBGE')
subdet_output = subdet_output[['Cod.IBGE','NOME DO MUNICÍPIO','UF','Patentes']]
subdet_output = subdet_output.rename(columns={'NOME DO MUNICÍPIO':'Município'})

################## Indicador Tamanho da indústria Inovadora ##################
list_cnae = tuple([
```

```
    'Fabricação de cloro e álcalis','Fabricação de intermediários para fertilizantes',
    'Fabricação de adubos e fertilizantes','Fabricação de gases industriais',
    'Fabricação de produtos químicos inorgânicos não especificados anteriormente',
    'Fabricação de produtos petroquímicos básicos','Fabricação de intermediários para plastificantes,
    'Fabricação de produtos químicos orgânicos não especificados anteriormente',
    'Fabricação de resinas termoplásticas','Fabricação de resinas termofixas',
    'Fabricação de elastômeros','Fabricação de fibras artificiais e sintéticas',
    'Fabricação de defensivos agrícolas','Fabricação de desinfestantes domissanitários',
    'Fabricação de sabões e detergentes sintéticos','Fabricação de produtos de limpeza e polimento',
    'Fabricação de cosméticos, produtos de perfumaria e de higiene pessoal',
    'Fabricação de tintas, vernizes, esmaltes e lacas','Fabricação de tintas de impressão',
    'Fabricação de impermeabilizantes, solventes e produtos afins',
    'Fabricação de adesivos e selantes','Fabricação de explosivos',
    'Fabricação de aditivos de uso industrial','Fabricação de catalisadores',
    'Fabricação de produtos químicos não especificados anteriormente',
    'Fabricação de produtos farmoquímicos','Fabricação de medicamentos para uso humano',
    'Fabricação de medicamentos para uso veterinário','Fabricação de preparações farmacêuticas',
    'Fabricação de aparelhos e equipamentos de medida, teste e controle','Fabricação de cronômetros e
    'Fabricação de aparelhos eletromédicos e eletroterapêuticos e equipamentos de irradiação',
    'Fabricação de equipamentos e instrumentos ópticos, fotográficos e cinematográficos',
    'Fabricação de geradores, transformadores e motores elétricos',
    'Fabricação de pilhas, baterias e acumuladores elétricos, exceto para veículos automotores',
    'Fabricação de baterias e acumuladores para veículos automotores',
    'Fabricação de aparelhos e equipamentos para distribuição e controle de energia elétrica',
    'Fabricação de material elétrico para instalações em circuito de consumo',
    'Fabricação de fios, cabos e condutores elétricos isolados',
    'Fabricação de lâmpadas e outros equipamentos de iluminação',
    'Fabricação de fogões, refrigeradores e máquinas de lavar e secar para uso doméstico',
    'Fabricação de aparelhos eletrodomésticos não especificados anteriormente',
    'Fabricação de equipamentos e aparelhos elétricos não especificados anteriormente',
    'Fabricação de motores e turbinas, exceto para aviões e veículos rodoviários',
    'Fabricação de equipamentos hidráulicos e pneumáticos, exceto válvulas',
    'Fabricação de válvulas, registros e dispositivos semelhantes','Fabricação de compressores',
    'Fabricação de equipamentos de transmissão para fins industriais',
    'Fabricação de aparelhos e equipamentos para instalações térmicas',
    'Fabricação de máquinas, equipamentos e aparelhos para transporte e elevação de cargas e pessoas'
    'Fabricação de máquinas e aparelhos de refrigeração e ventilação para uso industrial e comercial'
    'Fabricação de aparelhos e equipamentos de ar condicionado',
    'Fabricação de máquinas e equipamentos para saneamento básico e ambiental',
    'Fabricação de máquinas e equipamentos de uso geral não especificados anteriormente',
    'Fabricação de tratores agrícolas','Fabricação de equipamentos para irrigação agrícola',
    'Fabricação de máquinas e equipamentos para a agricultura e pecuária, exceto para irrigação',
    'Fabricação de máquinas-ferramenta','Fabricação de máquinas e equipamentos para a prospecção e ex
```

```
    'Fabricação de outras máquinas e equipamentos para uso na extração mineral, exceto na extração de
    'Fabricação de tratores, exceto agrícolas','Fabricação de máquinas e equipamentos para terraplena
    'Fabricação de máquinas para a indústria metalúrgica, exceto máquinas-ferramenta',
    'Fabricação de máquinas e equipamentos para as indústrias de alimentos, bebidas e fumo',
    'Fabricação de máquinas e equipamentos para a indústria têxtil',
    'Fabricação de máquinas e equipamentos para as indústrias do vestuário, do couro e de calçados',
    'Fabricação de máquinas e equipamentos para as indústrias de celulose, papel e papelão e artefato
    'Fabricação de máquinas e equipamentos para a indústria do plástico',
    'Fabricação de máquinas e equipamentos para uso industrial específico não especificados anteriorm
    'Fabricação de automóveis, camionetas e utilitários','Fabricação de caminhões e ônibus',
    'Fabricação de cabines, carrocerias e reboques para veículos automotores',
    'Fabricação de peças e acessórios para o sistema motor de veículos automotores',
    'Fabricação de peças e acessórios para os sistemas de marcha e transmissão de veículos automotore
    'Fabricação de peças e acessórios para o sistema de freios de veículos automotores',
    'Fabricação de peças e acessórios para o sistema de direção e suspensão de veículos automotores',
    'Fabricação de material elétrico e eletrônico para veículos automotores, exceto baterias',
    'Fabricação de peças e acessórios para veículos automotores não especificados anteriormente',
    'Recondicionamento e recuperação de motores para veículos automotores',
    'Fabricação de locomotivas, vagões e outros materiais rodantes',
    'Fabricação de peças e acessórios para veículos ferroviários','Fabricação de aeronaves',
    'Fabricação de turbinas, motores e outros componentes e peças para aeronaves',
    'Fabricação de motocicletas','Fabricação de bicicletas e triciclos não-motorizados',
    'Fabricação de equipamentos de transporte não especificados anteriormente'])

variaveis = ('cnae_2, descricao')
base = '`basedosdados.br_bd_diretorios_brasil.cnae_2`'
project_id = 'double-balm-306418'
query = (f'SELECT {variaveis} FROM {base} WHERE descricao IN {list_cnae}')
cnae_2 = bd.read_sql(query=query, billing_project_id=project_id)
cnae_2['cnae_2'] = cnae_2['cnae_2'].str.replace(r"(\d)\.",r"\1").str.replace(r"(\d)\-",r"\1")
cnae_2 = tuple(cnae_2['cnae_2'])

variaveis = ('id_municipio, count(*)')
base = '`basedosdados.br_me_rais.microdados_estabelecimentos`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = (f'SELECT {variaveis} FROM {base} WHERE ano = 2020 AND id_municipio'
         f' IN {cod_ibge} AND cnae_2 IN {cnae_2} GROUP BY id_municipio')
df_rais_inova = bd.read_sql(query=query, billing_project_id=project_id)
df_rais_inova = df_rais_inova.merge(df_rais, left_on='id_municipio', right_on='Cod.IBGE')
df_rais_inova['Tamanho da Indústria Inovadora'] = df_rais_inova['f0__x']/df_rais_inova['f0__y']

subdet_output = subdet_output.merge(df_rais_inova, how='left', on='Cod.IBGE')
```

```
interesse=['Cod.IBGE','Município','UF','Patentes','Tamanho da Indústria Inovadora']
subdet_output=subdet_output[interesse]

################### Indicador Tamanho da indústria Criativa ####################
list_cnae = tuple([
    'Lapidação de gemas e fabricação de artefatos de ourivesaria e joalheria',
    'Fabricação de bijuterias e artefatos semelhantes','Fabricação de instrumentos musicais',
    'Edição de livros','Edição de jornais','Edição de revistas',
    'Edição de Cadastros, Listas e de Outros Produtos Gráficos',
    'Edição integrada à impressão de livros','Edição integrada à impressão de jornais',
    'Edição integrada à impressão de revistas', 'Atividades de televisão aberta',
    'Edição integrada à impressão de cadastros, listas e de outros produtos gráficos',
    'Atividades de produção cinematográfica, de vídeos e de programas de televisão',
    'Atividades de pós-produção cinematográfica, de vídeos e de programas de televisão',
    'Distribuição cinematográfica, de vídeo e de programas de televisão',
    'Atividades de exibição cinematográfica','Agências de notícias',
    'Atividades de gravação de som e de edição de música','Atividades de rádio',
    'Programadoras e atividades relacionadas à televisão por assinatura',
    'Serviços de arquitetura','Agências de publicidade',
    'Pesquisa e desenvolvimento experimental em ciências físicas e naturais',
    'Pesquisa e desenvolvimento experimental em ciências sociais e humanas',
    'Atividades de publicidade não especificadas anteriormente',
    'Design e decoração de interiores','Atividades fotográficas e similares',
    'Aluguel de fitas de vídeo, DVDs e similares','Ensino de arte e cultura',
    'Ensino de idiomas','Artes cênicas, espetáculos e atividades complementares',
    'Criação artística','Gestão de espaços para artes cênicas, espetáculos e outras atividades artíst
    'Atividades de bibliotecas e arquivos',
    'Atividades de museus e de exploração, restauração artística e conservação de lugares e prédios h
    'Atividades de jardins botânicos, zoológicos, parques nacionais, reservas ecológicas e áreas de p
    'Atividades de organizações associativas ligadas à cultura e à arte'])

variaveis = ('cnae_2, descricao')
base = '`basedosdados.br_bd_diretorios_brasil.cnae_2`'
project_id = 'double-balm-306418'
query = (f'SELECT {variaveis} FROM {base} WHERE descricao IN {list_cnae}')
cnae_2 = bd.read_sql(query=query, billing_project_id=project_id)
cnae_2['cnae_2'] = cnae_2['cnae_2'].str.replace(r"(\d)\.",r"\1").str.replace(r"(\d)\-",r"\1")
cnae_2 = tuple(cnae_2['cnae_2'])

variaveis = ('id_municipio, count(*)')
base = '`basedosdados.br_me_rais.microdados_estabelecimentos`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
```

```
query = (f'SELECT {variaveis} FROM {base} WHERE ano = 2020 AND id_municipio'
         f' IN {cod_ibge} AND cnae_2 IN {cnae_2} GROUP BY id_municipio')
df_rais_cria = bd.read_sql(query=query, billing_project_id=project_id)
df_rais_cria = df_rais_cria.merge(df_rais, left_on='id_municipio', right_on='Cod.IBGE')
df_rais_cria['Tamanho da Indústria Criativa'] = df_rais_cria['f0__x']/df_rais_cria['f0__y']

subdet_output = subdet_output.merge(df_rais_cria, how='left', on='Cod.IBGE')
interesse=['Cod.IBGE','Município','UF','Patentes','Tamanho da Indústria Inovadora',
           'Tamanho da Indústria Criativa']
subdet_output=subdet_output[interesse]

###################### Indicador Tamanho das Empresas TIC ######################
list_cnae = tuple([
    'Fabricação de componentes eletrônicos','Fabricação de equipamentos de informática',
    'Fabricação de periféricos para equipamentos de informática',
    'Fabricação de equipamentos transmissores de comunicação',
    'Fabricação de aparelhos telefônicos e de outros equipamentos de comunicação',
    'Fabricação de aparelhos de recepção, reprodução, gravação e amplificação de áudio e vídeo',
    'Fabricação de mídias virgens, magnéticas e ópticas',
    'Comércio atacadista de computadores, periféricos e suprimentos de informática',
    'Comércio atacadista de componentes eletrônicos e equipamentos de telefonia e comunicação',
    'Telecomunicações sem fio','Operadoras de televisão por assinatura por cabo',
    'Telecomunicações por satélite','Operadoras de televisão por assinatura por microondas',
    'Operadoras de televisão por assinatura por satélite','Outras atividades de telecomunicações',
    'Desenvolvimento de programas de computador sob encomenda',
    'Desenvolvimento e licenciamento de programas de computador customizáveis',
    'Desenvolvimento e licenciamento de programas de computador não-customizáveis',
    'Consultoria em tecnologia da informação',
    'Suporte técnico, manutenção e outros serviços em tecnologia da informação',
    'Tratamento de dados, provedores de serviços de aplicação e serviços de hospedagem na internet',
    'Portais, provedores de conteúdo e outros serviços de informação na internet',
    'Reparação e manutenção de computadores e de equipamentos periféricos',
    'Reparação e manutenção de equipamentos de comunicação'])

variaveis = ('cnae_2, descricao')
base = '`basedosdados.br_bd_diretorios_brasil.cnae_2`'
project_id = 'double-balm-306418'
query = (f'SELECT {variaveis} FROM {base} WHERE descricao IN {list_cnae}')
cnae_2 = bd.read_sql(query=query, billing_project_id=project_id)
cnae_2['cnae_2'] = cnae_2['cnae_2'].str.replace(r"(\d)\.",r"\1").str.replace(r"(\d)\-",r"\1")
cnae_2 = tuple(cnae_2['cnae_2'])

variaveis = ('id_municipio, count(*)')
```

```
base = '`basedosdados.br_me_rais.microdados_estabelecimentos`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = (f'SELECT {variaveis} FROM {base} WHERE ano = 2020 AND id_municipio'
         f' IN {cod_ibge} AND cnae_2 IN {cnae_2} GROUP BY id_municipio')
df_rais_tic = bd.read_sql(query=query, billing_project_id=project_id)
df_rais_tic = df_rais_tic.merge(df_rais, left_on='id_municipio', right_on='Cod.IBGE')
df_rais_tic['Tamanho das Empresas TIC'] = df_rais_tic['f0__x']/df_rais_tic['f0__y']

subdet_output = subdet_output.merge(df_rais_tic, how='left', on='Cod.IBGE')
interesse=['Município','UF','Patentes','Tamanho da Indústria Inovadora',
           'Tamanho da Indústria Criativa','Tamanho das Empresas TIC']
subdet_output=subdet_output[interesse].set_index(['Município','UF'])

# Tratamentos de para os missing e outliers
missing_data(subdet_output)
extreme_values(subdet_output)

# Criando o subdeterminante
create_subindex(subdet_output, subdet)
inovacao[subdet] = subdet_output

######################## Criando o Determinante Inovação ##########################
inovacao = pd.concat(inovacao, axis=1)
create_detindex(inovacao, 'Inovação')

inovacao.to_csv('DETERMINANTES/det-INOVACAO.csv')
```

Determinante Capital Humano

- Para retornar ao relatório do Determinante Capital Humano clique aqui.

```
# Criando o ambiente que criará o determinante
capital_humano = {}
```

Subdeterminante Acesso e Qualidade da Mão de Obra Básica

- Para retornar ao relatório do Subdeterminante Acesso e Qualidade da Mão de Obra Básica clique aqui

```
# Indicando o subdeterminante
subdet = 'Acesso e Qualidade da Mão de Obra Básica'

############################### Indicador nota ideb ##############################
df_ideb = pd.read_excel('Arquivos ICE - 23/Ind_Originais_ICE_2022.xlsx', header=5,
```

```
                        usecols="B:C,BA")
subdet_acesso = df_ideb.rename(columns={df_ideb.columns[1]:'Município',
                                        df_ideb.columns[2]:'Nota do IDEB'})

subdet_acesso = subdet_acesso.merge(amostra, how='right', left_on='Município',
                                    right_on='NOME DO MUNICÍPIO')
subdet_acesso = subdet_acesso[['UF_x','Município','Nota do IDEB','Cod.IBGE']]
subdet_acesso = subdet_acesso.rename(columns={'UF_x':'UF'})

##### Indicador proporção de adultos com pelo menos o ensino médio completo #####
df_enem = pd.read_csv('DETERMINANTE CAPITAL HUMANO/ENEM_2021_100mun.csv')

alvo = ['E','F','M']

pai_EM,mae_EM,num_inscritos = pd.DataFrame(),pd.DataFrame(),pd.DataFrame()

num_inscritos['n_inscritos'] = df_enem.groupby('CO_MUNICIPIO_ESC').size()
pai_EM['pai_EM'] = df_enem[df_enem['Q001'].isin(alvo)].groupby('CO_MUNICIPIO_ESC').size()
mae_EM['mae_EM'] = df_enem[df_enem['Q002'].isin(alvo)].groupby('CO_MUNICIPIO_ESC').size()

pai_mae_EM = pai_EM.merge(mae_EM, how='inner', on='CO_MUNICIPIO_ESC')
pai_mae_EM = pai_mae_EM.merge(num_inscritos, how='inner', on='CO_MUNICIPIO_ESC')

pai_mae_EM['prop_pai_EM'] = pai_mae_EM['pai_EM']/pai_mae_EM['n_inscritos']
pai_mae_EM['prop_mae_EM'] = pai_mae_EM['mae_EM']/pai_mae_EM['n_inscritos']
pai_mae_EM['Proporção de Adultos com pelo menos o Ensino Médio Completo'] = (pai_mae_EM['prop_pai_EM'

interesse = ['Proporção de Adultos com pelo menos o Ensino Médio Completo']
pai_mae_EM = pai_mae_EM[interesse].reset_index()
pai_mae_EM['CO_MUNICIPIO_ESC'] = pai_mae_EM['CO_MUNICIPIO_ESC'].astype(str)

subdet_acesso = subdet_acesso.merge(pai_mae_EM, how='right', left_on='Cod.IBGE',
                                    right_on='CO_MUNICIPIO_ESC')

############## Indicador Taxa Líquida de Matrícula no Ensino Médio ###############
#### Pessoas entre 15 e 17 anos no município (população 2010)
variaveis = 'id_setor_censitario, sigla_uf, v049, v050, v051'
base = '`basedosdados.br_ibge_censo_demografico.setor_censitario_idade_total_2010`'
project_id = 'double-balm-306418'
query = (f'SELECT {variaveis} FROM {base}')

df_censo_15_17 = bd.read_sql(query=query, billing_project_id=project_id)
df_censo_15_17['Cod.IBGE'] = df_censo_15_17['id_setor_censitario'].str[:7]
```

```
df_censo_15_17['UF'] = df_censo_15_17['sigla_uf'].str[:2]
df_censo_15_17 = df_censo_15_17.merge(database, how='right', on='Cod.IBGE')
df_censo_15_17 = df_censo_15_17.dropna()

df_censo_15_17 = df_censo_15_17.iloc[:,2:7].set_index(['Cod.IBGE','UF'])
df_censo_15_17.iloc[:,0:3] = df_censo_15_17.iloc[:,0:3].apply(pd.to_numeric)
df_censo_15_17 = df_censo_15_17.groupby('Cod.IBGE').sum()
df_censo_15_17['pop_15_17'] = df_censo_15_17.sum(axis=1)

#### População 2010
variaveis = 'id_municipio, populacao'
base = '`basedosdados.br_ibge_populacao.municipio`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = (f'SELECT {variaveis} FROM {base} WHERE ano = 2010 AND id_municipio IN {cod_ibge}')

pop_2010 = bd.read_sql(query=query, billing_project_id=project_id)
populacao = pop_2010.merge(amostra, left_on='id_municipio', right_on='Cod.IBGE')
interesse = ['Cod.IBGE','populacao','POPULAÇÃO ESTIMADA']
populacao = populacao[interesse]
populacao['tx_crecimento'] = 1 + (populacao['POPULAÇÃO ESTIMADA'].astype(int)-populacao['populacao'])

df_censo_15_17 = df_censo_15_17.merge(populacao, how='right', on='Cod.IBGE')
df_censo_15_17['pop_15_17_atualizado'] = df_censo_15_17['pop_15_17']*df_censo_15_17['tx_crecimento']

interesse = ['Cod.IBGE','pop_15_17_atualizado']
df_censo_15_17 = df_censo_15_17[interesse]

#### censo escolar população entre 15 e 17 anos
df_ce_2021 = pd.read_csv('DETERMINANTE CAPITAL HUMANO/CE_2021_100mun.csv',
                         sep=',', encoding='latin-1')
df_ce_2021 = df_ce_2021[['CO_MUNICIPIO','QT_MAT_MED']].dropna()
df_ce_2021 = df_ce_2021.groupby('CO_MUNICIPIO').sum().reset_index()
df_ce_2021['CO_MUNICIPIO'] = df_ce_2021['CO_MUNICIPIO'].astype(str)

df_ce_2021 = df_ce_2021.merge(df_censo_15_17, left_on='CO_MUNICIPIO',
                              right_on='Cod.IBGE')

df_ce_2021['Taxa Líquida de Matrícula no Ensino Médio'] = df_ce_2021['QT_MAT_MED']/df_ce_2021['pop_15

interesse = ['Cod.IBGE','Taxa Líquida de Matrícula no Ensino Médio']
df_ce_2021 = df_ce_2021[interesse]
```

```
subdet_acesso = subdet_acesso.merge(df_ce_2021, how='right', on='Cod.IBGE')

########################## Indicador Nota Média no Enem ##########################
nota_enem = df_enem[['CO_MUNICIPIO_ESC','NU_NOTA_CH','NU_NOTA_CN',
                     'NU_NOTA_LC','NU_NOTA_MT','NU_NOTA_REDACAO']].dropna()
nota_enem = nota_enem.groupby('CO_MUNICIPIO_ESC').mean()
nota_enem['Nota Média no ENEM'] = nota_enem.mean(axis=1)
nota_enem = nota_enem['Nota Média no ENEM'].reset_index()
nota_enem['CO_MUNICIPIO_ESC'] = nota_enem['CO_MUNICIPIO_ESC'].astype(str)

subdet_acesso = subdet_acesso.merge(nota_enem, left_on='Cod.IBGE',
                                    right_on='CO_MUNICIPIO_ESC')

## Indicador Proporção de Matriculados no Ensino Técnico e Profissionalizante ##
#### População maior que 15 anos
base = '`basedosdados.br_ibge_censo_demografico.setor_censitario_idade_total_2010`'
project_id = 'double-balm-306418'
query = (f'SELECT * FROM {base}')

df_censo_15 = bd.read_sql(query=query, billing_project_id=project_id)
df_censo_15['Cod.IBGE'] = df_censo_15['id_setor_censitario'].str[:7]
df_censo_15['UF'] = df_censo_15['sigla_uf'].str[:2]
df_censo_15 = df_censo_15.set_index(['UF','Cod.IBGE'])
df_censo_15 = df_censo_15.iloc[:,50:137].reset_index()
df_censo_15 = df_censo_15.merge(database, how='right', on='Cod.IBGE').dropna()
df_censo_15.iloc[:,2:89] = df_censo_15.iloc[:,2:89].apply(pd.to_numeric)
df_censo_15 = df_censo_15.groupby('Cod.IBGE').sum().reset_index()
df_censo_15['pop_maior_15'] = df_censo_15.sum(axis=1)

interesse=['Cod.IBGE','pop_maior_15']
df_censo_15 = df_censo_15[interesse].merge(populacao, how='right', on='Cod.IBGE')
df_censo_15['atualizada_pop_maior_15'] = df_censo_15['pop_maior_15']*df_censo_15['tx_crecimento']
interesse=['Cod.IBGE','atualizada_pop_maior_15']
df_censo_15=df_censo_15[interesse]

#### Censo escolar
df_ce_tec = pd.read_csv('DETERMINANTE CAPITAL HUMANO/CE_2021_100mun.csv',
                        sep=',', encoding='latin-1')
df_ce_tec = df_ce_tec[['CO_MUNICIPIO','QT_MAT_PROF_TEC']].dropna()
df_ce_tec = df_ce_tec.groupby('CO_MUNICIPIO').sum().reset_index()
df_ce_tec['CO_MUNICIPIO'] = df_ce_tec['CO_MUNICIPIO'].astype(str)

df_ce_tec = df_ce_tec.merge(df_censo_15, left_on='CO_MUNICIPIO',right_on='Cod.IBGE')
```

```
df_ce_tec['Proporção de Matriculados no Ensino Técnico e Profissionalizante'] = df_ce_tec['QT_MAT_PRO
interesse = ['Cod.IBGE','Proporção de Matriculados no Ensino Técnico e Profissionalizante']
df_ce_tec = df_ce_tec[interesse]

subdet_acesso = subdet_acesso.merge(df_ce_tec, how='right', on='Cod.IBGE')
subdet_acesso = subdet_acesso.set_index(['Município','UF'])
del subdet_acesso['Cod.IBGE']
del subdet_acesso['CO_MUNICIPIO_ESC_x']
del subdet_acesso['CO_MUNICIPIO_ESC_y']

# Tratamentos de para os missing e outliers
missing_data(subdet_acesso)
extreme_values(subdet_acesso)

# Criando o subdeterminante
create_subindex(subdet_acesso, subdet)
capital_humano[subdet] = subdet_acesso
```

Subdeterminante Acesso e Qualidade da Mão de Obra Qualificada

- Para retornar ao relatório do Subdeterminante Acesso e Qualidade da Mão de Obra Qualificada clique aqui

```
# Indicando o subdeterminante
subdet = 'Acesso e Qualidade da Mão de Obra Qualificada'

### Indicador Proporção de Adultos com Pelo Menos o Ensino Superior Completo ###
alvo = ['F','G']
pai_SUP,mae_SUP = pd.DataFrame(),pd.DataFrame()

pai_SUP['pai_SUP'] = df_enem[df_enem['Q001'].isin(alvo)].groupby('CO_MUNICIPIO_ESC').size()
mae_SUP['mae_SUP'] = df_enem[df_enem['Q002'].isin(alvo)].groupby('CO_MUNICIPIO_ESC').size()

pai_mae_SUP = pai_SUP.merge(mae_SUP, how='inner', on='CO_MUNICIPIO_ESC')
subdet_acesso_quali = pai_mae_SUP.merge(num_inscritos, how='inner', on='CO_MUNICIPIO_ESC')

subdet_acesso_quali['prop_pai_SUP'] = subdet_acesso_quali['pai_SUP']/subdet_acesso_quali['n_inscritos
subdet_acesso_quali['prop_mae_SUP'] = subdet_acesso_quali['mae_SUP']/subdet_acesso_quali['n_inscritos
subdet_acesso_quali['Proporção de Adultos com pelo menos os Ensino Superior Completo'] = (subdet_aces
subdet_acesso_quali = subdet_acesso_quali['Proporção de Adultos com pelo menos os Ensino Superior Com

##### Indicador Proporção de Alunos Concluintes em Cursos de Alta Qualidade #####
df_enade = pd.read_excel('Arquivos ICE - 23/Ind_Originais_ICE_2022.xlsx', header=5,
```

```
                        usecols="B:C,BH")
df_enade = df_enade.rename(columns={df_enade.columns[2]:'Proporção de Alunos Concluintes em Cursos de
df_enade = df_enade.merge(amostra, how='right', left_on='Município',right_on='NOME DO MUNICÍPIO')
df_enade = df_enade[['UF_x','Município','Proporção de Alunos Concluintes em Cursos de Alta Qualidade'
df_enade = df_enade.rename(columns={'UF_x':'UF'})

subdet_acesso_quali['CO_MUNICIPIO_ESC'] = subdet_acesso_quali['CO_MUNICIPIO_ESC'].astype(str)
subdet_acesso_quali = subdet_acesso_quali.merge(df_enade, right_on='Cod.IBGE',
                                                left_on='CO_MUNICIPIO_ESC')
order = ['Cod.IBGE','Município','UF','Proporção de Adultos com pelo menos os Ensino Superior Completo
subdet_acesso_quali = subdet_acesso_quali[order]

################ Indicador Custo Médio de Salários de Dirigentes #################
cbo_2002 = tuple(['121005','121010','122105','122110','122115','122120','122205',
                  '122305','122405','122505','122510','122515','122520','122605',
                  '122610','122615','122620','122705','122710','122715','122720',
                  '122725','122730','122735','122740','122745','122750','122755',
                  '123105','123110','123115','123205','123210','123305','123310',
                  '123405','123410','123605','123705','123805','131105','131110',
                  '131115','131120','131205','131210','131215','131220','131225',
                  '131305','131310','131315','131320','141105','141110','141115',
                  '141120','141205','141305','141405','141410','141415','141420',
                  '141505','141510','141515','141520','141525','141605','141610',
                  '141615','141705','141710','141715','141720','141725','141730',
                  '141735','141805','141810','141815','141820','141825','141830',
                  '142105','142110','142115','142120','142125','142130','142205',
                  '142210','142305','142310','142315','142320','142325','142330',
                  '142335','142340','142345','142350','142405','142410','142415',
                  '142505','142510','142515','142520','142525','142530','142535',
                  '142605','142610','142705','142710'])

variaveis = 'valor_remuneracao_media,id_municipio'
base = '`basedosdados.br_me_rais.microdados_vinculos`'
project_id = 'double-balm-306418'
cod_ibge = tuple(database['Cod.IBGE'].astype(str))
query = (f'SELECT {variaveis} FROM {base} WHERE ano = 2019 AND cbo_2002 IN {cbo_2002}'
         f' AND id_municipio IN {cod_ibge}')

df_rais = bd.read_sql(query=query, billing_project_id=project_id)
df_rais = df_rais.groupby('id_municipio').agg(['count','sum'])
df_rais['Custo Médio de Salários de Dirigentes'] = df_rais.iloc[:,1]/df_rais.iloc[:,0]
interesse = ['Custo Médio de Salários de Dirigentes']
df_rais = df_rais[interesse].reset_index().droplevel(level=1, axis=1)
```

```
df_rais['Custo Médio de Salários de Dirigentes'] = negative(df_rais['Custo Médio de Salários de Dirig

subdet_acesso_quali = subdet_acesso_quali.merge(df_rais, right_on='id_municipio',
                                                left_on='Cod.IBGE')
subdet_acesso_quali = subdet_acesso_quali.set_index(['Município','UF'])
del subdet_acesso_quali['Cod.IBGE']
del subdet_acesso_quali['id_municipio']

# Tratamentos de para os missing e outliers
missing_data(subdet_acesso_quali)
extreme_values(subdet_acesso_quali)

# Criando o subdeterminante
create_subindex(subdet_acesso_quali, subdet)
capital_humano[subdet] = subdet_acesso_quali

################# Criando o determinante de Ambiente Regulatório #################
capital_humano = pd.concat(capital_humano, axis=1)
create_detindex(capital_humano, 'Capital Humano')

capital_humano.to_csv('DETERMINANTES/det-CAPITAL HUMANO.csv')
```

Determinante Cultura