

✓ ETL com PySpark

Escola: SoulCode Academy

Curso: Analista de Dados e Dashboard

Assunto: ETL

Professora: Franciane Rodrigues

Aluno (a):

✓ Solicitação

Faça uma ETL no conjunto de dados usando PySpark

Dicionário de Dados

- RowNumber: Número da linha no conjunto de dados.
- CustomerId: Identificação única do cliente.
- Surname: Sobrenome do cliente.
- CreditScore: Pontuação de crédito do cliente, uma medida de sua credibilidade financeira.
- Geography: Localização geográfica do cliente (por exemplo, país ou região).
- Gender: Gênero do cliente.
- Age: Idade do cliente.
- Tenure: Tempo que o cliente permaneceu como cliente (em anos).
- Balance: Saldo na conta do cliente.
- NumOfProducts: Número de produtos financeiros que o cliente possui.
- HasCrCard: Indicação se o cliente possui um cartão de crédito (1 para "sim", 0 para "não").
- IsActiveMember: Indicação se o cliente é um membro ativo (1 para "sim", 0 para "não").
- EstimatedSalary: Salário estimado do cliente.
- Exited: Indicação se o cliente encerrou sua conta ou não (1 para "sim", 0 para "não").

Fonte: <https://www.kaggle.com/datasets/mervetorkan/churndataset>

✓ Infraestrutura

```
1 # Abertura Google Drive
2 from google.colab import drive
3 drive.mount('/content/drive')
```

Mounted at /content/drive

```
1 # Instalação da biblioteca
2 !pip install pyspark
```

Collecting pyspark
Downloading pyspark-3.5.1.tar.gz (317.0 MB)
317.0/317.0 MB 4.5 MB/s eta 0:00:00
Preparing metadata (setup.py) ... done
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-packages (from pyspark) (0.10.9.7)
Building wheels for collected packages: pyspark
Building wheel for pyspark (setup.py) ... done
Created wheel for pyspark: filename=pyspark-3.5.1-py2.py3-none-any.whl size=317488491 sha256=e490fca5aec74a614e84d51fa9fcf7235c1f5
Stored in directory: /root/.cache/pip/wheels/80/1d/60/2c256ed38dddce2fdd93be545214a63e02fbd8d74fb0b7f3a6
Successfully built pyspark
Installing collected packages: pyspark
Successfully installed pyspark-3.5.1

```
1 # Abertura de biblioteca
2 from pyspark.sql import SparkSession # utilizado para criação da sessão pyspark no colab
3 from pyspark.sql.functions import col, when, countDistinct
```

```
1 # Configurando ambiente
2 spark = (SparkSession.builder.master('local') # máquina local do colab
3         .appName('comandos_basicos') # Nome da aplicação
4         .config('spark.ui.port', '4050') # Porta padrão do colab;
5         .getOrCreate())
```

```
1 # Checando o ambiente criado
2 spark
```

 **SparkSession - in-memory**

SparkContext

[Spark UI](#)

Version
v3.5.1


Master
local

AppName
comandos_basicos

Extração

```
1 # Extração
2 df = (spark.read.format('csv')
3       .option('delimiter', ',')
4       .option('header', 'true')
5       .option('inferSchema', 'true')
6       .load('/content/drive/MyDrive/CHAVES/churn.csv'))
```

```
1 # Visualizando a conjunto de dados
2 df.show()
```




RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
1	15634602	Hargrave	619	France	Female	42	2	0.0	1	1	1	10134
2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	11254
3	15619304	Onio	502	France	Female	42	8	159660.8	3	1	0	11393
4	15701354	Boni	699	France	Female	39	1	0.0	2	0	0	9382
5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	796
6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	1	0	14975
7	15592531	Bartlett	822	France	Male	50	7	0.0	2	1	1	106
8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	1	0	11934
9	15792365	He	501	France	Male	44	4	142051.07	2	0	1	745
10	15592389	H?	684	France	Male	27	2	134603.88	1	1	1	7172
11	15767821	Bearce	528	France	Male	31	6	102016.72	2	0	0	8018
12	15737173	Andrews	497	Spain	Male	24	3	0.0	2	1	0	7635
13	15632264	Kay	476	France	Female	34	10	0.0	2	1	0	2626
14	15691483	Chin	549	France	Female	25	5	0.0	2	0	0	19085
15	15600882	Scott	635	Spain	Female	35	7	0.0	2	1	1	6595
16	15643966	Goforth	616	Germany	Male	45	3	143129.41	2	0	1	6432
17	15737452	Romeo	653	Germany	Male	58	1	132602.88	1	1	0	505
18	15788218	Henderson	549	Spain	Female	24	9	0.0	2	1	1	1446
19	15661507	Muldrow	587	Spain	Male	45	6	0.0	1	0	0	15868
20	15568982	Hao	726	France	Female	24	6	0.0	2	1	1	5472

only showing top 20 rows

Pré-Análise

```
1 # Visualizando a conjunto de dados
2 df.show()
```



RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
1	15634602	Hargrave	619	France	Female	42	2	0.0	1	1	1	10134
2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	11254
3	15619304	Onio	502	France	Female	42	8	159660.8	3	1	0	11393
4	15701354	Boni	699	France	Female	39	1	0.0	2	0	0	9382
5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	796
6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	1	0	14975
7	15592531	Bartlett	822	France	Male	50	7	0.0	2	1	1	106
8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	1	0	11934
9	15792365	He	501	France	Male	44	4	142051.07	2	0	1	745
10	15592389	H?	684	France	Male	27	2	134603.88	1	1	1	7172
11	15767821	Bearce	528	France	Male	31	6	102016.72	2	0	0	8018
12	15737173	Andrews	497	Spain	Male	24	3	0.0	2	1	0	7635
13	15632264	Kay	476	France	Female	34	10	0.0	2	1	0	2626
14	15691483	Chin	549	France	Female	25	5	0.0	2	0	0	19085
15	15600882	Scott	635	Spain	Female	35	7	0.0	2	1	1	6595
16	15643966	Goforth	616	Germany	Male	45	3	143129.41	2	0	1	6432
17	15737452	Romeo	653	Germany	Male	58	1	132602.88	1	1	0	505
18	15788218	Henderson	549	Spain	Female	24	9	0.0	2	1	1	1446
19	15661507	Muldrow	587	Spain	Male	45	6	0.0	1	0	0	15868

```
|      20| 15568982| Hao|      726| France|Female| 24|      6|      0.0|      2|      1|      1|      547|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows
```

```
1 # Visualização do Cabeçalho
2 df.head()
```

```
Row(RowNumber=1, CustomerId=15634602, Surname='Hargrave', CreditScore=619, Geography='France', Gender='Female', Age=42, Tenure=2,
Balance=0.0, NumOfProducts=1, HasCrCard=1, IsActiveMember=1, EstimatedSalary=101348.88, Exited=1)
```

```
1 # Visualizando as últimas posições do DataFrame
2 # Usar count() para determinar o número total de linhas e então mostrar as últimas linhas
3 total_rows = df.count()
4 df.limit(5).show(total_rows - 5) # Utilize + se quiser aparecer as primeiras posições
```

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|RowNumber|CustomerId| Surname|CreditScore|Geography|Gender|Age|Tenure| Balance|NumOfProducts|HasCrCard|IsActiveMember|EstimatedSal|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|      1| 15634602|Hargrave|      619| France|Female| 42|      2|      0.0|      1|      1|      1|      101348|
|      2| 15647311| Hill|      608| Spain|Female| 41|      1| 83807.86|      1|      0|      1|      112547|
|      3| 15619304| Onio|      502| France|Female| 42|      8| 159660.8|      3|      1|      0|      113931|
|      4| 15701354| Boni|      699| France|Female| 39|      1|      0.0|      2|      0|      0|      93826|
|      5| 15737888|Mitchell|      850| Spain|Female| 43|      2|125510.82|      1|      1|      1|      7908|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

```
1 # As cinco ultimas posições
2 df.tail(5)
```

```
[Row(RowNumber=9996, CustomerId=15606229, Surname='Obijiaku', CreditScore=771, Geography='France', Gender='Male', Age=39, Tenure=5,
Balance=0.0, NumOfProducts=2, HasCrCard=1, IsActiveMember=0, EstimatedSalary=96270.64, Exited=0),
Row(RowNumber=9997, CustomerId=15569892, Surname='Johnstone', CreditScore=516, Geography='France', Gender='Male', Age=35,
Tenure=10, Balance=57369.61, NumOfProducts=1, HasCrCard=1, IsActiveMember=1, EstimatedSalary=101699.77, Exited=0),
Row(RowNumber=9998, CustomerId=15584532, Surname='Liu', CreditScore=709, Geography='France', Gender='Female', Age=36, Tenure=7,
Balance=0.0, NumOfProducts=1, HasCrCard=0, IsActiveMember=1, EstimatedSalary=42085.58, Exited=1),
Row(RowNumber=9999, CustomerId=15682355, Surname='Sabbatini', CreditScore=772, Geography='Germany', Gender='Male', Age=42,
Tenure=3, Balance=75075.31, NumOfProducts=2, HasCrCard=1, IsActiveMember=0, EstimatedSalary=92888.52, Exited=1),
Row(RowNumber=10000, CustomerId=15628319, Surname='Walker', CreditScore=792, Geography='France', Gender='Female', Age=28,
Tenure=4, Balance=130142.79, NumOfProducts=1, HasCrCard=1, IsActiveMember=0, EstimatedSalary=38190.78, Exited=0)]
```

```
1 # Visualização do DataFrame de forma aleatória
2 df.sample(False, 0.1).show(4) # 0.1 indica a fração das linhas a serem amostradas
```

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|RowNumber|CustomerId|Surname|CreditScore|Geography|Gender|Age|Tenure| Balance|NumOfProducts|HasCrCard|IsActiveMember|EstimatedSal|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|      25| 15625047| Yen|      846| France|Female| 38|      5|      0.0|      1|      1|      1|      187616|
|      38| 15729599|Lorenzo|      804| Spain| Male| 33|      7| 76548.6|      1|      0|      1|      98453|
|      40| 15585768|Cameron|      582| Germany| Male| 41|      6| 70349.48|      2|      0|      1|      178074|
|      45| 15684171|Bianchi|      660| Spain|Female| 61|      5|155931.11|      1|      1|      1|      158338|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 4 rows
```

```
1 # Verificando o tamanho do DataFrame (linhas, colunas)
2 print((df.count(), len(df.columns)))
```

```
(10000, 14)
```

```
1 # Verificando o tipo de dados de cada coluna
2 df.dtypes
```

```
[('RowNumber', 'int'),
 ('CustomerId', 'int'),
 ('Surname', 'string'),
 ('CreditScore', 'int'),
 ('Geography', 'string'),
 ('Gender', 'string'),
 ('Age', 'int'),
 ('Tenure', 'int'),
 ('Balance', 'double'),
 ('NumOfProducts', 'int'),
 ('HasCrCard', 'int'),
 ('IsActiveMember', 'int'),
 ('EstimatedSalary', 'double'),
 ('Exited', 'int')]
```

```
1 # Contar as observações em cada coluna
2 column_counts = [(col, df.select(col).count()) for col in df.columns]
```

```

3
4 # Exibir os resultados
5 for col, count in column_counts:
6     print(f"{col}: {count}")

```

```

↗ RowNumber: 10000
CustomerId: 10000
Surname: 10000
CreditScore: 10000
Geography: 10000
Gender: 10000
Age: 10000
Tenure: 10000
Balance: 10000
NumOfProducts: 10000
HasCrCard: 10000
IsActiveMember: 10000
EstimatedSalary: 10000
Exited: 10000

```

```

1 # Informações detalhadas do conjunto de dados
2 df.printSchema()

```

```

↗ root
|-- RowNumber: integer (nullable = true)
|-- CustomerId: integer (nullable = true)
|-- Surname: string (nullable = true)
|-- CreditScore: integer (nullable = true)
|-- Geography: string (nullable = true)
|-- Gender: string (nullable = true)
|-- Age: integer (nullable = true)
|-- Tenure: integer (nullable = true)
|-- Balance: double (nullable = true)
|-- NumOfProducts: integer (nullable = true)
|-- HasCrCard: integer (nullable = true)
|-- IsActiveMember: integer (nullable = true)
|-- EstimatedSalary: double (nullable = true)
|-- Exited: integer (nullable = true)

```

+ Código

+ Texto

▼ Transformação

```

1 # Visualização do conjunto de dados
2 df.show()

```

```

↗ +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|RowNumber|CustomerId| Surname|CreditScore|Geography|Gender|Age|Tenure| Balance|NumOfProducts|HasCrCard|IsActiveMember|EstimatedSalary|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|1|15634602|Hargrave|619|France|Female|42|2|0.0|1|1|1|10134|
|2|15647311|Hill|608|Spain|Female|41|1|83807.86|1|0|1|11254|
|3|15619304|Onio|502|France|Female|42|8|159660.8|3|1|0|11393|
|4|15701354|Boni|699|France|Female|39|1|0.0|2|0|0|9387|
|5|15737888|Mitchell|850|Spain|Female|43|2|125510.82|1|1|1|796|
|6|15574012|Chu|645|Spain|Male|44|8|113755.78|2|1|0|14975|
|7|15592531|Bartlett|822|France|Male|50|7|0.0|2|1|1|106|
|8|15656148|Obinna|376|Germany|Female|29|4|115046.74|4|1|0|11934|
|9|15792365|He|501|France|Male|44|4|142051.07|2|0|1|745|
|10|15592389|H?|684|France|Male|27|2|134603.88|1|1|1|7172|
|11|15767821|Bearce|528|France|Male|31|6|102016.72|2|0|0|8018|
|12|15737173|Andrews|497|Spain|Male|24|3|0.0|2|1|0|7635|
|13|15632264|Kay|476|France|Female|34|10|0.0|2|1|0|2626|
|14|15691483|Chin|549|France|Female|25|5|0.0|2|0|0|19085|
|15|15600882|Scott|635|Spain|Female|35|7|0.0|2|1|1|6595|
|16|15643966|Goforth|616|Germany|Male|45|3|143129.41|2|0|1|6437|
|17|15737452|Romeo|653|Germany|Male|58|1|132602.88|1|1|0|505|
|18|15788218|Henderson|549|Spain|Female|24|9|0.0|2|1|1|1446|
|19|15661507|Muldrow|587|Spain|Male|45|6|0.0|1|0|0|15866|
|20|15568982|Hao|726|France|Female|24|6|0.0|2|1|1|5472|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

```

▼ Renomeando colunas

```

1 # Renomear as colunas
2 df = df \
3     .withColumnRenamed("RowNumber", "numero_linha") \
4     .withColumnRenamed("CustomerId", "id") \
5     .withColumnRenamed("Surname", "sobrenome") \
6     .withColumnRenamed("CreditScore", "pontuacao_credito") \
7     .withColumnRenamed("Geography", "localizacao") \

```

```

8 .withColumnRenamed("Gender", "genero") \
9 .withColumnRenamed("Age", "idade") \
10 .withColumnRenamed("Tenure", "tempo_permanencia") \
11 .withColumnRenamed("Balance", "saldo") \
12 .withColumnRenamed("NumOfProducts", "num_produtos") \
13 .withColumnRenamed("HasCrCard", "tem_cartao_credito") \
14 .withColumnRenamed("IsActiveMember", "membro_ativo") \
15 .withColumnRenamed("EstimatedSalary", "salario_estimado") \
16 .withColumnRenamed("Exited", "encerrou_conta")

```

```

1 # Visualização do conjunto de dados
2 df.show()

```

```

↗ +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|numero_linha|      id|sobrenome|pontuacao_credito|localizacao|genero|idade|tempo_permanencia|      saldo|num_produtos|tem_cartao_credito|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|1|15634602|Hargrave|619|France|Female|42|2|0.0|1|
|2|15647311|Hill|608|Spain|Female|41|1|83807.86|1|
|3|15619304|Onio|502|France|Female|42|8|159660.8|3|
|4|15701354|Boni|699|France|Female|39|1|0.0|2|
|5|15737888|Mitchell|850|Spain|Female|43|2|125510.82|1|
|6|15574012|Chu|645|Spain|Male|44|8|113755.78|2|
|7|15592531|Bartlett|822|France|Male|50|7|0.0|2|
|8|15656148|Obinna|376|Germany|Female|29|4|115046.74|4|
|9|15792365|He|501|France|Male|44|4|142051.07|2|
|10|15592389|H?|684|France|Male|27|2|134603.88|1|
|11|15767821|Bearce|528|France|Male|31|6|102016.72|2|
|12|15737173|Andrews|497|Spain|Male|24|3|0.0|2|
|13|15632264|Kay|476|France|Female|34|10|0.0|2|
|14|15691483|Chin|549|France|Female|25|5|0.0|2|
|15|15600882|Scott|635|Spain|Female|35|7|0.0|2|
|16|15643966|Goforth|616|Germany|Male|45|3|143129.41|2|
|17|15737452|Romeo|653|Germany|Male|58|1|132602.88|1|
|18|15788218|Henderson|549|Spain|Female|24|9|0.0|2|
|19|15661507|Muldrow|587|Spain|Male|45|6|0.0|1|
|20|15568982|Hao|726|France|Female|24|6|0.0|2|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

```

Valores nulos

```

1 # Verificar valores nulos em cada coluna
2 for column in df.columns:
3     null_count = df.filter(df[column].isNull()).count()
4     print(f"{column}: {null_count} nulos")

```

```

↗ numero_linha: 0 nulos
id: 0 nulos
sobrenome: 0 nulos
pontuacao_credito: 0 nulos
localizacao: 0 nulos
genero: 0 nulos
idade: 0 nulos
tempo_permanencia: 0 nulos
saldo: 0 nulos
num_produtos: 0 nulos
tem_cartao_credito: 0 nulos
membro_ativo: 0 nulos
salario_estimado: 0 nulos
encerrou_conta: 0 nulos

```

```

1 # Antes de eliminar de vez, por favor filtre os dados
2 # Eliminação de dados nulos - como não tem deixei comentado
3 #df.dropna()

```

Valores únicos e duplicados

```

1 # Verificando se os dados são únicos na coluna 'id'
2 # Pelo fato de ter aparecido 'false', significa que temos dados duplicados
3 print("A coluna id possui valores únicos? Resposta:", df.select("id").distinct().count() == df.count())

```

```

↗ A coluna id possui valores únicos? Resposta: True

```

```

1 # Verificando se os dados são únicos na coluna 'sobrenome'
2 # Pelo fato de ter aparecido 'false', significa que temos dados duplicados
3 print("A coluna sobrenome possui valores únicos? Resposta:", df.select("sobrenome").distinct().count() == df.count())

```

```

↗ A coluna sobrenome possui valores únicos? Resposta: False

```

```
1 # Visualizando dados duplicados no conjunto de dados
2 df.orderBy("sobrenome").show()
```

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|numero_linha|id|sobrenome|pontuacao_credito|localizacao|genero|idade|tempo_permanencia|saldo|num_produtos|tem_cartao_cre|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|3272|15708791|Abazu|584|Spain|Male|32|9|85534.83|1|
|5110|15576156|Abazu|710|Spain|Female|28|6|0.0|1|
|842|15737792|Abbie|818|France|Female|31|1|186796.37|1|
|2538|15723706|Abbott|573|France|Female|33|0|90124.64|1|
|2989|15684801|Abbott|689|France|Male|47|1|93871.95|3|
|4574|15693906|Abbott|645|France|Female|24|3|34547.82|1|
|6839|15680804|Abbott|850|France|Male|29|6|0.0|2|
|8485|15601012|Abdullah|802|France|Female|60|3|92887.06|1|
|6085|15619494|Abdulov|562|Germany|Female|31|9|117153.0|1|
|6328|15793856|Abdulov|667|Spain|Female|36|3|121542.57|2|
|4457|15724428|Abel|544|France|Male|40|8|0.0|2|
|9374|15807457|Abernathy|641|Spain|Female|36|1|0.0|2|
|8396|15586069|Abernathy|560|France|Female|30|0|108883.29|1|
|1162|15781802|Abramov|755|France|Male|41|6|104817.41|1|
|5461|15668894|Abramova|661|Germany|Male|41|5|122552.48|2|
|7659|15666297|Abramova|706|Spain|Female|53|3|0.0|3|
|5665|15661723|Abramovich|667|Spain|Male|71|4|137260.78|1|
|3381|15601184|Abramovich|604|Spain|Female|26|3|0.0|2|
|3825|15728167|Abramovich|667|France|Male|44|2|122806.95|1|
|1043|15593969|Abramovich|630|Spain|Female|39|7|135483.17|1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

```

```
1 # Verificando a quantidade de sobrenomes repetidos na base de dados em ordem decrescente
2 df.groupBy("sobrenome").count().orderBy("count", ascending=False).show()
```

```

+-----+-----+
|sobrenome|count|
+-----+-----+
|Smith|32|
|Scott|29|
|Martin|29|
|Walker|28|
|Brown|26|
|Genovese|25|
|Yeh|25|
|Shih|25|
|Wright|24|
|Maclean|24|
|Ma|23|
|Fanucci|23|
|Wilson|23|
|White|23|
|Lu|22|
|Chu|22|
|Wang|22|
|Moore|22|
|Johnson|22|
|Sun|21|
+-----+-----+
only showing top 20 rows

```

```
1 # Verificando a quantidade de sobrenomes repetidos na base de dados
2 df.groupBy("sobrenome").count().show()
```

```

+-----+-----+
|sobrenome|count|
+-----+-----+
|Tyler|4|
|Palermo|12|
|Piccio|13|
|Lazareva|3|
|Kaminachi|5|
|Virgo|2|
|Baryshnikov|2|
|Wofford|1|
|Lavrov|2|
|Bezrukova|2|
|Avdeev|1|
|Clunie|1|
|Duigan|2|
|Sokolova|2|
|Azarov|1|
|Rawlings|1|
|Zox|1|
|Rubeo|1|
|Arbour|1|
|Rapuluchukwu|1|
+-----+-----+

```

```
+-----+-----+
only showing top 20 rows
```

```
1 # Filtrando pelo nome
2 df.filter(df["sobrenome"] == "Piccio").show()
```

numero_linha	id	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_cred
128	15782688	Piccio	625	Germany	Male	56	0	148507.24	1	
1146	15601688	Piccio	546	France	Male	28	8	0.0	1	
3815	15642093	Piccio	646	France	Male	30	7	0.0	2	
4470	15692443	Piccio	612	Spain	Male	33	5	69478.57	1	
4569	15672875	Piccio	584	Germany	Male	32	8	40172.91	1	
5305	15671345	Piccio	531	Spain	Female	42	6	75302.85	2	
5364	15663410	Piccio	771	Spain	Male	51	5	135506.58	3	
6768	15654964	Piccio	608	Spain	Male	48	7	75801.74	1	
6863	15736287	Piccio	586	France	Male	33	9	0.0	1	
8216	15743236	Piccio	687	France	Female	61	7	80538.56	1	
8340	15658100	Piccio	695	France	Female	42	0	0.0	2	
8547	15686957	Piccio	553	Germany	Male	35	2	158584.28	2	
9313	15658946	Piccio	579	Germany	Male	40	10	45408.85	2	

```
1 # Salvando o filtro da familia Piccio
2 df_filtro1 = df.filter(df["sobrenome"] == "Piccio")
```

```
1 # Mostrando resultado do filtro
2 df_filtro1.show()
```

numero_linha	id	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_cred
128	15782688	Piccio	625	Germany	Male	56	0	148507.24	1	
1146	15601688	Piccio	546	France	Male	28	8	0.0	1	
3815	15642093	Piccio	646	France	Male	30	7	0.0	2	
4470	15692443	Piccio	612	Spain	Male	33	5	69478.57	1	
4569	15672875	Piccio	584	Germany	Male	32	8	40172.91	1	
5305	15671345	Piccio	531	Spain	Female	42	6	75302.85	2	
5364	15663410	Piccio	771	Spain	Male	51	5	135506.58	3	
6768	15654964	Piccio	608	Spain	Male	48	7	75801.74	1	
6863	15736287	Piccio	586	France	Male	33	9	0.0	1	
8216	15743236	Piccio	687	France	Female	61	7	80538.56	1	
8340	15658100	Piccio	695	France	Female	42	0	0.0	2	
8547	15686957	Piccio	553	Germany	Male	35	2	158584.28	2	
9313	15658946	Piccio	579	Germany	Male	40	10	45408.85	2	

```
1 #Filtrar as linhas onde encerrou_conta é igual a 0
2 encerrou_conta_0 = df.filter(df.encerrou_conta == 0)
```

```
1 # Filtrar as linhas onde encerrou_conta é igual a 1
2 encerrou_conta_1 = df.filter(df.encerrou_conta == 1)
```

```
1 # Verificando o tamanho do DataFrame (linhas, colunas)
2 print((df.count(), len(df.columns)))
```

```
(10000, 14)
```

```
1 # Backup do conjunto original
2 dfback = df
```

```
1 # mostrando o conjunto
2 dfback.show()
```

numero_linha	id	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_cred
1	15634602	Hargrave	619	France	Female	42	2	0.0	1	
2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	
3	15619304	Onio	502	France	Female	42	8	159660.8	3	
4	15701354	Boni	699	France	Female	39	1	0.0	2	
5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	
6	15574012	Chu	645	Spain	Male	44	8	113755.78	2	
7	15592531	Bartlett	822	France	Male	50	7	0.0	2	
8	15656148	Obinna	376	Germany	Female	29	4	115046.74	4	
9	15792365	He	501	France	Male	44	4	142051.07	2	

10	15592389	H?	684	France	Male	27	2	134603.88	1
11	15767821	Bearce	528	France	Male	31	6	102016.72	2
12	15737173	Andrews	497	Spain	Male	24	3	0.0	2
13	15632264	Kay	476	France	Female	34	10	0.0	2
14	15691483	Chin	549	France	Female	25	5	0.0	2
15	15600882	Scott	635	Spain	Female	35	7	0.0	2
16	15643966	Goforth	616	Germany	Male	45	3	143129.41	2
17	15737452	Romeo	653	Germany	Male	58	1	132602.88	1
18	15788218	Henderson	549	Spain	Female	24	9	0.0	2
19	15661507	Muldrow	587	Spain	Male	45	6	0.0	1
20	15568982	Hao	726	France	Female	24	6	0.0	2

only showing top 20 rows

```
1 # Eliminando valores duplicados na coluna sobrenome
2 df = df.dropDuplicates(['sobrenome'])
```

```
1 # Checando o tamanho do DataFrame (linhas, colunas)
2 print((df.count(), len(df.columns)))
```

↗ (2932, 14)

✓ Seleção de Colunas Importantes

```
1 df.show()
```

↗

numero_linha	id	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_cre
3272	15708791	Abazu	584	Spain	Male	32	9	85534.83	1	
842	15737792	Abbie	818	France	Female	31	1	186796.37	1	
2538	15723706	Abbott	573	France	Female	33	0	90124.64	1	
8485	15601012	Abdullah	802	France	Female	60	3	92887.06	1	
6085	15619494	Abdulov	562	Germany	Female	31	9	117153.0	1	
4457	15724428	Abel	544	France	Male	40	8	0.0	2	
8396	15586069	Abernathy	560	France	Female	30	0	108883.29	1	
1162	15781802	Abramov	755	France	Male	41	6	104817.41	1	
5461	15668894	Abramova	661	Germany	Male	41	5	122552.48	2	
1043	15593969	Abramovich	630	Spain	Female	39	7	135483.17	1	
3269	15611430	Abramowitz	690	France	Male	54	5	0.0	1	
8952	15636388	Abrego	702	Germany	Female	23	7	98775.23	1	
8387	15641110	Abron	708	France	Male	41	0	0.0	1	
683	15775238	Achebe	651	Germany	Female	41	4	133432.59	1	
3467	15631339	Adams	791	France	Male	28	4	0.0	1	
3601	15573599	Adamson	506	France	Female	57	6	0.0	2	
7743	15571940	Afamefula	579	Spain	Male	22	3	118680.57	1	
1794	15777922	Afamefuna	629	Spain	Male	36	1	161757.87	2	
7471	15602456	Afanasyev	850	Germany	Female	47	4	99219.47	2	
5765	15791851	Afanasyeva	726	France	Female	34	0	185734.75	1	

only showing top 20 rows

```
1 # Simulação
2 df.drop("numero_linha", "id").show()
```

↗

sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito	membro_ativo	sal
Abazu	584	Spain	Male	32	9	85534.83	1	0	0	
Abbie	818	France	Female	31	1	186796.37	1	0	0	
Abbott	573	France	Female	33	0	90124.64	1	1	0	
Abdullah	802	France	Female	60	3	92887.06	1	1	0	
Abdulov	562	Germany	Female	31	9	117153.0	1	1	1	
Abel	544	France	Male	40	8	0.0	2	1	0	
Abernathy	560	France	Female	30	0	108883.29	1	1	0	
Abramov	755	France	Male	41	6	104817.41	1	1	0	
Abramova	661	Germany	Male	41	5	122552.48	2	0	1	
Abramovich	630	Spain	Female	39	7	135483.17	1	1	0	
Abramowitz	690	France	Male	54	5	0.0	1	1	0	
Abrego	702	Germany	Female	23	7	98775.23	1	1	0	
Abron	708	France	Male	41	0	0.0	1	1	0	
Achebe	651	Germany	Female	41	4	133432.59	1	0	1	
Adams	791	France	Male	28	4	0.0	1	1	0	
Adamson	506	France	Female	57	6	0.0	2	0	1	
Afamefula	579	Spain	Male	22	3	118680.57	1	1	1	
Afamefuna	629	Spain	Male	36	1	161757.87	2	1	1	
Afanasyev	850	Germany	Female	47	4	99219.47	2	1	1	
Afanasyeva	726	France	Female	34	0	185734.75	1	1	1	

only showing top 20 rows


```
1 # Aplicação
2 df = df.drop("numero_linha", "id")
```

```
1 # Verificação
2 df.show()
```

```

↳ +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| sobrenome|pontuacao_credito|localizacao|genero|idade|tempo_permanencia|saldo|num_produtos|tem_cartao_credito|membro_ativo|sal:
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Abazu|584|Spain|Male|32|9|85534.83|1|0|0|
| Abbie|818|France|Female|31|1|186796.37|1|0|0|
| Abbott|573|France|Female|33|0|90124.64|1|1|0|
| Abdullah|802|France|Female|60|3|92887.06|1|1|0|
| Abdulov|562|Germany|Female|31|9|117153.0|1|1|1|
| Abel|544|France|Male|40|8|0.0|2|1|0|
| Abernathy|560|France|Female|30|0|108883.29|1|1|0|
| Abramov|755|France|Male|41|6|104817.41|1|1|0|
| Abramova|661|Germany|Male|41|5|122552.48|2|0|1|
| Abramovich|630|Spain|Female|39|7|135483.17|1|1|0|
| Abramowitz|690|France|Male|54|5|0.0|1|1|0|
| Abrego|702|Germany|Female|23|7|98775.23|1|1|0|
| Abron|708|France|Male|41|0|0.0|1|1|0|
| Achebe|651|Germany|Female|41|4|133432.59|1|0|1|
| Adams|791|France|Male|28|4|0.0|1|1|0|
| Adamson|506|France|Female|57|6|0.0|2|0|1|
| Afamefula|579|Spain|Male|22|3|118680.57|1|1|1|
| Afamefuna|629|Spain|Male|36|1|161757.87|2|1|1|
| Afanasyev|850|Germany|Female|47|4|99219.47|2|1|1|
| Afanasyeva|726|France|Female|34|0|185734.75|1|1|1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows

```

```
1 # Selecionando colunas
2 df.select("genero", "pontuacao_credito", "localizacao", "saldo").show()
```

```

↳ +-----+-----+-----+-----+
| genero|pontuacao_credito|localizacao|saldo|
+-----+-----+-----+-----+
| Male|584|Spain|85534.83|
| Female|818|France|186796.37|
| Female|573|France|90124.64|
| Female|802|France|92887.06|
| Female|562|Germany|117153.0|
| Male|544|France|0.0|
| Female|560|France|108883.29|
| Male|755|France|104817.41|
| Male|661|Germany|122552.48|
| Female|630|Spain|135483.17|
| Male|690|France|0.0|
| Female|702|Germany|98775.23|
| Male|708|France|0.0|
| Female|651|Germany|133432.59|
| Male|791|France|0.0|
| Female|506|France|0.0|
| Male|579|Spain|118680.57|
| Male|629|Spain|161757.87|
| Female|850|Germany|99219.47|
| Female|726|France|185734.75|
+-----+-----+-----+-----+
only showing top 20 rows

```

```
1 # Caso queira usar: Selecionando colunas
2 df_selec = df.select("genero", "pontuacao_credito", "localizacao", "saldo")
```

```
1 # Verificando
2 df_selec.show()
```

```

↳ +-----+-----+-----+-----+
| genero|pontuacao_credito|localizacao|saldo|
+-----+-----+-----+-----+
| Male|584|Spain|85534.83|
| Female|818|France|186796.37|
| Female|573|France|90124.64|
| Female|802|France|92887.06|
| Female|562|Germany|117153.0|
| Male|544|France|0.0|
| Female|560|France|108883.29|
| Male|755|France|104817.41|
| Male|661|Germany|122552.48|
| Female|630|Spain|135483.17|

```

```

| Male|      690| France|    0.0|
| Female|    702| Germany| 98775.23|
| Male|    708| France|    0.0|
| Female|    651| Germany| 133432.59|
| Male|    791| France|    0.0|
| Female|    506| France|    0.0|
| Male|    579| Spain| 118680.57|
| Male|    629| Spain| 161757.87|
| Female|    850| Germany| 99219.47|
| Female|    726| France| 185734.75|
+-----+-----+-----+-----+

```

only showing top 20 rows

Tradução de Categorias

```
1 df.show()
```

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| sobrenome|pontuacao_credito|localizacao|genero|idade|tempo_permanencia|  saldo|num_produtos|tem_cartao_credito|membro_ativo|salari
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Abazu|      584| Spain| Male| 32|      9| 85534.83|      1|      0|      0|
| Abbie|      818| France| Female| 31|      1| 186796.37|      1|      0|      0|
| Abbott|      573| France| Female| 33|      0| 90124.64|      1|      1|      0|
| Abdullah|      802| France| Female| 60|      3| 92887.06|      1|      1|      0|
| Abdulov|      562| Germany| Female| 31|      9| 117153.0|      1|      1|      1|
| Abel|      544| France| Male| 40|      8|      0.0|      2|      1|      0|
| Abernathy|      560| France| Female| 30|      0| 108883.29|      1|      1|      0|
| Abramov|      755| France| Male| 41|      6| 104817.41|      1|      1|      0|
| Abramova|      661| Germany| Male| 41|      5| 122552.48|      2|      0|      1|
| Abramovich|      630| Spain| Female| 39|      7| 135483.17|      1|      1|      0|
| Abramowitz|      690| France| Male| 54|      5|      0.0|      1|      1|      0|
| Abrego|      702| Germany| Female| 23|      7| 98775.23|      1|      1|      0|
| Abron|      708| France| Male| 41|      0|      0.0|      1|      1|      0|
| Achebe|      651| Germany| Female| 41|      4| 133432.59|      1|      0|      1|
| Adams|      791| France| Male| 28|      4|      0.0|      1|      1|      0|
| Adamson|      506| France| Female| 57|      6|      0.0|      2|      0|      1|
| Afamefula|      579| Spain| Male| 22|      3| 118680.57|      1|      1|      1|
| Afamefuna|      629| Spain| Male| 36|      1| 161757.87|      2|      1|      1|
| Afanasyev|      850| Germany| Female| 47|      4| 99219.47|      2|      1|      1|
| Afanasyeva|      726| France| Female| 34|      0| 185734.75|      1|      1|      1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+

```

only showing top 20 rows

A expressão `rdd.map(lambda x: x[0]).collect()` em PySpark faz o seguinte:

- RDD (rdd): Representa um Resilient Distributed Dataset, uma estrutura de dados fundamental em PySpark, que permite operações distribuídas.
- `map(lambda x: x[0])`: Aplica uma função lambda a cada elemento do RDD, onde lambda `x: x[0]` extrai o primeiro elemento de cada tupla ou lista `x`.
- `collect()`: Coleta todos os elementos do RDD de volta para o driver (o ambiente principal onde o código é executado), retornando uma lista de Python contendo todos os elementos processados pelo map.

Portanto, `rdd.map(lambda x: x[0]).collect()` retorna uma lista com o primeiro elemento de cada elemento do RDD rdd

```

1 # Verificação de valores unicos
2 sorted(df.select("localizacao").distinct().rdd.map(lambda x: x[0]).collect())

```

```
['France', 'Germany', 'Spain']
```

```

1 # Verificação de valores unicos
2 sorted(df.select("genero").distinct().rdd.map(lambda x: x[0]).collect())

```

```
['Female', 'Male']
```

```

1 # Simulação
2 df.withColumn("localizacao", when(df["localizacao"] == "France", "França") \
3                               .when(df["localizacao"] == "Germany", "Alemanha")\
4                               .when(df["localizacao"] == "Spain", "Espanha")\
5                               .otherwise(df["localizacao"])).show()

```

```

+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| sobrenome|pontuacao_credito|localizacao|genero|idade|tempo_permanencia|  saldo|num_produtos|tem_cartao_credito|membro_ativo|salari
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Abazu|      584| Espanha| Male| 32|      9| 85534.83|      1|      0|      0|
| Abbie|      818| França| Female| 31|      1| 186796.37|      1|      0|      0|
| Abbott|      573| França| Female| 33|      0| 90124.64|      1|      1|      0|
| Abdullah|      802| França| Female| 60|      3| 92887.06|      1|      1|      0|

```

	Abdulov	562	Alemanha	Female	31	9	117153.0	1	1	1
	Abel	544	França	Male	40	8	0.0	2	1	0
	Abernathy	560	França	Female	30	0	108883.29	1	1	0
	Abramov	755	França	Male	41	6	104817.41	1	1	0
	Abramova	661	Alemanha	Male	41	5	122552.48	2	0	1
	Abramovich	630	Espanha	Female	39	7	135483.17	1	1	0
	Abramowitz	690	França	Male	54	5	0.0	1	1	0
	Abrego	702	Alemanha	Female	23	7	98775.23	1	1	0
	Abron	708	França	Male	41	0	0.0	1	1	0
	Achebe	651	Alemanha	Female	41	4	133432.59	1	0	1
	Adams	791	França	Male	28	4	0.0	1	1	0
	Adamson	506	França	Female	57	6	0.0	2	0	1
	Afamefula	579	Espanha	Male	22	3	118680.57	1	1	1
	Afamefuna	629	Espanha	Male	36	1	161757.87	2	1	1
	Afanasyev	850	Alemanha	Female	47	4	99219.47	2	1	1
	Afanasyeva	726	França	Female	34	0	185734.75	1	1	1

only showing top 20 rows

```
1 # Aplicação
2 df = df.withColumn("localizacao", when(df["localizacao"] == "France", "França") \
3     .when(df["localizacao"] == "Germany", "Alemanha")\
4     .when(df["localizacao"] == "Spain", "Espanha")\
5     .otherwise(df["localizacao"]))
```

```
1 # Simulação
2 df.withColumn("genero", when(df["genero"] == "Female", "Feminino") \
3     .when(df["genero"] == "Male", "Masculino") \
4     .otherwise(df["genero"])).show()
```

	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito	membro_ativo
	Abazu	584	Espanha	Masculino	32	9	85534.83	1	0	0
	Abbie	818	França	Feminino	31	1	186796.37	1	0	0
	Abbott	573	França	Feminino	33	0	90124.64	1	1	0
	Abdullah	802	França	Feminino	60	3	92887.06	1	1	0
	Abdulov	562	Alemanha	Feminino	31	9	117153.0	1	1	1
	Abel	544	França	Masculino	40	8	0.0	2	1	0
	Abernathy	560	França	Feminino	30	0	108883.29	1	1	0
	Abramov	755	França	Masculino	41	6	104817.41	1	1	0
	Abramova	661	Alemanha	Masculino	41	5	122552.48	2	0	1
	Abramovich	630	Espanha	Feminino	39	7	135483.17	1	1	0
	Abramowitz	690	França	Masculino	54	5	0.0	1	1	0
	Abrego	702	Alemanha	Feminino	23	7	98775.23	1	1	0
	Abron	708	França	Masculino	41	0	0.0	1	1	0
	Achebe	651	Alemanha	Feminino	41	4	133432.59	1	0	1
	Adams	791	França	Masculino	28	4	0.0	1	1	0
	Adamson	506	França	Feminino	57	6	0.0	2	0	1
	Afamefula	579	Espanha	Masculino	22	3	118680.57	1	1	1
	Afamefuna	629	Espanha	Masculino	36	1	161757.87	2	1	1
	Afanasyev	850	Alemanha	Feminino	47	4	99219.47	2	1	1
	Afanasyeva	726	França	Feminino	34	0	185734.75	1	1	1

only showing top 20 rows

```
1 # Aplicação
2 df = df.withColumn("genero", when(df["genero"] == "Female", "Feminino") \
3     .when(df["genero"] == "Male", "Masculino") \
4     .otherwise(df["genero"]))
```

```
1 # Verificação geral
2 df.show()
```

	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito	membro_ativo
	Abazu	584	Espanha	Masculino	32	9	85534.83	1	0	0
	Abbie	818	França	Feminino	31	1	186796.37	1	0	0
	Abbott	573	França	Feminino	33	0	90124.64	1	1	0
	Abdullah	802	França	Feminino	60	3	92887.06	1	1	0
	Abdulov	562	Alemanha	Feminino	31	9	117153.0	1	1	1
	Abel	544	França	Masculino	40	8	0.0	2	1	0
	Abernathy	560	França	Feminino	30	0	108883.29	1	1	0
	Abramov	755	França	Masculino	41	6	104817.41	1	1	0
	Abramova	661	Alemanha	Masculino	41	5	122552.48	2	0	1
	Abramovich	630	Espanha	Feminino	39	7	135483.17	1	1	0
	Abramowitz	690	França	Masculino	54	5	0.0	1	1	0
	Abrego	702	Alemanha	Feminino	23	7	98775.23	1	1	0
	Abron	708	França	Masculino	41	0	0.0	1	1	0
	Achebe	651	Alemanha	Feminino	41	4	133432.59	1	0	1
	Adams	791	França	Masculino	28	4	0.0	1	1	0

Adamson	506	França	Feminino	57	6	0.0	2	0	1
Afamefula	579	Espanha	Masculino	22	3	118680.57	1	1	1
Afamefuna	629	Espanha	Masculino	36	1	161757.87	2	1	1
Afanasyev	850	Alemanha	Feminino	47	4	99219.47	2	1	1
Afanasyeva	726	França	Feminino	34	0	185734.75	1	1	1

only showing top 20 rows

✓ Verificação de inconsistências

```
1 # Iterando sobre as colunas do DataFrame
2 for column, dtype in df.dtypes:
3     print(f"Valores únicos na coluna '{column}':")
4
5     # Verificando se a coluna contém valores inteiros
6     if dtype == 'int':
7         valores_unicos = sorted(df.select(column).distinct().rdd.map(lambda x: x[0]).collect())
8         print(valores_unicos)
9
10    # Verificando se a coluna contém valores de ponto flutuante
11    elif dtype == 'double':
12        valores_unicos = sorted(df.select(column).distinct().rdd.map(lambda x: x[0]).collect())
13        print(valores_unicos)
14
15    # Verificando se a coluna contém valores de texto (string)
16    elif dtype == 'string':
17        valores_unicos = sorted(df.select(column).distinct().rdd.map(lambda x: x[0]).collect())
18        print(valores_unicos)
19
20    print("-" * 50) # Linha de separação
```

```
↩️ Valores únicos na coluna 'sobrenome':
['Abazu', 'Abbie', 'Abbott', 'Abdullah', 'Abdulov', 'Abel', 'Abernathy', 'Abramov', 'Abramova', 'Abramovich', 'Abramowitz', 'Abrego']

Valores únicos na coluna 'pontuacao_credito':
[358, 359, 363, 365, 376, 386, 399, 401, 408, 411, 413, 414, 416, 417, 418, 420, 421, 422, 427, 428, 429, 430, 431, 432, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494, 495, 496, 497, 498, 499, 500]

Valores únicos na coluna 'localizacao':
['Alemanha', 'Espanha', 'França']

Valores únicos na coluna 'genero':
['Feminino', 'Masculino']

Valores únicos na coluna 'idade':
[18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50]

Valores únicos na coluna 'tempo_permanencia':
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Valores únicos na coluna 'saldo':
[0.0, 16893.59, 33563.95, 36566.96, 37266.67, 40105.51, 40224.7, 40685.92, 40915.55, 42157.08, 43134.65, 44020.89, 44582.07, 45144.4]

Valores únicos na coluna 'num_produtos':
[1, 2, 3, 4]

Valores únicos na coluna 'tem_cartao_credito':
[0, 1]

Valores únicos na coluna 'membro_ativo':
[0, 1]

Valores únicos na coluna 'salario_estimado':
[123.07, 142.81, 371.05, 417.41, 428.23, 555.28, 582.53, 582.59, 598.8, 600.36, 645.61, 820.46, 823.36, 823.96, 878.87, 933.38, 944]

Valores únicos na coluna 'encerrou_conta':
[0, 1]
```



```
1 # Verificação
2 df.show()
```

```
↩️ +-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| sobrenome|pontuacao_credito|localizacao| genero|idade|tempo_permanencia| saldo|num_produtos|tem_cartao_credito|membro_ativo|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Abazu|584|Espanha|Masculino|32|9|85534.83|1|0|0|
| Abbie|818|França|Feminino|31|1|186796.37|1|0|0|
| Abbott|573|França|Feminino|33|0|90124.64|1|1|0|
| Abdullah|802|França|Feminino|60|3|92887.06|1|1|0|
| Abdulov|562|Alemanha|Feminino|31|9|117153.0|1|1|1|
| Abel|544|França|Masculino|40|8|0.0|2|1|0|
| Abernathy|560|França|Feminino|30|0|108883.29|1|1|0|
| Abramov|755|França|Masculino|41|6|104817.41|1|1|0|
```

	Abramova	661	Alemanha	Masculino	41	5	122552.48	2	0	1
	Abramovich	630	Espanha	Feminino	39	7	135483.17	1	1	0
	Abramowitz	690	França	Masculino	54	5	0.0	1	1	0
	Abrego	702	Alemanha	Feminino	23	7	98775.23	1	1	0
	Abrón	708	França	Masculino	41	0	0.0	1	1	0
	Achebe	651	Alemanha	Feminino	41	4	133432.59	1	0	1
	Adams	791	França	Masculino	28	4	0.0	1	1	0
	Adamson	506	França	Feminino	57	6	0.0	2	0	1
	Afamefula	579	Espanha	Masculino	22	3	118680.57	1	1	1
	Afamefuna	629	Espanha	Masculino	36	1	161757.87	2	1	1
	Afanasyev	850	Alemanha	Feminino	47	4	99219.47	2	1	1
	Afanasyeva	726	França	Feminino	34	0	185734.75	1	1	1

only showing top 20 rows

1 # Verificação

```
2 df.withColumn("tem_cartao_credito", when(df["tem_cartao_credito"] == 0, "Não")
3     .when(df["tem_cartao_credito"] == 1, "Sim")
4     .otherwise(df["tem_cartao_credito"])).show()
```

	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito	membro_ativo
	Abazu	584	Espanha	Masculino	32	9	85534.83	1	Não	0
	Abbie	818	França	Feminino	31	1	186796.37	1	Não	0
	Abbott	573	França	Feminino	33	0	90124.64	1	Sim	0
	Abdullah	802	França	Feminino	60	3	92887.06	1	Sim	0
	Abdulov	562	Alemanha	Feminino	31	9	117153.0	1	Sim	1
	Abel	544	França	Masculino	40	8	0.0	2	Sim	0
	Abernathy	560	França	Feminino	30	0	108883.29	1	Sim	0
	Abramov	755	França	Masculino	41	6	104817.41	1	Sim	0
	Abramova	661	Alemanha	Masculino	41	5	122552.48	2	Não	1
	Abramovich	630	Espanha	Feminino	39	7	135483.17	1	Sim	0
	Abramowitz	690	França	Masculino	54	5	0.0	1	Sim	0
	Abrego	702	Alemanha	Feminino	23	7	98775.23	1	Sim	0
	Abrón	708	França	Masculino	41	0	0.0	1	Sim	0
	Achebe	651	Alemanha	Feminino	41	4	133432.59	1	Não	1
	Adams	791	França	Masculino	28	4	0.0	1	Sim	0
	Adamson	506	França	Feminino	57	6	0.0	2	Não	1
	Afamefula	579	Espanha	Masculino	22	3	118680.57	1	Sim	1
	Afamefuna	629	Espanha	Masculino	36	1	161757.87	2	Sim	1
	Afanasyev	850	Alemanha	Feminino	47	4	99219.47	2	Sim	1
	Afanasyeva	726	França	Feminino	34	0	185734.75	1	Sim	1

only showing top 20 rows

1 # Com função

```
2 def substituir_valores(df, coluna):
3     return df.withColumn(coluna, when(df[coluna] == 0, "Não")
4         .when(df[coluna] == 1, "Sim")
5         .otherwise(df[coluna]))
6
```

7 # Aplicação

```
8 df = substituir_valores(df, "tem_cartao_credito")
9 df = substituir_valores(df, "membro_ativo")
10 df = substituir_valores(df, "encerrou_conta")
```

1 df.show()

	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito	membro_ativo
	Abazu	584	Espanha	Masculino	32	9	85534.83	1	Não	Não
	Abbie	818	França	Feminino	31	1	186796.37	1	Não	Não
	Abbott	573	França	Feminino	33	0	90124.64	1	Sim	Não
	Abdullah	802	França	Feminino	60	3	92887.06	1	Sim	Não
	Abdulov	562	Alemanha	Feminino	31	9	117153.0	1	Sim	Sim
	Abel	544	França	Masculino	40	8	0.0	2	Sim	Não
	Abernathy	560	França	Feminino	30	0	108883.29	1	Sim	Não
	Abramov	755	França	Masculino	41	6	104817.41	1	Sim	Não
	Abramova	661	Alemanha	Masculino	41	5	122552.48	2	Não	Sim
	Abramovich	630	Espanha	Feminino	39	7	135483.17	1	Sim	Não
	Abramowitz	690	França	Masculino	54	5	0.0	1	Sim	Não
	Abrego	702	Alemanha	Feminino	23	7	98775.23	1	Sim	Não
	Abrón	708	França	Masculino	41	0	0.0	1	Sim	Não
	Achebe	651	Alemanha	Feminino	41	4	133432.59	1	Não	Sim
	Adams	791	França	Masculino	28	4	0.0	1	Sim	Não
	Adamson	506	França	Feminino	57	6	0.0	2	Não	Sim
	Afamefula	579	Espanha	Masculino	22	3	118680.57	1	Sim	Sim
	Afamefuna	629	Espanha	Masculino	36	1	161757.87	2	Sim	Sim
	Afanasyev	850	Alemanha	Feminino	47	4	99219.47	2	Sim	Sim
	Afanasyeva	726	França	Feminino	34	0	185734.75	1	Sim	Sim

only showing top 20 rows

Tipos de Dados

```
1 # Verificando o tipo de dados de cada coluna
2 df.dtypes
```

```
[('sobrenome', 'string'),
 ('pontuacao_credito', 'int'),
 ('localizacao', 'string'),
 ('genero', 'string'),
 ('idade', 'int'),
 ('tempo_permanencia', 'int'),
 ('saldo', 'double'),
 ('num_produtos', 'int'),
 ('tem_cartao_credito', 'string'),
 ('membro_ativo', 'string'),
 ('salario_estimado', 'double'),
 ('encerrou_conta', 'string')]
```

```
1 # Informações detalhadas do conjunto de dados
2 df.printSchema()
```

```
root
 |-- sobrenome: string (nullable = true)
 |-- pontuacao_credito: integer (nullable = true)
 |-- localizacao: string (nullable = true)
 |-- genero: string (nullable = true)
 |-- idade: integer (nullable = true)
 |-- tempo_permanencia: integer (nullable = true)
 |-- saldo: double (nullable = true)
 |-- num_produtos: integer (nullable = true)
 |-- tem_cartao_credito: string (nullable = true)
 |-- membro_ativo: string (nullable = true)
 |-- salario_estimado: double (nullable = true)
 |-- encerrou_conta: string (nullable = true)
```

No PySpark, você pode trabalhar com vários tipos de dados para representar diferentes tipos de informações. Alguns dos tipos de dados comuns disponíveis no PySpark incluem:

- String: Para armazenar dados de texto.
- Integer: Para armazenar números inteiros.
- Double: Para armazenar números decimais de precisão dupla (64 bits).
- Float: Para armazenar números decimais de precisão simples (32 bits).
- Boolean: Para armazenar valores booleanos (True ou False).
- DateType: Para armazenar datas

```
1 df.show()
```

```
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| sobrenome|pontuacao_credito|localizacao| genero|idade|tempo_permanencia| saldo|num_produtos|tem_cartao_credito|membro_ativo|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Abazu|584|Espanha|Masculino|32|9|85534.83|1|Não|Não|
| Abbie|818|França|Feminino|31|1|186796.37|1|Não|Não|
| Abbott|573|França|Feminino|33|0|90124.64|1|Sim|Não|
| Abdullah|802|França|Feminino|60|3|92887.06|1|Sim|Não|
| Abdulov|562|Alemanha|Feminino|31|9|117153.0|1|Sim|Sim|
| Abel|544|França|Masculino|40|8|0.0|2|Sim|Não|
| Abernathy|560|França|Feminino|30|0|108883.29|1|Sim|Não|
| Abramov|755|França|Masculino|41|6|104817.41|1|Sim|Não|
| Abramova|661|Alemanha|Masculino|41|5|122552.48|2|Não|Sim|
| Abramovich|630|Espanha|Feminino|39|7|135483.17|1|Sim|Não|
| Abramowitz|690|França|Masculino|54|5|0.0|1|Sim|Não|
| Abrego|702|Alemanha|Feminino|23|7|98775.23|1|Sim|Não|
| Abron|708|França|Masculino|41|0|0.0|1|Sim|Não|
| Achebe|651|Alemanha|Feminino|41|4|133432.59|1|Não|Sim|
| Adams|791|França|Masculino|28|4|0.0|1|Sim|Não|
| Adamson|506|França|Feminino|57|6|0.0|2|Não|Sim|
| Afamefula|579|Espanha|Masculino|22|3|118680.57|1|Sim|Sim|
| Afamefuna|629|Espanha|Masculino|36|1|161757.87|2|Sim|Sim|
| Afanasyev|850|Alemanha|Feminino|47|4|99219.47|2|Sim|Sim|
| Afanasyeva|726|França|Feminino|34|0|185734.75|1|Sim|Sim|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 20 rows
```

```
1 # Convertendo tipos de dados necessários
2 df = df.withColumn("saldo", df["saldo"].cast("float"))
3 df = df.withColumn("salario_estimado", df["salario_estimado"].cast("float"))
```

```
1 # Verificação
2 df.dtypes
```

```
[('sobrenome', 'string'),
 ('pontuacao_credito', 'int'),
 ('localizacao', 'string'),
 ('genero', 'string'),
 ('idade', 'int'),
 ('tempo_permanencia', 'int'),
 ('saldo', 'float'),
 ('num_produtos', 'int'),
 ('tem_cartao_credito', 'string'),
 ('membro_ativo', 'string'),
 ('salario_estimado', 'float'),
 ('encerrou_conta', 'string')]
```

```
1 df.printSchema()
```

```
root
|-- sobrenome: string (nullable = true)
|-- pontuacao_credito: integer (nullable = true)
|-- localizacao: string (nullable = true)
|-- genero: string (nullable = true)
|-- idade: integer (nullable = true)
|-- tempo_permanencia: integer (nullable = true)
|-- saldo: float (nullable = true)
|-- num_produtos: integer (nullable = true)
|-- tem_cartao_credito: string (nullable = true)
|-- membro_ativo: string (nullable = true)
|-- salario_estimado: float (nullable = true)
|-- encerrou_conta: string (nullable = true)
```

▼ Integridade dos dados

```
1 # Instalação do pacote pandera
2 !pip install pandera
```

```
Collecting pandera
  Downloading pandera-0.19.3-py3-none-any.whl (251 kB)
    251.9/251.9 kB 5.1 MB/s eta 0:00:00
Collecting multimethod<=1.10.0 (from pandera)
  Downloading multimethod-1.10-py3-none-any.whl (9.9 kB)
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.10/dist-packages (from pandera) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from pandera) (24.0)
Requirement already satisfied: pandas>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from pandera) (2.0.3)
Requirement already satisfied: pydantic in /usr/local/lib/python3.10/dist-packages (from pandera) (2.7.1)
Collecting typeguard (from pandera)
  Downloading typeguard-4.2.1-py3-none-any.whl (34 kB)
Collecting typing-inspect>=0.6.0 (from pandera)
  Downloading typing_inspect-0.9.0-py3-none-any.whl (8.8 kB)
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from pandera) (1.14.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2.0->pandera) (2.8)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2.0->pandera) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.2.0->pandera) (2024.1)
Collecting mypy_extensions>=0.3.0 (from typing-inspect>=0.6.0->pandera)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Requirement already satisfied: typing_extensions>=3.7.4 in /usr/local/lib/python3.10/dist-packages (from typing-inspect>=0.6.0->pandera) (4.12.0)
Requirement already satisfied: annotated_types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic->pandera) (0.6.0)
Requirement already satisfied: pydantic-core==2.18.2 in /usr/local/lib/python3.10/dist-packages (from pydantic->pandera) (2.18.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas>=1.2.0->pandera) (1.16.0)
Installing collected packages: typeguard, mypy_extensions, multimethod, typing_inspect, pandera
Successfully installed multimethod-1.10 mypy_extensions-1.0.0 pandera-0.19.3 typeguard-4.2.1 typing_inspect-0.9.0
```

```
1 # Importando Biblioteca
2 import pandas as pd
3 import pandera as pa
```

```
1 # Coletando os dados do PySpark DataFrame para Pandas
2 df_pandas = df.toPandas()
```

```
1 # Vendo o nosso conjunto de dados
2 df_pandas.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2932 entries, 0 to 2931
Data columns (total 12 columns):
```

```
# Column          Non-Null Count  Dtype
---  -
0  sobrenome        2932 non-null    object
1  pontuacao_credito 2932 non-null    int32
2  localizacao      2932 non-null    object
3  genero           2932 non-null    object
4  idade            2932 non-null    int32
5  tempo_permanencia 2932 non-null    int32
6  saldo            2932 non-null    float32
7  num_produtos     2932 non-null    int32
8  tem_cartao_credito 2932 non-null    object
9  membro_ativo     2932 non-null    object
10 salario_estimado 2932 non-null    float32
11 encerrou_conta   2932 non-null    object
dtypes: float32(2), int32(4), object(6)
memory usage: 206.3+ KB
```

```
1 # Definindo o esquema de validação com Pandera
2 schema = pa.DataFrameSchema({
3     'sobrenome': pa.Column(pa.String),
4     'pontuacao_credito': pa.Column(pa.Int32),
5     'localizacao': pa.Column(pa.String),
6     'genero': pa.Column(pa.String),
7     'idade': pa.Column(pa.Int32),
8     'tempo_permanencia': pa.Column(pa.Int32),
9     'saldo': pa.Column(pa.Float32),
10    'num_produtos': pa.Column(pa.Int32),
11    'tem_cartao_credito': pa.Column(pa.String),
12    'membro_ativo': pa.Column(pa.String),
13    'salario_estimado': pa.Column(pa.Float32),
14    'encerrou_conta': pa.Column(pa.String),
15 })
```

```
1 # Validar o DataFrame
2 schema.validate(df_pandas)
```

	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito
0	Abazu	584	Espanha	Masculino	32	9	85534.828125	1	Não
1	Abbie	818	França	Feminino	31	1	186796.375000	1	Não
2	Abbott	573	França	Feminino	33	0	90124.640625	1	Sim
3	Abdullah	802	França	Feminino	60	3	92887.062500	1	Sim
4	Abdulov	562	Alemanha	Feminino	31	9	117153.000000	1	Sim
...
2927	Zubarev	749	França	Feminino	26	6	0.000000	2	Não
2928	Zubareva	592	França	Masculino	42	1	147249.296875	2	Sim
2929	Zuev	730	França	Masculino	39	1	116537.601562	1	Não
2930	Zuyev	621	Espanha	Masculino	53	9	170491.843750	1	Sim
2931	Zuyeva	606	Alemanha	Feminino	32	1	106301.851562	2	Não

2932 rows × 10 columns

```
1 # Validar o DataFrame do Pandas
2 validated_df = schema.validate(df_pandas)
```

```
1 # Criar DataFrame do Pandas validado como PySpark DataFrame
2 spark_df = spark.createDataFrame(validated_df)
```

```
1 # Verificando a conversão
2 spark_df.show()
```

	sobrenome	pontuacao_credito	localizacao	genero	idade	tempo_permanencia	saldo	num_produtos	tem_cartao_credito	membro_ativo
0	Abazu	584	Espanha	Masculino	32	9	85534.828125	1	Não	
1	Abbie	818	França	Feminino	31	1	186796.375	1	Não	
2	Abbott	573	França	Feminino	33	0	90124.640625	1	Sim	
3	Abdullah	802	França	Feminino	60	3	92887.0625	1	Sim	
4	Abdulov	562	Alemanha	Feminino	31	9	117153.0	1	Sim	
5	Abel	544	França	Masculino	40	8	0.0	2	Sim	
6	Abernathy	560	França	Feminino	30	0	108883.2890625	1	Sim	
7	Abramov	755	França	Masculino	41	6	104817.40625	1	Sim	
8	Abramova	661	Alemanha	Masculino	41	5	122552.4765625	2	Não	
9	Abramovich	630	Espanha	Feminino	39	7	135483.171875	1	Sim	
10	Abramowitz	690	França	Masculino	54	5	0.0	1	Sim	

Abrego	702	Alemanha	Feminino	23	7	98775.2265625	1	Sim
Abron	708	França	Masculino	41	0	0.0	1	Sim
Achebe	651	Alemanha	Feminino	41	4	133432.59375	1	Não
Adams	791	França	Masculino	28	4	0.0	1	Sim
Adamson	506	França	Feminino	57	6	0.0	2	Não
Afamefula	579	Espanha	Masculino	22	3	118680.5703125	1	Sim
Afamefuna	629	Espanha	Masculino	36	1	161757.875	2	Sim
Afanasyev	850	Alemanha	Feminino	47	4	99219.46875	2	Sim
Afanasyeva	726	França	Feminino	34	0	185734.75	1	Sim

only showing top 20 rows

Carregamento

```
1 spark_df.write.format('csv').save('/content/drive/MyDrive/CHAVES/churn_tratado')
```

Visualização de Dados

```
1 # Visualizando estatísticas descritivas para dados numéricos
2 spark_df.summary().show()
```

```

+-----+-----+-----+-----+-----+-----+-----+-----+
|summary|sobrenome|pontuacao_credito|localizacao|genero|idade|tempo_permanencia|saldo|num_produto|
+-----+-----+-----+-----+-----+-----+-----+-----+
|count|2932|2932|2932|2932|2932|2932|2932|2932|
|mean|NULL|652.1289222373806|NULL|NULL|38.98806275579809|4.9566848567530695|75829.90900249404|1.53683492496589|
|stddev|NULL|96.1372151188427|NULL|NULL|10.608420308348057|2.8903242730792122|62511.42718476476|0.589635786739854|
|min|Abazu|358|Alemanha|Feminino|18|0|0.0|
|25%|NULL|585|NULL|NULL|32|3|0.0|
|50%|NULL|655|NULL|NULL|37|5|96674.546875|
|75%|NULL|720|NULL|NULL|44|7|127283.78125|
|max|Zuyeva|850|França|Masculino|88|10|238387.5625|
+-----+-----+-----+-----+-----+-----+-----+-----+

```

```
1 # Instalando Plotly
2 !pip install Plotly
```

```

Requirement already satisfied: Plotly in /usr/local/lib/python3.10/dist-packages (5.15.0)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from Plotly) (8.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from Plotly) (24.0)

```

```

1 # Abertura de pacotes
2 import plotly.graph_objs as go
3 import plotly.io as pio
4 from plotly.subplots import make_subplots

```

```

1 # Coletando os dados do PySpark DataFrame para Pandas DataFrame para Plotly
2 df_pandas = spark_df.toPandas()

```

```

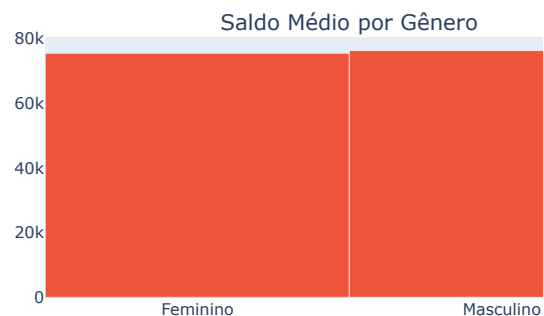
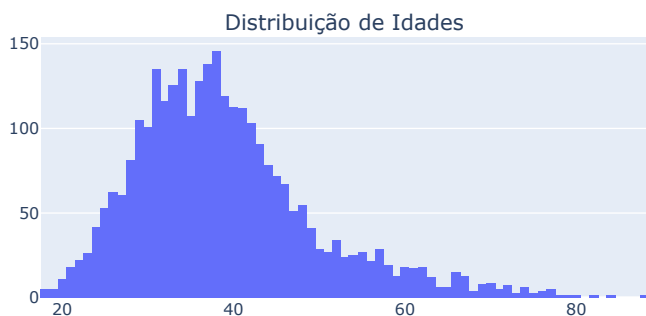
1 # Criando o layout do dashboard com 4 gráficos
2 fig = make_subplots(rows=2, cols=2, subplot_titles=("Distribuição de Idades", "Saldo Médio por Gênero",
3                                                     "Proporção de Clientes Ativos", "Distribuição de Pontuação de Crédito"),
4                     specs=[[{"type": "pie"}, {"type": "pie"}], [{"type": "pie"}, {"type": "pie"}]))
5
6 # Gráfico 1: Distribuição de Idades
7 fig.add_trace(
8     go.Histogram(x=df_pandas['idade'], name='Idade'),
9     row=1, col=1
10 )
11
12 # Gráfico 2: Saldo Médio por Gênero
13 saldo_por_genero = df_pandas.groupby('genero')['saldo'].mean().reset_index()
14 fig.add_trace(
15     go.Bar(x=saldo_por_genero['genero'], y=saldo_por_genero['saldo'], name='Saldo Médio'),
16     row=1, col=2
17 )
18
19 # Gráfico 3: Proporção de Clientes Ativos
20 ativos_counts = df_pandas['membro_ativo'].value_counts()
21 fig.add_trace(
22     go.Pie(labels=ativos_counts.index, values=ativos_counts.values, name='Clientes Ativos'),
23     row=2, col=1
24 )
25
26 # Gráfico 4: Distribuição de Pontuação de Crédito

```

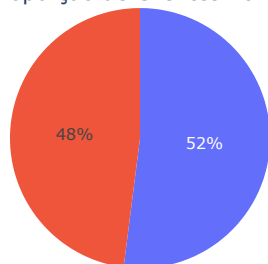
```
27 fig.add_trace(  
28     go.Box(y=df_pandas['pontuacao_credito'], name='Pontuação de Crédito'),  
29     row=2, col=2  
30 )  
31  
32 # Atualizando layout e configurações  
33 fig.update_layout(  
34     title='Dashboard Interativo com Plotly',  
35     showlegend=True,  
36     height=700,  
37     width=1300,  
38 )  
39  
40 # Exibindo o dashboard interativo  
41 pio.show(fig)
```



Dashboard Interativo com Plotly



Proporção de Clientes Ativos



Distribuição de Pontuação de Crédito

