

TRANSPORTE AÉREO

A photograph taken from an airplane window, showing a vast landscape of fields and roads below under a clear blue sky.

SOUL.CODE

TRANSPORTE AÉREO EM 2019

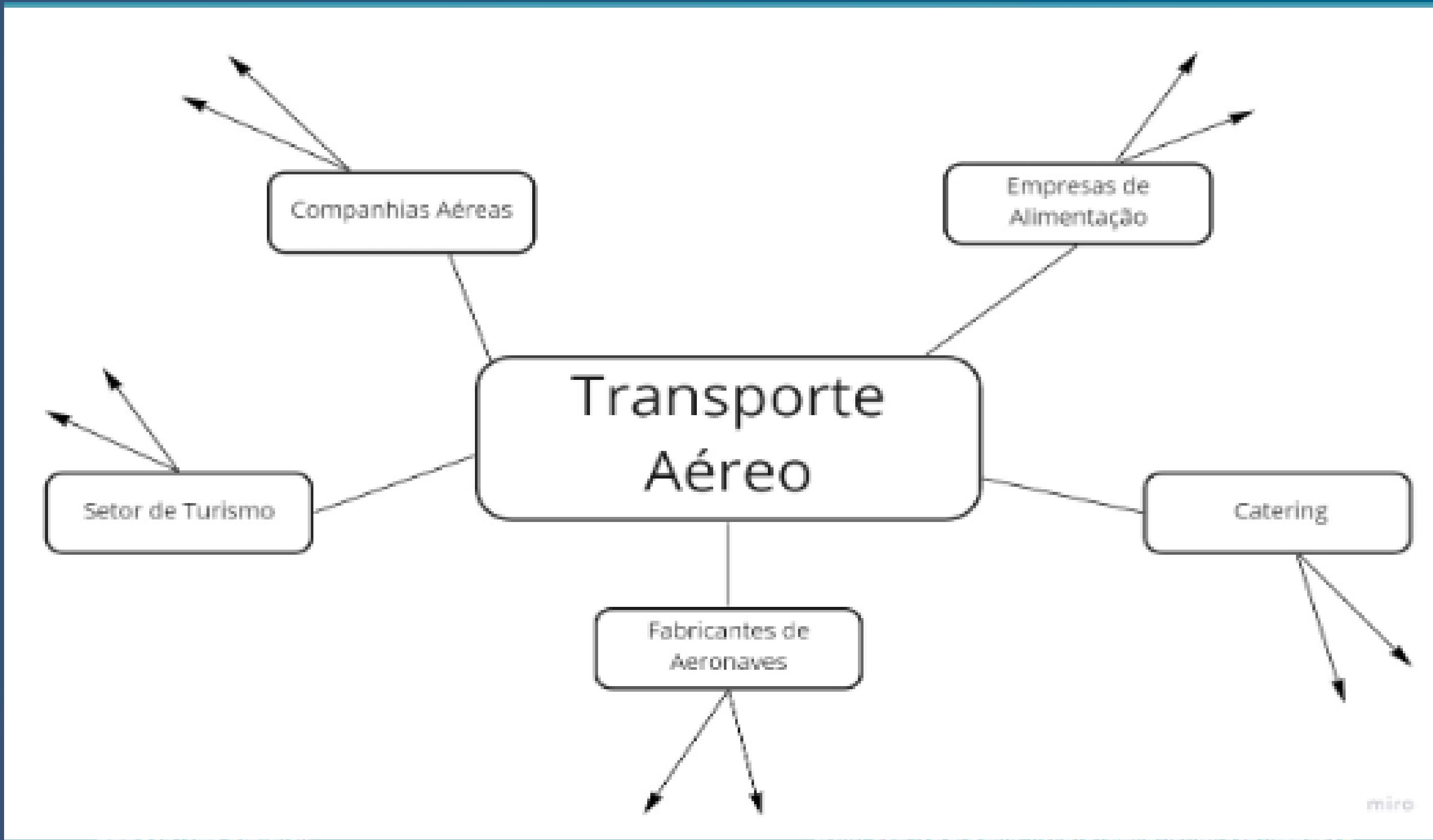


Agência Nacional de Aviação Civil (ANAC) Entrar

Mercado aéreo em 2019: maior número de passageiros transportados da série histórica

Mais de 119 milhões de passageiros foram transportadas ano passado no mercado doméstico e internacional.

Fonte: ANAC



TRANSPORTE AÉREO EM 2020



Covid-19:

Surgido na China no final de 2019, o SARS - COV -2, um vírus altamente contagioso, ficou conhecido como COVID - 19;

Pandemia:

- Caracterizada como Pandemia no dia **11 de março de 2020** pela OMS;
- Distanciamento social;
- Redução de viagens internacionais;
- Reduções de viagens nacionais.

IMPACTO DA COVID-19

No Transporte aéreo Brasileiro

Nosso objetivo neste projeto, é apresentar a vocês uma análise descritiva, trazendo algumas visões sobre o que aconteceu com o setor aéreo nas principais empresas aéreas do Brasil, respondendo as seguintes perguntas:

- Qual a Variação na demanda e oferta comparadas ao ano anterior?**
- Quais foram as Receitas? Houve Prejuízo dessas empresas?**
- O que os Dados estatísticos sobre a variação no valor e venda de passagens aéreas mostram?**



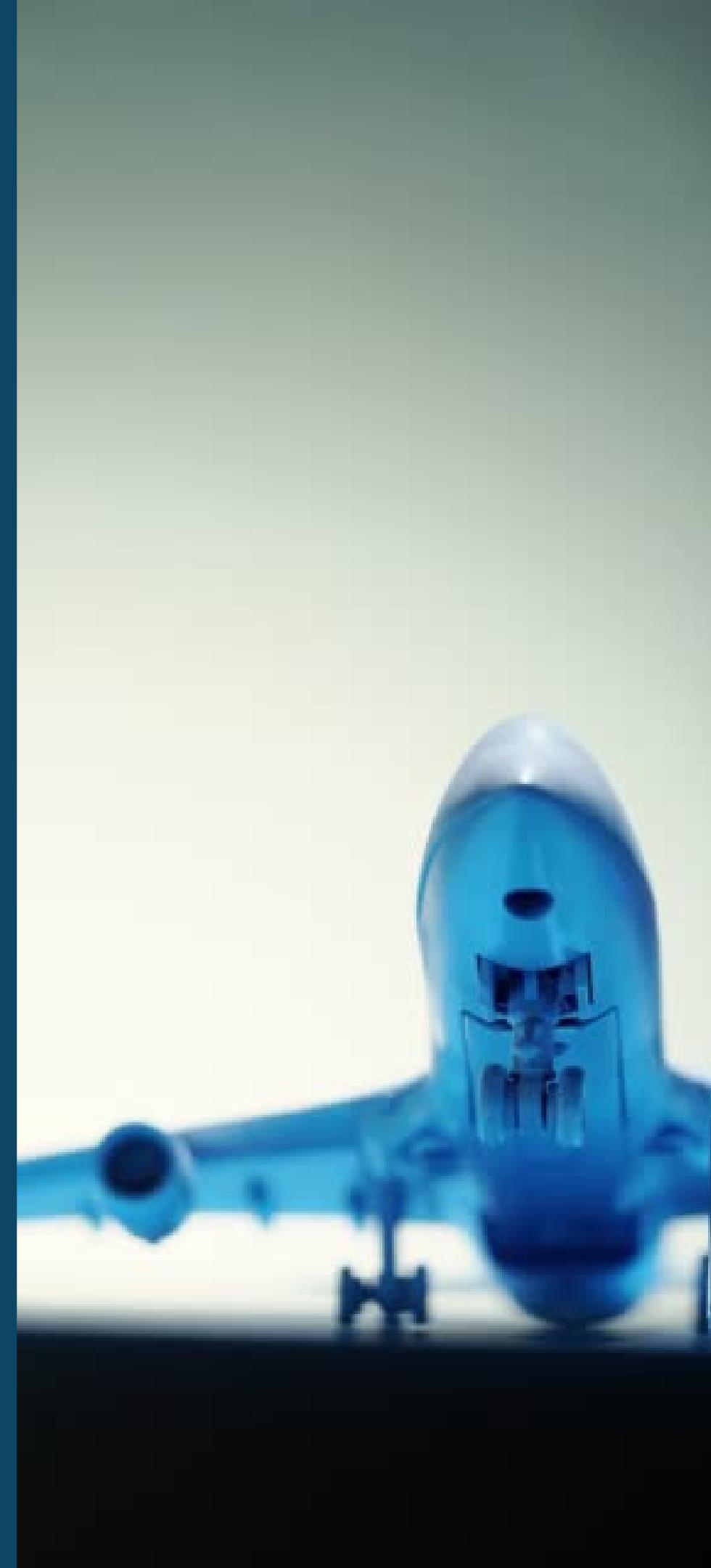
O PROJETO

Nosso objetivo é apresentar a vocês uma análise descritiva, trazendo algumas visões sobre o que aconteceu com o setor aéreo durante a pandemia do Coronavírus.

➤ Mapeamento da análise

Analisamos o impacto da Covid-19 nas principais empresas aéreas do Brasil:
Azul, Gol e Tam.

- Variação na demanda e oferta comparadas ao ano anterior.
- Receita e Prejuízo dessas empresas.
- Dados estatísticos sobre a variação no valor e venda de passagens aéreas.



METODOLOGIAS

aplicadas no desenvolvimento do projeto

Proj_Transportes_Aéreos

- ▶ Equipe
- ▶ Brainstorm Recorte Tema
- ▶ Fluxograma do projeto
- ▶ Daily Scrum (Todos os dias as 11:00)
- ▶ Mapeamento do problema de Pesquisa:
- ▶ DataSets que estamos trabalhando
- ▶ Colab Oficial
- ▶ Meet do projeto
- ▶ Budget do projeto

Workflow: Etapas de ETL

Board view

Etapas ETL

EXTRAÇÃO 1

- 📄 GCP - Cloud Storage
- 👤 Elidy IzzY 🏤 ricardo oliveira

+ New

VALIDAÇÃO (DATA VALIDATION) 2

- 📊 Pandas - Profiling
- 👤 Elidy IzzY

Análise Global dos Datasets -
PANDAS

Alguns aspectos do Scrum

01 Daily Scrum

02 Projeto dividido em Sprints

03 Acompanhamento pelo kanban

METODOLOGIAS

aplicadas no desenvolvimento do projeto

Cronograma

Board view

Acompanhamento das tarefas

Não Iniciado 3

[Nível SQL - SparkSQL](#)

 Luciana Bahia Sullca

[Análises no BIG QUERY](#)

[Criar Dashboard DATA STUDIO](#)

+ New

Fazendo 7

[Escolher os DataSets Iniciais](#)

 Elidy IzzY  Luciana Bahia Sullca

 ricardo oliveira  Jozi Moreira

[ETL PANDAS](#)

 Elidy IzzY  ricardo oliveira

[Workflow ETL](#)

 Elidy IzzY

[Análise PYSPARK](#)

 Jozi Moreira

Feito! 6

[Brainstorm com o tema](#)

 Elidy IzzY  ricardo oliveira

 Jozi Moreira  Luciana Bahia Sullca

[Definir Cronogramas, prazos e modelo de trabalho](#)

 Elidy IzzY  Luciana Bahia Sullca

 ricardo oliveira  Jozi Moreira

[Garimpar DataSets](#)

 Elidy IzzY  Luciana Bahia Sullca

 Jozi Moreira  ricardo oliveira

STACK DE FERRAMENTAS



Organização e produtividade:

NOTION
MIRO



Infraestrutura:

**GOOGLE CLOUD
STORAGE**
MYSQL

BIG QUERY
BANCOS NOSQL
MONGODB (MONGO ATLAS)

**GOOGLE COLAB (IDE DE
DESENVOLVIMENTO
COLABORATIVO)**

ORÇAMENTO



WORKFLOW ETL



ANAC
IBGE
GOV



EXTRAÇÃO

- Buscamos os datasets em repositórios oficiais
- Disponibilizamos no Cloud Storage
- Fizemos a conexão com o colab para trabalhar os dados



TRANSFORMAÇÃO

Transformação e geração dos novos dados em insights



VALIDAÇÃO

- Transformação em Pandas DataFrame
- Normalização dos dados para trabalhar a análise corretamente
- Limpeza dos Missing Values e Outliers



AGREGAÇÃO

Usamos o Big Query para obter análises estatísticas de dados em vários campos



CARREGAMENTO





EXTRAÇÃO

Navegador

[**+ CRIAR INTERVALO**](#) [EXCLUIR](#) [ATUALIZAR](#)

Filtro [Filtrar intervalos](#)

<input type="checkbox"/>	Nome	Criado em	Tipo de local	Local	Defa
<input type="checkbox"/>	transporte_aereo	28 de mar. de 2022 16:33:58	Region	us-central1 (lo...	Star

transporte_aereo

Local	Classe de armazenamento	Acesso público	Proteção
us-central1 (Iowa)	Standard	Sujeito a ACLs de objeto	Nenhum

OBJETOS **CONFIGURAÇÃO** **PERMISSÕES** **PROTEÇÃO** **CICLO DE VIDA**

Intervalos > transporte_aereo 

FAZER UPLOAD DE ARQUIVOS **CARREGAR PASTA** **criar pasta** **GERENCIAR RETENÇÕES** **FAZER O DOWNLOAD** **EXCLUIR**

Filtrar apenas pelo prefixo do nome ▾

 **Filtro** Filtrar objetos e pastas

<input type="checkbox"/>	Nome	Tamanho	Tipo	Criado 	Classe de armazenamento	Última modificação	Acesso público 
<input type="checkbox"/>	 chave_json/	—	Pasta	—	—	—	—
<input type="checkbox"/>	 dados_brutos/	—	Pasta	—	—	—	—
<input type="checkbox"/>	 dados_tratados/	—	Pasta	—	—	—	—

transporte_aereo

Local	Classe de armazenamento	Acesso público	Proteção
us-central1 (Iowa)	Standard	Sujeito a ACLs de objeto	Nenhum

OBJETOS CONFIGURAÇÃO PERMISSÕES PROTEÇÃO CICLO DE VIDA

Intervalos > transporte_aereo > dados_tratados □

FAZER UPLOAD DE ARQUIVOS CARREGAR PASTA CRIAR PASTA GERENCIAR RETENÇÕES FAZER O DOWNLOAD EXCLUIR

Filtrar apenas pelo prefixo do nome ▾

≡ Filtro Filtrar objetos e pastas

<input type="checkbox"/>	Nome	Tamanho	Tipo	Criado ?	Classe de armazenamento	Última modificação	Acesso público ?
<input type="checkbox"/>	df_demanda_normalizado.csv	39,5 KB	/dados_tratados	5 de abr...	Standard	8 de abr. de 202...	Público para a Internet
<input type="checkbox"/>	dfdemandatemporario.csv	292 B	text/plain	9 de abr...	Standard	9 de abr. de 202...	Não público
<input type="checkbox"/>	dfdemonst2019.csv	379 B	text/plain	9 de abr...	Standard	9 de abr. de 202...	Não público
<input type="checkbox"/>	dfdemonst2020.csv	347 B	text/plain	9 de abr...	Standard	9 de abr. de 202...	Não público
<input type="checkbox"/>	dfdiferencademanda.csv	16,1 KB	text/plain	9 de abr...	Standard	9 de abr. de 202...	Não público
<input type="checkbox"/>	dfpassagenstratado.csv	157,5 KB	text/plain	9 de abr...	Standard	9 de abr. de 202...	Não público



pandas

VALIDAÇÃO

▼ Tratamento de Dados do DF Demanda (Data Cleaning)

- O Python está lendo as informações no formato correto?
- Será que existe alguma coluna completamente vazia?
- Será que existe alguma informação em alguma linha vazia?

```
1 # Overview do DataSet com a função info()
2 df_demanda.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18747 entries, 0 to 18746
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   ano              18747 non-null   int64  
 1   mes              18747 non-null   int64  
 2   empresa_nome     18747 non-null   object  
 3   empresa_sigla    18747 non-null   object  
 4   empresa_pais     18714 non-null   object  
 5   natureza         18747 non-null   object  
 6   passageiros_pg    18742 non-null   float64 
 7   carga_correio_kg  18742 non-null   float64 
 8   rpk               18746 non-null   float64 
 9   ask               18746 non-null   float64 
dtypes: float64(4), int64(2), object(4)
```

Primeiros Tratamentos

```
1 df_demanda.info() # Overview do DataSet com a função info()  
  
[50] 1 # Estamos dropando as primeiras colunas que contém informações redundantes e outras que contém informações irrelevantes na construção das análises como sig  
2 df_demanda.drop(['carga_correio_kg'],axis=1,inplace=True)
```

```
[51] 1 df_demanda.columns
```

```
Index(['ano', 'mes', 'empresa_nome', 'empresa_sigla', 'empresa_pais',  
       'natureza', 'passageiros_pg', 'rpk', 'ask'],  
      dtype='object')
```

```
[52] 1 # Usamos a função dataframe.round() para arredondar todos os valores decimais no dataframe para 3 casas decimais.  
2 df_demanda.round(decimals=3)
```

	ano	mes	empresa_nome	empresa_sigla	empresa_pais	natureza	passageiros_pg	rpk	ask	editar
0	2000	1	AMERICAN AIRLINES, INC.	AAL	ESTADOS UNIDOS DA AMÉRICA	INTERNACIONAL	65102.0	419939072.0	667154432.0	
1	2000	1	ABAETÉ LINHAS AÉREAS S.A.	ABJ	BRASIL	DOMÉSTICA	428.0	233327.0	725085.0	
2	2000	1	AIR CANADA	ACA	CANADÁ	INTERNACIONAL	163.0	1335622.0	1679770.0	
3	2000	1	AEROFLOT-AEROLÍNEAS INTERNACIONAIS DA RÚSSIA	AFL	RÚSSIA	INTERNACIONAL	1634.0	15006158.0	22362600.0	
4	2000	1	SOCIÉTÉ AIR FRANCE	AFR	FRANÇA	INTERNACIONAL	18048.0	168573888.0	217613312.0	
...	
18742	2022	1	TOTAL LINHAS AÉREAS S.A.	TTL	BRASIL	DOMÉSTICA	0.0	0.0	0.0	

4s conclusão: 11:38

```
[ ] 1 # Para poder alterar valores utilizei 3 passos,
2 # 1 - removi os pontos
3 # 2 - tranformei as virgulas em pontos
4 # 3 - transformei o dado em float
5 df_demonst_2019['ValorApuradoInicioTrimestre'] = df_demonst_2019['ValorApuradoInicioTrimestre'].str.replace('.','')
6 df_demonst_2019['ValorApuradoInicioTrimestre'] = df_demonst_2019['ValorApuradoInicioTrimestre'].str.replace(',','.')
7 df_demonst_2019['ValorApuradoInicioTrimestre'] = df_demonst_2019['ValorApuradoInicioTrimestre'].astype(float)
8
9 df_demonst_2019['ValorApuradoFinalTrimestre'] = df_demonst_2019['ValorApuradoFinalTrimestre'].str.replace('.','')
10 df_demonst_2019['ValorApuradoFinalTrimestre'] = df_demonst_2019['ValorApuradoFinalTrimestre'].str.replace(',','.')
11 df_demonst_2019['ValorApuradoFinalTrimestre'] = df_demonst_2019['ValorApuradoFinalTrimestre'].astype(float)
```

```
<> /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: FutureWarning:
```

Usando Filtros para segmentar o DataFrame

```
1 # Segmentando o DataFrame,filtrando e dropando os vôos de natureza INTERNACIONAL  
2 df_demanda.drop(df_demanda.loc[df_demanda['natureza']!='DOMÉSTICA'].index, inplace=True)
```

```
[54] 1 # Conferindo a segmentação aplicada 'DOMÉSTICA'  
2 df_demanda['natureza'].unique()
```

```
array(['DOMÉSTICA'], dtype=object)
```

```
[55] 1 # Segmentando o DataFrame,filtrando e dropando as empresas que não são nacionais  
2 df_demanda.drop(df_demanda.loc[df_demanda['empresa_pais']!='BRASIL'].index, inplace=True)
```

```
[56] 1 # Conferindo a segmentação aplicada 'BRASIL'  
2 df_demanda['empresa_pais'].unique()
```

```
array(['BRASIL'], dtype=object)
```

```
[57] 1 # Segmentando o DataFrame,filtrando e dropando os dados que não correspondem aos anos de 2019 a 2021.  
2 df_demanda.drop(df_demanda.loc[df_demanda['ano'] < 2019].index, inplace=True)
```

Tratando Missin values

```
[65] 1 # Mostrar as colunas com mais valores ausentes
2 df_demanda.isnull().sum().sort_values(ascending=False)

ano          0
mes          0
empresa_nome 0
empresa_sigla 0
empresa_pais  0
natureza      0
passageiros_pg 0
rpk           0
ask            0
dtype: int64

[66] 1 #Exclusão controlada - excluir caso uma coluna tenha todos os valores missing, pois ela de fato não nos serve em nada
2 df_demanda = df_demanda.dropna(how = 'all')
```

```
[67] 1 # Contagem de Missing Values por coluna  
2 df_demanda.isnull().sum()
```

```
ano          0  
mes          0  
empresa_nome 0  
empresa_sigla 0  
empresa_pais  0  
natureza      0  
passageiros_pg 0  
rpk           0  
ask           0  
dtype: int64
```



TRANSFORMAÇÃO

▼ Análise detalhada com Pyspark

▼ Spark Session

```
[ ] 1 spark = (
2     SparkSession.builder
3         .master('local')
4         .appName('projetoFinal')
5         .config('spark.ui.port', '4050')
6         .config("spark.jars", 'https://storage.googleapis.com/hadoop-lib/gcs/gcs-connector-hadoop2-latest.jar')
7         .getOrCreate()
8     )
```

▼ Struct Type

*Obs: toda a etapa de Data Cleaning foi realizada com pandas: verificação de existência de dados inconsistentes, nulos e realização de limpeza. Assim como verificação de necessidade de drop em colunas ou linhas. *

```
[457] 1 #Criação do Schema de Dados
      2 schema = ( StructType([
      3     StructField('DescricaoIdentificador', StringType(), True),
      4     StructField('CodigoEmpresa', StringType(), True),
      5     StructField('ValorApuradoInicioTrimestre', DoubleType(), True),
      6     StructField('ValorApuradoFinalTrimestre', DoubleType(), False),
      7     StructField('AnoDeReferencia', IntegerType(), True),
      8     StructField('TrimestreDeReferencia', StringType(), True),
      9     ])
     10 )
```

Converterndo DataFrame Pandas em Pyspark

```
1 dfspk_demonst_2019 = spark.createDataFrame(df_demonst_2019,)
```

```
[101] 1 dfspk_demonst_2019.printSchema()
```

```
root
| -- DescricaoIdentificador: string (nullable = true)
| -- CodigoEmpresa: string (nullable = true)
| -- ValorApuradoInicioTrimestre: string (nullable = true)
| -- ValorApuradoFinalTrimestre: double (nullable = true)
| -- AnoDeReferencia: long (nullable = true)
| -- TrimestreDeReferencia: string (nullable = true)
```

Confirmação que os dados nulos foram tratados no pandas

```
[ ] 1 for cd in dfspk_demanda.columns:  
2 | print(cd, dfspk_demanda.filter(F.col(cd).isNull()).count())
```

```
ano 0  
mes 0  
empresa_nome 0  
empresa_sigla 0  
empresa_pais 0  
natureza 0  
passageiros_pg 0  
rpk 0  
ask 0
```

```
[ ] 1 for c in dfspk_demonst_2019.columns:  
2 | print(c, dfspk_demonst_2019.filter(F.col(c).isNull()).count())
```

```
DescricaoIdentificador 0  
CodigoEmpresa 0  
ValorApuradoInicioTrimestre 0  
ValorApuradoFinalTrimestre 0  
AnoReferencia 0
```

▼ dfspk_demanda

```
✓ [1] 1 dfspk_demanda1 = dfspk_demanda.select(F.col('ano')).distinct().show()
```

```
+---+  
| ano|  
+---+  
|2021|  
|2022|  
|2020|  
|2019|  
+---+
```

```
✓ [111] 1 dfspk_demanda2 = dfspk_demanda.select(F.col('empresa_nome')).distinct().show(truncate=False)
```

```
+-----+  
|empresa_nome|  
+-----+  
|AZUL LINHAS AÉREAS BRASILEIRAS S/A  
|GOL LINHAS AÉREAS S.A. (EX- VRG LINHAS AÉREAS S.A.)  
|CONNECT LINHAS AÉREAS S.A. (ANTIGA CONNECT TÁXI AÉREO LTDA.)  
|OCEANAIR LINHAS AÉREAS S.A. (AVIANCA)  
|MODERN TRANSPORTE AEREO DE CARGA S.A  
|AZUL CONECTA LTDA. (EX TWO TAXI AEREO LTDA)  
|PASSAREDO TRANSPORTES AÉREOS S.A.  
|SIDERAL LINHAS AÉREAS LTDA.  
|TOTAL LINHAS AÉREAS S.A.
```

▼ dfspk_demanدا

```
[108] 1 dfspk3 = dfspk2.withColumn('DiferençaEntreOfertaeDemandaa', F.col('ask') - F.col('rpk'))
```

```
[109] 1 dfspk3.show()
```

ano	passageiros_pg	ask	rpk	DiferençaEntreOfertaeDemandaa
2019	1124.0	708075.0	263657.0	444418.0
2019	2051290.0	2.335065344E9	1.914865024E9	4.2020032E8
2019	3265539.0	4.411227648E9	3.729938176E9	6.81289472E8
2019	0.0	0.0	0.0	0.0
2019	0.0	0.0	0.0	0.0
2019	928837.0	1.23053824E9	1.064538304E9	1.65999936E8
2019	9575.0	9166818.0	5882102.0	3284716.0
2019	28681.0	2.993802E7	2.0763424E7	9174596.0
2019	0.0	0.0	0.0	0.0
2019	2637462.0	3.41729152E9	2.867724288E9	5.49567232E8
2019	5522.0	3267360.0	2942152.0	325208.0
2019	1154.0	635886.0	278344.0	357542.0
2019	1868621.0	1.955350528E9	1.599448704E9	3.55901824E8
2019	2477819.0	3.258213376E9	2.684719104E9	5.73494272E8
2019	0.0	0.0	0.0	0.0
2019	0.0	0.0	0.0	0.0
2019	814189.0	1.080639872E9	9.27911232E8	1.5272864E8
2019	9742.0	8812376.0	5830416.0	2981960.0
2019	26908.0	2.9624268E7	1.7205092E7	1.2419176E7
2019	139.0	63856.0	57076.0	6780.0

only showing top 20 rows

▶ dfspk_passagens

[] 42 células ocultas

```
[ ] 1 #Diferença de valor das passagens entre 2019 e 2020
2 df2 = df1.withColumn('DiferençaTarifas', F.col('tarifa_med_2019') - F.col('tarifa_med_2020'))
3 df2.show()
```

qtd_assent_2019	qtd_assent_2020	tarifa_med_2019	tarifa_med_2020	DiferençaAssentos	DiferençaTarifas
1027294.0	455330.0	318.2	243.99	571964.0	74.2099999999998
1026248.0	455700.0	320.06	246.94	570548.0	73.12
626155.0	309355.0	312.93	254.07	316800.0	58.86000000000014
626063.0	308367.0	318.53	269.18	317696.0	49.34999999999966
518955.0	286796.0	414.05	359.07	232159.0	54.9800000000002
514890.0	285945.0	407.77	365.54	228945.0	42.2299999999996
503103.0	213790.0	263.6	215.78	289313.0	47.8200000000002
500765.0	213185.0	261.83	209.32	287580.0	52.5099999999999
476663.0	235973.0	393.82	281.96	240690.0	111.8600000000001
475169.0	235381.0	386.46	278.12	239788.0	108.3399999999997
445548.0	174857.0	249.26	209.31	270691.0	39.9499999999999
440816.0	171853.0	237.84	200.05	268963.0	37.7899999999999
405378.0	229264.0	355.49	240.2	176114.0	115.2900000000002
404581.0	287035.0	523.76	421.86	117546.0	101.8999999999998
402319.0	227217.0	343.74	230.33	175102.0	113.41
401973.0	287374.0	531.78	431.52	114599.0	100.2599999999999
362485.0	179296.0	262.47	244.35	183189.0	18.12000000000033
360890.0	177030.0	262.6	250.49	183860.0	12.11000000000014
341491.0	212186.0	477.74	448.03	129305.0	29.71000000000036
337759.0	213103.0	500.65	449.63	124656.0	51.0199999999998

only showing top 20 rows

Filtros por Empresas

▶ dfspk_demonst_2019

▶ 94 células ocultas

▶ dfspk_demonst_2020

▶ 96 células ocultas

Verificação dos dados da empresa Azul

```
[ ] 1 #Verificação dos Ativos da Empresa (tudo o que pode ser transformado em dinheiro em pouco tempo)
2 dfa2019 = dfspk_demonst_2019.filter((F.col('CodigoEmpresa') == 'AZU') & (F.col('DescricaoIdentificador') == 'Ativo'))
```

```
[ ] 1 dfa2019.show()
```

DescricaoIdentificador	CodigoEmpresa	ValorApuradoInicioTrimestre	ValorApuradoFinalTrimestre	AnoDeReferencia	TrimestreDeReferencia
Ativo	AZU	9.5541064E9	1.46722028E10	2019	T1 - 1º Trimestre
Ativo	AZU	1.49148785E10	1.59752407E10	2019	T2 - 2º Trimestre
Ativo	AZU	1.59752407E10	1.67896822E10	2019	T3 - 3º Trimestre
Ativo	AZU	1.39479747E10	1.71121623E10	2019	T4 - 4º Trimestre

```
[ ] 1 #Soma da coluna Valor Apurado Final por Trimestre, transformando em apenas uma coluna
2 dfazul_ativo2019 = dfa2019.agg(F.sum("ValorApuradoFinalTrimestre"))
```

```
[ ] 1 #Modificando o nome da coluna para melhor visualização ao final das análises
2 dfazul_ativo2019 = dfazul_ativo2019.withColumnRenamed('sum(ValorApuradoFinalTrimestre)', 'AtivoTotalAno')
```

```
[ ] 1 dfazul_ativo2019.show()
```

AtivoTotalAno
6.4549287936E10

Convertendo para Pandas dfspk_demonst_2019

Azul

```
[ ] 1 spark.conf.set('spark.sql.execution.arrow.enable','true')
2 df0azul_2019 = dfazul_ativo2019.select('*').toPandas()
```

```
[ ] 1 df0azul_2019
```

AtivoTotalAno

```
0 64549287936.00
```

```
[ ] 1 spark.conf.set('spark.sql.execution.arrow.enable','true')
2 df1azul_2019 = dfazul_lucro2019.select('*').toPandas()
```

```
[ ] 1 df1azul_2019
```

LucroTotalAno

```
0 3845668648.00
```

- Concatenando (juntando) os dataframes de tudo que a empresa perdeu, transformando assim em um dataframe

```
[429] 1 dftam22020 = pd.concat([df4tam_2020, df5tam_2020], axis = 1)
```

```
[430] 1 dftam22020
```

	Custos	Despesas
0	-29588210068.66	-4949436791.12

```
[431] 1 dftam2020 = pd.concat([dftam12020, dftam22020], axis = 1)
```

```
1 dftam2020
```

	Ativo	Lucro	Caixa	Custos	Despesas
0	60840512711.97	-3657355389.73	5150748072.90	-29588210068.66	-4949436791.12

- Transformando os dataframes das 3 empresas em um apenas

```
✓ [411] 1 df2020 = pd.concat([dfazul2020, dfgol2020, dftam2020])
```

```
✓ [412] 1 #Criando nova coluna com o nemes das empresas  
2 df2020.insert(0,'Empresa', ['Azul', 'Gol', 'Tam'])
```

```
✓ [413] 1 df2020
```

	Empresa	Ativo	Lucro	Caixa	Passivo	Custos	Despesas	⋮
0	Azul	60776337512.72	-6499077.66	5601163977.52	60776337512.72	-15678599762.95	-3199931816.18	
0	Gol	48117841867.49	1165633130.33	975247689.03	48117841867.45	-15218691312.62	-3530543358.01	
0	Tam	60840512711.97	-3657355389.73	5150748072.90	60840512711.97	-29588210068.66	-4949436791.12	



AGREGAÇÃO

```
[397] 1 #Criando uma exibição temporária do dataframe.  
2 dfspk_demonst_2019.createOrReplaceTempView("Demonst_2019")  
  
[398] 1 #Criando uma exibição temporária do dataframe.  
2 dfspk_demonst_2020.createOrReplaceTempView("Demonst_2020")  
  
[401] 1 spark.sql('SELECT * FROM Demonst_2019').show(truncate=False)  
[402] 1 spark.sql('SELECT * FROM Demonst_2020').show(truncate=False)  
  
[401] 1 #--EMPRESAS AÉREAS ANALIZADAS.  
2 spark.sql("SELECTCodigoEmpresa FROM Demonst_2019 WHERE CodigoEmpresa IN ('AZU','GLO','TAM') GROUP BY CodigoEmpresa").show()  
  
+-----+  
|CodigoEmpresa|  
+-----+  
|      GLO|  
|      TAM|  
|      AZU|  
+-----+
```

Analizando:

- Ativo (Ativo Circulante + Passivo Não Circulante)
- Passivo
- Patrimônio Líquido

```

2 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2019 WHERE
3 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2019 WHERE
4
5 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2020 WHERE
6 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2020 WHERE

```

2º TRIMESTRE

Codigo Empresa	Trimestre De Referencia	Descricao Identificador	Ano De Referencia	Valor Apurado Final Trimestre
AZU	T2 - 2º Trimestre	Ativo	2019	1.59752407E10
AZU	T2 - 2º Trimestre	Ativo Circulante	2019	3.5042473E9
AZU	T2 - 2º Trimestre	Ativo Não Circulante	2019	1.24709939E10

Codigo Empresa	Trimestre De Referencia	Descricao Identificador	Ano De Referencia	Valor Apurado Final Trimestre
AZU	T2 - 2º Trimestre	Passivo	2019	1.59752407E10
AZU	T2 - 2º Trimestre	Passivo Circulante	2019	5.7966479E9
AZU	T2 - 2º Trimestre	Passivo Não Circu...	2019	1.25558364E10
AZU	T2 - 2º Trimestre	Patrimônio Líquido	2019	-2.37724288E9

Codigo Empresa	Trimestre De Referencia	Descricao Identificador	Ano De Referencia	Valor Apurado Final Trimestre
AZU	T2 - 2º Trimestre	Ativo	2020	1.44652605E10
AZU	T2 - 2º Trimestre	Ativo Circulante	2020	3.14480896E9
AZU	T2 - 2º Trimestre	Ativo Não Circulante	2020	1.13204511E10

Codigo Empresa	Trimestre De Referencia	Descricao Identificador	Ano De Referencia	Valor Apurado Final Trimestre
AZU	T2 - 2º Trimestre	Passivo	2020	1.44652605E10
AZU	T2 - 2º Trimestre	Passivo Circulante	2020	9.7030697E9
AZU	T2 - 2º Trimestre	Passivo Não Circu...	2020	1.81939323E10
AZU	T2 - 2º Trimestre	Patrimônio Líquido	2020	-1.34317404E10

```

2 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2019")
3 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2019")
4
5 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2020")
6 spark.sql ("SELECTCodigoEmpresa, TrimestreDeReferencia, DescricaoIdentificador, AnoDeReferencia, ValorApuradoFinalTrimestre FROM Demonst_2020")

```

CodigoEmpresa	TrimestreDeReferencia	DescricaoIdentificador	AnoDeReferencia	ValorApuradoFinalTrimestre
AZU	T4 - 4º Trimestre	Ativo	2019	1.71121623E10
AZU	T4 - 4º Trimestre	Ativo Circulante	2019	3.68938163E9
AZU	T4 - 4º Trimestre	Ativo Não Circulante	2019	1.34227814E10

4º TRIMESTRE

CodigoEmpresa	TrimestreDeReferencia	DescricaoIdentificador	AnoDeReferencia	ValorApuradoFinalTrimestre
AZU	T4 - 4º Trimestre	Passivo	2019	1.71121623E10
AZU	T4 - 4º Trimestre	Passivo Circulante	2019	6.9694208E9
AZU	T4 - 4º Trimestre	Passivo Não Circu...	2019	1.54676183E10
AZU	T4 - 4º Trimestre	Patrimônio Líquido	2019	-5.3248758E9

CodigoEmpresa	TrimestreDeReferencia	DescricaoIdentificador	AnoDeReferencia	ValorApuradoFinalTrimestre
AZU	T4 - 4º Trimestre	Ativo	2020	1.47250995E10
AZU	T4 - 4º Trimestre	Ativo Circulante	2020	4.5220465E9
AZU	T4 - 4º Trimestre	Ativo Não Circulante	2020	1.02030531E10

CodigoEmpresa	TrimestreDeReferencia	DescricaoIdentificador	AnoDeReferencia	ValorApuradoFinalTrimestre
AZU	T4 - 4º Trimestre	Passivo	2020	1.47250995E10
AZU	T4 - 4º Trimestre	Passivo Circulante	2020	9.9262894E9
AZU	T4 - 4º Trimestre	Passivo Não Circu...	2020	1.76669532E10
AZU	T4 - 4º Trimestre	Patrimônio Líquido	2020	-1.28681431E10

1 #--TOTAL DE RECEITA POR TRIMESTRE SEPARADO POR COMPANHIA AÉREA.			
2 spark.sql("SELECT AnoDeReferencia, DescricaoIdentificador, CódigoEmpresa, TrimestreDeReferencia, ValorApuradoInicioTrimestre FROM Dem			
3 spark.sql("SELECT AnoDeReferencia, DescricaoIdentificador, CódigoEmpresa, TrimestreDeReferencia, ValorApuradoInicioTrimestre FROM Dem			
4 spark.sql("SELECT AnoDeReferencia, DescricaoIdentificador, CódigoEmpresa, TrimestreDeReferencia, ValorApuradoInicioTrimestre FROM Dem			
+-----+-----+-----+-----+			
AnoDeReferencia DescricaoIdentificador		CódigoEmpresa TrimestreDeReferencia ValorApuradoInicioTrimestre	
+-----+-----+-----+-----+			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU T1 - 1º Trimestre 7.9479488E8			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU T2 - 2º Trimestre 1.11656346E9			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU T3 - 3º Trimestre 1.45538458E9			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU T4 - 4º Trimestre 1.47213837E9			
+-----+-----+-----+-----+			
+-----+-----+-----+-----+			
AnoDeReferencia DescricaoIdentificador		CódigoEmpresa TrimestreDeReferencia ValorApuradoInicioTrimestre	
+-----+-----+-----+-----+			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM T1 - 1º Trimestre 5.17135968E8			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM T2 - 2º Trimestre 9.6515117E8			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM T3 - 3º Trimestre 1.01468224E9			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM T4 - 4º Trimestre 9.3626176E8			
+-----+-----+-----+-----+			
+-----+-----+-----+-----+			
AnoDeReferencia DescricaoIdentificador		CódigoEmpresa TrimestreDeReferencia ValorApuradoInicioTrimestre	
+-----+-----+-----+-----+			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO T1 - 1º Trimestre 1.00310616E8			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO T2 - 2º Trimestre 7.9859208E7			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO T3 - 3º Trimestre 2.761376E8			
2019 (=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO T4 - 4º Trimestre 3.40832E8			
+-----+-----+-----+-----+			

```
1 #--TOTAL DE RECEITA POR TRIMESTRE SEPARADO POR COMPANHIA AÉREA.
```

```
2 spark.sql("SELECT AnoDeReferencia, DescricaoIdentificador,CodigoEmpresa, TrimestreDeReferencia, ValorApuradoInicioTrimestre FROM Demanda WHERE DescricaoIdentificador = '(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO' AND CodigoEmpresa = 'AZU'")  
3 spark.sql("SELECT AnoDeReferencia, DescricaoIdentificador, CódigoEmpresa, TrimestreDeReferencia, ValorApuradoInicioTrimestre FROM Demanda WHERE DescricaoIdentificador = '(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO' AND CodigoEmpresa = 'TAM'")  
4 spark.sql("SELECT AnoDeReferencia, DescricaoIdentificador, CódigoEmpresa, TrimestreDeReferencia, ValorApuradoInicioTrimestre FROM Demanda WHERE DescricaoIdentificador = '(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO' AND CodigoEmpresa = 'GLO'")
```

AnoDeReferencia	DescricaoIdentificador	CodigoEmpresa	TrimestreDeReferencia	ValorApuradoInicioTrimestre
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU		T1 - 1º Trimestre	4.72797536E8
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU		T2 - 2º Trimestre	1.42733018E9
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU		T3 - 3º Trimestre	1.26632858E9
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO AZU		T4 - 4º Trimestre	2.43470771E9

AnoDeReferencia	DescricaoIdentificador	CodigoEmpresa	TrimestreDeReferencia	ValorApuradoIriicioTrimestre
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM		T1 - 1º Trimestre	1.2935881E9
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM		T2 - 2º Trimestre	2.00774106E9
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM		T3 - 3º Trimestre	9.640272E8
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO TAM		T4 - 4º Trimestre	8.8539174E8

AnoDeReferencia	DescricaoIdentificador	CodigoEmpresa	TrimestreDeReferencia	ValorApuradoIriicioTrimestre
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO		T1 - 1º Trimestre	1.89905232E8
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO		T2 - 2º Trimestre	2.778856E8
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO		T3 - 3º Trimestre	3.80456E8
2020	(=) CAIXA E EQUIVALENTES DE CAIXA NO FINAL DO PERÍODO GLO		T4 - 4º Trimestre	1.27001016E8

BigQuery

Analysis ^

- SQL workspace
- Data transfers
- Scheduled queries
- Analytics Hub

Migration ^

- SQL translation

Administration ^

- Monitoring
- Capacity management
- BI Engine

Release Notes

FEATURES & INFO SHORTCUT

Explorer EDITOR DEMAND...

Type to search

Viewing pinned projects.

Projeto_Final

- Demandas
- Demandas_ASK_2019 (selected)
- Demandas_ASK_2020
- Demandas_AZU_2020
- Demandas_AZU_2021
- Demandas_GLO_2020
- Demandas_GLO_2021
- Demandas_RPK_2019
- Demandas_RPK_2020
- Demandas_TAM_2020
- Demandas_TAM_2021
- Demonstrativo_2019
- Demonstrativo_2019...
- Demonstrativo_2019...
- Demonstrativo_2020
- Demonstrativo_2020...
- Demonstrativo_2020...
- Passagens
- Passagens_MedAss...
- Passagens_MedTarif...

Demandas_A... QUERY SHARE COPY SNAPSHOT

SCHEMA DETAILS PREVIEW

Table info

Table ID	projeto-transporte-aereo.Proyecto_Final.Demandas_ASK_2019
Table size	87 B
Long-term storage size	0 B
Number of rows	3
Created	Apr 7, 2022, 1:29:19 PM UTC-3
Last modified	Apr 7, 2022, 1:29:19 PM UTC-3
Table expiration	NEVER
Data location	us-central1
Description	

PERSONAL HISTORY PROJECT HISTORY SAVED QUERIES

Projeto-Transporte-Aereo ▾ Search Products, resources, docs (/) ▾

RTCUT DISABLE EDITOR TABS

DEMONS... X DEMONS... X

RUN SAVE SHARE SCHEDULE MORE This query will process

```
1 CREATE OR REPLACE TABLE `projeto-transporte-aereo.Projeto_Final.Demonstrativo_2020_Lucro-Caixa` AS (
2 SELECT Empresa, Lucro, Caixa FROM `projeto-transporte-aereo.Projeto_Final.Demonstrativo_2020`
3 );
```

...

DISABLE EDITOR TABS

EDITOR X MEDIA D... X

RUN SAVE SHARE SCHEDULE MORE This query will process 15

```
1 CREATE OR REPLACE TABLE `projeto-transporte-aereo.Projeto_Final.Passagens_MedTarifas_2019-2020` AS (
2 SELECT
3     AVG(tarifa_med_2019) AS Media_Tarifa_2019,
4     AVG(tarifa_med_2020) AS Media_Tarifa_2020,
5     AVG(Diferen_aTarifas) AS Media_Difrenca
6 FROM `projeto-transporte-aereo.Projeto_Final.Passagens`
7 );
```

DISABLE EDITOR TABS

COMPAR... ▾ DEMAND... ▾

RUN

SAVE ▾

SHARE ▾

SCHEDULE ▾

MORE ▾

This query will p

```
1 CREATE OR REPLACE TABLE `projeto-transporte-aereo.Projeto_Final.Demandas_RPK_2020` AS (
2   SELECT
3     empresa_sigla,
4     rpk,
5     mes,
6     ano
7   FROM `projeto-transporte-aereo.Projeto_Final.Demandas`
8   WHERE empresa_sigla IN ('AZU', 'GLO', 'TAM')
9   AND
10    mes = "Março"
11   AND
12    ano = 2020
13 );
```



CARREGAMENTO



Conexão GCP

Carregamento GCP

```
[ ] 1 dfdiferencademandato_csv("dfdiferencademandato.csv", index=False)
2 serviceAccount = '/content/projeto-transporte-aereo-e8f7fb440c38.json' #CHAVE
3 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
4 client = storage.Client()
5 bucket = client.get_bucket('transporte_aereo') #nome da sua bucket
6 bucket.blob('dados_tratados/dfdiferencademandato.csv').upload_from_string(dfdiferencademandato.to_csv(index=False)) #salva na bucket
```

```
[ ] 1 dfspkdemandatratado.to_csv("dfspkdemandatratado.csv", index=False)
2 serviceAccount = '/content/projeto-transporte-aereo-e8f7fb440c38.json' #CHAVE
3 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
4 client = storage.Client()
5 bucket = client.get_bucket('transporte_aereo') #nome da sua bucket
6 bucket.blob('dados_tratados/dfspkdemandatratado.csv').upload_from_string(dfspkdemandatratado.to_csv(index=False)) #salva na bucket
```

```
[ ] 1 df2019.to_csv("dfdemonst2019.csv", index=False)
2 serviceAccount = '/content/projeto-transporte-aereo-e8f7fb440c38.json' #CHAVE
3 os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
4 client = storage.Client()
5 bucket = client.get_bucket('transporte_aereo') #nome da sua bucket
6 bucket.blob('dados_tratados/dfdemonst2019.csv').upload_from_string(df2019.to_csv(index=False)) #salva na bucket
```

Conexão MongoDB

Mongo Atlas

DF Demandas

```
[336] 1 #Fazendo conexão com Mongo Atlas
2 client = pymongo.MongoClient("mongodb+srv://elidy:918273@cluster0.24szl.mongodb.net/myFirstDatabase?retryWrites=true&w=majority")

[473] 1 db = client['transportes_aereo'] # coloca o DB
2 colecao = db.dataset_Demandas
3 df_demanda_dici = df_demanda.to_dict('records')
4 colecao.insert_many(df_demanda_dici)

<pymongo.results.InsertManyResult at 0x7f9bbe10e820>
```

DF Diferença demanda

```
[475] 1 db = client['transportes_aereo'] # coloca o DB
2 colecao = db.dataset_Diferencademandad
3 dfdiferencademandad_dici = dfdiferencademandad.to_dict('records')
4 colecao.insert_many(dfdiferencademandad_dici)
```

er... :   

[+ Create Database](#)

 NAMESPACES

transportes_aereo

- dataset_Demandas**
- dataset_Demonst2019
- dataset_Demonst2020
- dataset_Diferencade...
- dataset_Passagens

transportes_aereo.dataset_Demandas

STORAGE SIZE: 888KB TOTAL DOCUMENTS: 15429 INDEXES TOTAL SIZE: 500KB

[Find](#) [Indexes](#) [Schema Anti-Patterns 0](#) [Aggregation](#) [Search Indexes •](#)

[INSERT DOCUMENT](#)

FILTER { field: 'value' } [» OPTIONS](#) [Apply](#) [Reset](#)

QUERY RESULTS 1-20 OF MANY

```
_id: ObjectId("624ca45689bcf9a8fa8147b0")
ano: 2019
mes: "Janeiro"
empresa_nome: "PASSAREDO TRANSPORTES AÉREOS S.A."
```

[!\[\]\(0f8b5a308a48a1a8a9a8168be0f729ec_img.jpg\) PREVIOUS](#) [1-20 of many results](#) [!\[\]\(86436a659492401fa4d8c9275c6e6372_img.jpg\) NEXT](#)

Conexão MySQL

```
✓ 0s 1 # realizando conexão com my sql no GCP
2 conexao = mysql.connector.connect(host='34.71.205.119',user='root',password='123456',db='transporteaereo')
3 cursor = conexao.cursor()
4 engine = create_engine("mysql+pymysql://root:123456@34.71.205.119/transporteaereo")  
+ Código + Texto ↑ ↓ ⌂ ⚙ ⌂ ⌂ ⌂ ⌂ :  
0 1 # Criação do Banco de Dados no Mysql
2 query = 'CREATE DATABASE transporteaereo'
3 cursor.execute(query)
4 conexao.commit()  
✓ [331] 1 cursor.execute('Show tables')
2 cursor.fetchall()  
✓ [321] 1 # comunicando com o DF para enviar para o Mysql
2 df1_demonst_2020 = pd.read_csv(path3, sep = ';',skiprows=1)  
✓ [322] 1 df1_demonst_2020.columns
```

Google Cloud Platform Projeto-Transporte-Aereo Search Products, resources, docs (/) 13

Overview EDIT IMPORT EXPORT RESTART STOP DELETE CLONE

All instances > projetotransportesaereo

projetotransportesaereo

MySQL 5.7

1 hour 6 hours 1 day **7 days** 30 days Custom

Chart CPU utilization

UTC-3 Apr 6 Apr 7 Apr 8 Apr 9 Apr 10 Apr 11

0% 5% 10%

Connect to this instance

Public IP address: 34.71.205.119

Connection name: projeto-transporte-aereo:us-central1:projetotransportesa...

Need help connecting?

Review the documentation to learn about the many ways to connect to your instance. [Learn more](#)

To connect using gcloud, [OPEN CLOUD SHELL](#)

Configuration

vCPUs	Memory	SSD storage
4	26 GB	100 GB

Database version is MySQL 5.7.36

Auto storage increase is enabled

Automated backups are enabled

Point-in-time recovery is enabled

Located in us-central1-f

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help:' or '\h' for help. Type '\c' to clear the current input statement.

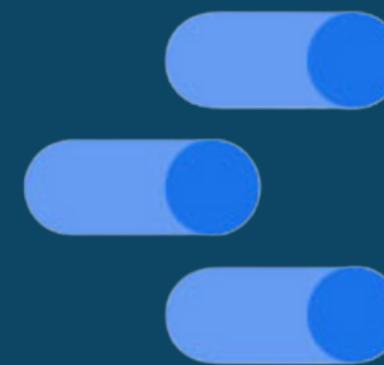
```
mysql> show database;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'database' at line 1
mysql> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
| sys            |
| transporteaereo   |
+-----+
5 rows in set (0.03 sec)

mysql> use transporteaereo
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

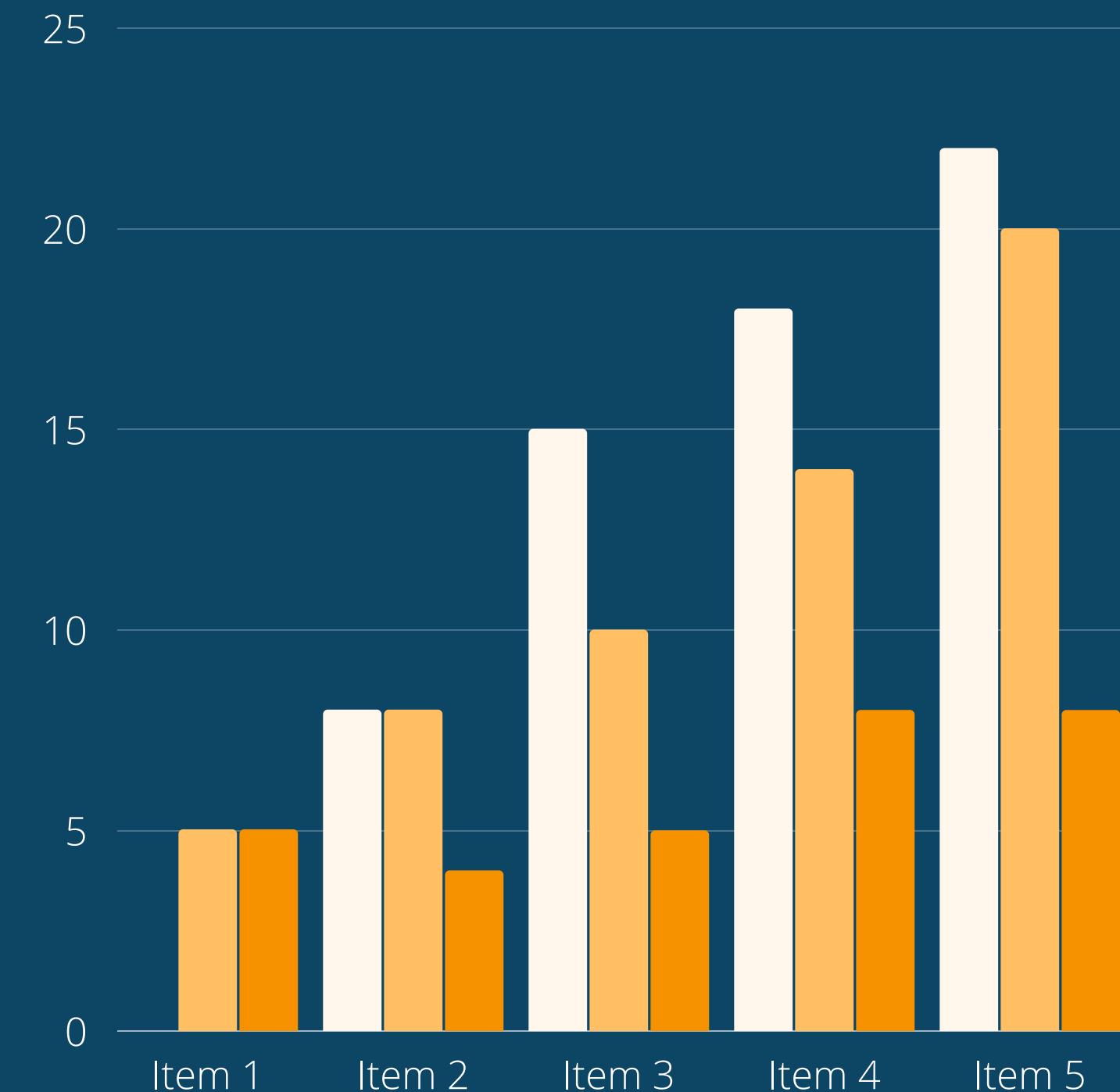
Database changed
mysql> show tables;
+-----+
| Tables_in_transporteaereo |
+-----+
| Demanda_e_Oferata_01      |
| demonstrativo_2019         |
| demonstrativo_2020         |
| passagens_2019_2020        |
+-----+
4 rows in set (0.03 sec)

mysql> |
```

VISUALIZAÇÃO DOS DADOS



Google
Data Studio



O setor passou por fortes turbulências e, assim como a maioria de nós, um dos piores períodos da sua história. A Covid-19 deixou sim, “sequelas” no transporte aéreo Brasileiro!

CONCLUSÃO

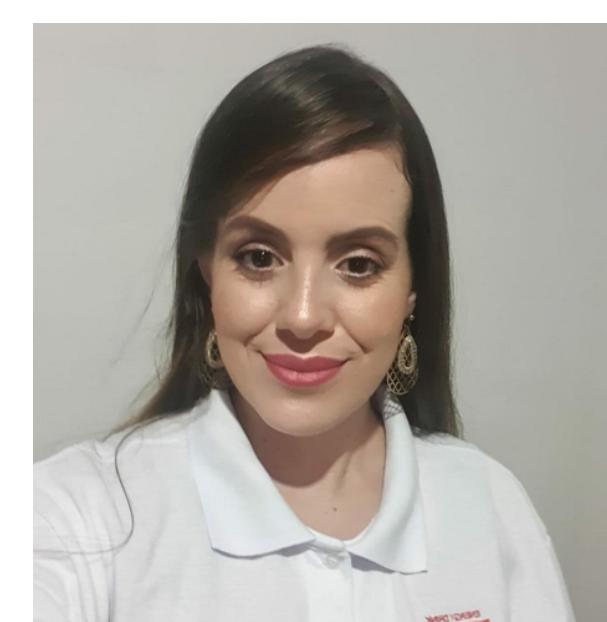
- Reduções de viagens;
- Turismo paralisado;
- Fronteiras fechadas;
- Viagens canceladas;
- Enormes prejuízos



OBRIGADA!



ÉLIDY IZIDIO



JOZI MOREIRA



**LUCIANA BAHIA
SULLCA**



RICARDO OLIVEIRA

EQUIPE

Autores:

Élidy Izidio

Jozi Moreira

Luciana Bahia Sullca

Ricardo Oliveira

Orientador:

Prof. Bismark William (SoulCode)