

MVP - Data Engineering

Estudo para oferta de produtos de energia para novos nichos de mercado

Aluna: Ana Paula Zampier Abreu Alvarenga

Email: anapaulazampier@gmail.com

GitHub: <https://github.com/anapaulazampier>

Discord: anapaulazampier

Sprint: Engenharia de Dados (40530010057_20240_01)

Estudo para oferta de produtos de energia para novos nichos de mercado.....	1
Contextualização e objetivos gerais.....	2
Detalhamento do projeto.....	2
Dados e ferramentas selecionadas.....	2
Coleta e carregamento de dados.....	3
Modelagem de dados.....	8
Catálogo de dados.....	12
Análise de dados.....	15
Qualidade de dados.....	15
Solução do problema proposto.....	17
Considerações finais.....	22
Pontos de melhoria.....	22
Autoavaliação.....	23

Contextualização e objetivos gerais

O mercado livre de energia brasileiro é marcado pelo processo de abertura de mercado, pulverização da concorrência decorrente do aumento exponencial de clientes e a consequente redução do ticket médio. Nesse cenário, faz-se necessário analisar potenciais segmentos de mercado para oferta de novos produtos de energia, que ultimamente podem gerar vantagem competitiva para empresas de comercialização de energia.

Dessa forma, o objetivo do presente trabalho é conduzir um processo de análise de mercado para buscar novos nichos de mercado para oferta de produtos de energia a consumidores com potencial de migração para o mercado livre de energia.

O setor selecionado para análise é o de empresas compradoras de crédito de carbono e nesse contexto, é desejável responder os seguintes questionamentos:

- 1) Das empresas listadas no Registro Público de Emissões, quantas estão no mercado livre de energia?
 - a) Para as companhias no mercado livre de energia, quais são os perfis de consumo (consumidor livre, consumidor especial, autoprodutor ou comercializador)? A análise de perfil é relevante para a oferta de produtos de energia.
 - b) Para empresas no mercado livre de energia, qual seu consumo de energia em 2024 em MWm?
- 2) As empresas compradoras de crédito de carbono podem ser consideradas consumidores em potencial de energia no mercado livre?
 - a) Quais tipos de produto são mais prováveis que essas empresas adquiram?

Detalhamento do projeto

Dados e ferramentas selecionadas

Para responder os questionamentos propostos, as principais fontes de dados escolhidas foram:

- **InfoMercado - Dados Individuais** ([lista_perfil_2024](#))
Arquivo completo de informações sobre todos os agentes participantes do mercado livre de energia. Será extraída a tabela dimensão de perfis de agentes.
- **Consumo Mensal de Energia** ([consumo_mensal_perfil_agente_2024](#))
Total de energia consumida por empresas no mercado livre de energia.
- **Registro Público de Emissões** ([Participantes | Registro Público de Emissões](#))
Lista de empresas, divulgada pelo Centro de Estudos de Sustentabilidade da Fundação Getúlio Vargas, que autorizaram o compartilhamento de suas emissões de gases de efeito estufa.

- **dCalendario**

Tabela dimensão de datas com o cálculo do total de horas no mês

O projeto proposto foi executado inteiramente na plataforma Databricks em sua versão Community Edition. Dessa forma, os scripts foram desenvolvidos em notebooks Python e posteriormente, na etapa de análise de dados, tabelas foram consultadas em SQL. É importante destacar algumas limitações devido ao uso da versão Community Edition, como o não agendamento da recorrência da pipeline de dados e não possibilidade de integração direta entre Databricks e GitHub.

Para a limitação de agendamento de pipelines, uma possível alternativa com custos mais reduzidos seria a utilização do Databricks com a Azure Cloud e o agendamento de pipelines integradas diretamente ao notebook e ao cluster do Databricks com a ferramenta Data Factory.

Quanto a limitação de integração direta com o GitHub, foi aplicada como alternativa uma requisição GET para cópia de arquivos e para o upload do script foi realizado um export manual na plataforma Databricks.

Coleta e carregamento de dados

Para os arquivos disponibilizados pela Câmara de Comercialização de Energia Elétrica (CCEE), utilizou-se a função `dbutils.fs.cp` que realiza cópia dos arquivos originais para o diretório do Databricks File System (dbfs) especificado.

A disponibilização de dados pela CCEE é feita através de seu portal de dados abertos e nele são disponibilizadas as URLs para download de dados.

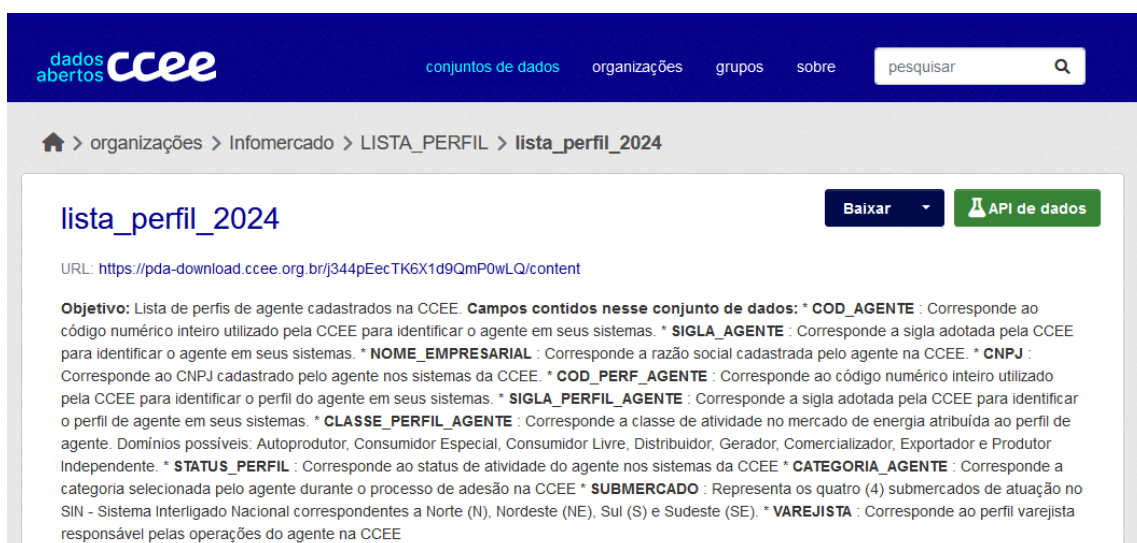


Figura 1 - Portal de dados abertos da CCEE

Ingestão e tratamento da tabela de perfis

```
catalog = "mvp_dataengineering"
schema = "ccee"
volume = "lista_perfis"
download_url = "https://pda-download.ccee.org.br/j344pEecTK6X1d9QmP0wLQ/content"
file_name = "lista_perfis.csv"
table_name = "lista_perfis"
layer = "bronze"
path_volume = "/" + catalog + "/" + schema + "/" + volume
path_table = "/" + catalog + "." + schema

dbutils.fs.cp(f"{download_url}", f"{path_volume}" + "/" + layer + "/" + f"{file_name}")

True
```

Figura 2 - Processo de ingestão do arquivo de Lista de Perfis

O processo de coleta dos arquivos do Registro Público de Emissões foi dado em duas etapas e se demonstrou consideravelmente mais complexo.

A primeira etapa consistiu na localização dos códigos das organizações listadas dentro do inspetor de elementos do HTML. Durante essa etapa, foram realizados testes com os módulos PhantomJS da biblioteca Selenium, bem como requisições com extração de texto da biblioteca BeautifulSoup. Entretanto, com nenhum dos métodos mencionados obteve-se o resultado desejado, dessa forma, foi necessário realizar uma cópia manual do elemento dinâmico e posterior armazenamento em um arquivo .txt disponibilizado no GitHub do projeto.

The image shows the 'Registro Público de Emissões' website interface on the left and its corresponding HTML structure on the right. The website has a search bar for 'Organização' and a timeline for 'Ano Inventariado' from 2008 to 2022. Below the search results, it lists several organizations like '2W Ecobank', 'A.C. Camargo Cancer Center', 'ACHE LABORATÓRIOS S/A', and 'Acumuladores Moura S/A'. The HTML view on the right shows the underlying structure, with a red box highlighting a list of participant links, each with a unique ID like ``.

Figura 3 - Elemento de HTML com os códigos das empresas participantes do Registro Público de Emissões

Em sequência foram realizadas as leituras e armazenamento do arquivo .txt em camada bronze e tratamentos de regex para consolidar uma base de dados de ids de participantes na camada silver.

Ingestão e tratamento das tabelas do registro público de emissões

```
#Ingestão do txt com códigos de participantes que serão utilizados para chamada de APIs
import requests
import pandas as pd

url = 'https://raw.githubusercontent.com/anapaulazampier/mvp-dataengineering/main/participant-box_html.txt'
res = requests.get(url, allow_redirects=True)

#Armazenamento do arquivo na camada bronze
dbutils.fs.put('/mvp_dataengineering/emissions_registry/participants/bronze/participant_box_html.txt', res.text,
overwrite=True)

Wrote 287448 bytes.
True
```

Figura 4 - Download do arquivo .txt do gitHub para a camada bronze da tabela de participantes

```
#Leitura do arquivo txt de participantes com spark
participants_sparkread = spark.read.text('dbfs:/mvp_dataengineering/emissions_registry/participants/bronze/
participant_box_html.txt', wholetext=True)

#Tratamento com regex para separação dos códigos de participantes do html da página
participants_sparkread = str(participants_sparkread.collect()[0][0])
import re
import pandas as pd

href_strings = re.findall(r'href="([^\"]+)"', participants_sparkread)
df_dParticipants = pd.DataFrame({'href': href_strings})
df_dParticipants['participant'] = df_dParticipants['href'].str.split('/')[2]

for i in range(len(df_dParticipants)):
    df_dParticipants['participant'][i] = df_dParticipants['participant'][i][2]

display(df_dParticipants)

#Armazenamento do arquivo tratado na camada silver
spark_participants = spark.createDataFrame(df_dParticipants, schema="href STRING, participant STRING")
spark_participants.write.format("parquet").save("/mvp_dataengineering/emissions_registry/participants/silver")
```

(2) Spark Jobs

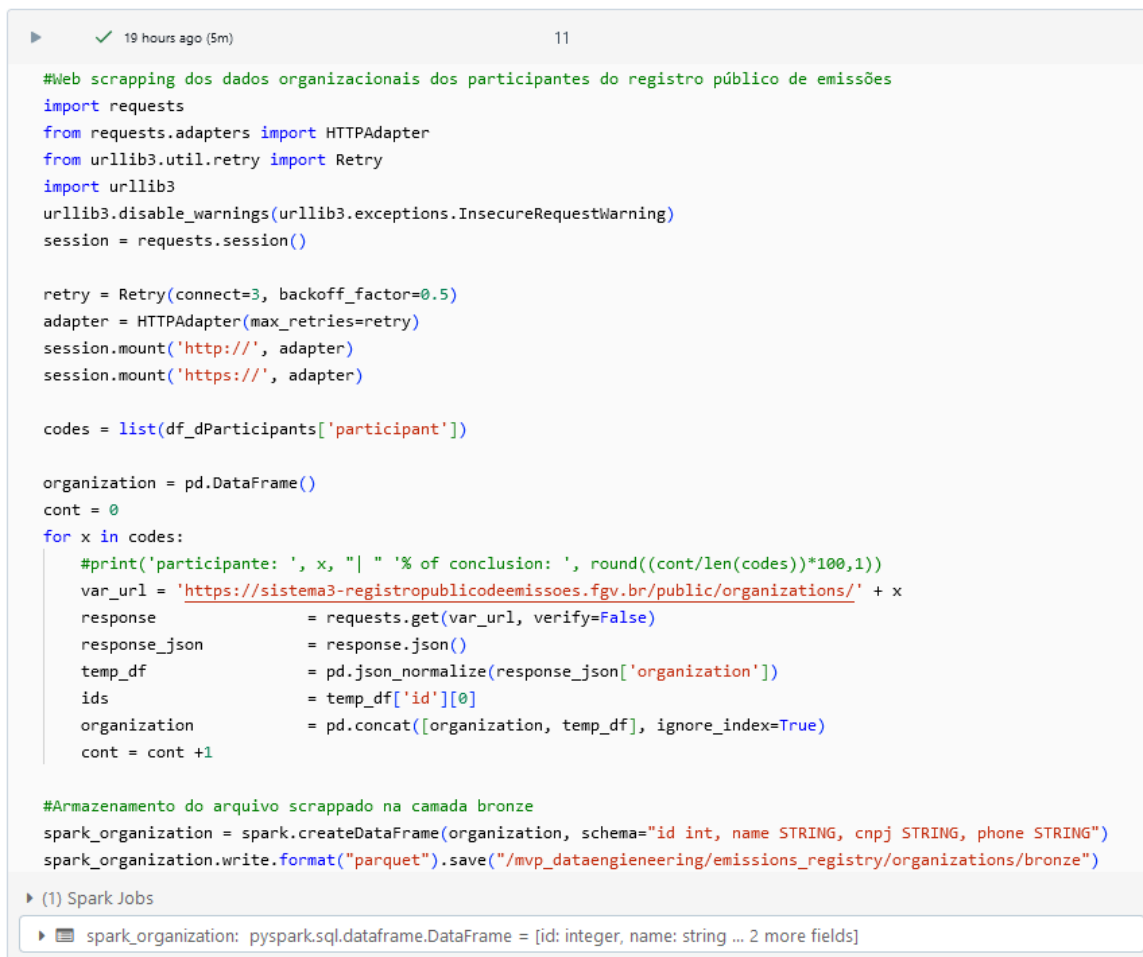
spark_participants: pyspark.sql.dataframe.DataFrame = [href: string, participant: string]

	href	participant
1	/participantes/53...	5372
2	/participantes/54...	5499
3	/participantes/20...	2098
4	/participantes/329	329
5	/participantes/36...	3699

575 rows | 2.57 seconds runtime Refreshed 19 hours ago

Figura 5 - Tratamento do arquivo .txt para construção de uma tabela em camada silver

Por fim, os códigos de id dos participantes foram utilizados para realizar requisições GET e obter informações dos dados organizacionais. Os resultados das requisições foram recebidos em formato json e tratados para construir um dataframe que foi armazenado em formato parquet na camada bronze da tabela de organizações.



```
#Web scrapping dos dados organizacionais dos participantes do registro público de emissões
import requests
from requests.adapters import HTTPAdapter
from urllib3.util.retry import Retry
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
session = requests.session()

retry = Retry(connect=3, backoff_factor=0.5)
adapter = HTTPAdapter(max_retries=retry)
session.mount('http://', adapter)
session.mount('https://', adapter)

codes = list(df_dParticipants['participant'])

organization = pd.DataFrame()
cont = 0
for x in codes:
    #print('participante: ', x, "| " '% of conclusion: ', round((cont/len(codes))*100,1))
    var_url = 'https://sistema3-registropublicodeemissoes.fgv.br/public/organizations/' + x
    response = requests.get(var_url, verify=False)
    response_json = response.json()
    temp_df = pd.json_normalize(response_json['organization'])
    ids = temp_df['id'][0]
    organization = pd.concat([organization, temp_df], ignore_index=True)
    cont = cont + 1

#Armazenamento do arquivo scrappado na camada bronze
spark_organization = spark.createDataFrame(organization, schema="id int, name STRING, cnpj STRING, phone STRING")
spark_organization.write.format("parquet").save("/mvp_dataengieneering/emissions_registry/organizations/bronze")

(1) Spark Jobs
spark_organization: pyspark.sql.dataframe.DataFrame = [id: integer, name: string ... 2 more fields]
```

Figura 6 - Processo de requisição, tratamento e armazenamento da base de dados de organizações

Conforme supracitado, não é possível realizar o agendamento de pipelines completarmos de maneira totalmente automatizada o processo de ETL do projeto, contudo, o notebook está organizado de forma escalável, de modo que, caso este recurso esteja disponível, o script possa ser executado conforme planejado. É recomendado atualizações mensais das bases de dados conforme o calendário de divulgação de resultados da Câmara de Comercialização de Energia.

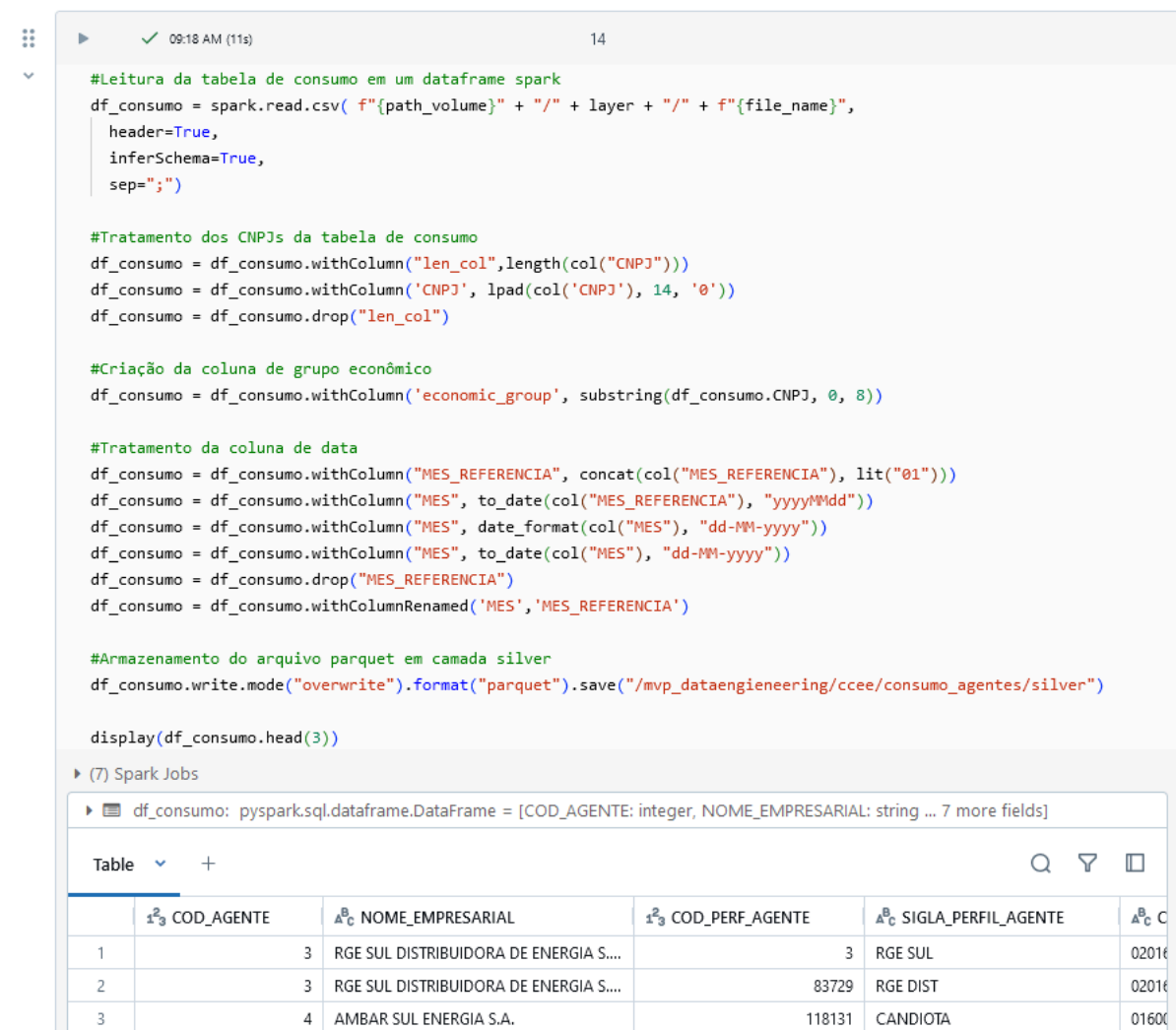
Para mais informações sobre o calendário, acessar: [Liquidações Financeiras do MCP - 2024](#)

Modelagem de dados

Após realizar a ingestão dos arquivos em camadas bronze, os dados passaram pelos seguintes processos de padronização:

- Padronização dos textos de CNPJ entre as tabelas através da remoção dos caracteres especiais "-", "/", "-" e preenchimento das strings com a adição de "0" no início de valores que possuem menos de 14 caracteres.
- Criação da coluna de grupo econômico através da extração dos 8 primeiros dígitos do CNPJ. Essa coluna será a chave primária entre as tabelas de consumo e organizações.
- Ajuste da coluna de data para que estejam todos valores no formato dd-mm-yyy
- Join entre tabelas para construir a camada ouro e iniciar o processo de análise.

Abaixo, tem-se a manipulação realizada para a tabela de consumo, que contempla as etapas de padronização de CNPJ, criação do grupo econômico e ajuste da coluna de data:



```
#Leitura da tabela de consumo em um dataframe spark
df_consumo = spark.read.csv( f"{path_volume}" + "/" + layer + "/" + f"{file_name}",
    header=True,
    inferSchema=True,
    sep=";")

#Tratamento dos CNPJs da tabela de consumo
df_consumo = df_consumo.withColumn("len_col", length(col("CNPJ")))
df_consumo = df_consumo.withColumn('CNPJ', lpad(col('CNPJ'), 14, '0'))
df_consumo = df_consumo.drop("len_col")

#Criação da coluna de grupo econômico
df_consumo = df_consumo.withColumn('economic_group', substring(df_consumo.CNPJ, 0, 8))

#Tratamento da coluna de data
df_consumo = df_consumo.withColumn("MES_REFERENCIA", concat(col("MES_REFERENCIA"), lit("01")))
df_consumo = df_consumo.withColumn("MES", to_date(col("MES_REFERENCIA"), "yyyyMMdd"))
df_consumo = df_consumo.withColumn("MES", date_format(col("MES"), "dd-MM-yyyy"))
df_consumo = df_consumo.withColumn("MES", to_date(col("MES"), "dd-MM-yyyy"))
df_consumo = df_consumo.drop("MES_REFERENCIA")
df_consumo = df_consumo.withColumnRenamed("MES", "MES_REFERENCIA")

#Armazenamento do arquivo parquet em camada silver
df_consumo.write.mode("overwrite").format("parquet").save("/mvp_dataengineering/ccee/consumo_agentes/silver")

display(df_consumo.head(3))
```

(7) Spark Jobs

df_consumo: pyspark.sql.dataframe.DataFrame = [COD_AGENTE: integer, NOME_EMPRESARIAL: string ... 7 more fields]

	¹ ₂ COD_AGENTE	^A ₀ NOME_EMPRESARIAL	¹ ₂ COD_PERF_AGENTE	^A ₀ SIGLA_PERFIL_AGENTE	^A ₀ C
1	3	RGE SUL DISTRIBUIDORA DE ENERGIA S...	3	RGE SUL	02016
2	3	RGE SUL DISTRIBUIDORA DE ENERGIA S...	83729	RGE DIST	02016
3	4	AMBAR SUL ENERGIA S.A.	118131	CANDIOTA	01600

Figura 7 - Leitura, tratamento e armazenamento da base de dados de consumo

Dessa forma, obtemos um banco de dados em esquema snowflake. Abaixo, foi realizada a representação do modelo relacional com auxílio da ferramenta [genmymodel](#) e as chaves utilizadas para relacionamento com suas respectivas cardinalidades também foram representadas.

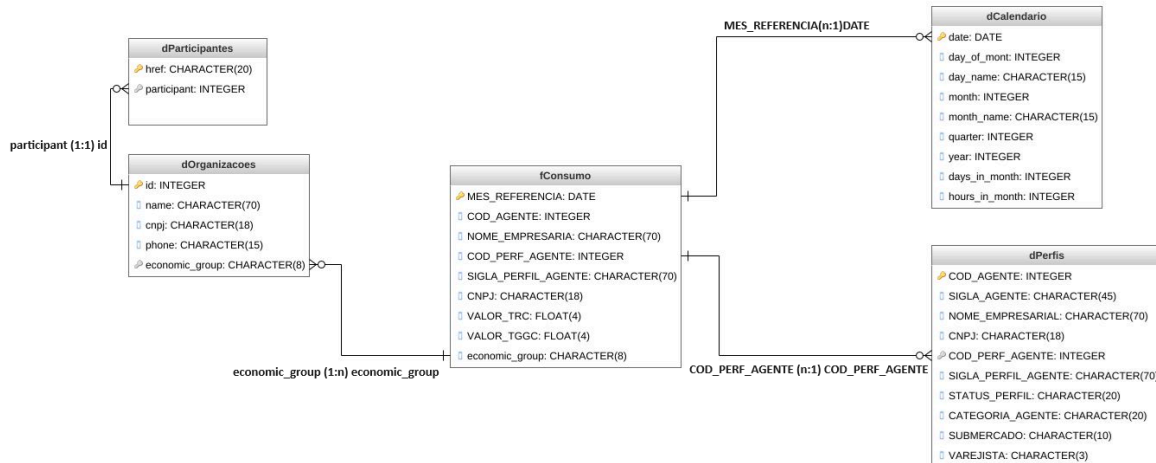


Figura 8 - Representação do modelo de dados relacional

As manipulações para construção de tabelas em camada gold, farão com que o modelo se reduza a uma tabela única (esquema flat), que será acrescida de flags condicionais para facilitar o processo de consulta e análise em SQL posteriormente.

Abaixo, o join entre as tabelas de consumo e perfis e a criação da coluna de validação para verificar se o perfil da tabela de consumo está listado na tabela de perfis, construindo a camada gold para a tabela de consumo:

```

09:20 AM (<1s) 17

#Criação da tabela gold de consumo com o merge entre as tabelas dPerfis e fConsumo
from pyspark.sql.functions import desc
fConsumo_gold = fConsumo_silver.join(dPerfis_gold, fConsumo_silver.consumo_COD_PERF_AGENTE
== dPerfis_gold.perfis_COD_PERF_AGENTE, 'left')

fConsumo_gold: pyspark.sql.dataframe.DataFrame = [consumo_COD_AGENTE: integer, consumo_NOME_EMPRESARIAL: string ... 19
more fields]

09:20 AM (<1s) 18

#Criação de colunas flag para análise de dados posterior
from pyspark.sql.functions import col, length, trim, lpad, regexp_replace, date_format, to_date, concat, lit,
substring, when

fConsumo_gold = fConsumo_gold.withColumn("PERFIL_ESTA_LISTADO", when(col("perfis_COD_PERF_AGENTE").isNull(), 0).
otherwise(1))

fConsumo_gold: pyspark.sql.dataframe.DataFrame = [consumo_COD_AGENTE: integer, consumo_NOME_EMPRESARIAL: string ... 20
more fields]

```

Figura 9 - Modelagem da camada gold da tabela de fConsumo

Por fim, para o join entre a camada gold da tabela de consumo e a camada silver da tabela de organizações foram utilizadas as colunas de grupo econômico preparadas anteriormente e foi adotado o método de junção 'outer'.

O método outer, ou full outer, foi aplicado pela necessidade de manter todas as linhas de ambas tabelas. Essa decisão permite com que os consumidores sejam classificados em três categorias:

- **Consumidor liberalizado que não compra crédito de carbono:**
Consumidores puramente no mercado livre de energia que não estão listados como compradores de crédito de carbono pelo Registro Público de Emissões
- **Consumidor liberalizado que compra crédito de carbono:**
Consumidores que estão listados como compradores de crédito de carbono e que também compram energia no mercado livre
- **Consumidor não liberalizado que compra crédito de carbono:**
Consumidores que apenas compram créditos de carbono e não migraram para o mercado livre de energia

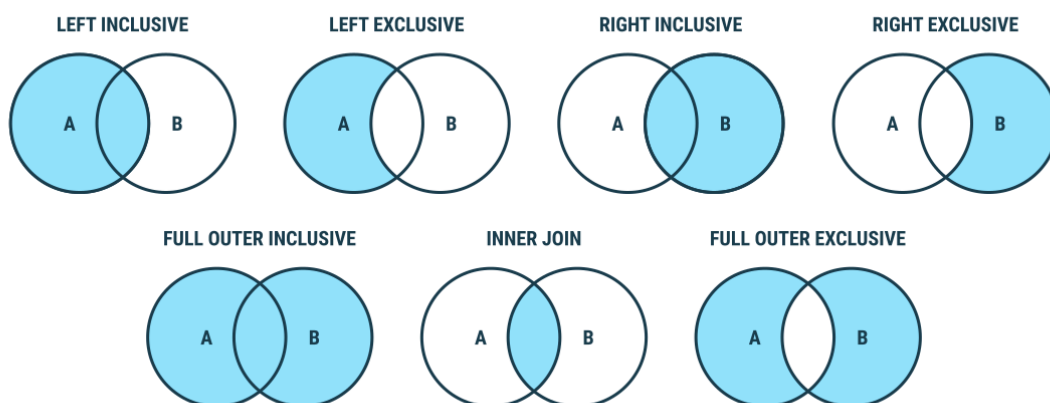


Figura 10 - Comparação entre técnicas de junção

```

09:20 AM (<1s) 20

#Merge entre compradores de credito de carbono e consumidores de energia no mercado livre
carbono_e_mercadolivre = fConsumo_gold.join(dCompradores, fConsumo_gold.consumo_economic_group == dCompradores.
economic_group, 'outer')

#Coluna flag para análises posteriores
carbono_e_mercadolivre = carbono_e_mercadolivre.withColumn(
    "PERFIL_COMPRA_CONSUMIDOR",
    when(col("id").isNull(), "Consumidor liberalizado que não compra crédito de carbono")
    .when((col("id").isNotNull()) & (col("consumo_COD_AGENTE").isNotNull()), "Consumidor liberalizado que compra
    crédito de carbono")
    .otherwise("Consumidor não liberalizado que compra crédito de carbono")
)

carbono_e_mercadolivre: pyspark.sql.dataframe.DataFrame = [consumo_COD_AGENTE: integer, consumo_NOME_EMPRESARIAL:
string ... 26 more fields]

```

Figura 11 - Modelagem da tabela final de consumidores com outer join

Catálogo de dados

1. Lista de perfis

Coluna	Tipo	Descrição
COD_AGENTE	int	Código numérico inteiro utilizado pela CCEE para identificar o agente em seus sistemas.
SIGLA_AGENTE	string	Sigla adotada pela CCEE para identificar o agente em seus sistemas.
NOME_EMPRESARIAL	string	Razão social cadastrada pelo agente na CCEE.
CNPJ	string	CNPJ cadastrado pelo agente nos sistemas da CCEE.
COD_PERF_AGENTE	int	Código numérico inteiro utilizado pela CCEE para identificar o perfil do agente em seus sistemas.
SIGLA_PERFIL_AGENTE	string	Sigla adotada pela CCEE para identificar o perfil de agente em seus sistemas.
CLASSE_PERFIL_AGENTE	string	Classe de atividade no mercado de energia atribuída ao perfil de agente. Domínios possíveis: Autoprodutor, Consumidor Especial, Consumidor Livre, Distribuidor, Gerador, Comercializador, Exportador e Produtor Independente.
STATUS_PERFIL	string	Status de atividade do agente nos sistemas da CCEE
CATEGORIA_AGENTE	string	Categoria selecionada pelo agente durante o processo de adesão na CCEE
SUBMERCADO	string	Representa os quatro (4) submercados de atuação no SIN - Sistema Interligado Nacional correspondentes a Norte (N), Nordeste (NE), Sul (S) e Sudeste (SE).
VAREJISTA	string	Perfil varejista responsável pelas operações do agente na CCEE
economic_group	string	O grupo econômico ou raiz do CNPJ é composto pelos 8 primeiros dígitos do CNPJ cadastrado pelo agente.

Tabela 1 - Catálogo de dados da tabela de lista de perfis

2. Consumo mensal por agente

Coluna	Tipo	Descrição
MES_REFERENCIA	datetime	Mês e o ano de referência na qual os dados foram utilizados.
COD_AGENTE	int	Código numérico inteiro utilizado pela CCEE para identificar o agente em seus sistemas.
NOME_EMPRESARIAL	string	Razão social cadastrada pelo agente na CCEE.
COD_PERF_AGENTE	int	Código numérico inteiro utilizado pela CCEE para identificar o perfil do agente em seus sistemas.
SIGLA_PERFIL_AGENTE	string	Sigla adotada pela CCEE para identificar o perfil de agente em seus sistemas.
CNPJ	string	CNPJ cadastrado pelo agente nos sistemas da CCEE.
VALOR_TRC	double	Consumo Total do perfil do agente “a”, por submercado “s”, no período de comercialização “j” TRCa,s,j [MWh]
VALOR_TGGC	double	Informação consolidada correspondente ao consumo da geração de cada perfil de agente “a” no submercado “s” (Sul, Sudeste/Centro-Oeste, Norte e Nordeste), por período de comercialização “j” TGGCa,s,j [MWh]
economic_group	string	O grupo econômico ou raiz do CNPJ é composto pelos 8 primeiros dígitos do CNPJ cadastrado pelo agente.

Tabela 2 - Catálogo de dados da tabela de consumo mensal por agente

3. Registro público de emissões

a. Códigos de participantes

Coluna	Tipo	Descrição
href	string	Trecho do html da página que contém o id de cada empresa participante do registro público de emissões.
participant	string	Código da organização.

Tabela 3 - Catálogo de dados da tabela de código de participantes

b. Organizações

Coluna	Tipo	Descrição
id	string	Código da organização
name	string	Nome da empresa, podendo ser preenchido como razão social ou nome fantasia.
cnpj	string	CNPJ da empresa.
phone	string	Telefone para contato do responsável pela empresa.
economic_group	string	O grupo econômico ou raiz do CNPJ é composto pelos 8 primeiros dígitos do CNPJ cadastrado.

Tabela 4 - Catálogo de dados da tabela de organizações participantes do Registro Público de Emissões

4. Calendário

Coluna	Tipo	Descrição
date	datetime	Data (data vigente - 2 anos, 31/12 do ano vigente)
day_of_month	int	Dia do mês (1-31)
day_name	string	Nome do dia (Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday)
month	in	Mês (1-12)
month_name	string	Mês (January, February, March, April, May, June, July, August, September, October, November, December)
quarter	int	Trimestres (1-4)
year	int	Ano
days_in_month	int	Quantidade de dias no mês
hours_in_month	int	Quantidade de horas no mês

Tabela 5 - Catálogo de dados da tabela de calendário

Análise de dados

A etapa de análise de dados será dividida em dois momentos, sendo o primeiro dedicado para análise de qualidade de dados e o segundo para responder às perguntas de negócio feitas na etapa de contextualização e objetivos do trabalho.

Qualidade de dados

Nessa etapa, espera-se validar que todas as bases ingeridas estejam adequadas para análise, pois todas as fontes utilizadas são curadas por órgãos responsáveis. Para essa etapa foi utilizado a própria ferramenta de [Data Profile do Databricks](#).

Para a tabela de consumo, observamos valores faltantes somente nos campos relacionados a consumo, contudo não é necessário removê-los, pois indicam simplesmente que determinado agente não consumiu e/ou gerou energia em determinado mês.

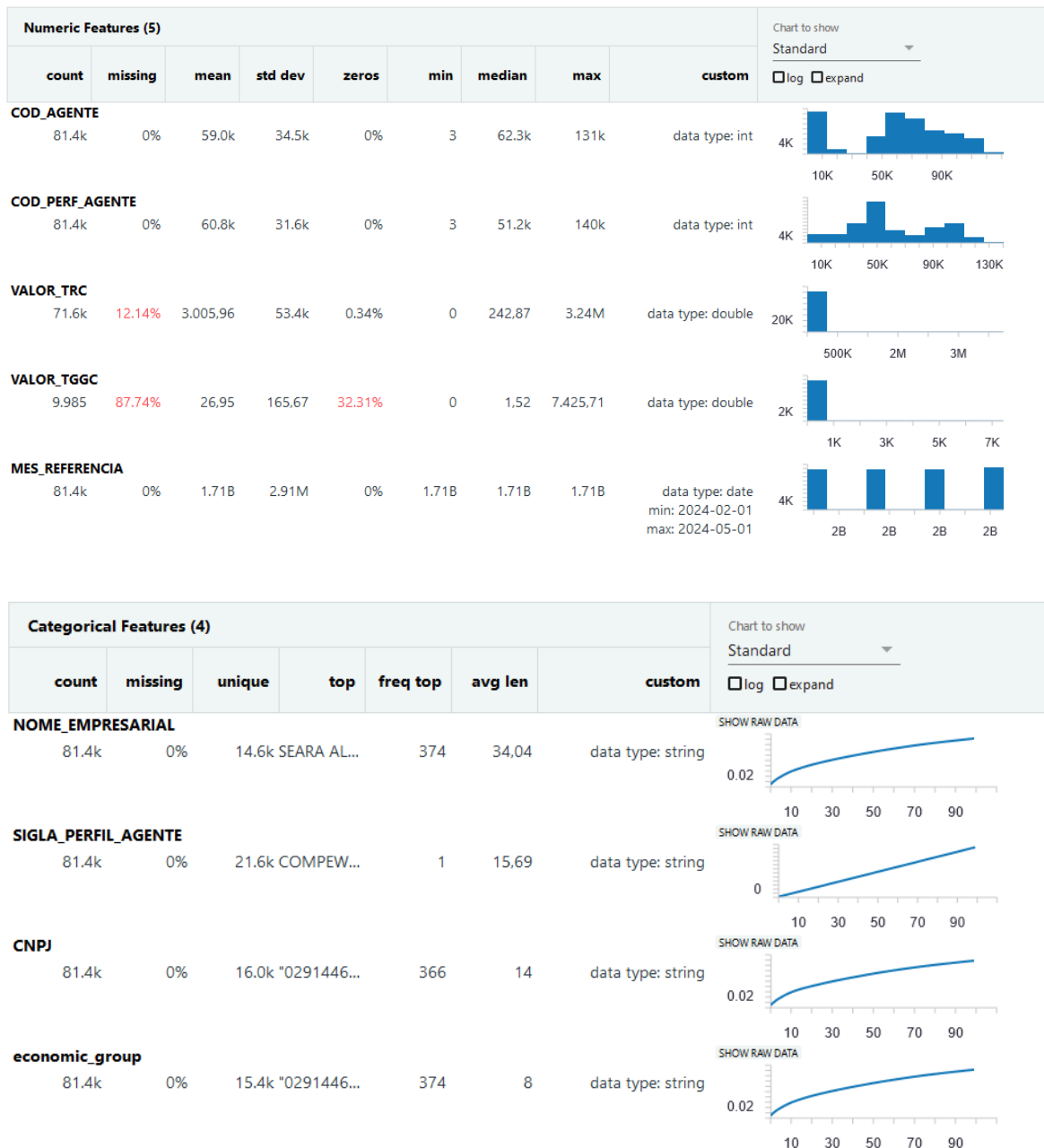


Figura 12 - Perfil de dados fConsumo_bronze

De forma análoga, observamos o mesmo padrão de qualidade de dados para a tabela de perfis e não foram detectados dados faltantes. O perfil de dados da tabela de perfis não será anexado para que o relatório não se torne demasiado redundante.

Por fim, para a tabela de organizações mantém-se a observação de ausência de dados faltantes e por sua natureza categórica as medidas estatísticas não se fazem necessárias. Analogamente, o mesmo é observado para a tabela de participantes e pelas mesmas razões anteriores seu perfil de dados não será anexado.

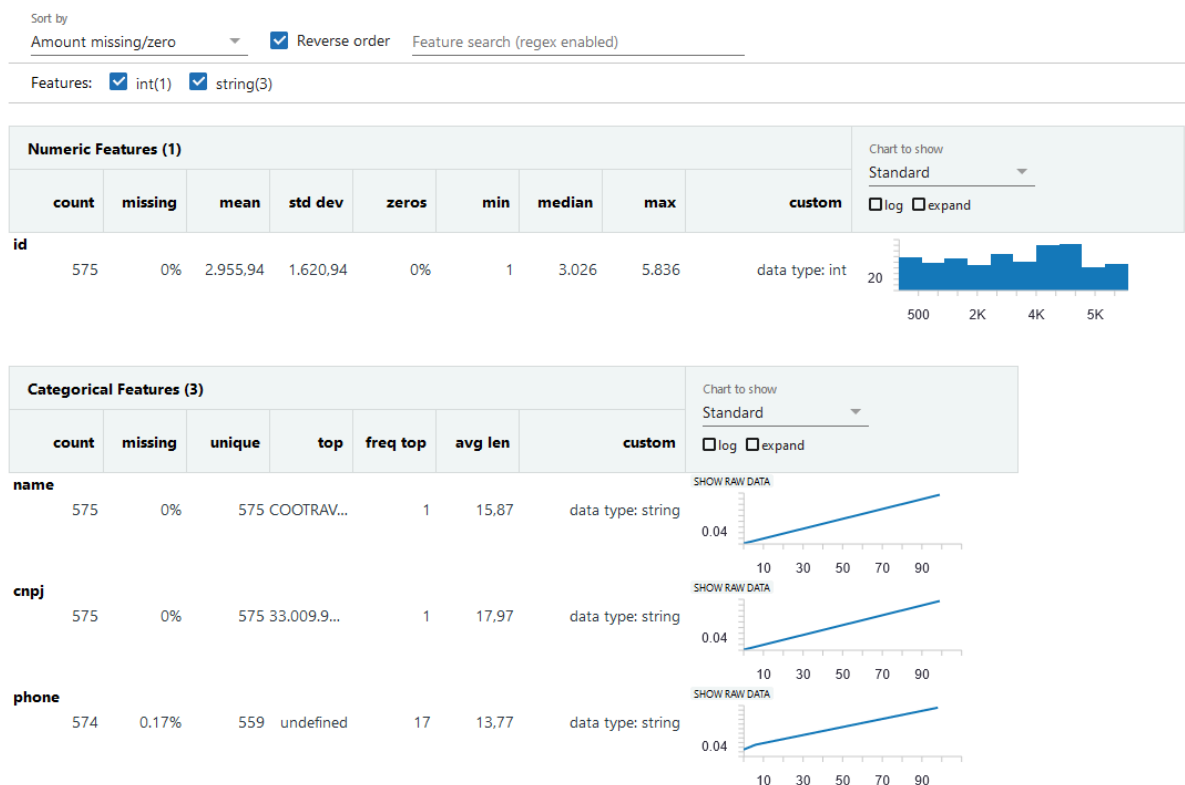


Figura 13 - Perfil de dados dOrganizacoes_bronze

Solução do problema proposto

A partir do outer join performedo nas tabelas `fConsumo_gold` e `dCompradores`, é possível responder as perguntas de negócio feitas na primeira sessão do trabalho. Para isso, criou-se uma tabela delta com o dataframe resultante do join e uma view temporária para executar as consultas em SQL e também um novo notebook com a criação de uma tabela `dCalendario`.

```
▶ 16 hours ago (1m) 23

#Leitura do arquivo final das transformações em spark dataframe
path = '/mvp_dataengiengineering/ccee/consumo_e_perfis/gold'
carbonte_e_mercadolivre = spark.read.format('parquet').options(header=True, inferSchema=True).load(path)

#Criação de uma delta table para consultas em SQL
carbonte_e_mercadolivre.write.mode('overwrite').format("delta").option("mergeSchema", "true").save("/mvp_dataengiengineering/tables/compradores_e_consumidores")

#Leitura da tabela delta
df = spark.read.format("delta").load("/mvp_dataengiengineering/tables/compradores_e_consumidores")

#Criação de uma tabela temporária
df.createOrReplaceTempView("compradores_e_consumidores")

▶ (11) Spark Jobs
▶ carbonte_e_mercadolivre: pyspark.sql.dataframe.DataFrame = [consumo_COD_AGENTE: integer, consumo_NOME_EMPRESARIAL: string ... 26 more fields]
▶ df: pyspark.sql.dataframe.DataFrame = [consumo_COD_AGENTE: integer, consumo_NOME_EMPRESARIAL: string ... 35 more fields]
```

Figura 14 - Criação da tabela delta e view temporária para consultas em SQL

```
▶ 3 days ago (1s) 1 Python

from datetime import *
from math import *
import calendar

start_date = date(datetime.today().year-2, 1, 1)
end_date = date(datetime.today().year, 12, 31)

# list of dates
dates = []
# delta of time to advanced on each loop
delta = timedelta(days=1)
loop_date = start_date

while loop_date <= end_date:
    # add loop_date with all attributes to list
    row = {
        "date": loop_date,
        "day_of_month": loop_date.day,
        "day_name": loop_date.strftime('%A'),
        "month": loop_date.month,
        "month_name": loop_date.strftime('%B'),
        "quarter": ceil(loop_date.month / 3),
        "year": loop_date.year,
        "days_in_month": calendar.monthrange(loop_date.year, loop_date.month)[1],
        "hours_in_month": calendar.monthrange(loop_date.year, loop_date.month)[1]*24
    }
    dates.append(row)
    # increment start date by timedelta
    loop_date += delta

dates_df = spark.createDataFrame(dates)
display(dates_df)
```

Figura 15 - Criação da tabela dCalendario

1) Das empresas listadas no Registro Público de Emissões, quantas estão no mercado livre de energia?

Para responder essa questão, é necessário filtrar a tabela `compradores_e_consumidores` para compradores de crédito de carbono e analisar a distribuição de grupos econômicos distintos por `PERFIL_COMPRA_CONSUMIDOR`

```
display(spark.sql("""
SELECT
    PERFIL_COMPRA_CONSUMIDOR, count(distinct(economic_group)) AS CLIENTES
FROM compradores_e_consumidores
WHERE id IS NOT NULL
group by PERFIL_COMPRA_CONSUMIDOR"""))
```

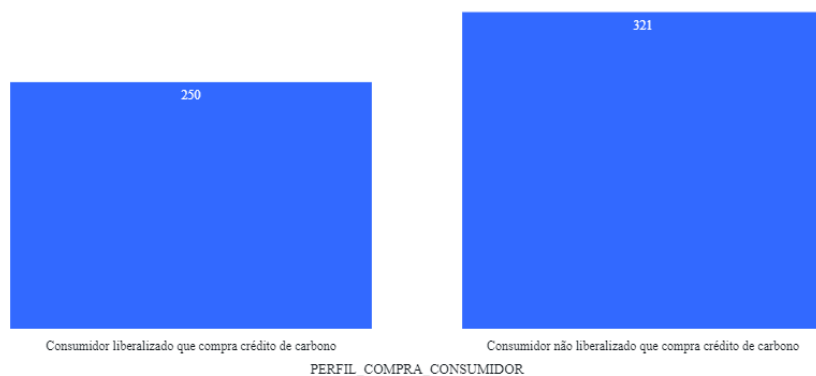


Figura 16 - Perfil do comprador de crédito de carbono

É possível observar que existe uma grande aderência para produtos de energia em empresas compradoras de crédito de carbono, pois aproximadamente 47% dos compradores já fazem parte do mercado livre de energia.

1.a) Para as companhias no mercado livre de energia, quais são os perfis de consumo (consumidor livre, consumidor especial, autoprodutor ou comercializador)?

Através do questionamento proposto, é possível identificar o perfil das empresas compradoras de ambos produtos. Para isso, foi executada a query abaixo que resultou na imagem em sequência:

```
display(spark.sql("""
SELECT
    COUNT(DISTINCT economic_group) AS distinct_economic_group,
    perfis_CLASSE_PERFIL_AGENTE
FROM compradores_e_consumidores
WHERE PERFIL_COMPRA_CONSUMIDOR = 'Consumidor liberalizado que compra crédito de
carbono'
GROUP BY perfis_CLASSE_PERFIL_AGENTE
ORDER BY distinct_economic_group ASC """))
```

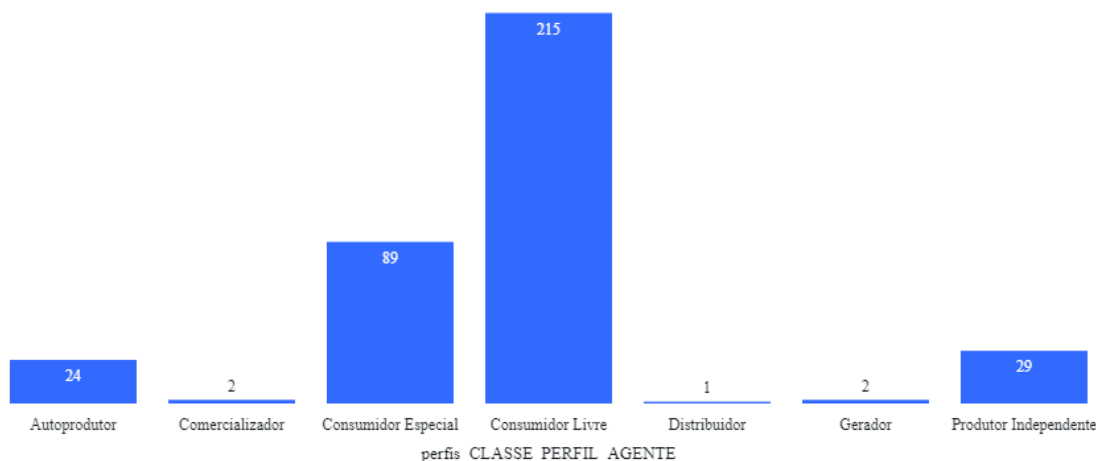


Figura 17 - Perfil de agente dos compradores de crédito de carbono que compram energia

Dessa forma, observou-se uma grande concentração de empresas com perfil *Consumidor Especial e Livre* (84%), isso significa um perfil de empresas de alto consumo energético que fazem parte do grupo tarifário de alta tensão e com demanda contratada acima de 500kW.

Analisando o histórico de migrações para o mercado livre de energia, esses perfis são compostos principalmente pelas indústrias metalúrgica, alimentícia, química, manufatura e minerais não metálicos. Uma parcela significativa também é compreendida pelo setor de serviços, principalmente construção, serviços financeiros (bancos) e relacionados à saúde (hospitais e clínicas). O setor de comércio também é fortemente representado por supermercados e hipermercados.

Assim, é possível definir que dentro dos compradores de crédito de carbono que ainda não migraram para o mercado livre, aqueles que pertencerem aos setores supracitados, possuem o perfil para oferta de produtos de energia.

1.b) Para empresas no mercado livre de energia, qual seu consumo de energia em 2024 em MWm?

O consumo em MWm é um importante balizador do potencial econômico do nicho de mercado estudado, pois define o tamanho dos contratos de energia e consequentemente o potencial financeiro que esses consumidores podem agregar às comercializadoras. Para realizar essa avaliação, executou-se a query e o seguinte resultado foi produzido:

```
display(spark.sql("""
SELECT
    COUNT(DISTINCT economic_group) AS distinct_economic_group,
    SUM(consumo_VALOR_TRC)/1000000 AS CONSUMO_TWh,
    (SUM(consumo_VALOR_TRC)/1000000) / COUNT(DISTINCT economic_group) AS AOV_TWh,
    PERFIL_COMPRA_CONSUMIDOR
FROM compradores_e_consumidores
WHERE PERFIL_COMPRA_CONSUMIDOR = 'Consumidor liberalizado que compra crédito de carbono'
GROUP BY PERFIL_COMPRA_CONSUMIDOR """))
```

perfis_CLASS E_PERFIL_AGE NTE	Grupo_Econo mico	Consumo_MWh	Consumo_MWm	Ticket_Medio_ MWm
Gerador	2,0	-		
Comercializador	2,0	2.542,0	0,9	0,4
Autoprodutor	24,0	5.997.920,0	2.065,4	86,0
null	2,0	5.658,0	1,9	1,0
Produtor Independente	29,0			
Consumidor Livre	215,0	19.057.064,0	6.562,4	30,5
Distribuidor	1,0	12.322.023,0	4.243,0	4.243,0
Consumidor Especial	89,0	1.193.531,0	410,0	4,2

Tabela 6 - Ticket Médio por perfil

Concentrando a análise nos perfis de Consumidor Livre e Especial, observa-se um ticket médio muito elevado em relação à média do mercado, que considera como clientes de grande porte a partir de 1 MWm.

2.) As empresas compradoras de crédito de carbono podem ser consideradas consumidores em potencial de energia no mercado livre?

A partir dos dados analisados nas perguntas anteriores, é possível considerar empresas compradoras de crédito de carbono como consumidores em potencial para o mercado livre de energia, pois apresentam grande aderência aos produtos e elevado ticket médio. Dessa forma, a prospecção de empresas listadas no registro que ainda não são liberalizadas apresentam um grande potencial econômico.

2.a) Quais tipos de produto são mais prováveis que essas empresas adquiram?

Por se tratarem de empresas com perfil de consumidor livre e especial e de alto ticket médio, recomenda-se a oferta de produtos de energia atacadista, para que dessa forma o consumidor tenha uma maior economia nas faturas de energia.

Considerações finais

Pontos de melhoria

Uma possível evolução para as análises apresentadas, seria a incorporação dos CNAEs provenientes da Receita Federal para os consumidores de crédito de carbono, dessa forma, seria possível elaborar uma lista de prospecção mais assertiva, pois seriam selecionados os setores elencados na pergunta 1.a.

A ingestão e tratamento das bases da Receita Federal não foi considerada no escopo do projeto por possuírem um volume muito elevado, podendo atingir as restrições de armazenamento e processamento do Cluster da versão Community Edition.

O segundo ponto desejável seria a utilização dos serviços Azure Databricks para o agendamento das pipelines, configurando atualizações diárias para a tabela calendário e atualizações mensais para o código principal. Esse serviço também permitiria a integração com o repositório do projeto no github, eliminando a etapa de upload manual dos arquivos na plataforma.

Finalmente, também é desejável que a etapa de coleta dos códigos de participantes do Registro Público de Emissões seja completamente automatizada. Devido às restrições de prazo do projeto não foi possível encontrar uma solução mais elaborada.

Autoavaliação

Em um âmbito pessoal, esse foi um projeto extremamente enriquecedor e prazeroso de se fazer! Gostei muito de ter a oportunidade de criar uma pipeline de ponta a ponta e estou muito feliz com meu trabalho. Independente de qualquer resultado final, acredito que foi um trabalho muito importante para que eu ganhasse confiança nos meus conhecimentos e “perdesse o medo” de aprofundar os estudos em conteúdos técnicos.

No que tange ao projeto realizado, acredito ter atingido os objetivos propostos, dentro de algumas limitações das ferramentas que foram discutidas em tópicos anteriores. Também acredito que após a entrega, o presente trabalho pode ser significativamente melhorado pelo aprendizado de boas práticas ao analisar as entregas dos demais colegas, principalmente por se tratar de meu primeiro projeto em engenharia de dados.