

# • Acidentes • Terrestres

Projeto Final do Curso de  
Engenharia de Dados

SoulCode Academy

**Projeto Final**  
**SoulCode Academy**

**Trabalho Final do Curso de Engenharia de Dados**  
**Turma BC-12**

Projeto desenvolvido pelos alunos:  
Alyne Cristina  
Eveline Marques  
Gustavo Santoro  
Matheus Reis

## Requisitos

Data sets públicos que abrangem o tema Acidentes Terrestres que contenham informações sobre acidentes, as rodovias onde mais acontecem e dos quais seja possível analisar por ano, rodovia e estados as características dos acidentes, suas causas e número de pessoas envolvidas.

Deverão ser encontrados e tratados no mínimo dois data sets sobre o assunto em questão. Seguindo alguns passos como:

- Os dados devem ser de formatos diferentes;
- Deve-se usar pandas, PySpark e Spark SQL para transformações e consultas;
- Armazenamento dos dados brutos em na Cloud SQL com MySql;
- Os dados devem estar disponíveis na Google Cloud Storage e BigQuery;
- Os dados tratados devem ser enviados para o MongoDB;

## Origem dos Dados

O dados utilizados no processo de ETL desse documento foram colhidos na plataforma de [Dados Abertos da Polícia Rodoviária Federal](#). Foram utilizados os dados sobre Acidentes no período de 2021 à 2017 disponibilizados em arquivos separados por 'Agrupados por Ocorrências' e 'Agrupados por Pessoas'. Os arquivos estão em formato 'zip', divididos por ano em que as ocorrências foram registradas.

## Destino dos Dados

Os arquivos foram armazenados na Google Cloud Storage (Datalake). Para isso foi criada a uma bucket com nome 'acidentes\_terrestres' e a pasta 'dados\_brutos'.

Apos os tratamentos feitos com pandas e PySpark os dados foram enviados para a BigQuery e para MongoDB. O dados tratados e normalizados foram armazenados no Datalake citado na pasta 'dados\_tratados.'





## Previsão de Gastos de acordo com a GCP

Estimativa de custo de acordo com a Google Cloud

Link calculadora: <https://cloud.google.com/products/calculator#id=e468548e-326a-44ae-a7ab-6b81cbbb4f6a>

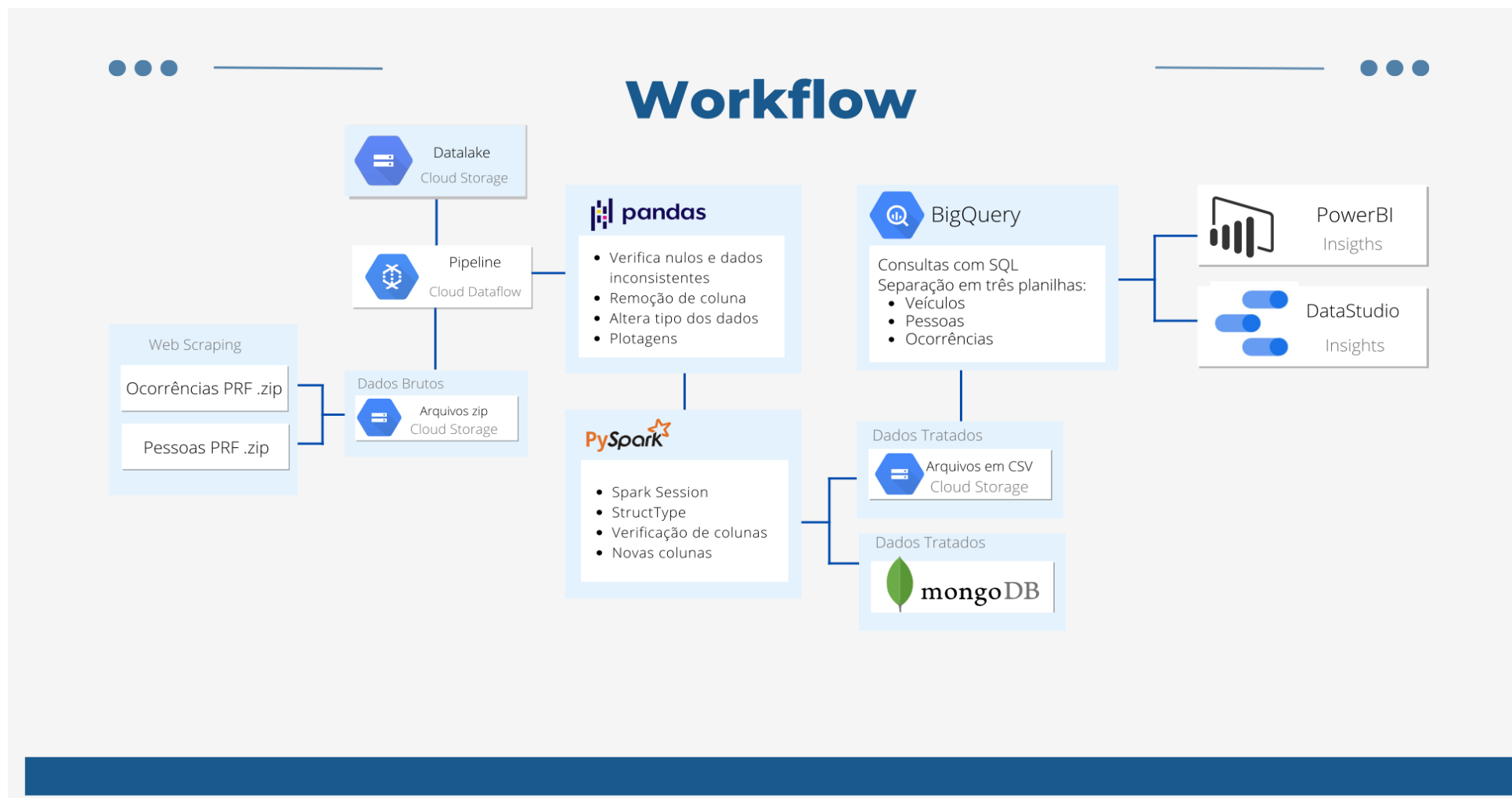
### Your Estimated Bill Projeto Final

**Estimated Monthly Cost: USD 22.74 (R\$ 102.00)**

 1x Standard Storage	Standard Storage	5 GiB	USD 0.18
info_ocorrencias	BigQuery	78.55 GiB	USD 1.37
info_pesspas	BigQuery	40 GiB	USD 0.00
info_veiculos	BigQuery	40.97 GiB	USD 0.00
Dataproc	projeto-final	48	USD 0.48
 1 x Dataproc master node	n1-standard-4	6 total hours per month	USD 2.18
 1 x Dataproc worker nodes	n1-standard-4	6 total hours per month	USD 2.18
 1x	Persistent Disk - Dataproc	8.219178082191782 GiB	USD 0.49
Dataflow	1 x n1-standard-1 workers in Batch Mode	1	USD 0.11
db-standard-1	10 GB	130.35714285714283 total hours per month	USD 15.75

**Total Estimated Monthly Cost**

**USD  
22.74**



## Importação/Extração dos dados

A extração dos dados se deu através de um algoritmo, através de bibliotecas de raspagem de dados, agrega todos os elementos HTML do site da PRF em um dicionário, onde é possível extrair os arquivos compactados.

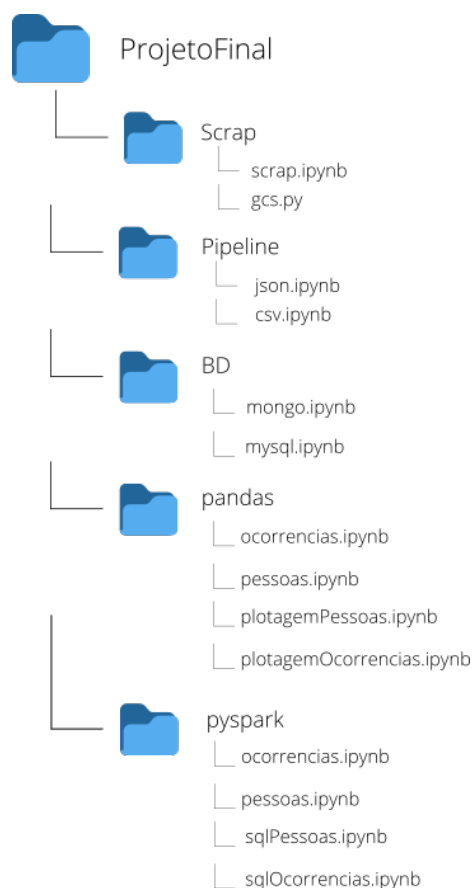
## Ferramentas Utilizadas

Foram utilizados pandas e PySpark para o tratamento, normalização dos dados e remoção de colunas e dados inconsistentes e plotagem de gráficos com análises iniciais.

Com SparkSql foram feitas buscas que guiaram a elaboração dos relatórios e insights para análise posterior no DataStudio e PowerBi. No BigQuery foram disponibilizados os dados tratados.

Para visualizar os relatórios acesse os links para o [PowerBI](#) e [DataStudio](#)

## Arquivos



## Dicionário dos Dados

### Acidentes Agrupados por ocorrências

NOME DA VARIÁVEL	DESCRIÇÃO
<i>id</i>	Variável com valores numéricos, representando o identificador do acidente.
<i>data_inversa</i>	Data da ocorrência no formato dd/mm/aaaa.
<i>dia_semana</i>	Dia da semana da ocorrência. Ex.: Segunda, Terça, etc.
<i>horario</i>	Horário da ocorrência no formato hh:mm:ss.
<i>uf</i>	Unidade da Federação. Ex.: MG, PE, DF, etc.
<i>br</i>	Variável com valores numéricos, representando o identificador da BR do acidente.
<i>km</i>	Identificação do quilômetro onde ocorreu o acidente, com valor mínimo de 0,1 km e com a casa decimal separada por ponto.
<i>municipio</i>	Nome do município de ocorrência do acidente
<i>causa_acidente</i>	Identificação da causa principal do acidente. Neste conjunto de dados são excluídos os acidentes com a variável causa principal igual a “Não”.
<i>tipo_acidente</i>	Identificação do tipo de acidente. Ex.: Colisão frontal, Saída de pista, etc. Neste conjunto de dados são excluídos os tipos de acidentes com ordem maior ou igual a dois. A ordem do acidente demonstra a sequência cronológica dos tipos presentes na mesma ocorrência.
<i>classificação_acidente</i>	Classificação quanto à gravidade do acidente: Sem Vítimas, Com Vítimas Feridas, Com Vítimas Fatais e Ignorado.
<i>fase_dia</i>	Fase do dia no momento do acidente. Ex. Amanhecer, Pleno dia, etc.
<i>sentido_via</i>	Sentido da via considerando o ponto de colisão: Crescente e decrescente.
<i>condição_meteorologica</i>	Condição meteorológica no momento do acidente: Céu claro, chuva, vento, etc.
<i>tipo_pista</i>	Tipo da pista considerando a quantidade de faixas: Dupla, simples ou múltipla.
<i>tracado_via</i>	Descrição do traçado da via.
<i>uso_solo</i>	Descrição sobre as características do local do acidente: Urbano=Sim;Rural=Não.
<i>latitude</i>	Latitude do local do acidente em formato geodésico decimal.
<i>longitude</i>	Longitude do local do acidente em formato geodésico decimal.
<i>pessoas</i>	Total de pessoas envolvidas na ocorrência.
<i>mortos</i>	Total de pessoas mortas envolvidas na ocorrência.
<i>feridos_leves</i>	Total de pessoas com ferimentos leves envolvidas na ocorrência.
<i>feridos_graves</i>	Total de pessoas com ferimentos graves envolvidas na ocorrência.

<i>feridos</i>	Total de pessoas feridas envolvidas na ocorrência (é a soma dos feridos leves com os graves).
<i>ilesos</i>	Total de pessoas ilesas envolvidas na ocorrência.
<i>ignorados</i>	Total de pessoas envolvidas na ocorrência e que não se soube o estado físico.
<i>veiculos</i>	Total de veículos envolvidos na ocorrência.

### *Acidentes Agrupados por pessoas*

NOME DA VARIÁVEL	DESCRIÇÃO
<i>id</i>	Variável com valores numéricos, representando o identificador do acidente.
<i>pesid</i>	Variável com valores numéricos, representando o identificador da pessoa envolvida.
<i>data_inversa</i>	Data da ocorrência no formato dd/mm/aaaa.
<i>dia_semana</i>	Dia da semana da ocorrência. Ex.: Segunda, Terça, etc.
<i>horario</i>	Horário da ocorrência no formato hh:mm:ss.
<i>uf</i>	Unidade da Federação. Ex.: MG, PE, DF, etc.
<i>br</i>	Variável com valores numéricos, representando o identificador da BR do acidente.
<i>km</i>	Identificação do quilômetro onde ocorreu o acidente, com valor mínimo de 0,1 km e com a casa decimal separada por ponto.
<i>municipio</i>	Nome do município de ocorrência do acidente
<i>causa_acidente</i>	Identificação da causa principal do acidente. Neste conjunto de dados são excluídos os acidentes com a variável causa principal igual a “Não”.
<i>tipo_acidente</i>	Identificação do tipo de acidente. Ex.: Colisão frontal, Saída de pista, etc. Neste conjunto de dados são excluídos os tipos de acidentes com ordem maior ou igual a dois. A ordem do acidente demonstra a sequência cronológica dos tipos presentes na mesma ocorrência.
<i>classificação_acidente</i>	Classificação quanto à gravidade do acidente: Sem Vítimas, Com Vítimas Feridas, Com Vítimas Fatais e Ignorado.
<i>fase_dia</i>	Fase do dia no momento do acidente. Ex. Amanhecer, Pleno dia, etc.
<i>sentido_via</i>	Sentido da via considerando o ponto de colisão: Crescente e decrescente.
<i>condição_meteorologica</i>	Condição meteorológica no momento do acidente: Céu claro, chuva, vento, etc.
<i>tipo_pista</i>	Tipo da pista considerando a quantidade de faixas: Dupla, simples ou múltipla.
<i>tracado_via</i>	Descrição do traçado da via.



<i>uso_solo</i>	Descrição sobre as características do local do acidente: Urbano=Sim;Rural=Não.
<i>id_veiculo</i>	Variável com valores numéricos, representando o identificador do veículo envolvido.
<i>tipo_veiculo</i>	Tipo do veículo conforme Art. 96 do Código de Trânsito Brasileiro. Ex.: Automóvel, Caminhão, Motocicleta, etc.
<i>marca</i>	Descrição da marca do veículo.
<i>ano_fabricacao_veiculo</i>	Ano de fabricação do veículo, formato aaaa
<i>tipo_envolvido</i>	Tipo de envolvido no acidente conforme sua participação no evento. Ex.: condutor, passageiro, pedestre, etc
<i>estado_fisico</i>	Condição do envolvido conforme a gravidade das lesões. Ex.: morto, ferido leve, etc.
<i>idade</i>	Idade do envolvido. O código “-1” indica que não foi possível coletar tal informação.
<i>sexo</i>	Sexo do envolvido. O valor “inválido” indica que não foi possível coletar tal informação.
<i>ilesos</i>	Valor binário que identifica se o envolvido foi classificado como ileso.
<i>feridos_leves</i>	Valor binário que identifica se o envolvido foi classificado como ferido leve.
<i>feridos_graves</i>	Valor binário que identifica se o envolvido foi classificado como ferido grave.
<i>mortos</i>	Valor binário que identifica se o envolvido foi classificado como morto.
<i>latitude</i>	Latitude do local do acidente em formato geodésico decimal.
<i>longitude</i>	Longitude do local do acidente em formato geodésico decimal.

## Pandas

### Remoção de nulos e colunas

- As colunas 'regional', 'delegacia' e 'uop' foram removidas por não serem necessárias para as análises
- A coluna 'horário' não é necessária, as análises serão feitas a partir da coluna 'fase\_dia'
- Ocorrências que serão removidas [109, 100046, 182369, 260357, 331815]. Na tabela Pessoas 'pesid' estava nulo

```
for coluna in df.columns:
    nulo= df[coluna].isna().sum() / df.shape[0]*100
    if(nulo!=0):
        print(f"{coluna:{2}}:-----> {nulo:.2f}%")
```

```
br:-----> 0.19%
km:-----> 0.19%
uop:-----> 3.13%
```

```
df.drop(['delegacia', 'uop', 'regional', 'horario'], axis = 1, inplace=True)
```

```
df.dropna(subset=['br', 'km'], inplace=True)
```

Remove ocorrencias que tiveram inconsistências na tabela pessoas

```
lista=[109, 100046, 182369, 260357, 331815, 1494, 1494, 2272, 2427]
df=(df[~df.id.isin(lista)]).reset_index(drop=True)
```

```
df.shape
```

```
(353605, 26)
```

### Transforma longitude e latitude para float

```
df['data_inversa']=pd.to_datetime(df['data_inversa'])
df['ano']=pd.DatetimeIndex(df['data_inversa']).year
```

```
df.loc[df['ano']<2021, 'latitude'] = df['latitude'].str.replace(',', '.')
df.loc[df['ano']<2021, 'longitude'] = df['longitude'].str.replace(',', '.')
df=df.drop('ano', axis=1)
df['latitude']=df['latitude'].astype(float)
df['longitude']=df['longitude'].astype(float)
```

## Pandas

```
#trocando valores nulos das colunas marca e ano_fabricacao_veiculo por 'Não informado'
df.marca.replace((pd.NA, 'Não informado'), inplace = True)
```

```
#preenchendo vazios para que se sejam possíveis as análises
df.ano_fabricacao_veiculo.replace((pd.NA, 0), inplace = True)
```

```
#Corrigindo a coluna Dia Semana
df['dia_semana'].replace(['sexta'],'sexta-feira',inplace=True)
df['dia_semana'].replace(['segunda'],'segunda-feira',inplace=True)
df['dia_semana'].replace(['quinta'],'quinta-feira',inplace=True)
df['dia_semana'].replace(['quarta'],'quarta-feira',inplace=True)
df['dia_semana'].replace(['terça'],'terça-feira',inplace=True)
```

Transformação de 'km' para float

```
df['km']=df['km'].str.replace(',','.')
df['km']=df['km'].astype(float)
```

```
#Corrigindo coluna Sexo
df['sexo'].replace(['Ignorado'],'Não Informado',inplace=True)
```

```
#Definindo os tipos de cada coluna
df['id']=df['id'].astype(int)
df['pesid']=df['pesid'].astype(int)
df['ano_fabricacao_veiculo']=df['ano_fabricacao_veiculo'].astype(int)
df['idade']=df['idade'].astype(str)
df['id_veiculo']=df['id_veiculo'].astype(int)
df['dia_semana']=df['dia_semana'].astype(str)
df['uf']=df['uf'].astype(str)
df['km']=df['km'].astype(float)
df['municipio']=df['municipio'].astype(str)
df['tipo_veiculo']=df['tipo_veiculo'].astype(str)
df['estado_fisico']=df['estado_fisico'].astype(str)
df['fase_dia']=df['fase_dia'].astype(str)
df['tipo_envolvido']=df['tipo_envolvido'].astype(str)
df['sexo']=df['sexo'].astype(str)
df['data_inversa'] = pd.to_datetime(df['data_inversa'])
```

```
# renomeando para melhor assimilação
df.rename(columns = {'pesid':'id_pessoa', 'data_inversa':'data', 'br':'rodovia', 'marca':'marca_modelo'}, inplace=True)
```

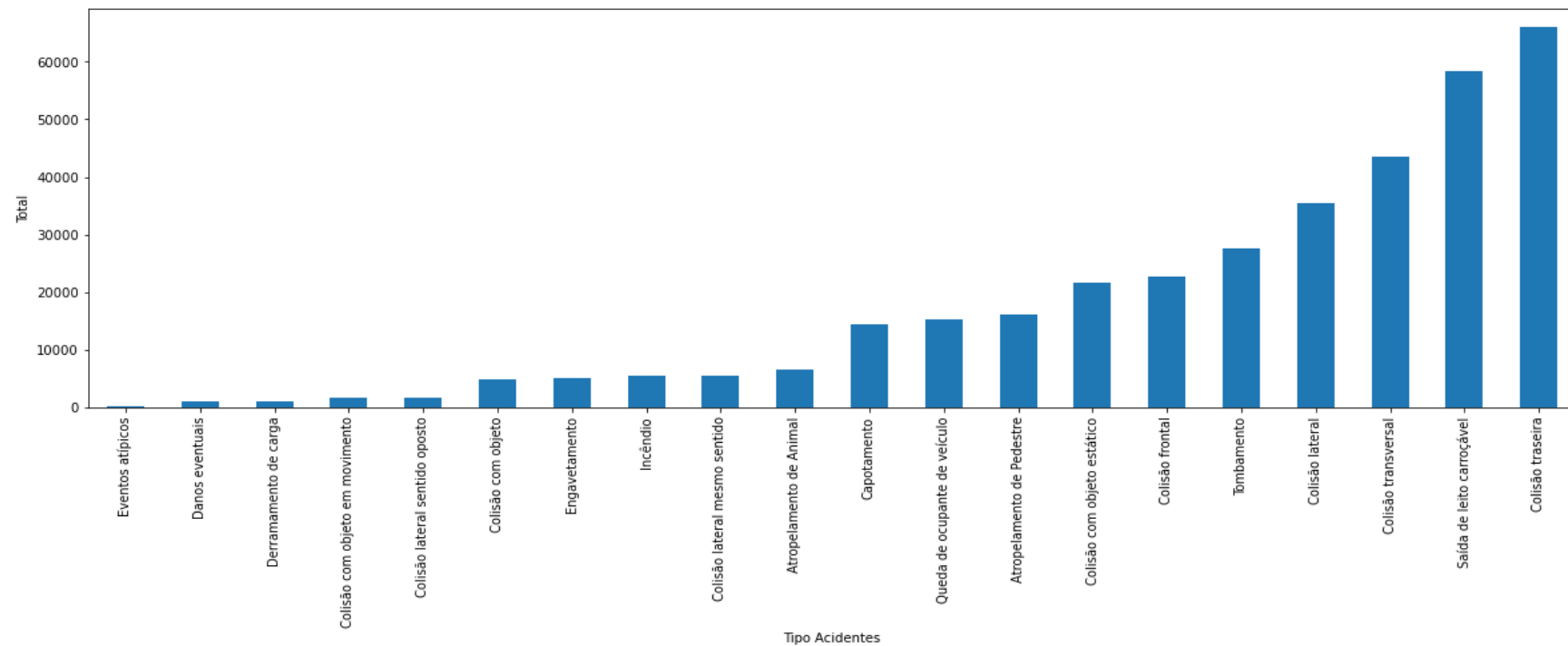
## Plotagens Pandas

### 4. Pandas

```
: from google.cloud import storage
import pandas as pd
import os
os.environ["GOOGLE_APPLICATION_CREDENTIALS"]="sca-at-b5179177a346.json"

: df1.groupby('tipo_acidente').id.count().sort_values().plot(kind='bar', figsize=(20,6), xlabel= "Tipo Acidentes",ylabel='Total' )

: <AxesSubplot:xlabel='Tipo Acidentes', ylabel='Total'>
```



## PySpark

### 5. Consultas

```
] : #Quantas pessoas se envolveram em acidentes durante o periodo de 2017 a 2021
dfs.select(F.countDistinct(F.col('id_pessoa')).alias('Pessoas envolvidas')).show(truncate=False)

+-----+
|Pessoas envolvidas|
+-----+
|749876           |
+-----+

] : #De que forma se envolveram
dfs.groupBy(dfs.tipo_envolvido.alias('Forma de Envolvimento')).count().orderBy('count', ascending=False).show()

+-----+-----+
|Forma de Envolvimento| count|
+-----+-----+
|          Condutor   |535778|
|        Passageiro   |199100|
|         Pedestre    | 14820|
|        Cavaleiro    |   178|
+-----+-----+

] : # Filtro que demonstra os acidentes ocorridos no fim de semana
dfs.filter("dia_semana = 'domingo' or dia_semana = 'sábado' and fase_dia = 'Plena Noite' ").orderBy("uf").show(5)
filtro_weekend = dfs.filter("dia_semana = 'domingo' or dia_semana = 'sábado' and fase_dia = 'Plena Noite' ")
```

## Spark SQL

```
#Criando a tabela temporaria que será utilizada pelo SQL
dfs.createOrReplaceTempView('pessoas')

#QUANTIDADE DE ACIDENTES AGRUPADO POR ESTADO FISICO, SEXO
spark.sql("SELECT estado_fisico AS Nivel_Lesao, sexo, count(DISTINCT id) AS Numero_Acidentes FROM pessoas GROUP BY estado_fisico, sexo ORDER BY estado_fisico ASC").show()

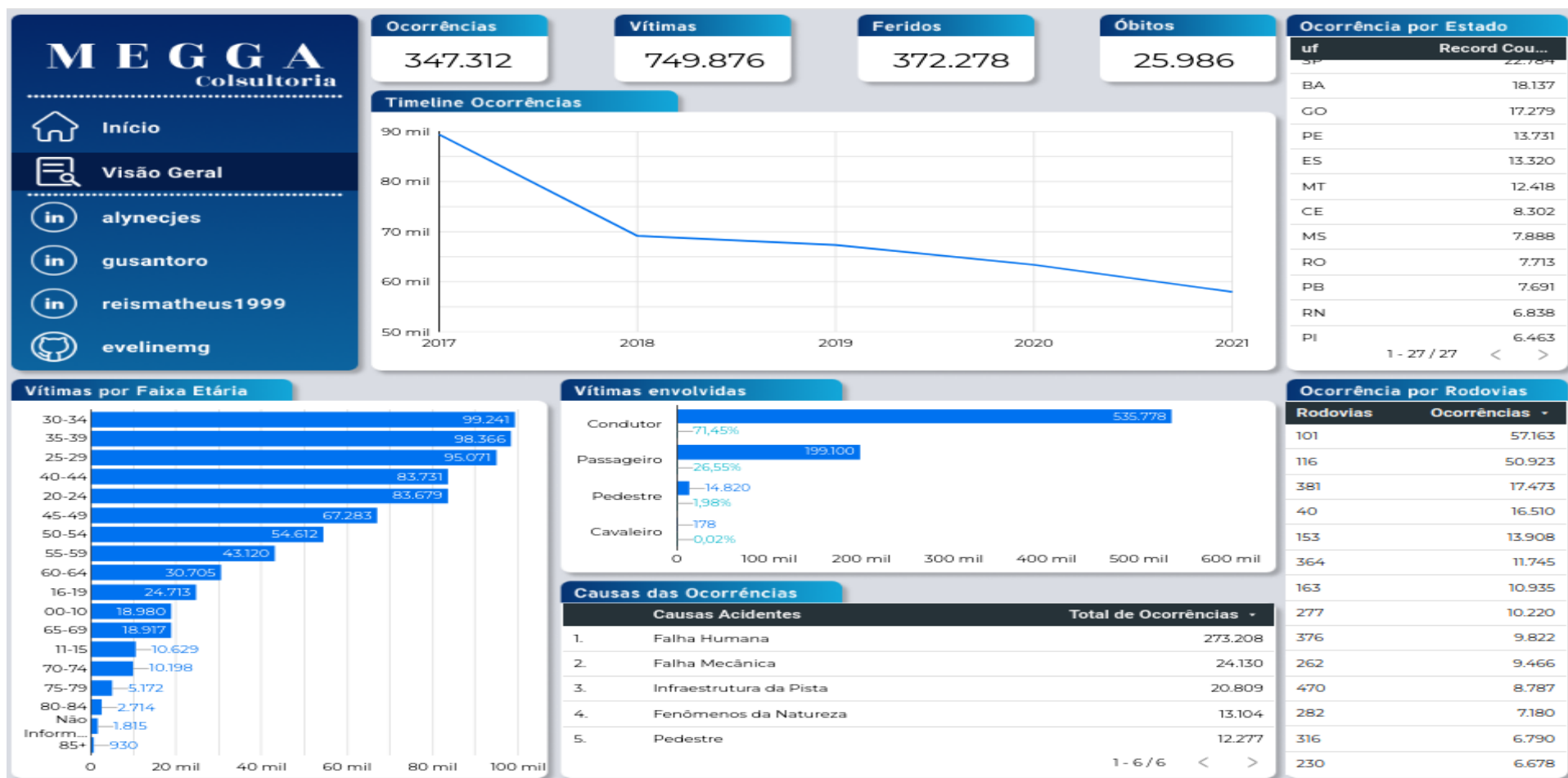
+-----+-----+-----+
| Nivel_Lesao|      sexo|Numero_Acidentes|
+-----+-----+-----+
| Ileso|Não Informado|      78|
| Ileso| Masculino|    206352|
| Ileso|  Feminino|    44678|
| Lesões Graves| Masculino|    57246|
| Lesões Graves|Não Informado|      12|
| Lesões Graves|  Feminino|    17851|
| Lesões Leves| Masculino|   163608|
| Lesões Leves|Não Informado|      81|
| Lesões Leves|  Feminino|   70220|
| Óbito| Masculino|    19319|
| Óbito|  Feminino|    4126|
| Óbito|Não Informado|       7|
+-----+-----+-----+

#QUANTIDADE DE ACIDENTES AGRUPADO POR SEXO
spark.sql("SELECT estado_fisico AS Nivel_Lesao, sexo, count(DISTINCT id) AS Numero_Acidentes FROM pessoas GROUP BY estado_fisico, sexo ORDER BY estado_fisico ASC").show()

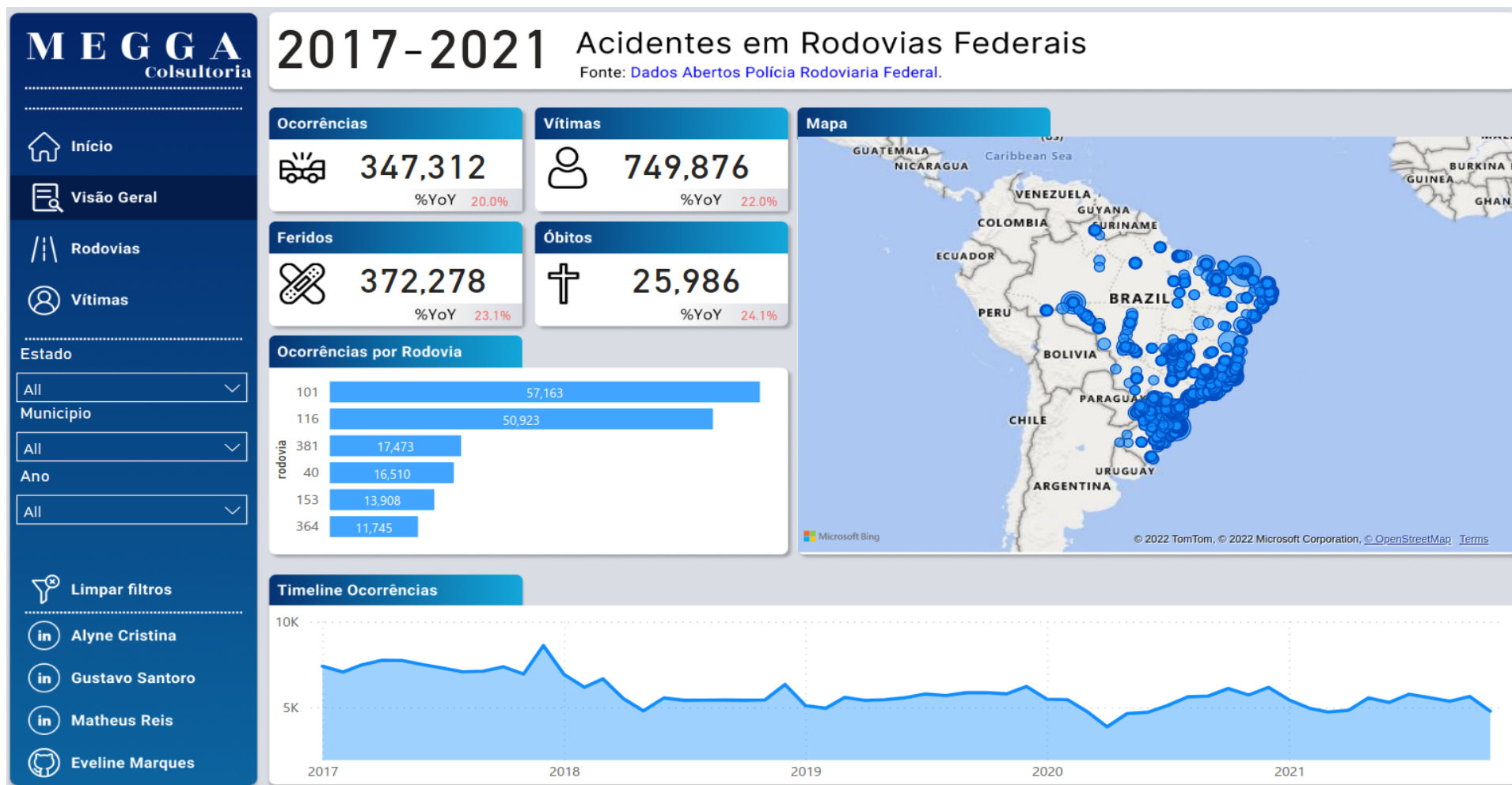
+-----+-----+-----+
| Nivel_Lesao|      sexo|Numero_Acidentes|
+-----+-----+-----+
| Ileso|Não Informado|      78|
| Ileso| Masculino|    206352|
| Ileso|  Feminino|    44678|
| Lesões Graves| Masculino|    57246|
| Lesões Graves|Não Informado|      12|
| Lesões Graves|  Feminino|    17851|
| Lesões Leves| Masculino|   163608|
| Lesões Leves|Não Informado|      81|
| Lesões Leves|  Feminino|   70220|
| Óbito| Masculino|    19319|
| Óbito|  Feminino|    4126|
| Óbito|Não Informado|       7|
+-----+-----+-----+
```

## Análises

## DataStudio



## PowerBI



## Dataflow

## Pipelines

<a href="#">← beamapp-root-0411213620-685435-hhxgyqjd</a> <a href="#">+ IMPORT AS PIPELINE</a> <a href="#">↻ SHARE</a>						
<a href="#">JOB GRAPH</a> <a href="#">EXECUTION DETAILS</a> <a href="#">JOB METRICS</a> <a href="#">RECOMMENDATIONS</a>						
<div>Job steps view <span>Table view</span> <span>▼</span></div> <div>CLEAR SELECTION</div>						
<div><span>≡ Filter</span> <span>Filter steps</span> <span>?</span></div>						
Step name	Status	Wall time	Stages	Input steps	Output steps	
▶ Procura arquivo	✓ Succeeded	0 seconds	✓ F8	—	Encontra os targets/ParDo(_ReadMatchesFn)	
▶ Encontra os targets	✓ Succeeded	0 seconds	✓ F8	Procura arquivo/.../ParDo(_MatchAllFn)	Unzipa	
Unzipa	✓ Succeeded	32 seconds	✓ F8	Encontra os targets/ParDo(_ReadMatchesFn)	Dataframe	
Dataframe	✓ Succeeded	29 seconds	✓ F8	Unzipa	Dicionario	
Dicionario	✓ Succeeded	35 seconds	✓ F8	Dataframe	Json	
Json	✓ Succeeded	20 seconds	✓ F8	Dicionario	Escreve arquivo/.../Windowinto(WindowIntoFn)	
▶ Escreve arquivo	✓ Succeeded	43 seconds	✓ F5	Json	—	▼



## Pipelines

[←](#) beamapp-root-0411213620-685435-hhxgyqjd [+ IMPORT AS PIPELINE](#) [↗ SHARE](#)

JOB GRAPH

EXECUTION DETAILS

JOB METRICS

RECOMMENDATIONS

Job steps view

Table view

CLEAR SELECTION

Filter

Filter steps

?

Step name	Status	Wall time	Stages	Input steps	Output steps	
▶ Procura arquivo	✓ Succeeded	0 seconds	✓ F8	—	Encontra os targets/ParDo(_ReadMatchesFn)	
▶ Encontra os targets	✓ Succeeded	0 seconds	✓ F8	Procura arquivo/.../ParDo(_MatchAllFn)	Unzipa	
Unzipa	✓ Succeeded	32 seconds	✓ F8	Encontra os targets/ParDo(_ReadMatchesFn)	Dataframe	
Dataframe	✓ Succeeded	29 seconds	✓ F8	Unzipa	Dicionario	
Dicionario	✓ Succeeded	35 seconds	✓ F8	Dataframe	Json	
Json	✓ Succeeded	20 seconds	✓ F8	Dicionario	Escreve arquivo/.../WindowInto(WindowIntoFn)	
▶ Escreve arquivo	✓ Succeeded	43 seconds	✓ F5	Json	—	▼

## **Mapeamento dos tratamentos e normalizações nos data sets Pessoas e Ocorrências**

Dataset Pessoas						
Database:	Acidentes por pessoas				colunas	35
Descrição	Tabela de acidentes agrupados por cada indivíduo envolvido na ocorrência.				linhas	829727
local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_pessoas.json					
Origem				Objetivo		
nº	campo	data type	descrição	campo	data type	transformação
1	id	inteiro	PK ocorrência	id	int	
2	pesid	float	PK pessoa	id_pessoa	int	
3	data_invertida	string	data YYYY-MM-DD	data	datetime	
4	dia_semana	string			string	df['dia_semana'].replace(['{dia}'], '{dia}-feira', inplace=True)
5	horario	string				remove
6	uf	string			string	
7	br	string		rodovia	string	
8	km	float			float	df['km'].str.replace(',', '.')
9	municipio	string			string	
10	causa_acidente	string				remove
11	tipo_acidente	string				remove
12	classificacao_acidente	string				remove
13	fase_dia	string				remove
14	sentido_via	string				remove
15	condicao_metereologica	string				remove

Dataset Pessoas							
Database:	Acidentes por pessoas					colunas	35
Descrição	Tabela de acidentes agrupados por cada indivíduo envolvido na ocorrência.					linhas	829727
local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_pessoas.json						
Origem				Objetivo			
nº	campo	data type	descrição	campo	data type	transformação	
16	tipo_pista	string				remove	
17	tracado_via	string				remove	
18	uso_solo	string	Urbano = sim; Rural=não		string		
19	id_veiculo	float			int		
20	tipo_veiculo	string			string		
21	marca	string		marca_mod elo	string		
22	ano_fabricacao_veiculo	float			int	Nível Pandas df.ano_fabricacao_veiculo.replace((pd.NA, 0))	
23	tipo_envolvido	string			string		
24	estado_fisico	string			string		
25	idade	float	-1 idade não foi coletada	faixa_idade	string	Nível PySpark dfs.withColumn('faixa_idade', F.when((col('idade')>100), lit('Não Informado'))  	

Dataset Pessoas						
Database:	Acidentes por pessoas				colunas	35
Descrição	Tabela de acidentes agrupados por cada indivíduo envolvido na ocorrência.				linhas	829727
local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_pessoas.json					
Origem				Objetivo		
nº	campo	data type	descrição	campo	data type	transformação
						<pre>.when((col('idade')&gt;=75), lit('75-79')) .when((col('idade')&gt;=70), lit('70-74')) .when((col('idade')&gt;=65), lit('65-69')) .when((col('idade')&gt;=60), lit('60-64')) .when((col('idade')&gt;=55), lit('55-59')) .when((col('idade')&gt;=50), lit('50-54')) .when((col('idade')&gt;=45), lit('45-49')) .when((col('idade')&gt;=40), lit('40-44')) .when((col('idade')&gt;=35), lit('35-39')) .when((col('idade')&gt;=30), lit('30-34')) .when((col('idade')&gt;=25), lit('25-29')) .when((col('idade')&gt;=20), lit('20-24')) .when((col('idade')&gt;=16), lit('16-19')) .when((col('idade')&gt;=11), lit('11-15')) .when((col('idade')&lt;=10), lit('00-10'))</pre>
26	sexo	string			string	Nível Pandas <code>df['sexo'].replace(['Ignorado'],'Não Informado',inplace=True)</code>

Dataset Pessoas						
Database:	Acidentes por pessoas				colunas	35
Descrição	Tabela de acidentes agrupados por cada indivíduo envolvido na ocorrência.				linhas	829727
local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_pessoas.json					
Origem				Objetivo		
nº	campo	data type	descrição	campo	data type	transformação
27	ilesos	int	0- não; 1-sim booleano			remove
28	feridos_leves	int	0- não; 1-sim booleano			remove
29	feridos_graves	int	0- não; 1-sim booleano			remove
30	mortos	int	0- não; 1-sim booleano			remove
31	latitude	string			float	Nível Pandas df['latitude'].str.replace(',','.') )
32	longitude				float	Nível Pandas df['longitude'].str.replace(',','.') )
33	regional	string				remove
34	delegacia	string				remove
35	uop	string				remove

Dataset Ocorrências						
Database:	Acidentes por ocorrências				colunas	30
Descrição	Tabela de acidentes agrupados por cada ocorrência				linhas	354293
Local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_ocorrendia.csv					
Origem				Objetivo		
nº	campo	data type	descrição	campo	data type	transformação
1	id	inteiro	PK ocorrência	id	int	
2	data_inversa	string	data YYYY-MM-DD	data	datetime	
3	dia_semana	string			string	Nível df['dia_semana'].replace(['{dia}'],'{dia}-feira',inplace=True)
4	horario	string				remove
5	uf	string			string	
6	br	string		rodovia	string	
7	km	float			float	Nível Pandas df['km'].str.replace(',','.') )
8	municipio	string			string	
9	causa_acidente	string			string	
10	tipo_acidente	string			string	
11	classificacao_acidente	string			string	
12	fase_dia	string			string	

Dataset Ocorrências						
Database:	Acidentes por ocorrências				colunas	30
Descrição	Tabela de acidentes agrupados por cada ocorrência				linhas	354293
Local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_ocorrendia.csv					
Origem				Objetivo		
nº	campo	data type	descrição	campo	data type	transformação
13	sentido_via	string			string	
14	condicao_metereologica	string			string	
15	tipo_pista	string			string	
16	tracado_via	string			string	
17	uso_solo	string	Urbano = sim; Rural=não		string	
18	pessoas	int			int	
19	veiculos	string			string	
20	ilesos	int	0- não; 1-sim booleano		int	
21	feridos_leves	int	0- não; 1-sim booleano		int	
22	feridos_graves	int	0- não; 1-sim booleano		int	
23	ignorados	int			int	
24	mortos	int	0- não; 1-sim booleano		int	
25	latitude	string			float	Nível Pandas df['latitude'].str.replace(',','.') )



Dataset Ocorrências						
Database:	Acidentes por ocorrências					colunas 30
Descrição	Tabela de acidentes agrupados por cada ocorrência					linhas 354293
Local:	gs://projetofinalsc/dadosbrutos/pessoas/acidentes_ocorrendia.csv					
Origem				Objetivo		
nº	campo	data type	descrição	campo	data type	transformação
26	longitude				float	Nível Pandas df['longitude'].str.replace(',', '.')
26	regional	string				remove
27	delegacia	string				remove
28	uop	string				remove
29			A partir da coluna causa_acidente	consumo_alcool	string	Nível PySpark: df=df.withColumn("consumo_alcool", F.col('causa_acidente').rlike("Álcool")) df=df.withColumn('consumo_alcool', when(df.consumo_alcool=='true', 'sim').otherwise('não'))
30			A partir de da coluna causa_acidente	grupo_causas	string	Nível PySpark b=[x[0] for x in df.select('causa_acidente').distinct().collect()] df=df.withColumn("causa_grupos", condicao)

## Conclusão

Com os passos descritos nesse documento é possível repetir as operações de tratamento nos datasets utilizados de acordo com a necessidade do trabalho. Documentar cada transformação necessária nos diversos campos e a geração dos dados finais colabora para a organização e manutenção do trabalho.

Durante a elaboração desse trabalho também foi possível notar que as decisões tomadas em uma planilha impactaria a outra, sendo assim necessário, fazer remoções e alterações compatíveis em ambos datasets.

Além disso, automatizar processos como a pipeline pode impactar na organização e redução de tempo demandado em certas tarefas, como descomprimir e concatenar todos os arquivos.

Ao final, foi possível fazer várias análises sobre os dados trabalhados e a construção dos relatórios em ferramentas diferentes trouxe a possibilidade de visualização dos dados em diversas perspectivas, com filtros trazendo diversos níveis de análises, por estado, rodovia, por ano, entre outros.