

ST IT Cloud - Data and Analytics Test LV.4

Esse teste deve avaliar alguns conceitos de big data e a qualidade técnica na manipulação de dados, otimização de performance, trabalho com arquivos grandes e tratamento de qualidade.

Passo a passo

- *Parte teórica:* responda as questões abaixo preenchendo as células em branco.
- *Parte prática:* disponibilizamos aqui 2 cases para, leia os enunciados dos problemas, desenvolver os programas, utilizando a **stack definida durante o processo seletivo**, para entregar os dados de acordo com os requisitos descritos abaixo.

Faz parte dos critérios de avaliação a pontualidade da entrega. Implemente até onde for possível dentro do prazo acordado.

Os dados de pessoas foram gerados de forma aleatória, utilizando a biblioteca FakerJS, FakerJS-BR e Faker

LEMBRE-SE: A entrega deve conter TODOS os passos para o avaliador executar o programa (keep it simple).

Questão 1 - Descreva de forma detalhada quais são as etapas na construção de um pipeline de dados, sem considerar ferramentas específicas, imagine que é seu primeiro contato com o cliente e você precisa entender a demanda dele e explicar quais são os passos que você terá que implementar para entregar a demanda.

1- Avaliar o Problema de Negocio 2- Avaliar as possíveis soluções e abordagens para resolver o problema 3- Escolher um caminho a seguir entre as abordagens 5- Desenhar a Solução 4- Definir quais Ferramentas serão utilizadas 5- Desenvolver um protótipo inicial 6 - Fazer as Validações internas e testes com o Usuário 7- Fazer Deploy em Produção, é um processo que tem um inicio , meio e fim , ou do ponto de vista dos dados tem uma entrada , processamento e saída ou seja é a montagem de um fluxo operacional a ser seguido que tem um inicio e um fim em termos operacionais , e que pode ser aplicado a qualquer coisa , desde de um gerenciamento de produção de uma fábrica que fabrica determinado produto , ou a uma Fábrica de Software que desenvolve aplicativos ou mesmo a uma arquitetura de fluxo operacional de dados, ou seja se trata de gerenciamento organizado de fluxo operacional de trabalho.

Questão 2 - Defina com suas palavras um processamento em streaming e processamento em batch. Qual sua experiência com cada uma delas.

Processamento em Streaming é o processamento em tempo real ou seja no momento da produção da informação (como por exemplo a coleta de uma informação de venda online que acabou de acontecer), processamento em batch é um processamento através de lotes ou seja é enviado lotes de dados com tamanhos pre-definidos para processamento em intervalos de tempo pre definidos e pode ser muito tempo depois do fato ter ocorrido ou pode ser próximo ao tempo real.

Questão 3 - Quais são as camadas de um Data Lake?

Ingestão , Armazenamento em Cache e Processamento

Questão 4 - Quais as diferenças de um Data Lake e um DW?

Basicamente o Data Lake suporta dados não estruturados e o Data Warehouse não suporta , mas ambos suportam dados estruturados e agregados por segmentos e tambem um Data Lake por ser projetado para Big Data suporta armazenamento e processamento distribuido com volumes de dados gigantescos e alta integridade das informações ou seja atende ao requisito de grandes volumes de dados com alta integridade e tambem ao requisito de processamento de dados em alta velocidade, pois trabalha com armazenamento e processamento distribuido , operando através de um cluster de computadores como se fosse uma unica maquina , coisa que um DW não comporta pois não foi projetado para isso.

Questão 5 - O que é arquitetura Lambda e Kappa? Descreva com suas palavras.

Arquitetura Lambda Foi desenvolvida para contornar uma limitação do MapReduce que ao processar grandes volumes de dados demorava muito para dar um retorno do resultado. Isso foi resolvido pelo menos parcialmente ,através da arquiteura Kappa que dá dois caminhos para o resultado dos dados , o caminho quente que é mais rapido mas menos preciso e em uma janela pequena de tempo e o caminho frio que é mais lento e mais demorada e em uma janela de tempo maior mas é mais preciso em relação as informações, este é processado em Batch , tudo é unificado na camada de serviço que possui as informações completas unificadas entre os processamento em streeaming de dados e o processamento em batch , os dados dos eventos são imutaveis.Dessa forma acaba atendendo a necessidade de ver mais rapidamente os dados. A Arquitetura Kappa foi desenvolvida para suprir uma deficiencia da arquitetura Lambda pois havia a necessidade de dois codigos diferentes para tratar a camada em batch e a camada em streming , ja a Arquitetura Kappa pensa em termos de unificação dos dados na origem e para isso usa o Log para fazer essa unificação onde não importa de onde venha o dado , se de streaming de dados , de um csv , xml de um banco de dados , todas essas fontes de dados são tratados como eventos no arquivo de log e é um log imutavel todo o dado que entra entra como um simbolo de + e os updates e reduções de dados entram com um simbolo de - , e assim o seu processamento se torna muito mais rapido, e dessa forma tudo é processado em tempo real e armazenado em um repositório de Longo Termo e possui uma camada de serviço que serve para oferecer os dados para as ferramentas de visualização ou de extração de informações que foram processadas.

Questão 6 - O que é Data Quality para você e como você implementa isso nos seus processos?

É um processo de que visa validar os dados afim de manter a qualidade dos mesmos . Existe varias formas de fazer isso , como processos que validam os nomes das colunas, os tipos de dados e que compararam dados de periodos diferentes para detectar anomalizas ou falhas na produção da informação na origem dos dados

Questão 7 - Em uma escala de 0 a 10, qual seria seu nível de experiência com PySpark?

6, Utilizei no curso de Python com Spark que fiz.

Questão 8 - Em uma escala de 0 a 10, qual seria seu nível de experiência com SQL?

8 , Já trabalho com a linguagem SQL e banco SQL a muito tempo

Questão 9 - Descreva suas experiências com banco de dados SQL e NoSQL.

Ja trabalhei com o Banco SQL e atualmente Trabalho com Impala banco NoSQL usando a ferramenta Hue para rodar as consultas,tendo tambem o Hive como opção adicional.

Questão 10 - Tem experiência com versionamento de código? Com quais ferramentas já trabalhou? Descreva.

Só conheço GitHub , utilizamos na empresa que trabalho , mas depende muito do projeto , tem projeto que usamos mais , outros usamos menos e outros não usamos.

Questão 11 - Tem experiência em desenvolvimento em cloud? Se sim, especifique a(s) plataforma(s) que já trabalhou e suas principais implementações e conhecimentos em cada serviço.

Não tenho , a unica ferramenta que conheço que trabalha com este Viés é o Azure Machine Learning que aprendi ela no curso que fiz de Cientista de Dados na DSA.

Questão 12 - Tem experiência com metodologia ágil? Qual?

Sim , A maioria dos projetos que trabalhei e que estou trabalhando na Cargill utiliza a metodologia Scrum , então temos o Scrum Master que conduz a reunião em conjunto com os líderes do projeto normalmente alguém de negócio e alguém da área de TI. São realizadas reuniões diárias denominadas Dailys e as Plenys a cada 15 dias e mais uma reunião Geral no Final da Sprint e também pode ser marcada uma reunião após o término da sprint de retrospectiva para análise de pontos de dificuldades e o que pode ser melhorado para a próxima Sprint, cada Sprint são atividades de trabalhos vinculados a um prazo , são determinadas metas de trabalhos e delegadas as atividades aos responsáveis que terão como meta concluir as atividades até o final da Sprint, o prazo da Sprint pode variar em relação ao tempo , normalmente 15 ou 30 dias, é o que eu conheço. Mas sei que a Metodologia Scrum é bem maleável então esta estrutura pode variar um pouco de empresa para empresa.

TESTE PRÁTICO

Problema 1: Você está recebendo o arquivo 'dados_cadastrais_fake.csv' que contém dados cadastrais de clientes, mas para que análises ou relatórios sejam feitos é necessário limpar e normalizar os dados. Além disso, existe uma coluna com o número de cpf e outra com cnpj, você

precisará padronizar deixando apenas dígitos em formato string (sem caracteres especiais), implementar uma forma de verificar se tais documentos são válidos sendo que a informação deve se adicionada ao dataframe em outras duas novas colunas.

Após a normalização, gere reports que respondam as seguintes perguntas:

- Quantos clientes temos nessa base?
- Qual a média de idade dos clientes?
- Quantos clientes nessa base pertencem a cada estado?
- Quantos CPFs válidos e inválidos foram encontrados?
- Quantos CNPJs válidos e inválidos foram encontrados?

Ao final gere um arquivo no formato csv e um outro arquivo no formato parquet chamado (problema1_normalizado), eles serão destinados para pessoas distintas.

EXTRA: executar as mesmas validações no *1E8.csv.gz

```
In [1]: # Instalando e Importando Bibliotecas e Modulos
# !pip install --user pandas -U
import pandas as pd
import numpy as np

import sys
import os.path

#Importa Modulo Adicional
import validacpfnpj

#Desativa os avisos
import warnings
warnings.filterwarnings("ignore")
```

```
In [2]: # Cria função secundaria

def roda_validacao_cpf_cnpj(a):

    cpf_cnpj = validacpfnpj.ValidaCpfCnpj(str(a))

    return cpf_cnpj.valida()
```

```
In [3]: #Testa Função com CPF - True
roda_validacao_cpf_cnpj(97566536800)
```

Out[3]: True

```
In [4]: #Testa Função com CPF - False
roda_validacao_cpf_cnpj(97566536801)
```

Out[4]: False

In [5]: #Testa Função com CNPJ - True
roda_validacao_cpf_cnpj("06589184909526")

Out[5]: True

In [6]: #Testa Função com CNPJ - false
roda_validacao_cpf_cnpj("06589184909521")

Out[6]: False

In [7]: # Usando o método read_csv
df =pd.read_table('dados_cadastrais_fake.csv', sep = ';',encoding="utf8")
df

Out[7]:

	nomes	idade	cidade	estado	cpf	cnpj
0	Dennis Daniels	31	ACRELÂNDIA	AC	97566536800	06589184909526
1	Leah Becker	42	ÁGUA BRANCA	AL	425.263.807-07	25.673.336/2350-20
2	Sally Ford	18	ALVARÃES	AM	34647754103	26543101702989
3	Colleen Duncan	21	SERRA DO NAVIO	AP	252.531.560-03	19.062.080/5100-98
4	Jeff Stephenson	73	ABAÍRA	BA	49668886542	97794530015384
...
9995	Rebekah Mitchell PhD	55	ABAÍARA	CE	744.822.622-34	16.740.076/9329-75
9996	Lisa Parrish Jr.	73	Brasília	DF	10683395190	32246978843482
9997	Michael Young MD	87	AFONSO CLÁUDIO	ES	538.223.638-04	86.601.303/7580-88
9998	Kevin Watson DDS	82	ABADIA DE GOIÁS	GO	11632512408	08651414023648
9999	Mr. Joseph Wilson MD	50	AÇAILÂNDIA	MA	192.134.492-08	08.908.871/5161-91

10000 rows × 6 columns

In [8]: # Quantos clientes temos nessa base? R: 33 clientes distintos
max(df.nomes.value_counts())

Out[8]: 33

In [9]: # Qual a média de idade dos clientes? R: 53 anos
df.idade.mean()

Out[9]: 53.7831

Quantos clientes nessa base pertencem a cada estado?

R: Segue a lista abaixo

In [10]:

```
df['estado'].replace({' ':''},inplace=True, regex=True)
df.replace({'MINASGERAI':'MG','MINASGERAIs':'MG','distritofederal':'DF','riodejaro
    'sãopaulo':'SP'},inplace=True)
df[['estado','nomes']].groupby(['estado']).nunique()
```

Out[10]:

estado	nomes
AC	365
AL	362
AM	360
AP	365
BA	364
CE	364
DF	361
ES	359
GO	365
MA	364
MG	360
MS	361
MT	365
PA	361
PB	366
PE	362
PI	363
PR	362
RJ	362
RN	359
RO	363
RR	362
RS	360
SC	361
SE	363
SP	363
TO	363

In []:

Quantos CPFs válidos e inválidos foram encontrados?

R: Foram encontrado 10 mil cpfs validos ou seja todos os cpfs da base csv fornecida são validos.Não foram encontrados nenhum cpf invalido.

--> Segue detalhes do código abaixo

```
In [11]: #df[['cpf_Limpo']] = df[['cpf']].replace({'.':'', '-':'', '/':''}, inplace=True, regex=True)

df['cpf'] = df['cpf'].str.replace('.', '')
df['cpf'] = df['cpf'].str.replace('-', '')

df['valida_cpf'] = df['cpf'].apply(roda_validacao_cpf_cnpj)

df
```

Out[11]:

	nomes	idade	cidade	estado	cpf	cnpj	valida_cpf
0	Dennis Daniels	31	ACRELÂNDIA	AC	97566536800	06589184909526	True
1	Leah Becker	42	ÁGUA BRANCA	AL	42526380707	25.673.336/2350-20	True
2	Sally Ford	18	ALVARÃES	AM	34647754103	26543101702989	True
3	Colleen Duncan	21	SERRA DO NAVIO	AP	25253156003	19.062.080/5100-98	True
4	Jeff Stephenson	73	ABAÍRA	BA	49668886542	97794530015384	True
...
9995	Rebekah Mitchell PhD	55	ABAIARA	CE	74482262234	16.740.076/9329-75	True
9996	Lisa Parrish Jr.	73	Brasília	DF	10683395190	32246978843482	True
9997	Michael Young MD	87	AFONSO CLÁUDIO	ES	53822363804	86.601.303/7580-88	True
9998	Kevin Watson DDS	82	ABADIA DE GOIÁS	GO	11632512408	08651414023648	True
9999	Mr. Joseph Wilson MD	50	AÇAILÂNDIA	MA	19213449208	08.908.871/5161-91	True

10000 rows × 7 columns

```
In [12]: # Todos os 10 mil registros são True ou seja todos os Cnpjs são validos  
df[['cpf','valida_cpf']].groupby(['valida_cpf']).nunique()
```

Out[12]:

cpf	valida_cpf
True	10000

```
In [13]: # Os 10 mil registros possuem no campo CPF 11 caracteres validos , ou seja não existem numeros de caracteres invalidos
```

```
dfx=df[df['cpf'].str.len() == 11]  
max(dfx.valida_cpf.value_counts())
```

Out[13]: 10000

In []:

Quantos CNPJs válidos e inválidos foram encontrados?

R: Todos os 10 mil registros de CNPJ existentes no arquivo CSV analisados, são validos, não forma encontrados nenhum cnpj invalido.

```
In [14]: #df[['cpf_Limpo']] = df[['cpf']].replace({'.':'', '-':'', '/':''}, inplace=True, regex=True)

df['cnpj'] = df['cnpj'].str.replace('.', '')
df['cnpj'] = df['cnpj'].str.replace('/', '')
df['cnpj'] = df['cnpj'].str.replace('-', '')

#df[['valida_cpf']] = df[['cpf']].apply(roda_validacao_cpf_cnpj)
df['valida_cnpj'] = df['cnpj'].apply(roda_validacao_cpf_cnpj)

df
```

	nomes	idade	cidade	estado	cpf	cnpj	valida_cpf	valida_cnpj
0	Dennis Daniels	31	ACRELÂNDIA	AC	97566536800	06589184909526	True	True
1	Leah Becker	42	ÁGUA BRANCA	AL	42526380707	25673336235020	True	True
2	Sally Ford	18	ALVARÃES	AM	34647754103	26543101702989	True	True
3	Colleen Duncan	21	SERRA DO NAVIO	AP	25253156003	19062080510098	True	True
4	Jeff Stephenson	73	ABAÍRA	BA	49668886542	97794530015384	True	True
...
9995	Rebekah Mitchell PhD	55	ABAIARA	CE	74482262234	16740076932975	True	True
9996	Lisa Parrish Jr	73	Brasília	DF	10683395190	32246978843482	True	True

```
In [15]: df[['cnpj', 'valida_cnpj']].groupby(['valida_cnpj']).nunique()
```

Out[15]:

cnpj	valida_cnpj
True	10000

```
In [16]: # Os 10 mil registros possuem no campo CNPJ 14 caracteres validos , ou seja não existem numeros de caracteres invalidos
# Numeros de caracteres validos
dfx=df[df['cnpj'].str.len() == 14]
max(dfx.valida_cnpj.value_counts())
```

Out[16]: 10000

Ao final gera um arquivo no formato csv e um outro arquivo no formato parquet chamado (problema1_normalizado), eles serão destinados para pessoas distintas

--> Gerado Segue o código abaixo

```
In [17]: # Exportando Para CSV
df.to_csv('problema1_normalizado.csv', index = False, sep=';', encoding='utf-8')
```

```
In [18]: # Exportando Para Parquet
df.to_parquet("./problema1_normalizado.pq")
```

```
In [ ]:
```

Problema 2: Você deverá implementar um programa, para ler, tratar e particionar os dados.

O arquivo fonte está disponível em https://st-it-cloud-public.s3.amazonaws.com/people-v2_1E6.csv.gz

Data Quality

- Higienizar e homogenizar o formato da coluna document
- Detectar através da coluna document se o registro é de uma Pessoa Física ou Pessoa Jurídica, adicionando uma coluna com essa informação
- Higienizar e homogenizar o formato da coluna birthDate
- Existem duas colunas nesse dataset que em alguns registros estão trocadas. Quais são essas colunas?
- Corrigir os dados com as colunas trocadas
- Além desses pontos, existem outras tratamentos para homogenizar esse dataset. Aplique todos que conseguir.

Agregação dos dados

- Quais são as 5 PF que mais gastaram (totalSpent)?
- Qual é o valor de gasto médio por estado (state)?
- Qual é o valor de gasto médio por jobArea ?
- Qual é a PF que gastou menos (totalSpent)?
- Quantos nomes e documentos repetidos existem nesse dataset?
- Quantas linhas existem nesse dataset?

Particionamento de dados tratados com as regras descritas em DATA QUALITY

- Particionar em arquivos PARQUET por estado (state)
- Particionar em arquivos CSV por ano/mes/dia de nascimento (birthDate)

Data Quality

```
In [882]: # Carregando DataSet
# Usando o método read_csv
df_pl = pd.read_table('people-v2_1E6.csv', sep = ';', encoding="utf8")
df_pl
```

Out[882]:

	document	name	job	jobArea	jobType	phoneNumber	birth
0	76684148787	Charlleny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	20-05-
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	10-Jun-
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	05/16/
3	02.328.238/0877-86	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	19810
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	04/07/
...
999995	20297054201	Aurilo Franco Neto	Agente Funcionalidade Investidor	Otimização	NaN	(54) 8742-5107	16-04-
999996	76245248035484	Iata Martins	Técnico A infraestrutura Central	Rede	NaN	(72) 8066-0110	May.22.
999997	15787105200206	Erica Melo	Associado Operações Cliente	Garantia	NaN	(00) 5316-2453	23-01-
999998	12922681661	Jurema Carvalho Filho Neto	Agente Operações Legado	Programa	Engenheiro	+55 (44) 8691-2473	Apr.11.
999999	97145411318	Sr. Jair Martins	Produtor Comunicações Frente	Interações	Analista	(28) 0064-5483	11/27/

1000000 rows × 10 columns

1- Higienizar e homogenizar o formato da coluna document

R: Segue a resposta abaixo:

```
In [367]: df_pl['document'].replace({'-':'','/':'',' ':''},regex=True, inplace=True)

df_pl['document'] = df_pl['document'].str.replace('.','')

df_pl[['document']]
```

Out[367]:

	document
0	76684148787
1	85704855733
2	15664328373377
3	02328238087786
4	30073687408740
..	..
999995	20297054201
999996	76245248035484
999997	15787105200206
999998	12922681661
999999	97145411318

1000000 rows × 1 columns

2- Detectar através da coluna document se o registro é de uma Pessoa Física ou Pessoa Jurídica, adicionando uma coluna com essa informação

R: Segue a resposta abaixo:

In [368]: # Resposta abaixo:

```
df_pl.loc[df_pl['document'].str.len() == 14,'tipo_doc']='PJ'
df_pl.loc[df_pl['document'].str.len() == 11,'tipo_doc']='PF'
df_pl[['document','tipo_doc']].head()
```

Out[368]:

	document	tipo_doc
0	76684148787	PF
1	85704855733	PF
2	15664328373377	PJ
3	02328238087786	PJ
4	30073687408740	PJ

```
In [369]: # Verificando nulos em tipo_doc
df_pl[['tipo_doc']].isnull().sum()
```

```
Out[369]: tipo_doc    0
dtype: int64
```

In []:

3 - Higienizar e homogenizar o formato da coluna birthDate

R: Segue a resposta abaixo:

```
In [496]: # Resposta abaixo
df_pl['birthDate']
```

```
Out[496]: 0      1972-05-20
1      1982-10-06
2      1968-05-16
3      1981-04-17
4      1980-07-04
...
999995  1976-04-16
999996  1981-05-22
999997  1988-01-23
999998  1968-11-04
999999  1988-11-27
Name: birthDate, Length: 1000000, dtype: datetime64[ns]
```

```
In [497]: # Verificando o tipo de dados da coluna birthDate
df_pl.dtypes
```

```
Out[497]: document          object
name            object
job             object
jobArea         object
jobType         object
phoneNumber     object
birthDate       datetime64[ns]
city            object
state           object
totalSpent      float64
tipo_doc        object
dtype: object
```

```
In [498]: # Convertendo a coluna birthDate para o formato datetime , para poder aplicar a f
df_pl['birthDate'] = df_pl['birthDate'].astype('datetime64[ns]')

# Aplicando a formatação na coluna
df_pl['birthDate'] = df_pl['birthDate'].dt.strftime('%d/%m/%Y')

# Após a aplicação da formatação ele volta para texto , dessa forma é preciso cor
df_pl['birthDate'] = df_pl['birthDate'].astype('datetime64[ns]')

# Visualizando o Resultado
df_pl['birthDate']
```

```
Out[498]: 0      1972-05-20
1      1982-06-10
2      1968-05-16
3      1981-04-17
4      1980-04-07
...
999995  1976-04-16
999996  1981-05-22
999997  1988-01-23
999998  1968-04-11
999999  1988-11-27
Name: birthDate, Length: 1000000, dtype: datetime64[ns]
```

```
In [499]: #Confirmando o tipo de dados
df_pl.dtypes
```

```
Out[499]: document          object
name            object
job             object
jobArea         object
jobType         object
phoneNumber     object
birthDate       datetime64[ns]
city            object
state           object
totalSpent      float64
tipo_doc        object
dtype: object
```

```
In [500]: df_pl['birthDate'] = df_pl['birthDate'].astype('datetime64[ns]')
df_pl['birthDate']
```

```
Out[500]: 0      1972-05-20
1      1982-06-10
2      1968-05-16
3      1981-04-17
4      1980-04-07
...
999995  1976-04-16
999996  1981-05-22
999997  1988-01-23
999998  1968-04-11
999999  1988-11-27
Name: birthDate, Length: 1000000, dtype: datetime64[ns]
```

```
In [ ]:
```

4 - Existem duas colunas nesse dataset que em alguns registros estão trocadas. Quais são essas colunas?

R: Identificado valores trocados entre as colunas jobArea com phoneNumber

In [540]: # Visualizando todas as colunas
df_pl

Out[540]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate
0	76684148787	Charlleny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	1972-01-02
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	1982-01-01
2	15664328373377	Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-01-01
3	02328238087786	Celina Carvalho Jr	NaN	Qualidade	Ligaçao	+55 (58) 4136-5577	1981-01-01
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	1980-01-01
...
999995	20297054201	Aurilo Franco Neto	Agente Funcionalidade Investidor	Otimização	NaN	(54) 8742-5107	1976-01-01
999996	76245248035484	Iata Martins	Técnico A Infraestrutura Central	Rede	NaN	(72) 8066-0110	1981-01-02
999997	15787105200206	Erica Melo	Associado Operações Cliente	Garantia	NaN	(00) 5316-2453	1988-01-02
999998	12922681661	Jurema Carvalho Filho Neto	Agente Operações Legado	Programa	Engenheiro	+55 (44) 8691-2473	1968-01-01
999999	97145411318	Jair Martins	Produtor Comunicações Frente	Interações	Analista	(28) 0064-5483	1988-01-02

1000000 rows × 12 columns



In [541]: # Visualizando valores distintos de jobType
df_pl[['jobType']].groupby(['jobType']).nunique().head(20)

Out[541]:

jobType
Administrador
Agente
Analista
Arquiteto
Assistente
Associado
Consultor
Coordenador
Desenvolvedor
Designer
Diretor
Engenheiro
Especialista
Estrategista
Executivo
Facilitador
Gerente
Ligação
Oficial
Orquestrador

In [958]: # Visualizando valores distintos de job
df_pl[['job']].groupby(['job']).nunique().head(10)

Out[958]:

job
Administrador A infraestrutura Central
Administrador A infraestrutura Chefe
Administrador A infraestrutura Cliente
Administrador A infraestrutura Corporativo
Administrador A infraestrutura Dinâmico
Administrador A infraestrutura Direto
Administrador A infraestrutura Diretor
Administrador A infraestrutura Distrito
Administrador A infraestrutura Frente
Administrador A infraestrutura Global

In [957]: # Visualizando valores distintos de jobArea
Identificado valores trocados de jobArea com phoneNumber
df_pl[['jobArea']].groupby(['jobArea']).nunique().head(10)

Out[957]:

jobArea
A infraestrutura
Aplicações
Branding
Comunicações
Configuração
Contas
Criativo
Dados
Diretivas
Divisão

In [548]: df_pl_pn = df_pl[['phoneNumber']].groupby(['phoneNumber']).nunique()

In [607]: `df_pl_pn.sort_values(by=['phoneNumber'], ascending=False)`

phoneNumber

Usabilidade

Táticas

Soluções

Segurança

Resposta

...

(00) 0011-7959

(00) 0010-3495

(00) 0008-5759

(00) 0001-1538

(00) 0001-0003

In []:

5 - Corrigir os dados com as colunas trocadas

In [651]: `# Visualizando valores distintos de jobArea
Identificado valores trocados de jobArea com phoneNumber
df_pl[['jobArea']].groupby(['jobArea']).nunique().head(5)`

Out[651]:

jobArea

(00) 00435-1738

(00) 01430-7785

(00) 01525-7613

(00) 01837-4277

(00) 01870-0192

In [634]:

```
In [667]: df_pl_pn2 = df_pl[['jobArea', 'jobArea_nb']].groupby(['jobArea']).nunique()  
df_pl_pn2.sort_values(by=['jobArea', 'jobArea_nb'], ascending=False)
```

Out[667]:

jobArea	jobArea_nb
Usabilidade	0
Táticas	0
Soluções	0
Segurança	0
Resposta	0
...	...
(00) 01870-0192	1
(00) 01837-4277	1
(00) 01525-7613	1
(00) 01430-7785	1
(00) 00435-1738	1

50028 rows × 1 columns

```
In [701]: # Avaliando o comando  
df_pl_tmp = df_pl[df_pl['jobArea'].str.contains('\d+', regex=True, na=True)].head()
```

In [702]: #Avalindo o resultado
df_pl_tmp

Out[702]:

		document	name	job	jobArea	jobType	phoneNumber	birthDate	city
22	35171180369	Vanoil Xavier Filho Filho		Analista Comunicações Corporativo	+55 (47) 2953-9641	Analista	Operações	1979-09-25	Velha Fátima do Descoberto
42	27511213308	Leila Melo		Ligação Resposta Frente	+55 (35) 3837-9102	Analista	(39) 7849-1494	1965-06-05	Deise do Sul
44	18273161285	Sinezio Saraiva Filho		Produtor Fatores Diretor	(71) 49283-9575	Associado	(40) 6785-5749	1980-01-11	Angelino do Norte
49	91131730810	Rizia Carvalho		Assistente Integração Frente	+55 (45) 0085-5603	Oficial	(20) 6875-6735	1985-09-23	NaN
63	27534718198	a. Emerson Neto		Técnico Aplicações Internacional	(16) 39224-3386	NaN	+55 (57) 4449-3385	1970-11-19	NaN

In [883]: # Cria coluna vazia jobAreaTmp , que recebera os ajustes
df_pl['jobAreaTmp'] = np.nan

In [884]: # Cria coluna vazia phoneNumberTmp , que recebera os ajustes
df_pl['phoneNumberTmp'] = np.nan

In [889]: # Cria uma tabela com os valores que serao substituidos no Campo jobArea
df_pl_tmp = df_pl.loc[df_pl['jobArea'].str.contains('\d+', regex=True, na=True)]
Contem comente os numeros de telefone em JobArea

In [891]: # Adiciona uma nova coluna jobArea_mk como um marcador true para o campo jobArea
df_pl_tmp.loc[df_pl_tmp['jobArea'].str.contains('\d+', regex=True, na=True), 'jobArea_mk'] = True
Atribuo para df_pl_tmp a coluna jobArea_mk com o marcador true

In [892]: # Adiciona uma nova coluna phoneNumber_mk como um marcador true para o campo phoneNumber
df_pl_tmp.loc[df_pl_tmp['phoneNumber'].str.contains('\d+', regex=True, na=True), 'phoneNumber_mk'] = True
Atribuo para df_pl_tmp a coluna phoneNumber_mk com o marcador true que identifica os numeros de telefone

In [895]: # Remove do dataset de replace os campos onde phoneNumber é do tipo numero
df_pl_tmp = df_pl_tmp[df_pl_tmp['phoneNumber_mk'] != True]
removo do datasete as informaçoes onde phoneNumber_mk for diferente de numero de telefone
Tenhp agora no dataset phoneNumber apenas com descricoes e JobArea apenas com numero

In [896]: df_pl_tmp

Out[896]:

job	jobArea	jobType	phoneNumber	birthDate	city	state	totalSpent	job.
Analista municções Corporativo	+55 (47) 2953- 9641	Analista	Operações	09/25/1979	Velha Fátima do Descoberto	Paraná	402.10	
Técnico A infraestrutura Distrito	(24) 00844- 8380	Supervisor	Métricas	19880717,	Grande Neilton	AM	533.76	
Associado restação de contas Investidor	(51) 8225- 7885	Produtor	Funcionalidade	12/21/1967	Moraes de Nossa Senhora	RO	772.42	
Planejador rupo Frente	+55 (32) 1541- 0032	Analista	Funcionalidade	19741014,	Grande Arturo do Descoberto	Ceará	464.09	
Designer municções Nacional	(05) 2163- 5818	Executivo	Otimização	11-Mar- 1980	Nova Graziela do Norte	São Paulo	758.21	
...
NaN	(91) 8786- 0629	Gerente	Intranet	Sat, Jan.22.1966	NaN	PR	493.49	
Designer Contas Regional	(62) 7523- 4994	Administrador	Mercados	15-Jan- 1986	NaN	MS	26.02	
Arquiteto ados Global	(21) 64124- 7648	Coordenador	Grupo	08/25/1986	Grande Onivaldo de Nossa Senhora	Paraíba	14.85	
Consultor Intranet Dinâmico	(56) 17382- 5023	Estrategista	Programa	19680517,	Vila Yuriê	Amazonas	834.43	
Supervisor Aplicações Interno	(27) 0367- 6105	Diretor	Criativo	Wed, Jan.25.1978	NaN	Rio Grande do Norte	222.06	

In [901]: # Adiciona uma nova coluna jobArea_mk como um marcador true para o campo jobArea
df_pl.loc[df_pl['jobArea'].str.contains('\d+', regex=True, na=True), 'jobArea_mk']

In [902]: # Adiciona uma nova coluna phoneNumber_mk como um marcador true para o campo phone
df_pl.loc[df_pl['phoneNumber'].str.contains('\d+', regex=True, na=True), 'phoneNumber_mk']

In [856]: # Confere a novaa colunas
df_pl.head(5)

Out[856]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
0	76684148787	Charllyny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	1972-05-20	Mi
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	1982-10-06	Mi di
2	15664328373377	Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	Mi di
3	02328238087786	Celina Carvalho Jr	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	1981-04-17	
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	1980-07-04	Mi An

In [937]: # Atribui o valor da coluna phoneNumber do dataset df_pl_tmp na coluna jobArea da
Nessa operacāo estou substituindo o valor de jobArea pelo valor de phoneNumber
df_pl.loc[(df_pl['jobArea_mk'] == True), 'jobArea'] = df_pl_tmp['phoneNumber']

In [938]: # Atribui o valor da lista pre filtrada em jobArea com numeros e atribui a coluna
df_pl.loc[(df_pl['phoneNumber_mk'] != True), 'phoneNumber'] = df_pl_tmp['jobArea']

In [924]: # Quando jobArea_mk tras numero e phoneNumber_mk tras numero deve ser atribuido r
Valido na coluna phoneNumber
df_pl.loc[(df_pl['jobArea_mk'] == df_pl['phoneNumber_mk']), 'jobArea'] = np.nan

In [925]: # Para este Resultado deve ser atribuido nulo em JobArea pois em phoneNumber já t
O que foi feito
df_pl[df_pl['jobArea_mk'] == df_pl['phoneNumber_mk']]

Out[925]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate
42	27511213308	Leila Melo	Ligação Resposta Frente	NaN	Analista	(39) 7849-1494	06-05-19
44	182.731.612-85	Sinezio Saraiva Filho	Produtor Fatores Diretor	NaN	Associado	(40) 6785-5749	01/11/19
49	91131730810	Rizia Carvalho	Assistente Integração Frente	NaN	Oficial	(20) 6875-6735	Sep/23/19
63	27534718198	Srta. Emerson Neto	Técnico Aplicações Internacional	NaN	NaN	+55 (57) 4449-3385	19-11-19
70	39481762483	Dr. Valdirene Costa Neto	Executivo Marca Regional	NaN	Produtor	(08) 99447-5247	Mar/30/19
...
999921	54123014188503	Sra. Natalino Melo	Designer Comunicações Global	NaN	Administrador	+55 (55) 0751-2140	23-05-19
999927	81504285778	Sra. Sra. Lêda Martins	Designer Integração Regional	NaN	Facilitador	+55 (96) 1017-1489	10/03/19
999949	617.283.367-75	Laurenildo Carvalho Filho	Facilitador Métricas Cliente	NaN	Supervisor	(41) 79072-4273	07/02/19
999983	870.643.287-55	Sr. Werner Santos	Planejador Rede Nacional	NaN	Gerente	(79) 2661-4712	20-Oct-19
999989	46.563.086/3513-80	Maricy Braga	Desenvolvedor Interações Distrito	NaN	Orquestrador	(01) 7411-1296	30-Dec-19

47507 rows × 14 columns



```
In [914]: # Conferindo substituicao
# Efetuado as substituicoes em JobAreaTMP somente o que foi numero em jobAreaTMP
df_pl.loc[df_pl['jobArea_mk'] != True].head(5)
```

Out[914]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate
0	76684148787	Charlleny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	20-05-1972
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	10-Jun-1982
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	05/16/1968
3	02.328.238/0877-86	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	19810417,
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	04/07/1980



In [939]: # Verifica alterações onde jobArea tinha um numero
df_pl[df_pl['jobArea_mk'] == True]

Out[939]:

job	jobArea	jobType	phoneNumber	birthDate	city	state	totalSpent	job/
Analista Inovações Operativo	Operações	Analista	+55 (47) 2953-9641	09/25/1979	Velha Fátima do Descoberto	Paraná	402.10	O
Ligaçāo Resposta Frente	NaN	Analista	(39) 7849-1494	06-05-1965	Deise do Sul	Tocantins	521.93	
Produtor es Diretor	NaN	Associado	(40) 6785-5749	01/11/1980	Angelino do Norte	Rio de Janeiro	195.40	
Assistente Integração Frente	NaN	Oficial	(20) 6875-6735	Sep/23/1985	NaN	Roraima	717.84	
Técnico aplicações Internacional	NaN	NaN	+55 (57) 4449-3385	19-11-1970	NaN	AC	615.94	
...
Designer Inovações Global	NaN	Administrador	+55 (55) 0751-2140	23-05-1972	NaN	PR	615.98	
Designer Integração Regional	NaN	Facilitador	+55 (96) 1017-1489	10/03/1988	NaN	Rondônia	803.42	
Facilitador Métricas Cliente	NaN	Supervisor	(41) 79072-4273	07/02/1964	Messias do Norte	Rondônia	181.59	
lanejador Nacional	NaN	Gerente	(79) 2661-4712	20-Oct-1969	Costa do Sul	Distrito Federal	392.31	
envolvedor Interações Distrito	NaN	Orquestrador	(01) 7411-1296	30-Dec-1979	Barros de Nossa Senhora	SC	479.49	

In [949]: # Conferindo um documento no dataset backup nao modificado para ver a coerencia da
df_pl_bk[df_pl_teste['document'] == '23896888455']

Out[949]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	city
29	23896888455	Vilmar Braga	Produtor Contas Direto	Comunicações	Estrategista	Segurança	1964-03-07	Moreira do Descoberto

In [941]: `#df_pl.loc[df_pl['phoneNumber_mk'] == True].head(6)`
`df_pl[df_pl['phoneNumber_mk'] != True]`

Out[941]:

		document	name	job	jobArea	jobType	phoneNumber	bir
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador		NaN	10-Jul
22	351.711.803-69	Vanoil Xavier Filho Filho	Analista Comunicações Corporativo	Operações	Analista	+55 (47) 2953-9641	09/2	
29	23896888455	Vilmar Braga	Produtor Contas Direto	Comunicações	Estrategista		NaN	07/0
75	72681165782460	Fernanda Pereira Neto	Analista Fatores Central	Dados	Diretor		NaN	Aug.1
88	437.485.730-30	Eleonora	Ligaçāo Diretivas Interno	Resposta	Diretor		NaN	May/0
...
999877	63157341525662	Mitiko Martins Jr.	Estrategista Programa Frente	Branding		NaN	NaN	29-0:
999917	46323515806	Silmara Moraes	Diretor Divisão Líder	Marketing		NaN	NaN	08/1
999924	75736701050610	Juana Moraes Neto	Arquiteto Integração Regional	Aplicações		NaN	NaN	01-De
999943	70875946640	Sokuchin	Gerente Garantia Produtos	Funcionalidade	Supervisor		NaN	196
999947	75.804.866/2521-29	Sra. Hilca Franco	Gerente Contas Chefe	Operações	Estrategista		NaN	197

49836 rows × 14 columns



In [953]: `df_pl_tmp_4 = df_pl[df_pl['jobArea'].str.contains('\d+', regex=True, na=True)]`

In [954]: df_pl_tmp_4

Out[954]:

		document	name	job	jobArea	jobType	phoneNumber	birthDate
42	27511213308	Leila Melo		Ligação Resposta Frente	NaN	Analista	(39) 7849-1494	06-05-19
44	182.731.612-85	Sinezio Saraiva Filho		Produtor Fatores Diretor	NaN	Associado	(40) 6785-5749	01/11/19
49	91131730810	Rizia Carvalho		Assistente Integração Frente	NaN	Oficial	(20) 6875-6735	Sep/23/19
63	27534718198	Srta. Emerson Neto		Técnico Aplicações Internacional	NaN	NaN	+55 (57) 4449-3385	19-11-19
70	39481762483	Dr. Valdirene Costa Neto		Executivo Marca Regional	NaN	Produtor	(08) 99447-5247	Mar/30/19
...
999921	54123014188503	Sra. Natalino Melo		Designer Comunicações Global	NaN	Administrador	+55 (55) 0751-2140	23-05-19
999927	81504285778	Sra. Lêda Martins		Designer Integração Regional	NaN	Facilitador	+55 (96) 1017-1489	10/03/19
999949	617.283.367-75	Laurenildo Carvalho Filho		Facilitador Métricas Cliente	NaN	Supervisor	(41) 79072-4273	07/02/19
999983	870.643.287-55	Sr. Werner Santos		Planejador Rede Nacional	NaN	Gerente	(79) 2661-4712	20-Oct-19
999989	46.563.086/3513-80	Maricy Braga		Desenvolvedor Interações Distrito	NaN	Orquestrador	(01) 7411-1296	30-Dic-19

47507 rows × 14 columns

In []:

In []:

6- Além desses pontos, existem outras tratamentos para homogenizar esse dataset. Aplique todos que conseguir.

6.1 - Padronizando as Síntese dos Estados

~~... - , autorizando as origens dos estados~~

R: Segue a resposta abaixo:

In [377]: # Verificando o padrao dos dados do Campo Estado
df_pl[['state']].groupby(['state']).nunique()

Out[377]:

state
AC
AL
AM
AP
Acre
Alagoas
Amapá
Amazonas
BA
Bahia
CE
Ceará
DF
Distrito Federal
ES
Espírito Santo
GO
Goiás
MA
MS
MT
Maranhão
Mato Grosso
Mato Grosso do Sul
Minas Gerais
PA
PB
PE
PI
PR
Paraná
Paraíba
Pará

state
Pernambuco
Piauí
RJ
RN
RO
RR
RS
Rio Grande do Norte
Rio Grande do Sul
Rio de Janeiro
Rondônia
Roraima
SC
SP
Santa Catarina
Sergipe
São Paulo
Tocantins

```
In [378]: df_pl['state'].replace({' ':''},inplace=True, regex=True)
df_pl.replace({'Alagoas':'AL','Amapá':'AP','Amazonas':'AM','Bahia':'BA','Ceará':'CE',
               'DistritoFederal':'DF','EspíritoSanto':'ES','Goiás':'GO','Maranhão':'MA',
               'MinasGerais':'MG','Paraná':'PR','Pará':'PA','Pernambuco':'PE',
               'RioGrandedoNorte':'RN','RioGrandedoSul':'RS','Rio de Janeiro':'RJ',
               'SantaCatarina':'SC','SãoPaulo':'SP','Tocantins':'TO','Acre':'AC',
               'Sergipe':'SE','Rio de Janeiro':'RJ'},inplace=True)
df_pl[['state']].groupby(['state']).nunique()
```

Out[378]:

state

AC

AL

AM

AP

BA

CE

DF

ES

GO

MA

MG

MS

MT

PA

PB

PE

PI

PR

RJ

RN

RO

RR

RS

SC

SE

SP

TO

In [379]: `df_pl.head()`

Out[379]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
0	76684148787	Charlleny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	1972-05-20	Mi di
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	1982-10-06	Mi di
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	Mi di
3	02328238087786	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	1981-04-17	
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	1980-07-04	Mi An

In []:

6.2 - Padronizando o campo name

R: Segue a resposta abaixo:

In [380]: `# Verificando o padrao dos dados do Campo name`
`df_pl[['name']].groupby(['name']).nunique()`

Out[380]:

name
Abdias
Abdias Albuquerque
Abdias Albuquerque Filho
Abdias Barros
Abdias Barros Filho
...
Érico Silva Neto
Érico Souza
Érico Souza Filho
Érico Souza Neto Filho
Érico Xavier

455014 rows × 0 columns

```
In [381]: df_pl_title=pd.DataFrame(df_pl['name'].str[:3])
df_pl_title['name'].head(10)
```

```
Out[381]: 0    Cha
1    New
2    Dr.
3    Cel
4    Aur
5    Sr.
6    Sr.
7    Mar
8    Ben
9    Iza
Name: name, dtype: object
```

```
In [382]: df_pl_title.describe()
```

```
Out[382]:
```

	name
count	1000000
unique	956
top	Sr.
freq	91743

```
In [383]: df_pl.dtypes
```

```
Out[383]: document          object
name            object
job             object
jobArea         object
jobType         object
phoneNumber     object
birthDate       datetime64[ns]
city            object
state           object
totalSpent      float64
tipo_doc        object
dtype: object
```

In [384]: # Testando Pesquisa no Campo Name com A , funciona
df_pl[df_pl['name'].str.contains('A', na=False)].head(3)

Out[384]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	1980-07-04	Mu Am
5	77145788233	Sr. Sávio Albuquerque	Estrategista Branding Interno	Resposta	Planejador	+55 (06) 0512-5291	1983-06-18	Mu E
14	23487182365693	Edith Albuquerque Neto	Diretor Interações Líder	Garantia	Estrategista	+55 (67) 9793-9050	1986-02-12	

In [385]: # Testando Pesquisa no Campo Name com ".", Tem algum Bug e não esta funcionando
df_pl[df_pl['name'].str.contains('. ', na=False)].head(5)

Out[385]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
0	76684148787	Charllyny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	1972-05-20	Mi
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	1982-10-06	Mi d
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	Mi d
3	02328238087786	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	1981-04-17	
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	1980-07-04	Mi An

```
In [386]: # Vou substituir tudo ponto por traço , depois trazer somente os 3 primeiros caracteres e depois aplicar um ajuste somente nas colunas com branco
# Objetivo montar uma lista com os termos principais que contenha ponto e que ser um titulo e por isso nao tem que estar no nome de uma pessoa
df_pl['filtro_name'] = df_pl['name'].str.replace('.', '_')
df_pl.head(5)
```

Out[386]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
0	76684148787	Charlleny Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	1972-05-20	M
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	1982-10-06	M
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	M
3	02328238087786	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	1981-04-17	
4	30073687408740	Aurilo Martins	Especialista Paradigma Internacional	Programa	Executivo	(96) 47498-7325	1980-07-04	M
								An

```
In [387]: df_pl[['filtro_name']].groupby('filtro_name').nunique().head(5)
```

Out[387]:

filtro_name
Abdias
Abdias Albuquerque
Abdias Albuquerque Filho
Abdias Barros
Abdias Barros Filho

In [388]: #Fazendo a busca pelo novo campo filtro_name pelo caractere underline, esta traz
df_pl[df_pl['filtro_name'].str.contains('_', regex=True, na=True)].head(5)

Out[388]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	Mi de
3	02328238087786	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	1981-04-17	I
5	77145788233	Sr. Sávio Albuquerque	Estrategista Branding Interno	Resposta	Planejador	+55 (06) 0512-5291	1983-06-18	Mi
6	75752983045	Sr. Everaldo Saraiva Filho	Técnico Táticas Produtos	Marketing	Diretor	(33) 78784-4156	1980-10-27	(
11	58412777743	Sra. Lhirton Batista Jr.	Produtor Intranet Nacional	Configuração	Arquiteto	(33) 70042-1213	1985-12-05	

In [389]: # Fazendo a busca pelo campo JR
O campo Jr no final do processo esta me dando problemas por causa do ponto , po
df_pl[df_pl['filtro_name'].str.contains('Jr', regex=True, na=True)].head(5)

Out[389]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
3	02328238087786	Celina Carvalho Jr.	NaN	Qualidade	Ligaçāo	+55 (58) 4136-5577	1981-04-17	
11	58412777743	Sra. Lhirton Batista Jr.	Produtor Intranet Nacional	Configuração	Arquiteto	(33) 70042-1213	1985-12-05	B
13	23285262680	Dov Carvalho Jr.	Administrador Pesquisa Legado	Resposta	Analista	(65) 37328-4182	1989-02-08	E
25	35288067880	Vivaldo Pereira Jr.	Planejador Intranet Corporativo	Operações	Engenheiro	+55 (58) 8631-1622	1966-06-11	De
47	77982195016	Srta. Cerilo Oliveira Jr.	Estrategista Marca Central	Mercados	Arquiteto	(89) 37080-3645	1987-09-06	Vil

```
In [390]: # Substitui Jr_ Por Jr. na coluna filtro_name
# Ja havia feito alguns testes e essa palavra Jr estava atrapalhando o resultado
# trata-la previamente

df_pl['filtro_name'].replace({'Jr.':'Jr'},inplace=True, regex=True)
df_pl[['filtro_name']].groupby('filtro_name').nunique().head(10)
```

Out[390]:

filtro_name
Abdias
Abdias Albuquerque
Abdias Albuquerque Filho
Abdias Barros
Abdias Barros Filho
Abdias Barros Jr
Abdias Batista
Abdias Batista Filho
Abdias Batista Jr
Abdias Braga

In [391]: #Fazendo a busca pelo campo JR no campo filtro_name , para confirmar o ajuste feito
df_pl[df_pl['filtro_name'].str.contains('Jr', regex=True, na=True)].head(5)

Out[391]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate
3	02328238087786	Celina Carvalho Jr.	NaN	Qualidade	Ligaçao	+55 (58) 4136-5577	1981-04-17
11	58412777743	Sra. Lhirton Batista Jr.	Produtor Intranet Nacional	Configuração	Arquiteto	(33) 70042-1213	1985-12-05
13	23285262680	Dov Carvalho Jr.	Administrador Pesquisa Legado	Resposta	Analista	(65) 37328-4182	1989-02-08
25	35288067880	Vivaldo Pereira Jr.	Planejador Intranet Corporativo	Operações	Engenheiro	+55 (58) 8631-1622	1966-06-11
47	77982195016	Srta. Cerilo Oliveira Jr.	Estrategista Marca Central	Mercados	Arquiteto	(89) 37080-3645	1987-09-06

```
In [392]: # Aplicando nova transformação na coluna filtro_name
# Atualizando a tabela 'filtro_name' com apenas os 7 primeiros caracteres
# Montando uma coluna marcador para determinar o que será marcado como item a ser
df_pl['filtro_name_tmp']=pd.DataFrame(df_pl['filtro_name'].str[:7])
df_pl[['filtro_name_tmp']].head(10)
```

Out[392]:

	filtro_name_tmp
0	Charlle
1	Newton
2	Dr_ Sr_
3	Celina
4	Aurilo
5	Sr_ Sáv
6	Sr_ Eve
7	Martina
8	Benedik
9	Izamar

```
In [393]: # Atribuindo branco para valores invalidos em
# O que me interessa é o que tem underline o resto já esta correto entao sera red
# uma unica linha no meus distinct
df_pl.loc[df_pl['filtro_name_tmp'].str.contains('_', regex=True, na=True), 'filtro_name']=np.nan

df_pl[['filtro_name_tmp']].head(5)
```

Out[393]:

	filtro_name_tmp
0	Charlle
1	Newton
2	
3	Celina
4	Aurilo

In [394]: `df_pl[['filtro_name','filtro_name_tmp']].head(5)`

Out[394]:

	filtro_name	filtro_name_tmp
0	Charleny Braga	Charlle
1	Newton Saraiva	Newton
2	Dr_ Sr_ Solange Macedo	
3	Celina Carvalho Jr	Celina
4	Aurilo Martins	Aurilo

In [395]: `# Quando a coluna filtro_name_tmp for diferente de branco , atribui branco ao campo filtro_name`
`# Marcando correto, as linhas relevantes em filtro_name estao marcadas como brancas`
`df_pl.loc[(df_pl['filtro_name_tmp']!=""),'filtro_name']=''`
`df_pl[['filtro_name','filtro_name_tmp']].head(20)`

Out[395]:

	filtro_name	filtro_name_tmp
0		Charlle
1		Newton
2	Dr_ Sr_ Solange Macedo	
3		Celina
4		Aurilo
5	Sr_ Sávio Albuquerque	
6	Sr_ Everaldo Saraiva Filho	
7		Martina
8		Benedik
9		Izamar
10		Débora
11	Sra_ Lhirton Batista Jr	
12	Sra_ Rosildo Santos	
13		Dov Car
14		Edith A
15		Deilson
16		Viviana
17		Dione O
18		Kauy Fi
19	Dr_ Valdete Franco	

In [396]: # Filtra os 4 primeiros caracteres do campo filtro_name e atribui ao campo filtro
Montando primeira lista com todos os valores relevantes em uma nova coluna filtro
df_p1[['filtro_name_p1']] = pd.DataFrame(df_p1['filtro_name'].str[:4])
df_p1[['filtro_name', 'filtro_name_p1']]

Out[396]:

	filtro_name	filtro_name_p1
0		
1		
2	Dr_ Sr_ Solange Macedo	Dr_
3		
4		
...
999995		
999996		
999997		
999998		
999999	Sr_ Jair Martins	Sr_

In [397]: #Verifica valores unicos
Já tenho minha primeira lista de valores a serem substituídos no campo name
df_p1[['filtro_name_p1']].groupby(['filtro_name_p1']).nunique()

Out[397]:

filtro_name_p1
Dr_
Dr_
Sr_
Sra_
Srta

```
In [398]: #Verifica todos os nulos do dataset
# Estou vendo se tem valores nulos na coluna name
df_pl.isnull().sum()
```

```
Out[398]: document          0
name            0
job           99915
jobArea        0
jobType       100189
phoneNumber      0
birthDate        0
city          200001
state            0
totalSpent     9775
tipo_doc         0
filtro_name      0
filtro_name_tmp    0
filtro_name_p1      0
dtype: int64
```

```
In [399]: #Fazendo a busca pelo campo name valores branco
df_pl[( df_pl['name'] == '')]
```

```
Out[399]: document  name  job  jobArea  jobType  phoneNumber  birthDate  city  state  totalSpent  tipo_d

```

```
In [400]: # Confirmando valores entre name e filtro name para posterior verificacao apos ap
df_pl[df_pl['filtro_name'].str.contains('Dr.', regex=True, na=True)].head(5)
```

```
Out[400]:
```

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
2	15664328373377	Dr. Sr. Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	M d
19	13243665857133	Dr. Valdete Franco	NaN	Integração	NaN	+55 (21) 6949-7240	1972-03-21	J
23	40367582414	Dr. Márcia Barros	Consultor Integração Diretor	Configuração	Arquiteto	+55 (00) 0960-8178	1966-06-03	M c
46	28673547067	Dr. Laura Nogueira Filho	Ligaçāo Operações Global	Configuração	Coordenador	+55 (68) 1398-9342	1977-10-23	C s
52	71743405804	Dr. Tānia	Supervisor A infraestrutura Direto	Integração	Designer	(75) 6119-5294	1983-12-09	

```
In [401]: #tratamento na coluna original parte 01
df_pl['name'].replace({'Jr.':'Jr','Dr.':'','Sr.':'','Sra.':'','Srta.':''},regex=True)
```

In [402]: #Fazendo a busca pelo campo Sr no campo filtro_name e comparando com name , os valores

```
df_pl[df_pl['filtro_name'].str.contains('Dr', regex=True, na=True)].head(5)
```

Out[402]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
2	15664328373377	Solange Macedo	Designer Identidade Direto	Métricas	NaN	+55 (95) 7143-3307	1968-05-16	M d
19	13243665857133	Valdete Franco	NaN	Integração	NaN	+55 (21) 6949-7240	1972-03-21	J
23	40367582414	Márcia Barros	Consultor Integração Diretor	Configuração	Arquiteto	+55 (00) 0960-8178	1966-06-03	M c
46	28673547067	Laura Nogueira Filho	Ligaçāo Operações Global	Configuração	Coordenador	+55 (68) 1398-9342	1977-10-23	C S
52	71743405804	Tânia	Supervisor A Infraestrutura Direto	Integração	Designer	(75) 6119-5294	1983-12-09	

In [403]: # Vendo se ficou alguma informação faltante

```
df_pl['filtro_name2'] = df_pl['name'].str.replace('.', '_')
df_pl['filtro_name2'].head()
```

Out[403]:

```
0      Charlleny Braga
1      Newton Saraiva
2      Solange Macedo
3      Celina Carvalho Jr
4      Aurilo Martins
Name: filtro_name2, dtype: object
```

In [404]: # Fazendo a busca pelo campo _ no campo filtro_name2 , para confirmar o ajuste feito
Detectei que tem alguns nomes que o primeiro caracter esta vindo com ponto
df_pl[df_pl['filtro_name2'].str.contains('_', regex=True, na=True)].head(5)

Out[404]:

		document	name	job	jobArea	jobType	phoneNumber	birthDate
11	58412777743	. Lhirton Batista Jr		Produtor Intranet Nacional	Configuração	Arquiteto	(33) 70042-1213	1985-12-05
12	43256426510740	. Rosildo Santos		Desenvolvedor Funcionalidade Corporativo	Integração	Engenheiro	(12) 3014-4120	1980-12-14
34	77234489100	a. Walquiria Oliveira		Diretor Identidade Internacional	Paradigma	Especialista	(72) 60926-7940	1966-02-10
41	02741370621	a. a. Valeria Moraes Neto		Ligaçao Aplicações Investidor	Pesquisa	Orquestrador	(87) 1655-1370	1989-02-14
47	77982195016	a. Cerilo Oliveira Jr		Estrategista Marca Central	Mercados	Arquiteto	(89) 37080-3645	1987-09-06

In [405]: # Fazendo copia do dataset
df_pl_bk = df_pl.copy()

In [749]: # Recuperando a copia do dataset
df_pl = df_pl_bk.copy()

In [407]: # Remocao do ponto inicial na coluna name
df_pl['name']=df_pl.name.str.replace('^.\s','')

In [408]: # Confirmando a remoção do ponto inicial
`df_pl[df_pl['filtro_name2'].str.contains('_', regex=True, na=True)].head(5)`

Out[408]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate
11	58412777743	Lhirton Batista Jr	Produtor Intranet Nacional	Configuração	Arquiteto	(33) 70042-1213	1985-12-05
12	43256426510740	Rosildo Santos	Desenvolvedor Funcionalidade Corporativo	Integração	Engenheiro	(12) 3014-4120	1980-12-14
34	77234489100	a. Walquiria Oliveira	Diretor Identidade Internacional	Paradigma	Especialista	(72) 60926-7940	1966-02-10
41	02741370621	a. a. Valeria Moraes Neto	Ligaçao Aplicações Investidor	Pesquisa	Orquestrador	(87) 1655-1370	1989-02-14
47	77982195016	a. Cerilo Oliveira Jr	Estrategista Marca Central	Mercados	Arquiteto	(89) 37080-3645	1987-09-06

In [409]: # Confere nulos
`df_pl.isnull().sum()`

Out[409]:

document	0
name	0
job	99915
jobArea	0
jobType	100189
phoneNumber	0
birthDate	0
city	200001
state	0
totalSpent	9775
tipo_doc	0
filtro_name	0
filtro_name_tmp	0
filtro_name_p1	0
filtro_name2	0
dtype:	int64

In [410]: # Atribuindo nulos para os valores em branco do campo name para facilitar sua ideia
tratamento na coluna original
`df_pl['name'].replace({' ':np.nan},regex=True, inplace=True)`

```
In [411]: # Agora temos 19 valores nulos no campo name é um valor insignificante perto do volume de dados
# É uma decisão de negocio remover ou não , depende da analise que esta sendo feita
# removido pois tem cpf valido associado e tenho que fazer depois agrupamentos para o totalSpent
df_pl.isnull().sum()
```

```
Out[411]: document          0
name            19
job           99915
jobArea         0
jobType        100189
phoneNumber      0
birthDate        0
city           200001
state            0
totalSpent      9775
tipo_doc          0
filtro_name        0
filtro_name_tmp      0
filtro_name_p1        0
filtro_name2          0
dtype: int64
```

```
In [414]: # Reavaliando campo name para ver se ficou algum nome sem tratar
df_pl['filtro_name'] = df_pl['name'].str.replace('.', '_')
df_pl[df_pl['filtro_name'].str.contains('_', regex=True, na=True)].head(5)
df_pl['filtro_name'].replace({'Jr.':'Jr'}, inplace=True, regex=True)
df_pl['filtro_name_tmp']=pd.DataFrame(df_pl['filtro_name'].str[:3])
df_pl.loc[df_pl['filtro_name_tmp'].str.contains('_', regex=True, na=True), 'filtro_name']=df_pl['filtro_name'].str[:3]
df_pl.loc[(df_pl['filtro_name_tmp']!=''), 'filtro_name']=''
df_pl[['filtro_name_p1']] = pd.DataFrame(df_pl['filtro_name'].str[:4])
df_pl['filtro_name_p1']=df_pl.filtro_name_p1.str.replace('^\s+', '')
```

```
In [415]: # Nao retornou nada fora do padrao
df_pl[['filtro_name_p1']].groupby(['filtro_name_p1']).nunique().head()
```

Out[415]:

<u>filtro_name_p1</u>
B
-
_A
_B

In [416]: # Conferindo colunas
df_pl.dtypes

Out[416]:

document	object
name	object
job	object
jobArea	object
jobType	object
phoneNumber	object
birthDate	datetime64[ns]
city	object
state	object
totalSpent	float64
tipo_doc	object
filtro_name	object
filtro_name_tmp	object
filtro_name_p1	object
filtro_name2	object
dtype:	object

In [417]: # Limpa as colunas temporárias do dataset
df_pl.drop(['filtro_name', 'filtro_name_tmp', 'filtro_name_p1', 'filtro_name2'],

In [418]: df_pl.dtypes

Out[418]:

document	object
name	object
job	object
jobArea	object
jobType	object
phoneNumber	object
birthDate	datetime64[ns]
city	object
state	object
totalSpent	float64
tipo_doc	object
dtype:	object

In [419]: df_pl.head(2)

Out[419]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	c
0	76684148787	Charlley Braga	Oficial Criativo Dinâmico	Configuração	Estrategista	(62) 4216-9799	1972-05-20	Munic de I
1	85704855733	Newton Saraiva	Administrador Comunicações Internacional	Prestação de contas	Facilitador	Aplicações	1982-10-06	Munic de Ne do

In [420]: df_pl.shape

Out[420]: (1000000, 11)

In []:

Agregação dos dados

1 - Quais são as 5 PF que mais gastaram (totalSpent)?

R: Segue a resposta abaixo:

CPF	Valor
78155376770	1973.11
57913684291	1889.39
08174237704	1870.75
70795760671	1857.11
31307158277	1854.80

In [421]: #Filtrar os dados somente por PF e grava em um novo dataframe
df_pl_2 = df_pl[(df_pl['tipo_doc'] == 'PF')]
df_pl_2

6	75752983045	Saraiva Filho	Táticas Produtos	Marketing	Diretor	(33) 78784-4156	19	▲		
7	88363035521	Martina Moreira	Associado Contas Legado	Marketing	Designer	+55 (30) 1195-8461	19	▼		
...	
999993	22756200603	Cíntia	Administrador Grupo Internacional	Mercados	Representante	(24) 6082-7393	19	▼		
999994	51988920922	Ivaldo Reis	Orquestrador Divisão Regional	Mercados	Técnico	+55 (23) 2395-2235	19	▼		
999995	20297054201	Aurilo Franco Neto	Agente Funcionalidade Investidor	Otimização	Nan	(54) 8742-5107	19	▼		

In [422]: #Efetua a soma agrupando por document
df_pl_grp_pf = pd.DataFrame(df_pl_2.groupby(['document'])['totalSpent'].sum())

In [423]: df_pl_grp_pf

Out[423]:

totalSpent

document	totalSpent
00000229423	757.46
00000668265	196.59
00001356402	901.16
00003036235	325.36
00003620697	502.82
...	...
99991168842	440.78
99991430598	155.60
99992271396	511.88
99994518100	130.37
99997206762	177.26

749807 rows × 1 columns

In [424]: # Tras os 5 cpfs com maiores totais para o campo Total Spent
Estes são os 5 cpfs com maiores gastos

df_pl_grp_pf.sort_values(by=['totalSpent'], ascending=False).head(5)

Out[424]:

totalSpent

document	totalSpent
78155376770	1973.11
57913684291	1889.39
08174237704	1870.75
70795760671	1857.11
31307158277	1854.80

In []:

2 - Qual é o valor de gasto médio por estado (state)?

R: Segue a resposta abaixo:

In [425]: df_pl.dtypes

Out[425]:

document	object
name	object
job	object
jobArea	object
jobType	object
phoneNumber	object
birthDate	datetime64[ns]
city	object
state	object
totalSpent	float64
tipo_doc	object
dtype:	object

In [426]: # Segue o calculo da média por estado.

```
df_pl_grpes_2 = pd.DataFrame(df_pl.groupby(['state'])['totalSpent'].mean())
df_pl_grpes_2
```

state	
AC	502.478424
AL	500.515519
AM	498.770663
AP	504.259137
BA	498.774140
CE	499.300300
DF	499.107366
ES	501.631747
GO	501.292493
MA	500.435299
MG	499.532094
MS	499.509741

In []:

3 - Qual é o valor de gasto médio por jobArea?

R: Segue a resposta abaixo:

In [427]: `df_pl_grpjob = pd.DataFrame(df_pl.groupby(['jobArea'])['totalSpent'].mean())
df_pl_grpjob`

Out[427]:

	totalSpent
jobArea	
(00) 00435-1738	219.350000
(00) 01430-7785	799.860000
(00) 01525-7613	128.480000
(00) 01837-4277	125.280000
(00) 01870-0192	340.800000
...	...
Resposta	499.409451
Segurança	503.234670
Soluções	501.432619
Táticas	498.413775
Usabilidade	506.597001

50028 rows × 1 columns

In []:

4 - Qual é a PF que gastou menos (totalSpent)?

R: Foram os CPFs 82121396624, 61782608494, 28876278214, 12037751729 com os valores de 0.01 e 44922380850 com 0.02 , foram os que gastaram menos

In [428]: `# Remove os valores com zero que nao me interessa ver
df_pl_grp_pf_2 = df_pl_grp_pf[df_pl_grp_pf['totalSpent'] != 0]`

In [429]: `# Conferindo o menor valor
df_pl_grp_pf_2.min()`

Out[429]: `totalSpent 0.01
dtype: float64`

In [430]: `#Listando os 5 menores valores
df_pl_grp_pf_2.sort_values(by=['totalSpent'], ascending=True).head(5)`

Out[430]:

totalSpent	
document	
82121396624	0.01
61782608494	0.01
28876278214	0.01
12037751729	0.01
44922380850	0.02

In [431]: `# Conferindo os valores sem soma para estes Cpf's
df_pl[df_pl['document'].isin(['82121396624', '61782608494', '28876278214', '12037751729'])]`

Out[431]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	
65196	28876278214	Christina	Orquestrador Divisão Líder	Qualidade	Técnico	(80) 96132-1590	1982-09-30	Mac Nc Senf
392327	82121396624	Walcar Pereira	Executivo Garantia Distrito	Otimização	Arquiteto	+55 (03) 9698-0097	1962-10-16	Osi N
448089	12037751729	Vanderlene Martins	Assistente Branding Internacional	(47) 3427-8645	Designer	(09) 5993-9544	1987-07-13	Per do
476990	44922380850	Izabel Braga	Planejador Métricas Cliente	Prestação de contas	NaN	+55 (92) 0543-9605	1968-10-23	Lc
619650	61782608494	Zulamar Barros	Diretor Fatores Cliente	Qualidade	Oficial	(38) 8210-9979	1981-07-16	Det N

In []:

5 - Quantos nomes e documentos repetidos existem nesse dataset?

R: Temos 430 nomes repetidos por documento

E 173.571 documentos repetidos por nomes

Existe divergência entre as duas contagens, mas é justificável, pois temos como exemplo o nome Érico Xavier que possui 5 documentos diferentes associados a ele, então quando conto o nome e agrupo por documento retornara apenas 1 para cada documento e quando conto os documentos e agrupo por nome trara 5, mas esta correto

In [445]: # Criando dataset com a contagem de nomes por documentos PF e CNPJ
`df_pl_count_doc = df_pl[['document', 'name']].groupby(['document']).count()`

In [447]: # Verificando dataset
`df_pl_count_doc.head()`

Out[447]:

document	name
00000229423	1
00000315842849	1
00000416348025	1
00000463865561	1
00000668265	1

In [448]: `df_pl_count_doc.max()`

Out[448]: name 2
 dtype: int64

In [449]: #Listando os 5 menores valores
`df_pl_count_doc.sort_values(by=['name'], ascending=False).head(5)`

Out[449]:

document	name
11814323406	2
23751867104	2
55136447540	2
15851338504	2
81265535884	2

In [450]: `df_pl_count_doc_2=df_pl_count_doc[df_pl_count_doc['name'].isin([2])]`

In [451]: #Verificando se o menor valor é o 2 , após o ajuste
`df_pl_count_doc_2.min()`

Out[451]: name 2
 dtype: int64

In [452]: # Contando todos os nomes repetidos por documentos
`df_pl_count_doc_2.count()`

Out[452]: name 430
 dtype: int64

In []:

```
In [454]: # Criando dataset com a contagem de documentos PF e CNPJ por nomes
df_pl_count_doc_b = df_pl[['document', 'name']].groupby(['name']).count()
df_pl_count_doc_b
```

name	
	5
. Carvalho	1
. Nogueira	1
. Abdias Carvalho	1
. Abdias Costa Neto	1
...	...
Érico Silva Neto	3
Érico Souza	4
Érico Souza Filho	1
Érico Souza Neto Filho	1
Érico Xavier	5

272252 rows × 1 columns

```
In [455]: # Pegando apenas os que tiverem a contagem maior que 1 que são os repetidos
df_pl_count_doc_b_2 = df_pl_count_doc_b[df_pl_count_doc_b['document'] > 1]
```

```
In [456]: #Verificando se o menor valor é o 2 , após o ajuste
df_pl_count_doc_b_2.min()
```

```
Out[456]: document    2
dtype: int64
```

```
In [457]: # Contando todos os documentos repetidos por nome
df_pl_count_doc_b_2.count()
```

```
Out[457]: document    173571
dtype: int64
```

In [458]: df_pl_count_doc_b_2

Out[458]:

document

	name
	5
. Albina Reis	2
. Amarai Reis	2
. Anailza Silva	2
. Angelica Reis	2
...	...
Érico Saraiva Neto	2
Érico Silva	4
Érico Silva Neto	3
Érico Souza	4
Érico Xavier	5

173571 rows × 1 columns

In [459]: df_pl[df_pl['name']=='Érico Xavier']

Out[459]:

	document	name	job	jobArea	jobType	phoneNumber	birthDate	city
387525	18916214599	Érico Xavier	Facilitador Fatores Cliente	Táticas	Consultor	(30) 18908-3306	1981-06-11	Natal
458770	26608122753840	Érico Xavier	Facilitador Mercados Produtos	Integração	Arquiteto	+55 (09) 5458-0541	1983-10-12	Natal
830977	47304644121430	Érico Xavier	Analista Segurança Produtos	Marca	Consultor	(78) 36021-6258	1987-06-15	Fabiano de Nossa Senhora
897572	54623511430	Érico Xavier	Arquiteto Aplicações Direto	Programa	Associado	+55 (83) 7155-0549	1982-07-26	Natal
998300	56314984050	Érico Xavier	Executivo Táticas Líder	Resposta	Diretor	(63) 69007-6804	1985-03-31	Macedo do Norte

In []:

6 - Quantas linhas existem nesse dataset?

R: Existe 1 Milhão de linhas neste dataset

```
In [460]: df_pl['document'].count()
```

```
Out[460]: 1000000
```

```
In [461]: len(df_pl.index)
```

```
Out[461]: 1000000
```

```
In [ ]:
```

Particionamento de dados tratados com as regras descritas em DATA QUALITY**1- Particionar em arquivos PARQUET por estado (state)**

R:Segue Resposta Abaixo

```
In [465]: # Carrega modulos adicionais
import pyarrow as pa
import pyarrow.parquet as pq
```

```
In [476]: type(df_pl)
```

```
Out[476]: pandas.core.frame.DataFrame
```

```
In [479]: # Converte do pandas para o formato pyarrow
table = pa.Table.from_pandas(df)
```

```
In [480]: # Exportando para o formato parquet particionado por state
pq.write_to_dataset(
    table,
    root_path='output.parquet_state',
    partition_cols=['state'],
)
```

```
In [ ]:
```

2 - Particionar em arquivos CSV por ano/mes/dia de nascimento (birthDate)

```
In [504]: df_p1[['birthDate']].head()
```

Out[504]:

	birthDate
0	1972-05-20
1	1982-06-10
2	1968-05-16
3	1981-04-17
4	1980-04-07

```
In [521]: df_pl['birthDate_txt'] = df_pl['birthDate'].astype(str)
```

```
In [535]: lista_birth = df_pl['birthDate_txt'].tolist()
```

```
In [537]: lista_birth = sorted(lista_birth)
```

```
In [538]: lista_birth
```

```
Out[538]: ['1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-02',  
          '1962-01-03',  
          '1962-01-03',  
          '1962-01-03',  
          '1962-01-03']
```

```
In [539]: for birthDate in lista_birth:  
    #Chaves é um marcador recebe a varavel em format  
    caminho = "output.csv_birth/{}_export.csv".format(birthDate)  
    df_pl[df_pl['birthDate']==birthDate].to_csv(caminho)
```

In [1]:

