SOUL CODE

PROJETO FINAL BC 26 – ENG. DE DADOS

TEMA: COMBUSTÍVEIS

EQUIPE

AUGUSTO TONELLI

in/augusto-tonelli

CAIO ALVES

in/caio-italo-alves

ÉRICA MARÇAL

in/erica-marcal

LUAN SAGARA

in/luan-sagara

NAYARA BERNARDO

in/nayyara-bernardo

ESCOPO

REALIZAR O PROCESSO DE EXTRAÇÃO, TRANSFORMAÇÃO E CARREGAMENTO E ANÁLISES DOS DADOS UTILIZANDO LINGUAGEM PYTHON E BIBLIOTECAS, CLOUD E BANCO DE DADOS

REQUISITOS

- Uso mínimo de 2 (dois) Datasets em formatos diferentes, um obrigatoriamente em CSV
- Procedimento de ETL e análises através de Pandas, PySpark
- Armazenamento dos dados brutos em Cloud SQL (MySQL) e os tratados no MongoDB, Cloud Storage e/ou Big Query
- Análises realizadas através do BigQuery em linguagem padrão SQL
- Criação de Dashboard com dados tratados no Google Looker Studio

OBJETIVOS

Análise de dados públicos do Brasil sobre petróleo e derivados, biocombustíveis e gás natural no período de 2012 a 2021, em relação aos impactos:

- nos preços de revenda e margem de ganho
- no volume de produção
- nas importações e exportações

DataRoot

LINHA DO TEMPO

2012 — Início dos dados

2016 — Preço de Paridade de Importação (PPI)

2018 — Greve dos Caminhoneiros

2020 — Início da Pandemia

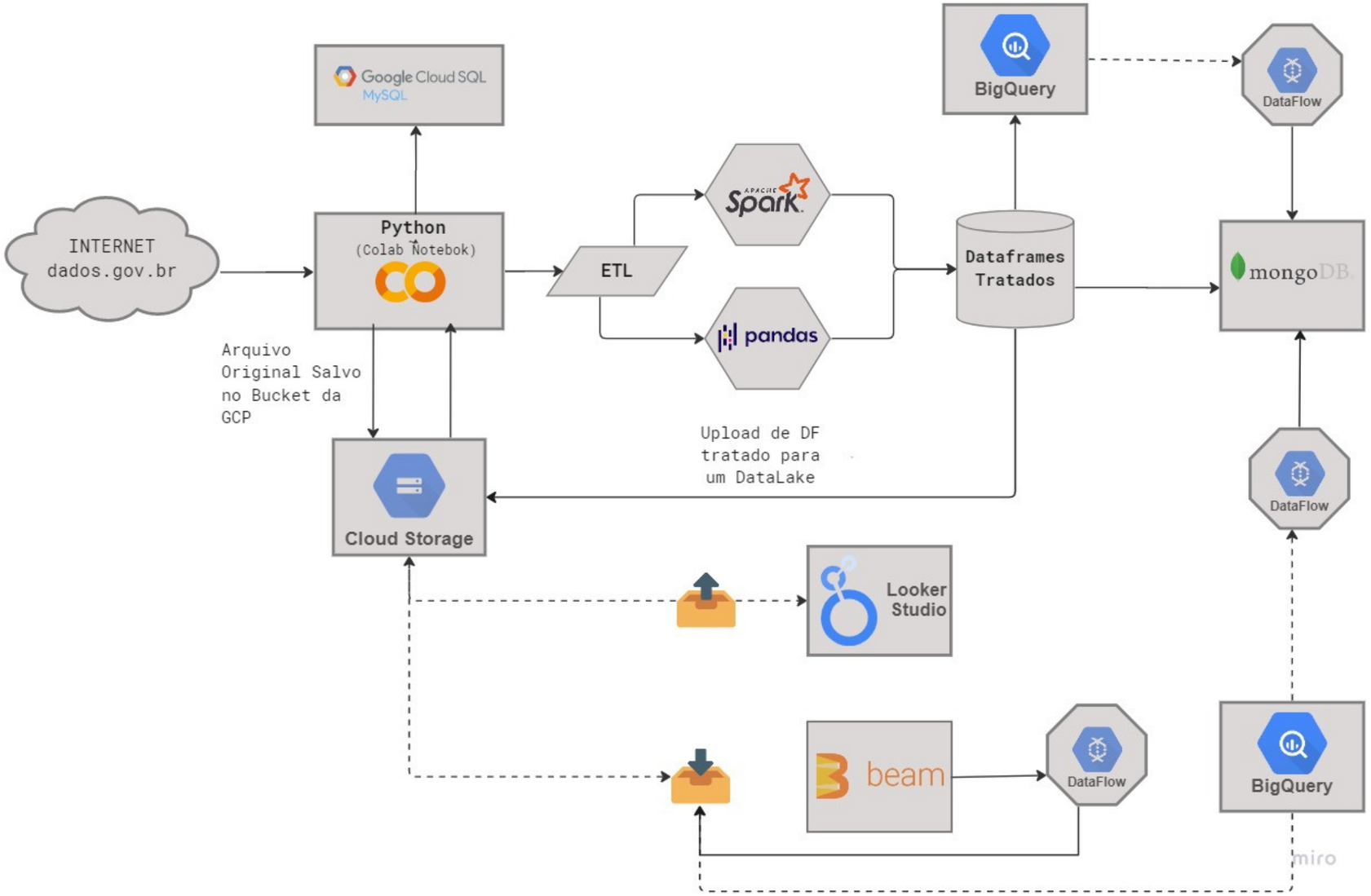
2021 — Situação Atual

DATASETS

Os dados brutos foram obtidos através da plataforma dados.gov.br. Ao todo, foram selecionados 11 (onze) datasets:

- Importação e Exportação de combustíveis
- Produção de Petróleo e Gás Natural por estado
- Produção de Derivados de Petróleo por refinaria
- Produção de Biodiesel, Etanol Anidro e Hidratado
- Série histórica de preços de combustíveis

WORKFLOW



CÓDIGOS

Tratamentos utilizando a biblioteca Pandas

> **Drop da coluna de produtor para igualar os Dataframes. Não fará falta nas análises**

```
[15]  1 dfbdi.drop('PRODUTOR',axis=1,inplace=True)
```

> **União dos dois dataframes a fim de facilitar o tratamento**

```
[16]  1 dfbio = pd.concat([dfbdi,dfeta])
```

CÓDIGOS

Tratamentos utilizando a biblioteca Pandas

Tratamento das inconsistências detectadas

```
[ ]  1 dfbio.volume_m3.replace(',','.',regex=True, inplace=True)
     2 dfbio.regiao.replace({'REGIÃO CENTRO OESTE':'REGIÃO CENTRO-OESTE'},inplace=True)
```

```
[ ]  1 dfbio.uf.replace({'BRASILIA':'DISTRITO FEDERAL'},inplace=True)
```

Restrição do dataframe para o intervalo desejado

```
[22]  1 dfbio = dfbio.loc[(dfbio['ano'] >= 2012) & (dfbio['ano'] <= 2021)]
```

CÓDIGOS

Tratamentos utilizando a Biblioteca Pandas

Ajuste da coluna de data unindo as colunas de mês e ano

```
[ ]  1 dfbio.mes.replace(({'JAN':'01','FEV':'02','MAR':'03','ABR':'04','MAI':'05','JUN':'06',
     2                     'JUL':'07','AGO':'08', 'SET':'09','OUT':'10','NOV':'11','DEZ':'12'}),
     3                    regex=True,inplace=True)
```

```
[ ]  1 dfbio['ano'] = dfbio['ano'].astype(str)
     2 dfbio['data'] = dfbio['ano'] + '-' + dfbio['mes']
     3 dfbio['data'] = pd.to_datetime(dfbio['data'], format = '%Y-%m')
```

```
[ ]  1 dfbio.drop(['ano','mes'],axis=1,inplace=True)
```

Filtragem de linhas com somente os produtos que interessam ao escopo

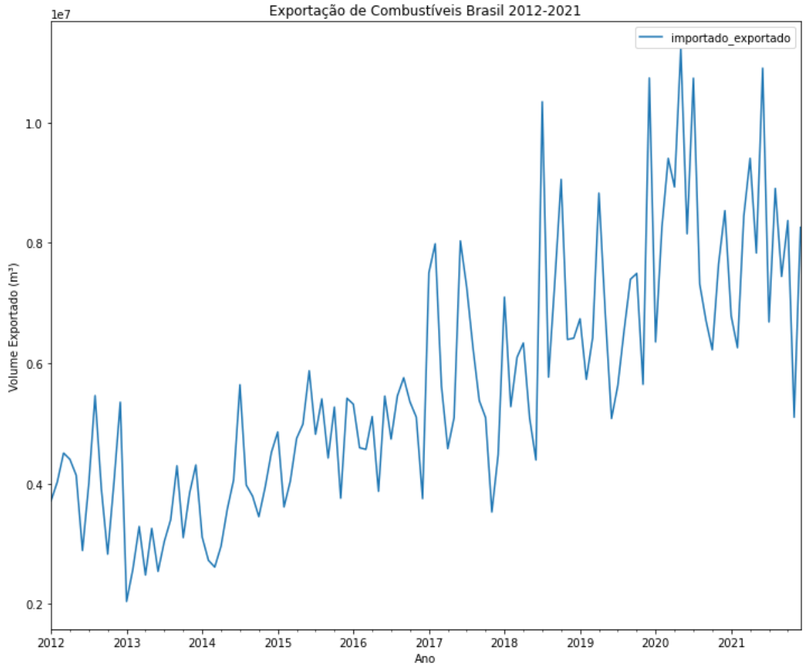
```
[ ]  1 dfder = dfder.loc[(dfder['produto'] == 'OUTROS ENERGÉTICOS') |
     2                    (dfder['produto'] == 'COQUE') | (dfder['produto'] == 'GASOLINA A') |
     3                    (dfder['produto'] == 'ÓLEO COMBUSTÍVEL') |
     4                    (dfder['produto'] == 'GASOLINA DE AVIAÇÃO') |
     5                    (dfder['produto'] == 'ÓLEO DIESEL') | (dfder['produto'] == 'GLP') |
     6                    (dfder['produto'] == 'QUEROSENE DE AVIAÇÃO')]
```

CÓDIGOS

Plotagem de gráficos utilizando a Biblioteca Pandas

Plot de Combustíveis do Brasil 2012-2021

```
[36]  1 ft = dfimex.loc[dfimex.operacao_comercial == 'EXPORTAÇÃO']
      2 ft.groupby('data').sum().plot.line(title='Exportação de Combustíveis Brasil 2012-2021',
      3                                    xlabel='Ano', ylabel='Volume Exportado (m³)', figsize=(9,7))
```



CÓDIGOS

Tratamentos utilizando a Biblioteca Pyspark

Conversão de tipos de colunas

```
[ ]
# Inserção do nome de todas as colunas que deveriam ser float mas são string em uma lista
colunas = ['margem_rev', 'media_dist', 'dp_dist',
           'preco_min_dist', 'preco_max_dist', 'coef_var_dist']

# Substituição de todas as strings '-' por 0
for i in colunas:
    dfpreco = dfpreco.withColumn(i, F.regexp_replace(i, '-', '0'))

dfpreco = dfpreco.withColumn("margem_rev",F.col("margem_rev").cast(FloatType()))\
                 .withColumn("media_dist",F.col("media_dist").cast(FloatType()))\
                 .withColumn("dp_dist",F.col("dp_dist").cast(FloatType()))\
                 .withColumn("preco_min_dist",F.col("preco_min_dist").cast(FloatType()))\
                 .withColumn("preco_max_dist",F.col("preco_max_dist").cast(FloatType()))\
                 .withColumn("coef_var_dist",F.col("coef_var_dist").cast(FloatType()))
```

CÓDIGOS

Envio de Datasets tratados para o MongoDB

Conexão ao usuário MongoDB a partir de certificado X.509

```
[ ]  1 uri = "mongodb+srv://erica-soulcode.7jfrfcs.mongodb.net/?authSource=%24external&authMechanism=MONGODB-X509&retryWrites=true&w=majority"
     2 client = MongoClient(uri,tls=True,tlsCertificateKeyFile='/content/X509-cert-8089168769125785989.pem')
     3
     4 db = client['projeto_final_combustiveis_viacolab']
```

Utilização de python puro para inserir e checar se os dados tratados foram enviados

```
[ ]  1 datasets = ['dfderivados_tratado','dfbio_tratado','dfproducaopetgas_tratado']
     2
     3 for x in datasets:
     4   colecao = db[f'{x}']
     5   df = pd.read_csv(f'https://storage.googleapis.com/projeto-final-equipe4/arquivos_trat/{x}')
     6   df_dict = df.to_dict('records')
     7   colecao.insert_many(df_dict)
     8   print(f'Database "{x}" adicionado ao mongoDB. Número de documentos criados: {colecao.count_documents({})}')
```

```
Database "dfderivados_trat" adicionado ao mongoDB. Número de documentos criados: 17264
Database "dfbio_trat" adicionado ao mongoDB. Número de documentos criados: 18051
Database "dfproducaopetgas_trat" adicionado ao mongoDB. Número de documentos criados: 5275
```

CÓDIGOS

Envio de Datasets para o Big Query

Utilização da biblioteca pandas_gbq para enviar o dataframe diretamente ao Big Query

```
[ ]  1 esquema = [{'name':'data','type':'DATETIME'},{'name':'produto','type':'STRING'},
     2            {'name':'operacao_comercial','type':'STRING'},
     3            {'name':'importado_exportado','type':'BIGNUMERIC'},
     4            {'name':'dispendio_receita','type':'BIGNUMERIC'}]
     5
     6 pandas_gbq.to_gbq(dfimex, 'projetofinal.df_impoexpo_tratado', project_id = 'sc-bc26-ed7',
     7                   if_exists = 'replace',table_schema=esquema, api_method="load_csv")
```

```
1it [00:03,  3.42s/it]
```

```
[ ]  1 esquema = [{'name':'data','type':'DATE'}]
     2 pandas_gbq.to_gbq(dfPandas, 'projetofinal.df_precos_tratado', project_id = 'sc-bc26-ed7',
     3                   if_exists = 'replace', table_schema = esquema)
```

BIG QUERY

Foram realizadas 10 (dez) queries para melhor compreensão dos dados constantes nos datasets

RUN SAVE SHARE SCHEDULE MORE

```
1  # Relação volume exportação/importartação e receita/custo - 2012-2021.
2
3  SELECT operacao_comercial,SUM(importado_exportado) AS volume_m3,SUM(dispendio_receita) AS
   receita_custo
4  FROM projetofinal.df_impoexpo_tratado
5  GROUP BY operacao_comercial
```

Row	operacao_comercial	volume_m3	receita_custo
1	EXPORTAÇÃO	569709706...	268567451...
2	IMPORTAÇÃO	568873278...	653500223...

PIPELINE APACHE BEAM

Instalação e importação das bibliotecas do Apache Beam no Colab Notebook

```
import apache_beam as beam
import os
from apache_beam.options.pipeline_options import PipelineOptions
from apache_beam.io.textio import WriteToText

colunas_bio = ['','regiao','uf','produto','volume_m3','data']
colunas_preco = ['','data','regiao','estado','produto','postos_pesquisados','uni_medida','media_rev','desvio_rev','preco_min_rev','preco_max_rev','margem_rev'

def lista_dicionario(elemento, colunas):
  return dict(zip(colunas, elemento))

def trata_data(elemento):
  # Recebe um dicionario e cria um novo campo com ANO-MES -  Retorna o mesmo dicionario com novo campo
  elemento['ano_mes']= '-'.join(elemento['data'].split('-')[:2])
  return elemento

def chave_uf(elemento):
#  Receber um dicionario -   Retorna uma tupla com estado e o elemento(UF, dicionario )
  chave = elemento['uf']
  return (chave, elemento)
```

PIPELINE APACHE BEAM

Parâmetros de conexão do Beam e conexão com Google Cloud

```
pipeline_options = {
    'project':'sc-bc26-ed7',
    'runner':'DataflowRunner',
    'region':'southamerica-east1',
    'staging_location':'gs://projeto-final-equipe4/beam/staging/',
    'temp_location':'gs://projeto-final-equipe4/beam/temp/',
    'template_location':'gs://projeto-final-equipe4/beam/models/modelo_batch'
}

serviceAccount = '/content/sc-bc26-ed7-adb0dc2607d9.json'
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount

pipeline_options = PipelineOptions.from_dictionary(pipeline_options)

p1 = beam.Pipeline(options=pipeline_options)
```

PIPELINE APACHE BEAM

A PCollection representa um conjunto de dados distribuídos no qual o pipeline do Beam opera

```
precos = (
    p1
    |'Extrair do CSV Preços'>> beam.io.ReadFromText('gs://projeto-final-equipe4/arquivos_trat/dfprecos.csv', skip_header_lines=1)
    |'Sep de dados Preços'>> beam.Map(lambda record: record.split(','))
    |'Filt por prod Preços'>> beam.Filter(lambda record: str(record[4])== 'ETANOL HIDRATADO')
    |'Tranformar lista para dic Preços'>>beam.Map(lista_dicionario, colunas_preco)
    |'Criar Campo ano_mes Preços'>>beam.Map(trata_data)
    |'Criar chave pelo est Preços'>> beam.Map(chave_estado)
    |'Agrupar estado Preços'>>beam.GroupByKey()
    |'Descompactar vol Preços'>>beam.FlatMap(media_rev)
    |'Media preços'>> beam.combiners.Mean.PerKey()
    |'Arredondar preços'>>beam.Map(arredonda)
    #|'Imprimir o resultado Dataset Preços'>> beam.Map(print)
)

resultado = (
    ({'volume_m3':biocombustiveis,'valor_media_rev':precos})
    |'Mesclar pcol'>>beam.CoGroupByKey()
    |'Filtrar dados vazios'>>beam.Filter(filtra_campos_vazios)
    |'Descompactar'>>beam.Map(descompactar_elementos)
    |'Preparar csv'>>beam.Map(preparar_csv, deliminator=',')
```

PIPELINE APACHE BEAM

Funcionamento da Pipeline

projeto-comb-final	Batch	Jan 10, 2023, 5:52:37 PM	5 min 21 sec	Jan 10, 2023, 5:47:16 PM	Succeeded

Job info

Resource metrics

Current vCPUs	1
Total vCPU time	0.055 vCPU hr
Current memory	3.75 GB
Total memory time	0.207 GB hr
Current HDD PD	25 GB
Total HDD PD time	1.378 GB hr
Current SSD PD	0 B
Total SSD PD time	0 GB hr
Total Shuffle data processed	12.61 MB
Billable Shuffle data processed	3.15 MB

PIPELINE BIG QUERY – MONGO DB

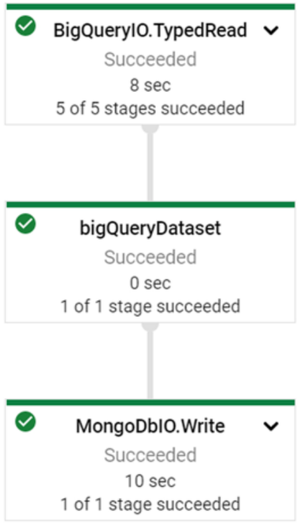
Modelo de pipeline do Dataflow pré-definido que integra o Big Query com o Mongo

Name	Type	End time	Elapsed time	Start time
dfcomb-etanol-trat	Batch	Jan 10, 2023, 6:16:19 PM	4 min 59 sec	Jan 10, 2023, 6:11:20 PM
dfprecos-tratado	Batch	Jan 10, 2023, 5:03:10 PM	8 min 12 sec	Jan 10, 2023, 4:54:58 PM
dfimport-export-trat	Batch	Jan 10, 2023, 4:24:57 PM	4 min 49 sec	Jan 10, 2023, 4:20:08 PM

Job info

Resource metrics

Current vCPUs	1
Total vCPU time	0.031 vCPU hr
Current memory	3.75 GB
Total memory time	0.116 GB hr
Current HDD PD	25 GB
Total HDD PD time	0.771 GB hr
Current SSD PD	0 B
Total SSD PD time	0 GB hr
Total Shuffle data processed	200 B
Billable Shuffle data processed	50 B



projeto_final_combustiveis_viabigquery

CREATE

LOGICAL DATA SIZE:	STORAGE SIZE:	INDEX SIZE:	TOTAL COLLECTIONS:
33.8MB	9.29MB	3.09MB	3

Collection Name	Documents	Logical Data Size	Avg Document Size	Storage Size	Indexes	Index Size	Avg Index Size
dfcomb_etanol_trat	3106	365.96KB	121B	152KB	1	120KB	120KB
dfimport-export_trat	2754	669.87KB	250B	204KB	1	120KB	120KB
dfprecos_tratado	87034	32.79MB	396B	8.94MB	1	2.86MB	2.86MB

INSIGHTS

- Na entressafra da cana-de-açúcar altera a importação e produção do Etanol
- A PPI em 2016 afetou o montante de importação de combustíveis
- Rio de Janeiro aparece em destaque em produção de Petróleo e Gás
- O ano de 2021 foi marcado por sucessivos aumentos nos preços dos combustíveis

Looker Studio

CUSTOS

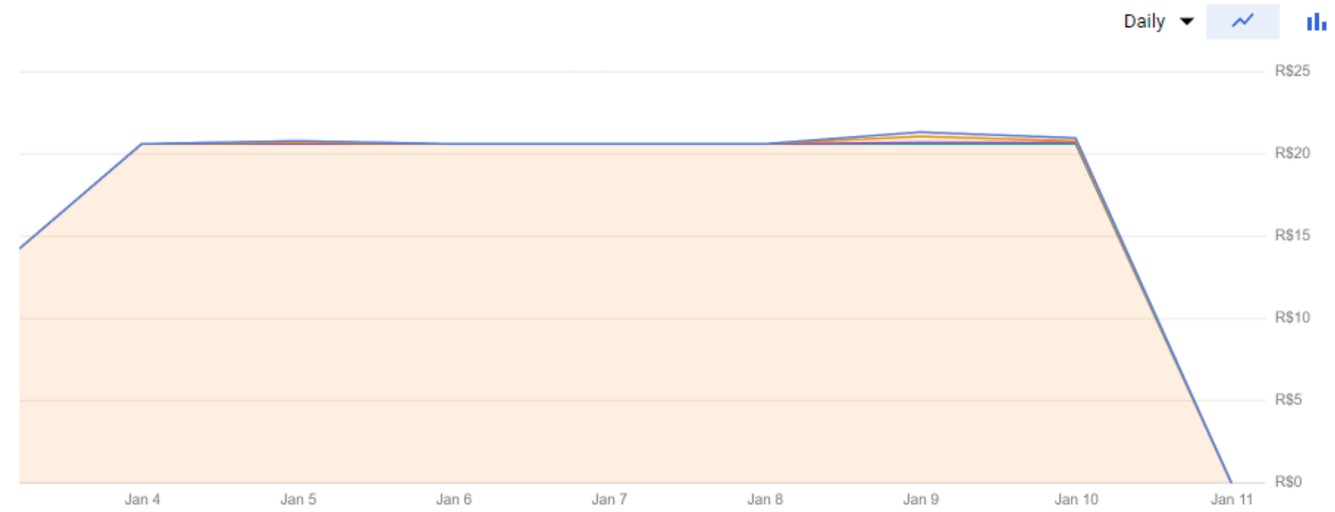
January 3 – 11, 2023 (total cost)

R$158.01

includes R$0.00 in credits

↑ –

R$158.01 over December 25, 2022 – January 2, 2023



Service	Cost	Discounts	Promotions and others	↓ Subtotal
Cloud SQL	R$156.80	–	–	R$156.80
Cloud Dataflow	R$0.59	–	–	R$0.59
Networking	R$0.52	–	–	R$0.52
Cloud Storage	R$0.09	–	–	R$0.09
Compute Engine	R$0.00	–	–	R$0.00

Sugestão de melhoria

- Melhor análise e utilização de máquina Cloud SQL

DataRoot

Contatos

Augusto Tonelli

augustoatonelli@gmail.com

augusto-tonelli

augustoTonelli

Caio Alves

caioitaloalves@gmail.com

caio-italo-alves

caioitalo

Érica Marçal

erica.elom@gmail.com

erica-marcal

Luan Sagara

luan.sagara@gmail.com

luan-sagara

LuanSagara

Nayara Bernardo

nayyarabernardo@gmail.com

nayyarabernardo

nayyarabernardo

SOUL CODE

Obrigado