Projeto de Banco de Dados final

Lucas Fernandes André - 182495
Mariana Alves de Sousa - 241201

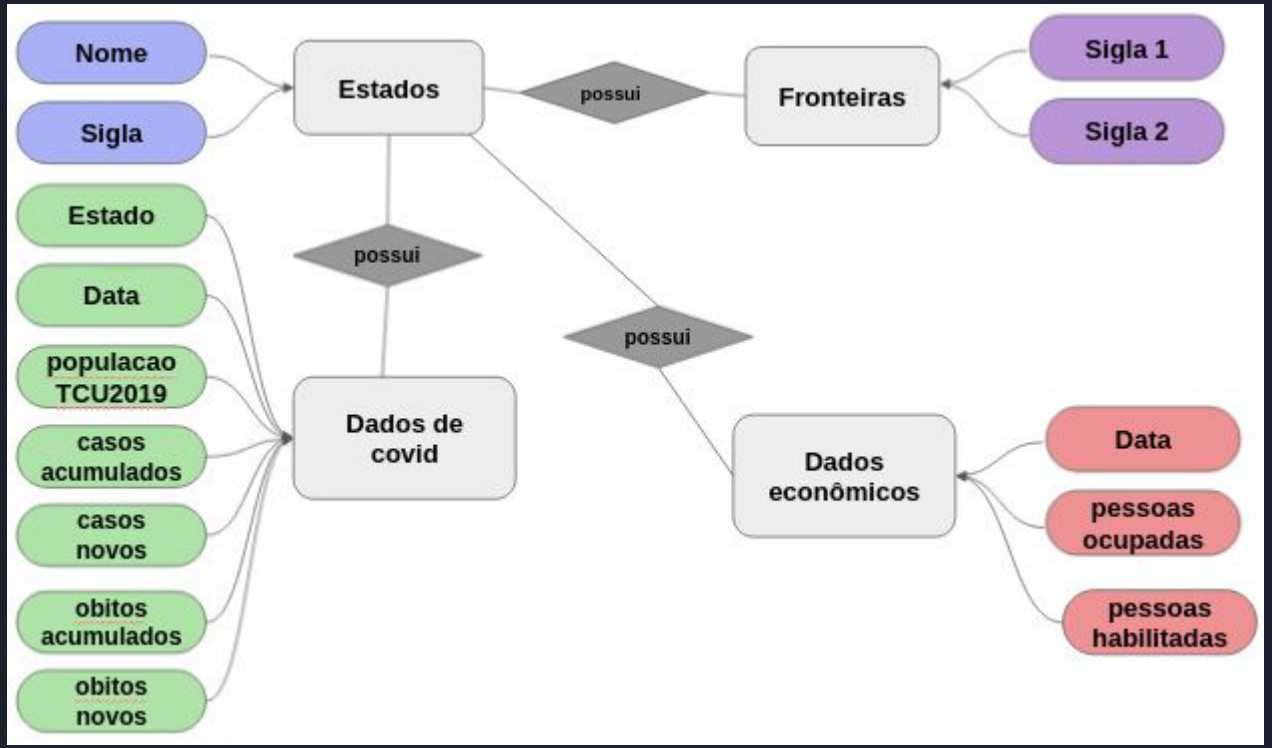
Contexto e motivação do dataset

Infelizmente, o covid-19 gerou milhões óbitos no mundo inteiro. Devido à pandemia e ao isolamento social, diversas esferas da sociedade foram afetadas. É importante analisar em como esses problemas afetaram a sociedade sob uma ótica econômica nestes últimos anos.

Descrição do objetivo do dataset

- Análise comparativa entre os estados brasileiros com relação ao dados de covid-19.
- Análise econômica da sociedade brasileira devido ao covid-19.

Modelo conceitual



Modelos Relacional

Dados divididos por tabelas.

estados (sigla, nome)

fronteiras (sigla_1, sigla_2)

dados (estado, data, semanaEpi, populacaoTCU2019, casosAcumulados, casosNovos, obitosAcumulados, obitosNovos)

dados_economcios(data, contagem_pessoas_habilitadas, contagem_pessoas_ocupadas)

Tratamento de dados

1) Função auxiliar:

Converte um dataframe em uma tabela de SQL.

```python
from tqdm import tqdm
def dataframe_to_sql(
  df,
  table_name,
  conn,
  cursor,
  map_columns
):

  TABLE_VALUES = ",".join(list(map_columns.values()))

  INSERT_STRING = f"INSERT INTO {table_name}({{}}) VALUES ({{}})"


  for _, i in tqdm(df.loc[:, map_columns.keys()].iterrows()):

    TABLE_VALUES = ",".join(
        value for key, value in map_columns.items() if not pd.isnull(i[key])
    )

    VALUES = ",".join(
      [f'"{value}"' if (type(value) is str) else str(value) for value in i.values if not pd.isnull(value)]
    )

    cursor.execute(INSERT_STRING.format(TABLE_VALUES, VALUES))

  conn.commit()
```

Tratamento de dados

2) Estados:

```
request = requests.get("https://raw.githubusercontent.com/leogermani/estados-e-municipios-ibge/master/estados.csv")
```

```
cursor.execute(
    """
    CREATE TABLE estados_nao_processado (
      sigla CHAR[3] PRIMARY KEY,
      nome TEXT
    );
    """
)
```

```
dataframe_to_sql(
    df=df,
    table_name="estados_nao_processado",
    conn=conn,
    cursor=cursor,
    map_columns={
        "NOME": "nome",
        "SIGLA": "sigla"
    }
)
```

```
cursor.execute(
    "CREATE TABLE estados AS "\
    "SELECT substr(sigla, 2, 3) AS sigla, nome from estados_nao_processado"
)
```

Tratamento de dados

3) Fronteiras:

```
df = pd.read_csv("fronteiras.csv")

dataframe_to_sql(
    df=df,
    table_name="fronteiras",
    conn=conn,
    cursor=cursor,
    map_columns={
        "sigla1": "sigla_1",
        "sigla2": "sigla_2"
    }
)
```

```
cursor.execute("""
  CREATE TABLE fronteiras (
    id INTEGER PRIMARY KEY,
    sigla_1 CHAR[2],
    sigla_2 CHAR[2]
  )
""")
```

Tratamento de dados

3) Dados:

```
cursor.execute("""CREATE TABLE dados_nao_processados (
    estado TEXT,
    data TEXT,
    semanaEpi INTEGER,
    populacaoTCU2019 REAL,
    casosAcumulado INTEGER,
    casosNovos INTEGER,
    obitosAcumulado INTEGER,
    obitosNovos INTEGER
);""")
```

```
file_list = [
  "/content/HIST_PAINEL_COVIDBR_2020_Parte1_10nov2021.csv",
  "/content/HIST_PAINEL_COVIDBR_2020_Parte2_10nov2021.csv",
  "/content/HIST_PAINEL_COVIDBR_2021_Parte1_10nov2021.csv",
  "/content/HIST_PAINEL_COVIDBR_2021_Parte2_10nov2021.csv",
]

for file_ in file_list:
  df = pd.read_csv(file_, sep=";")

  dataframe_to_sql(
      df=df,
      conn=conn,
      cursor=cursor,
      table_name="dados_nao_processados",
      map_columns={
          "estado": "estado",
          "data": "data",
          "semanaEpi": "semanaEpi",
          "populacaoTCU2019": "populacaoTCU2019",
          "casosAcumulado": "casosAcumulado",
          "casosNovos": "casosNovos",
          "obitosAcumulado": "obitosAcumulado",
          "obitosNovos": "obitosNovos"
      }
  )
```

```
cursor.execute(
    "CREATE TABLE dados AS "\
    "SELECT * FROM dados_nao_processados "\
    "WHERE estado IS NOT NULL"
)
```

Tratamento de dados

4) Dados resumidos:

```
cursor.execute(
    "CREATE TABLE dados_estados_total AS "\
    "SELECT estado, MAX(casosAcumulado) AS casosAcumulados, MAX(obitosAcumulado) as obitosAcumulados "\
    "FROM dados GROUP BY estado"
)
```

Tratamento de dados

4) Dados econômicos:

```
cursor.execute(
    """
    CREATE TABLE pessoas_habilitadas_nao_processado (
      id INTEGER PRIMARY KEY AUTOINCREMENT,
      data VARCHAR(8),
      contagem_pessoas INTEGER
    );
    """
)
```

```
df = pd.read_csv("pessoas_habilitadas.csv")

dataframe_to_sql(
    df=df,
    conn=conn,
    cursor=cursor,
    table_name="pessoas_habilitadas_nao_processado",
    map_columns={
        'Data': 'data',
         'Pessoas de 14 anos ou mais de idade - Pessoa - Instituto Brasileiro de Geografia e Estatística': 'contagem_pessoas'
    }
)
```

Tratamento de dados

4) Dados econômicos:

```
cursor.execute(
    """
    CREATE TABLE pessoas_ocupadas_nao_processado (
      id INTEGER PRIMARY KEY AUTOINCREMENT,
      data VARCHAR(8),
      contagem_pessoas INTEGER
    );
    """
)
```

```
df = pd.read_csv("pessoas_ocupadas.csv")

dataframe_to_sql(
    df=df,
    conn=conn,
    cursor=cursor,
    table_name="pessoas_ocupadas_nao_processado",
    map_columns={
        'Data': 'data',
        'Pessoas ocupadas - Pessoa - Instituto Brasileiro de Geografia e Estatística': 'contagem_pessoas'
    }
)
```

Tratamento de dados

4) Dados econômicos:

```
cursor.execute(
    "CREATE TABLE pessoas_habilitadas AS "\
    "SELECT * from pessoas_habilitadas_nao_processado WHERE data > \"2019\";"
)
```

```
cursor.execute(
    "CREATE TABLE pessoas_ocupadas AS "\
    "SELECT * from pessoas_ocupadas_nao_processado WHERE data > \"2019\";"
)
```

```
cursor.execute(
    "CREATE TABLE data_economico AS "\
    "SELECT pessoas_ocupadas.data as data, "\
    "pessoas_ocupadas.contagem_pessoas as contagem_pessoas_ocupadas, "\
    "pessoas_habilitadas.contagem_pessoas as contagem_pessoas_habilitadas "\
    "FROM pessoas_ocupadas INNER JOIN pessoas_habilitadas ON pessoas_ocupadas.data = pessoas_habilitadas.data"
)
```

- O mês de 2020 que teve maior número de óbitos foi o mesmo mês que teve menor número de pessoas ocupadas?

```
for i in cursor.execute(
    "SELECT data, MIN(contagem_pessoas_ocupadas) from data_economico"
):
  print(i)
```

```
('2020.08', 81666)
```

```
for i in cursor.execute("SELECT data, MAX(obitosNovos) from dados where data LIKE '%2020%' "):
  print(i)
```

```
('2020-07-29', 713)
```

```
for i in cursor.execute(
    "SELECT data, MIN(contagem_pessoas_ocupadas) FROM data_economico WHERE "\
    "contagem_pessoas_ocupadas > "\
    "(SELECT MIN(contagem_pessoas_ocupadas) FROM data_economico)"
):
  print(i)
```

```
('2020.07', 82027)
```

Possíveis perguntas

Comparação entre estados vizinhos quanto à letalidade do covid19

```
for i in cursor.execute(
    "SELECT fronteiras.sigla_1 "\
    ", CAST(dados_estado1_total.obitosAcumulados AS REAL) / CAST(dados_estado1_total.casosAcumulados AS REAL) , " \
    "fronteiras.sigla_2 " \
    ", CAST(dados_estado2_total.obitosAcumulados AS REAL) / CAST(dados_estado2_total.casosAcumulados AS REAL) " \
    "from fronteiras JOIN dados_estados_total dados_estado1_total ON fronteiras.sigla_1 = dados_estado1_total.estado " \
    "JOIN dados_estados_total dados_estado2_total ON fronteiras.sigla_2 = dados_estado2_total.estado " \
):
  print(i)
```

```
('RR', 0.015941121202630755, 'AM', 0.0321665818601112)
('RR', 0.015941121202630755, 'PA', 0.027911482154626615)
('AC', 0.020947342128567862, 'AM', 0.0321665818601112)
('AP', 0.016082437623059293, 'PA', 0.027911482154626615)
('RO', 0.02415141687333989, 'AM', 0.0321665818601112)
('RO', 0.02415141687333989, 'MT', 0.025105069909259887)
('PA', 0.027911482154626615, 'AM', 0.0321665818601112)
('PA', 0.027911482154626615, 'MA', 0.028295063684596996)
('PA', 0.027911482154626615, 'MT', 0.025105069909259887)
('PA', 0.027911482154626615, 'TO', 0.016931313579409295)
('BA', 0.021717758727299342, 'ES', 0.02128139496111487)
('BA', 0.021717758727299342, 'SE', 0.021655012993725683)
('BA', 0.021717758727299342, 'AL', 0.02621991999435059)
('BA', 0.021717758727299342, 'PE', 0.031652797401619574)
('BA', 0.021717758727299342, 'PI', 0.02174769136838892)
('BA', 0.021717758727299342, 'TO', 0.016931313579409295)
('BA', 0.021717758727299342, 'GO', 0.026663055302126464)
('BA', 0.021717758727299342, 'MG', 0.0254435528717211)
('SE', 0.021655012993725683, 'BA', 0.021717758727299342)
('SE', 0.021655012993725683, 'AL', 0.02621991999435059)
('AL', 0.02621991999435059, 'SE', 0.021655012993725683)
```

Possíveis perguntas

Comparação entre as semanas epidemiológicas dos anos de covid19

```
list(cursor.execute("SELECT data, semanaEpi, SUM(casosNovos), sum(obitosNovos) from dados GROUP BY data"))
```

```
('2020-04-02', 14, 2147, 115),
('2020-04-03', 14, 2330, 120),
('2020-04-04', 14, 2425, 146),
('2020-04-05', 15, 1686, 108),
('2020-04-06', 15, 1860, 136),
('2020-04-07', 15, 3457, 238),
('2020-04-08', 15, 4385, 268),
('2020-04-09', 15, 3779, 270),
('2020-04-10', 15, 3625, 237),
('2020-04-11', 15, 2177, 134),
('2020-04-12', 16, 2830, 193),
('2020-04-13', 16, 2659, 213),
('2020-04-14', 16, 3599, 412),
('2020-04-15', 16, 6219, 408),
('2020-04-16', 16, 4022, 376),
('2020-04-17', 16, 6567, 427),
('2020-04-18', 16, 5897, 422),
('2020-04-19', 17, 4004, 224),
('2020-04-20', 17, 3855, 229),
('2020-04-21', 17, 5013, 333),
('2020-04-22', 17, 5523, 331),
('2020-04-23', 17, 7470, 810),
```