Ventas DataFrames originales

Obtención de los datos y librerías necesarias.

In [1]:
```
# Importamos las librerías que usaremos durante la limpieza:
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import re
import scipy.stats as sts
from bokeh.plotting import figure
from bokeh.io import output_notebook, show
from IPython.display import display
from pandas import ExcelWriter
from pandas import ExcelFile
```

In [2]:
```
# Importamos el dataset

with pd.ExcelFile('dwh.xlsx') as xlsx:
    df_fac = pd.read_excel(xlsx, 'FactTableOrderDetail')
    df_dim = pd.read_excel(xlsx, 'DimOrderHead')
```

1. Relevamiento inicial y diagnóstico del estado del dataset

In [3]:
```
# Conocemos cual es el tamaño del dataset
df_fac.shape
```

Out[3]: (558074, 62)

In [4]:
```
# Aca el segundo archivo df_dim
df_dim.shape
```

Out[4]: (262620, 50)

Arrancamos a analizar la df_fac

In [5]:
```
df_fac.isnull().sum().sort_values()
```

Out[5]:
```
UniqueId                          0
OrderItemId                       0
UserProfileId                     0
ClientProfileId                   0
State                             0
EmailVtex                         0
DocumentType                      0
Document                          0
ShippingId                        0
DimShippingSK                     0
AddressId                         0
SelectedAddressType               0
SelectedAddressCity               0
SelectedAdressCountry             0
SelectedAddressShippingId         0
SelectedAddressState              0
SelectedAddressStreet             0
AddressType                       0
City                              0
Country                           0
Lat                               0
Lng                               0
OrderDate                         0
PaymentDataId                     0
OrderId                           0
MarketingDataId                   0
Id                                0
ProductId                         0
EAN                               0
LockId                            0
                                 ...
Name                              0
RefId                             0
ShippingDataId                    0
Price                             0
ListPrice                         0
ManualPrice                       0
Street                            0
ImageUrl                          0
ClientProfileDataOrderId          0
RawItemDiscount                   0
Ispercentual                      0
PromoName                         0
Commission                        0
PriceValidUnit                    0
GrossItemAmountDiscount           0
SellerSKU                         0
DetailUrl                         0
SelectedAddressPostalCode        12
PostalCode                       12
Number                        15121
SelectedAdressNumber          15121
SelectedAddressReceiverName   35455
ReceiverName                  35455
Complemet                    409438
SelectedAddressComplement    409438
SelectedAddressNeighborhood  429123
Neighborhood                 429123
SelectedAddressReference     556307
TrackingHints                558074
GiftRegistryData             558074
Length: 62, dtype: int64
```

In [6]:
```
df_fac.columns
```

Out[6]:
```
Index(['UniqueId', 'Id', 'ProductId', 'EAN', 'LockId', 'Quantity', 'Seller',
       'Name', 'RefId', 'Price', 'ListPrice', 'ManualPrice', 'ImageUrl',
       'DetailUrl', 'SellerSKU', 'PriceValidUnit', 'Commission', 'PromoName',
       'GrossItemAmountDiscount', 'Ispercentual', 'RawItemDiscount',
       'ClientProfileDataOrderId', 'GiftRegistryData', 'MarketingDataId',
       'ShippingDataId', 'PaymentDataId', 'OrderDate', 'OrderItemId',
       'UserProfileId', 'ClientProfileId', 'OrderId', 'EmailVtex',
       'DocumentType', 'Document', 'ShippingId', 'DimShippingSK', 'AddressId',
       'TrackingHints', 'SelectedAddressType', 'SelectedAddressCity',
       'SelectedAddressComplement', 'SelectedAdressCountry',
       'SelectedAddressNeighborhood', 'SelectedAdressNumber',
       'SelectedAddressPostalCode', 'SelectedAddressReceiverName',
       'SelectedAddressReference', 'SelectedAddressShippingId',
       'SelectedAddressState', 'SelectedAddressStreet', 'AddressType', 'City',
       'Complemet', 'Country', 'Lat', 'Lng', 'Neighborhood', 'Number',
       'PostalCode', 'ReceiverName', 'State', 'Street'],
      dtype='object')
```

In [7]:
```
df_fac.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 558074 entries, 0 to 558073
Data columns (total 62 columns):
UniqueId                        558074 non-null object
Id                              558074 non-null int64
ProductId                       558074 non-null int64
EAN                             558074 non-null object
LockId                          558074 non-null object
Quantity                        558074 non-null int64
Seller                          558074 non-null int64
Name                            558074 non-null object
RefId                           558074 non-null object
Price                           558074 non-null float64
ListPrice                       558074 non-null float64
ManualPrice                     558074 non-null int64
ImageUrl                        558074 non-null object
DetailUrl                       558074 non-null object
SellerSKU                       558074 non-null int64
PriceValidUnit                  558074 non-null int64
Commission                      558074 non-null int64
PromoName                       558074 non-null object
GrossItemAmountDiscount         558074 non-null float64
Ispercentual                    558074 non-null bool
RawItemDiscount                 558074 non-null float64
ClientProfileDataOrderId        558074 non-null object
GiftRegistryData                0 non-null float64
MarketingDataId                 558074 non-null object
ShippingDataId                  558074 non-null object
PaymentDataId                   558074 non-null object
OrderDate                       558074 non-null object
OrderItemId                     558074 non-null object
UserProfileId                   558074 non-null object
ClientProfileId                 558074 non-null object
OrderId                         558074 non-null object
EmailVtex                       558074 non-null object
DocumentType                    558074 non-null object
Document                        558074 non-null object
ShippingId                      558074 non-null object
DimShippingSK                   558074 non-null int64
AddressId                       558074 non-null object
TrackingHints                   0 non-null float64
SelectedAddressType             558074 non-null object
SelectedAddressCity             558074 non-null object
SelectedAddressComplement       148636 non-null object
SelectedAdressCountry           558074 non-null object
SelectedAddressNeighborhood     128951 non-null object
SelectedAdressNumber            542953 non-null object
SelectedAddressPostalCode       558062 non-null float64
SelectedAddressReceiverName     522619 non-null object
SelectedAddressReference        1767 non-null object
SelectedAddressShippingId       558074 non-null object
SelectedAddressState            558074 non-null object
SelectedAddressStreet           558074 non-null object
AddressType                     558074 non-null object
City                            558074 non-null object
Complemet                       148636 non-null object
Country                         558074 non-null object
Lat                             558074 non-null float64
Lng                             558074 non-null float64
Neighborhood                    128951 non-null object
Number                          542953 non-null object
PostalCode                      558062 non-null float64
ReceiverName                    522619 non-null object
State                           558074 non-null object
Street                          558074 non-null object
dtypes: bool(1), float64(10), int64(9), object(42)
memory usage: 260.3+ MB
```

In [8]:
```
# Analisis de registros duplicados (todas las columnas)
duplicados = df_fac[df_fac.duplicated(keep='first')]
len(duplicados)
```

Out[8]: 0

In [9]:
```
df_fac.nunique().sort_values(ascending=True)
```

Out[9]:
```
TrackingHints                     0
GiftRegistryData                  0
Commission                        1
Seller                            1
ManualPrice                       1
Ispercentual                      1
PriceValidUnit                    1
DocumentType                      2
EAN                               3
AddressType                       3
SelectedAddressType               3
Country                           4
SelectedAdressCountry             4
Quantity                         11
State                           129
SelectedAddressState            129
ListPrice                       249
Price                           251
SelectedAddressReference        348
OrderDate                       453
Neighborhood                    768
SelectedAddressNeighborhood     768
SelectedAddressCity            2155
City                           2155
SelectedAddressPostalCode      2183
PostalCode                     2183
ProductId                      2361
DetailUrl                      2387
Name                           4054
SellerSKU                      4198
                              ...
ImageUrl                       4284
SelectedAdressNumber          10245
Number                        10245
SelectedAddressComplement     11522
Complemet                     11522
GrossItemAmountDiscount       14043
Lat                           15276
Lng                           15511
RawItemDiscount               23634
PromoName                     32615
SelectedAddressStreet         40869
Street                        40869
SelectedAddressReceiverName   156492
ReceiverName                  156492
Document                      159156
UserProfileId                 161836
MarketingDataId               204610
OrderId                       262620
AddressId                     262620
DimShippingSK                 262620
ShippingId                    262620
EmailVtex                     262620
ClientProfileId               262620
PaymentDataId                 262620
ShippingDataId                262620
ClientProfileDataOrderId      262620
LockId                        262620
SelectedAddressShippingId     262620
OrderItemId                   558074
UniqueId                      558074
Length: 62, dtype: int64
```

In [10]:
```
# Borro las columnas en 0 o 1, no agregan info para analisis
df_fac.drop(['TrackingHints','GiftRegistryData','Commission','Seller','ManualPrice','Ispercentual','PriceValidUnit
            axis=1, inplace=True)
```

In [11]:
```
# Conocemos cual es el tamaño del dataset
df_fac.shape
```

Out[11]:
```
(558074, 55)
```

In [12]:
```
df_fac.head()
```

Out[12]:

	UniqueId	Id	ProductId	EAN	LockId	Quantity	Name	RefId	Price	ListPri
0	59431C45D7674B3A95FC1257C05B9E86	3483	1930	0	00-937403242935-01	1	Juego de Toalla y Toallón Liso Color Blanco	30002E76784%H	699.0	699
1	9DA40701E57C40F197FC1230EB1953B6	694	447	0	00-937403242935-01	1	Juego de Toalla y Toallón Línea Turbat Color B...	30012E76782%H	699.0	699
2	0A099F6548294BF692C2F7B66D9CA403	4527	2478	0	00-937403242935-01	1	Juego de Sábanas Cuna Funcional Milo Safari Co...	10033F16310%VE	1199.0	1199
					00-		Juego de Sábanas			

Analizo que valores hay en columnas y si hay repetidas para volver a achicarlas

In [13]:
```
# Hay 2 tipos de documentos segun pais
df_fac.DocumentType.value_counts()
```

Out[13]:
```
dni          526536
cedulaURY     31538
Name: DocumentType, dtype: int64
```

In [14]:
```
df_fac.EAN.value_counts()
```

Out[14]:
```
0               557742
10034K15926%A      320
82460Z65019%U       12
Name: EAN, dtype: int64
```

In [15]:
```
df_fac.AddressType.value_counts()
```

Out[15]:
```
residential    336045
pickup         221870
commercial        159
Name: AddressType, dtype: int64
```

In [16]:
```
df_fac.SelectedAddressType.value_counts()
```

Out[16]:
```
residential    336045
pickup         221870
commercial        159
Name: SelectedAddressType, dtype: int64
```

In [17]:
```
# Intuyo que las columnas 'AddressType' y 'SelectedAddressType'son iguales y si es asi, borrare una
a = df_fac['AddressType']
b = df_fac['SelectedAddressType']
c = a == b
c.all()
```

Out[17]:
```
True
```

In [18]:
```
df_fac.Country.value_counts()
```

Out[18]:
```
ARG    526528
URY     31539
ESP         4
ITA         3
Name: Country, dtype: int64
```

In [19]:
```
df_fac.SelectedAdressCountry.value_counts()
```

Out[19]:
```
ARG    526528
URY     31539
ESP         4
ITA         3
Name: SelectedAdressCountry, dtype: int64
```

In [20]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['Country']
b = df_fac['SelectedAdressCountry']
c = a == b
c.all()
```

Out[20]: True

In [21]:
```
# Intuyo que las columnas pueden ser iguales, porque coinciden las cantidades unicas... son iguales y si es asi, b
a = df_fac['State']
b = df_fac['SelectedAddressState']
c = a == b
c.all()
```

Out[21]: True

In [22]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['City']
b = df_fac['SelectedAddressCity']
c = a == b
c.all()
```

Out[22]: True

In [23]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['Street']
b = df_fac['SelectedAddressStreet']
c = a == b
c.all()
```

Out[23]: True

In [24]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['ReceiverName']
b = df_fac['SelectedAddressReceiverName']
c = a == b
c.all()
```

Out[24]: False

In [25]:
```
c.value_counts()
```

Out[25]:
```
True     522619
False     35455
dtype: int64
```

In [26]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['Number']
b = df_fac['SelectedAdressNumber']
c = a == b
c.all()
```

Out[26]: False

In [27]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['Complemet']
b = df_fac['SelectedAddressComplement']
c = a == b
c.all()
```

Out[27]: False

In [28]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['Neighborhood']
b = df_fac['SelectedAddressNeighborhood']
c = a == b
c.all()
```

Out[28]: False

In [29]:
```
c.value_counts()
```

Out[29]:
```
False    429123
True     128951
dtype: int64
```

In [30]:
```
# Intuyo que las columnas ... son iguales y si es asi, borrare una
a = df_fac['PostalCode']
b = df_fac['SelectedAddressPostalCode']
c = a == b
c.all()
```

Out[30]: False

In [31]:
```
c.value_counts()
```

Out[31]:
```
True     558062
False        12
dtype: int64
```

In [32]:
```
df_fac.Quantity.value_counts()
```

Out[32]:
```
1     483654
2      62192
3       6020
4       3921
6       1390
5        634
10        97
8         90
7         69
9          6
20         1
Name: Quantity, dtype: int64
```

In [33]:
```
# Intuyo que las columnas ... son casi iguales ????
a = df_fac['Price']
b = df_fac['ListPrice']
c = a == b
c.all()
```

Out[33]:
```
False
```

In [34]:
```
c.value_counts()
```

Out[34]:
```
True     547790
False     10284
dtype: int64
```

In [1]:
```
df_fac.Document.value_counts()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-6b7ac504fb22> in <module>
----> 1 df_fac.Document.value_counts()

NameError: name 'df_fac' is not defined
```

In [2]:
```
df_fac.ReceiverName.value_counts()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-123de387f2a9> in <module>
----> 1 df_fac.ReceiverName.value_counts()

NameError: name 'df_fac' is not defined
```

In [41]:
```
# Procedo a ir borrando las columnas iguales y las que no generan valor por ahora como las URL a modo de ejemplo
df_fac.drop(['SelectedAddressType', 'SelectedAdressCountry', 'SelectedAddressState','SelectedAddressCity','DetailU
             'ImageUrl','SelectedAddressStreet','EmailVtex'], axis=1, inplace=True)

df_fac.columns
```

Out[41]:
```
Index(['UniqueId', 'Id', 'ProductId', 'EAN', 'LockId', 'Quantity', 'Name',
       'RefId', 'Price', 'ListPrice', 'SellerSKU', 'PromoName',
       'GrossItemAmountDiscount', 'RawItemDiscount',
       'ClientProfileDataOrderId', 'MarketingDataId', 'ShippingDataId',
       'PaymentDataId', 'OrderDate', 'OrderItemId', 'UserProfileId',
       'ClientProfileId', 'OrderId', 'DocumentType', 'Document', 'ShippingId',
       'DimShippingSK', 'AddressId', 'SelectedAddressComplement',
       'SelectedAddressNeighborhood', 'SelectedAdressNumber',
       'SelectedAddressPostalCode', 'SelectedAddressReceiverName',
       'SelectedAddressReference', 'SelectedAddressShippingId', 'AddressType',
       'City', 'Complemet', 'Country', 'Lat', 'Lng', 'Neighborhood', 'Number',
       'PostalCode', 'ReceiverName', 'State', 'Street'],
      dtype='object')
```

In [42]:
```
# Conocemos cual es el tamaño del dataset
df_fac.shape
```

Out[42]:
```
(558074, 47)
```

In [43]:
```
# Busco a los extranjeros por DNI en el pais, y pondre dni2
d = df_fac.Document > '90000000'
d.sum()
```

Out[43]: 11921

In [3]:
```
df_fac.Document[d].unique()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-de5b8d471010> in <module>
----> 1 df_fac.Document[d].unique()

NameError: name 'df_fac' is not defined
```

In [45]:
```
df_fac.DocumentType[df_fac.DocumentType == 'dni'] = 'dni1'
df_fac.DocumentType[d] ='dni2'
df_fac.DocumentType[df_fac.DocumentType == 'cedulaURY'] = 'dni3'
```

```
C:\Users\hmarq\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  """Entry point for launching an IPython kernel.
C:\Users\hmarq\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy)

C:\Users\hmarq\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy (http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy)
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [46]:
```
df_fac.DocumentType.unique()
```

Out[46]: array(['dni1', 'dni2', 'dni3'], dtype=object)

In [4]:
```
a = df_fac.Document.unique()
a
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-4-057cec16cfe5> in <module>
----> 1 a = df_fac.Document.unique()
      2 a

NameError: name 'df_fac' is not defined
```

In [5]:
```
# Genero data frame de documentos para sacar el valor del dataframe con el que trabajare.
df_doc = pd.DataFrame({'Documento': [i for i in range(len(a))], 'docu': [i for i in (a)]})

df_doc.sample(10)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-737378c116c9> in <module>
      1 # Genero data frame de documentos para sacar el valor del dataframe con el que trabajare.
----> 2 df_doc = pd.DataFrame({'Documento': [i for i in range(len(a))], 'docu': [i for i in (a)]})
      3
      4 df_doc.sample(10)

NameError: name 'pd' is not defined
```

In [116]:
```
df_doc.Documento = df_doc.Documento.astype(str)
```

In [52]:
```
df_fac = pd.merge(df_fac, df_doc, left_on = ['Document'], right_on =['docu'])
```

In [6]:
```
df_fac[{'Document', 'docu', 'Documento', 'ReceiverName'}].sample(5)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-6-d0d06277b787> in <module>
----> 1 df_fac[{'Document', 'docu', 'Documento', 'ReceiverName'}].sample(5)

NameError: name 'df_fac' is not defined
```

In [54]:
```
df_doc = pd.merge(df_doc, df_fac[{'Document', 'DocumentType', 'ReceiverName', 'SelectedAddressReceiverName',\
                  'SelectedAdressNumber', 'Number', 'Street','OrderId'}], left_on = ['docu'], right_on =['Document
```

In [7]:
```
df_doc.sample(10)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-7-323a7d786ad3> in <module>
----> 1 df_doc.sample(10)

NameError: name 'df_doc' is not defined
```

In [56]:
```
df_doc.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 558074 entries, 0 to 558073
Data columns (total 10 columns):
Documento                     558074 non-null object
docu                          558074 non-null object
Document                      558074 non-null object
DocumentType                  558074 non-null object
SelectedAdressNumber          542953 non-null object
Street                        558074 non-null object
ReceiverName                  522619 non-null object
Number                        542953 non-null object
OrderId                       558074 non-null object
SelectedAddressReceiverName   522619 non-null object
dtypes: object(10)
memory usage: 46.8+ MB
```

In [57]:
```
# Intuyo que las columnas 'Document' y 'docu'son iguales y si es asi, borrare una
a = df_doc['docu']
b = df_doc['Document']
c = a == b
c.all()
```

Out[57]:
```
True
```

In [58]:
```
df_doc.drop(['docu'], axis=1, inplace=True)
df_doc.columns
```

Out[58]:
```
Index(['Documento', 'Document', 'DocumentType', 'SelectedAdressNumber',
       'Street', 'ReceiverName', 'Number', 'OrderId',
       'SelectedAddressReceiverName'],
      dtype='object')
```

In [59]:
```
# Procedo a ir borrando las columnas que no puedo usar por confidencialidad de datos
df_fac.drop(['Document', 'ReceiverName', 'SelectedAddressReceiverName', 'SelectedAdressNumber', 'Number', 'docu'],
            axis=1, inplace=True)

df_fac.columns
```

Out[59]:
```
Index(['UniqueId', 'Id', 'ProductId', 'EAN', 'LockId', 'Quantity', 'Name',
       'RefId', 'Price', 'ListPrice', 'SellerSKU', 'PromoName',
       'GrossItemAmountDiscount', 'RawItemDiscount',
       'ClientProfileDataOrderId', 'MarketingDataId', 'ShippingDataId',
       'PaymentDataId', 'OrderDate', 'OrderItemId', 'UserProfileId',
       'ClientProfileId', 'OrderId', 'DocumentType', 'ShippingId',
       'DimShippingSK', 'AddressId', 'SelectedAddressComplement',
       'SelectedAddressNeighborhood', 'SelectedAddressPostalCode',
       'SelectedAddressReference', 'SelectedAddressShippingId', 'AddressType',
       'City', 'Complemet', 'Country', 'Lat', 'Lng', 'Neighborhood',
       'PostalCode', 'State', 'Street', 'Documento'],
      dtype='object')
```

In [60]:
```
# Procedo a ir borrando las columnas que definimos no usar por codigos internos no utiles para el trabajo posteric
df_fac.drop(['UniqueId', 'EAN', 'LockId', 'ClientProfileDataOrderId', 'MarketingDataId', 'PaymentDataId', 'OrderIt
             'UserProfileId', 'ClientProfileId', 'ShippingId', 'DimShippingSK', 'SelectedAddressShippingId'],\
            axis=1, inplace=True)

df_fac.columns
```

Out[60]:
```
Index(['Id', 'ProductId', 'Quantity', 'Name', 'RefId', 'Price', 'ListPrice',
       'SellerSKU', 'PromoName', 'GrossItemAmountDiscount', 'RawItemDiscount',
       'ShippingDataId', 'OrderDate', 'OrderId', 'DocumentType', 'AddressId',
       'SelectedAddressComplement', 'SelectedAddressNeighborhood',
       'SelectedAddressPostalCode', 'SelectedAddressReference', 'AddressType',
       'City', 'Complemet', 'Country', 'Lat', 'Lng', 'Neighborhood',
       'PostalCode', 'State', 'Street', 'Documento'],
      dtype='object')
```

In [61]:
```
# Conocemos cual es el tamaño del dataset
df_fac.shape
```

Out[61]: (558074, 31)

Arrancamos a analizar la df_dim

In [62]:
```
df_dim.isnull().sum().sort_values()
```

Out[62]:
```
DimOrderSK                      0
UserProfileId                   0
ShippingCost                    0
TotalDiscountPrice              0
TotalOrderGross                 0
IsChequedIn                     0
AllowEdition                    0
AllowCancellation               0
StorePreferenceDataId           0
IsCompleted                     0
OrderFormId                     0
OrderDate                       0
HostName                        0
FollowUpEmail                   0
PaymentDataId                   0
ShippingDataId                  0
ShippingId                      0
Sequence                        0
ClientProfileDataOrderId        0
MarketPlaceServiceEndpoint      0
SellerOrderId                   0
Origin                          0
SalesChannel                    0
Status                          0
OrderId                         0
Value                           0
CreationDate                    0
LastChange                      0
OrderGroup                      0
StatusDescription               0
TimeStamp                       2
MarketingDataId             46543
IsCorporate                185955
Client Name                185955
DocumentType               185955
DocumentNumber             185955
Phone                      185955
Client LastName            185957
CustomerClass              262620
CorporatePhone             262620
StateInscription           262620
GiftRegistryData           262620
CorporateDocument          262620
CorporateName              262620
MarketPaceOrderId          262620
AffiliatedId               262620
MerchantName               262620
LastMessage                262620
TradeName                  262620
InvoiceData                262620
dtype: int64
```

In [63]:
```
df_dim.columns
```

Out[63]:
```
Index(['DimOrderSK', 'OrderId', 'Sequence', 'MarketPaceOrderId',
       'MarketPlaceServiceEndpoint', 'SellerOrderId', 'Origin', 'AffiliatedId',
       'SalesChannel', 'MerchantName', 'Status', 'StatusDescription', 'Value',
       'CreationDate', 'LastChange', 'OrderGroup', 'ClientProfileDataOrderId',
       'GiftRegistryData', 'MarketingDataId', 'ShippingDataId',
       'PaymentDataId', 'FollowUpEmail', 'LastMessage', 'HostName',
       'InvoiceData', 'OrderFormId', 'IsCompleted', 'StorePreferenceDataId',
       'AllowCancellation', 'AllowEdition', 'IsChequedIn', 'TotalOrderGross',
       'TotalDiscountPrice', 'ShippingCost', 'Client Name', 'Client LastName',
       'DocumentType', 'DocumentNumber', 'Phone', 'CorporateName',
       'CorporateDocument', 'TradeName', 'StateInscription', 'CorporatePhone',
       'IsCorporate', 'UserProfileId', 'CustomerClass', 'TimeStamp',
       'OrderDate', 'ShippingId'],
      dtype='object')
```

In [64]:
```
df_dim.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 262620 entries, 0 to 262619
Data columns (total 50 columns):
DimOrderSK                    262620 non-null int64
OrderId                       262620 non-null object
Sequence                      262620 non-null int64
MarketPaceOrderId             0 non-null float64
MarketPlaceServiceEndpoint    262620 non-null object
SellerOrderId                 262620 non-null object
Origin                        262620 non-null object
AffiliatedId                  0 non-null float64
SalesChannel                  262620 non-null int64
MerchantName                  0 non-null float64
Status                        262620 non-null object
StatusDescription             262620 non-null object
Value                         262620 non-null float64
CreationDate                  262620 non-null object
LastChange                    262620 non-null object
OrderGroup                    262620 non-null object
ClientProfileDataOrderId      262620 non-null object
GiftRegistryData              0 non-null float64
MarketingDataId               216077 non-null object
ShippingDataId                262620 non-null object
PaymentDataId                 262620 non-null object
FollowUpEmail                 262620 non-null object
LastMessage                   0 non-null float64
HostName                      262620 non-null object
InvoiceData                   0 non-null float64
OrderFormId                   262620 non-null object
IsCompleted                   262620 non-null bool
StorePreferenceDataId         262620 non-null object
AllowCancellation             262620 non-null bool
AllowEdition                  262620 non-null bool
IsChequedIn                   262620 non-null bool
TotalOrderGross               262620 non-null float64
TotalDiscountPrice            262620 non-null float64
ShippingCost                  262620 non-null float64
Client Name                   76665 non-null object
Client LastName               76663 non-null object
DocumentType                  76665 non-null object
DocumentNumber                76665 non-null float64
Phone                         76665 non-null float64
CorporateName                 0 non-null float64
CorporateDocument             0 non-null float64
TradeName                     0 non-null float64
StateInscription              0 non-null float64
CorporatePhone                0 non-null float64
IsCorporate                   76665 non-null float64
UserProfileId                 262620 non-null object
CustomerClass                 0 non-null float64
TimeStamp                     262618 non-null datetime64[ns]
OrderDate                     262620 non-null object
ShippingId                    262620 non-null object
dtypes: bool(4), datetime64[ns](1), float64(19), int64(3), object(23)
memory usage: 93.2+ MB
```

In [65]:
```
# Analisis de registros duplicados (todas las columnas)
duplicados = df_dim[df_dim.duplicated(keep='first')]
len(duplicados)
```

Out[65]: 0

In [66]:
```
df_dim.nunique().sort_values(ascending=True)
```

Out[66]:
```
InvoiceData                       0
CorporateName                     0
LastMessage                       0
CorporateDocument                 0
GiftRegistryData                  0
StateInscription                  0
CorporatePhone                    0
MerchantName                      0
TradeName                         0
AffiliatedId                      0
CustomerClass                     0
MarketPaceOrderId                 0
IsChequedIn                       1
AllowEdition                      1
IsCorporate                       1
Origin                            1
IsCompleted                       1
DocumentType                      2
AllowCancellation                 2
SalesChannel                      2
HostName                          2
MarketPlaceServiceEndpoint        4
Status                           12
StatusDescription                16
ShippingCost                     26
OrderDate                       453
TotalOrderGross                8628
Client Name                   13470
TimeStamp                     22393
TotalDiscountPrice            28964
Value                         34953
Client LastName               36250
DocumentNumber                60466
Phone                         61651
UserProfileId                161836
MarketingDataId              204610
OrderFormId                  258752
FollowUpEmail                262578
LastChange                   262594
DimOrderSK                   262620
PaymentDataId                262620
ShippingDataId               262620
ClientProfileDataOrderId     262620
OrderGroup                   262620
CreationDate                 262620
SellerOrderId                262620
Sequence                     262620
OrderId                      262620
StorePreferenceDataId        262620
ShippingId                   262620
dtype: int64
```

In [67]:
```
# Borro las columnas en 0 o 1, no agregan info para analisis
df_dim.drop(['InvoiceData', 'CorporateName','LastMessage', 'CorporateDocument','GiftRegistryData','StateInscriptic
             'CorporatePhone','MerchantName','TradeName','AffiliatedId','CustomerClass','MarketPaceOrderId', \
             'IsChequedIn','AllowEdition','IsCorporate','Origin', 'IsCompleted'], axis=1, inplace=True)
```

In [68]:
```
# Conocemos cual es el tamaño del dataset
df_dim.shape
```

Out[68]:
```
(262620, 33)
```

In [8]:
```
df_dim.head()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-8-3be2a6428787> in <module>
----> 1 df_dim.head()

NameError: name 'df_dim' is not defined
```

Analizo que valores hay en columnas y si hay repetidas para volver a achicarlas

In [70]:
```
# Hay 2 tipos de documentos segun pais
df_dim.DocumentType.value_counts()
```

Out[70]:
```
dni          71233
cedulaURY     5432
Name: DocumentType, dtype: int64
```

In [71]:
```
df_dim.AllowCancellation.value_counts()
```

Out[71]:
```
False    172167
True      90453
Name: AllowCancellation, dtype: int64
```

In [72]:
```
df_dim.SalesChannel.value_counts()
```

Out[72]:
```
1    248600
2     14020
Name: SalesChannel, dtype: int64
```

In [9]:
```
df_dim.HostName.value_counts()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-9-d1124b39d15b> in <module>
----> 1 df_dim.HostName.value_counts()

NameError: name 'df_dim' is not defined
```

In [10]:
```
df_dim.MarketPlaceServiceEndpoint.value_counts()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-10-a1246d1b780f> in <module>
----> 1 df_dim.MarketPlaceServiceEndpoint.value_counts()

NameError: name 'df_dim' is not defined
```

In [75]:
```
df_dim.HostName.replace({"hogar": 'Pippo', "hogar-uy": 'Pippo_uy'}, inplace=True)
```

In [76]:
```
df_dim.MarketPlaceServiceEndpoint.replace({'http://portal.vtexcommerce.com.br/api/oms?an=hogar': 'portal_Pippo', \
                                           'http://oms.vtexinternal.com.br/api/oms?an=hogar': 'oms_Pippo', \
                                           'http://portal.vtexcommerce.com.br/api/oms?an=hogar-uy': 'portal_Pippo_
                                         'http://oms.vtexinternal.com.br/api/oms?an=hogar-uy': 'oms_Pippo_uy'}, inplac
```

In [77]:
```
df_dim.HostName.value_counts()
```

Out[77]:
```
Pippo       248600
Pippo_uy     14020
Name: HostName, dtype: int64
```

In [78]:
```
df_dim.MarketPlaceServiceEndpoint.value_counts()
```

Out[78]:
```
portal_Pippo       164416
oms_Pippo           84184
portal_Pippo_uy      7334
oms_Pippo_uy         6686
Name: MarketPlaceServiceEndpoint, dtype: int64
```

In [79]:
```
# Borramos el SalesChanel, ya que es igual que HostName
a = (df_dim.SalesChannel == 1) & (df_dim.HostName == 'Pippo')
a.sum()
```

Out[79]:
```
248600
```

In [80]:
```
a = (df_dim.SalesChannel == 2) & (df_dim.HostName == 'Pippo_uy')
a.sum()
```

Out[80]:
```
14020
```

In [81]:
```
df_dim.DocumentNumber.isnull().value_counts()
```

Out[81]:
```
True     185955
False     76665
Name: DocumentNumber, dtype: int64
```

In [82]:
```
df_dim.DocumentNumber = df_dim.DocumentNumber.fillna(0).astype(int).astype(object).where(df_dim.DocumentNumber.not
```

In [83]:
```
# Busco a los extranjeros por DNI en el pais, y pondre dni2
d = df_dim.DocumentNumber > 90000000
d.sum()
```

Out[83]:
```
1765
```

In [11]:
```
df_dim.DocumentNumber[d].sample(5)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-11-bc0b955f7b49> in <module>
----> 1 df_dim.DocumentNumber[d].sample(5)

NameError: name 'df_dim' is not defined
```

In [85]:
```
df_dim.columns
```

Out[85]:
```
Index(['DimOrderSK', 'OrderId', 'Sequence', 'MarketPlaceServiceEndpoint',
       'SellerOrderId', 'SalesChannel', 'Status', 'StatusDescription', 'Value',
       'CreationDate', 'LastChange', 'OrderGroup', 'ClientProfileDataOrderId',
       'MarketingDataId', 'ShippingDataId', 'PaymentDataId', 'FollowUpEmail',
       'HostName', 'OrderFormId', 'StorePreferenceDataId', 'AllowCancellation',
       'TotalOrderGross', 'TotalDiscountPrice', 'ShippingCost', 'Client Name',
       'Client LastName', 'DocumentType', 'DocumentNumber', 'Phone',
       'UserProfileId', 'TimeStamp', 'OrderDate', 'ShippingId'],
      dtype='object')
```

In [86]:
```
df_dim.shape
```

Out[86]:
```
(262620, 33)
```

In [87]:
```
df_doc.columns
```

Out[87]:
```
Index(['Documento', 'Document', 'DocumentType', 'SelectedAdressNumber',
       'Street', 'ReceiverName', 'Number', 'OrderId',
       'SelectedAddressReceiverName'],
      dtype='object')
```

In [88]:
```
df_doc.shape
```

Out[88]:
```
(558074, 9)
```

In [89]:
```
df_dim = pd.merge(df_dim, df_doc[{'Documento', 'DocumentType', 'Document','OrderId'}],\
                  left_on = ['OrderId'], right_on = ['OrderId'], how ='outer')
```

In [90]:
```
df_dim.shape
```

Out[90]:
```
(558074, 36)
```

In [91]:
```
# Analisis de registros duplicados (todas las columnas)
duplicados = df_dim[df_dim.OrderId.duplicated(keep='first')]
len(duplicados)
```

Out[91]:
```
295454
```

In [92]:
```
df_dim.drop_duplicates(keep='first', inplace=True)

df_dim.shape
```

Out[92]:
```
(262620, 36)
```

In [12]:
```
df_dim[{'Documento', 'Client Name', 'Client LastName', 'DocumentNumber', 'Phone','Document'}].sample(10)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-12-e68c6039ff4f> in <module>
----> 1 df_dim[{'Documento', 'Client Name', 'Client LastName', 'DocumentNumber', 'Phone','Document'}].sample(10)

NameError: name 'df_dim' is not defined
```

In [94]:
```
df_doc = pd.merge(df_doc, df_dim[{'Client Name', 'Client LastName', 'Phone', 'OrderId'}], \
                  left_on = ['OrderId'], right_on =['OrderId'])
```

In [95]:
```
# Procedo a ir borrando las columnas que no puedo usar por confidencialidad de datos

df_dim.drop(['Client Name', 'Client LastName', 'DocumentType_x', 'DocumentNumber', 'Phone',
             'Document'], axis=1, inplace=True)
df_dim.columns
```

Out[95]:
```
Index(['DimOrderSK', 'OrderId', 'Sequence', 'MarketPlaceServiceEndpoint',
       'SellerOrderId', 'SalesChannel', 'Status', 'StatusDescription', 'Value',
       'CreationDate', 'LastChange', 'OrderGroup', 'ClientProfileDataOrderId',
       'MarketingDataId', 'ShippingDataId', 'PaymentDataId', 'FollowUpEmail',
       'HostName', 'OrderFormId', 'StorePreferenceDataId', 'AllowCancellation',
       'TotalOrderGross', 'TotalDiscountPrice', 'ShippingCost',
       'UserProfileId', 'TimeStamp', 'OrderDate', 'ShippingId', 'Documento',
       'DocumentType_y'],
      dtype='object')
```

In [96]:
```
# Procedo a ir borrando las columnas que definimos no usar por codigos internos no utiles para el trabajo posteric
df_dim.drop(['DimOrderSK', 'Sequence', 'SellerOrderId', 'StatusDescription', 'OrderGroup', 'ClientProfileDataOrder
       'MarketingDataId', 'ShippingDataId', 'PaymentDataId', 'FollowUpEmail', 'OrderFormId', 'StorePreferenceDataI
       'AllowCancellation','UserProfileId', 'ShippingId','SalesChannel'], axis=1, inplace=True)
df_dim.columns
```

Out[96]:
```
Index(['OrderId', 'MarketPlaceServiceEndpoint', 'Status', 'Value',
       'CreationDate', 'LastChange', 'HostName', 'TotalOrderGross',
       'TotalDiscountPrice', 'ShippingCost', 'TimeStamp', 'OrderDate',
       'Documento', 'DocumentType_y'],
      dtype='object')
```

In [97]:
```
df_dim.rename(columns={'DocumentType_y':'DocumentType'}, inplace=True)
```

In [98]:
```
df_dim.columns
```

Out[98]:
```
Index(['OrderId', 'MarketPlaceServiceEndpoint', 'Status', 'Value',
       'CreationDate', 'LastChange', 'HostName', 'TotalOrderGross',
       'TotalDiscountPrice', 'ShippingCost', 'TimeStamp', 'OrderDate',
       'Documento', 'DocumentType'],
      dtype='object')
```

In [99]:
```
df_dim.Status.value_counts()
```

Out[99]:
```
invoiced              251994
handling                7772
canceled                2501
ready-for-handling       264
payment-pending           49
payment-approved          21
invoice                    6
approve-payment            5
start-handling             4
on-order-completed         2
cancel                     1
window-to-cancel           1
Name: Status, dtype: int64
```

In [100]:
```
df_dim.isnull().sum().sort_values()
```

Out[100]:
```
OrderId                     0
MarketPlaceServiceEndpoint  0
Status                      0
Value                       0
CreationDate                0
LastChange                  0
HostName                    0
TotalOrderGross             0
TotalDiscountPrice          0
ShippingCost                0
OrderDate                   0
Documento                   0
DocumentType                0
TimeStamp                   2
dtype: int64
```

In [101]:
```
df_dim.nunique().sort_values(ascending=True)
```

Out[101]:
```
HostName                         2
DocumentType                     3
MarketPlaceServiceEndpoint       4
Status                          12
ShippingCost                    26
OrderDate                      453
TotalOrderGross               8628
TimeStamp                    22393
TotalDiscountPrice           28964
Value                        34953
Documento                   159156
LastChange                  262594
OrderId                     262620
CreationDate                262620
dtype: int64
```

Arrancamos a analizar la df_doc

In [102]:
```
df_doc.shape
```

Out[102]:
```
(558074, 12)
```

In [103]:
```
df_doc.isnull().sum().sort_values()
```

Out[103]:
```
Documento                       0
Document                        0
DocumentType                    0
Street                          0
OrderId                         0
SelectedAdressNumber        15121
Number                      15121
ReceiverName                35455
SelectedAddressReceiverName 35455
Client Name                395303
Phone                      395303
Client LastName            395305
dtype: int64
```

In [104]:
```
df_doc.nunique().sort_values(ascending=True)
```

Out[104]:
```
DocumentType                     3
SelectedAdressNumber         10245
Number                       10245
Client Name                  13470
Client LastName              36250
Street                       40869
Phone                        61651
ReceiverName                156492
SelectedAddressReceiverName 156492
Documento                   159156
Document                    159156
OrderId                     262620
dtype: int64
```

In [105]:
```
df_doc.columns
```

Out[105]:
```
Index(['Documento', 'Document', 'DocumentType', 'SelectedAdressNumber',
       'Street', 'ReceiverName', 'Number', 'OrderId',
       'SelectedAddressReceiverName', 'Client LastName', 'Client Name',
       'Phone'],
      dtype='object')
```

In [106]:
```
df_doc.shape
```

Out[106]:
```
(558074, 12)
```

In [13]:
```
df_doc.sample(10)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-13-323a7d786ad3> in <module>
----> 1 df_doc.sample(10)

NameError: name 'df_doc' is not defined
```

In [109]:
```
df_fac.columns
```

Out[109]:
```
Index(['Id', 'ProductId', 'Quantity', 'Name', 'RefId', 'Price', 'ListPrice',
       'SellerSKU', 'PromoName', 'GrossItemAmountDiscount', 'RawItemDiscount',
       'ShippingDataId', 'OrderDate', 'OrderId', 'DocumentType', 'AddressId',
       'SelectedAddressComplement', 'SelectedAddressNeighborhood',
       'SelectedAddressPostalCode', 'SelectedAddressReference', 'AddressType',
       'City', 'Complemet', 'Country', 'Lat', 'Lng', 'Neighborhood',
       'PostalCode', 'State', 'Street', 'Documento'],
      dtype='object')
```

In [110]:
```
df_dim.columns
```

Out[110]:
```
Index(['OrderId', 'MarketPlaceServiceEndpoint', 'Status', 'Value',
       'CreationDate', 'LastChange', 'HostName', 'TotalOrderGross',
       'TotalDiscountPrice', 'ShippingCost', 'TimeStamp', 'OrderDate',
       'Documento', 'DocumentType'],
      dtype='object')
```

In [111]:
```
df_doc.columns
```

Out[111]:
```
Index(['Documento', 'Document', 'DocumentType', 'SelectedAdressNumber',
       'Street', 'ReceiverName', 'Number', 'OrderId',
       'SelectedAddressReceiverName', 'Client LastName', 'Client Name',
       'Phone'],
      dtype='object')
```

In [112]:
```
# cargo cada base nueva
df_fac.to_csv('Pippo_fac.csv')
df_dim.to_csv('Pippo_dim.csv')
df_doc.to_csv('Pippo_doc.csv')
```