**CASO PRÁCTICO**

**Máster Análisis de Datos con Python**

Iván Pinar Domínguez ©

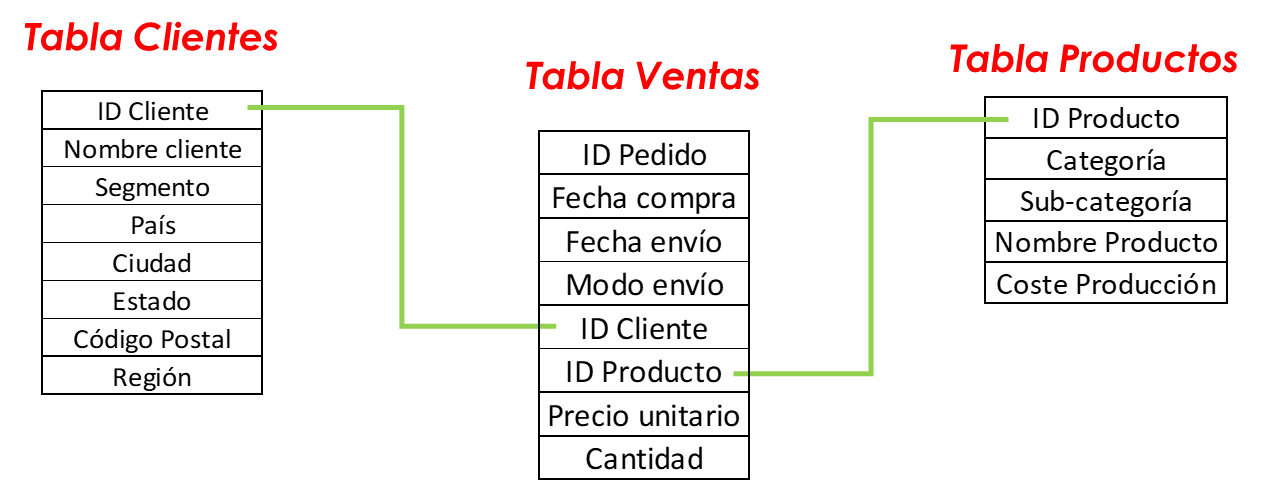
***Enunciado del Caso***

En este caso práctico usted es el Data Scientist de la empresa Mobpatec, dedicada a la fabricación de mobiliario de oficina y dispositivos tecnológicos.

Actualmente, su empresa dispone de una BBDD SQL donde registra lo siguiente:

* Tabla con las ventas de cada artículo.
* Tabla con la información de los productos que vende.
* Tabla con la información de sus clientes.

El modelo de datos de la BBDD relacional sería el siguiente:



Para facilidad de uso del caso práctico, dispone de estas 3 tablas en formato xlsx adjuntas, puesto que no conoce la IP del servidor de la BBDD SQL pero le han pasado el exportado con los datos.

Además, le han adelantado que va a tener que hacer una serie de estudios en función de la población del país de tus clientes, su PIB y la renta per cápita promedio, por tanto va a tener que buscar en la web estas tablas de datos:

* PIB / Población vs. País (la población es la columna "Population"):

<https://www.worldometers.info/gdp/gdp-by-country/>

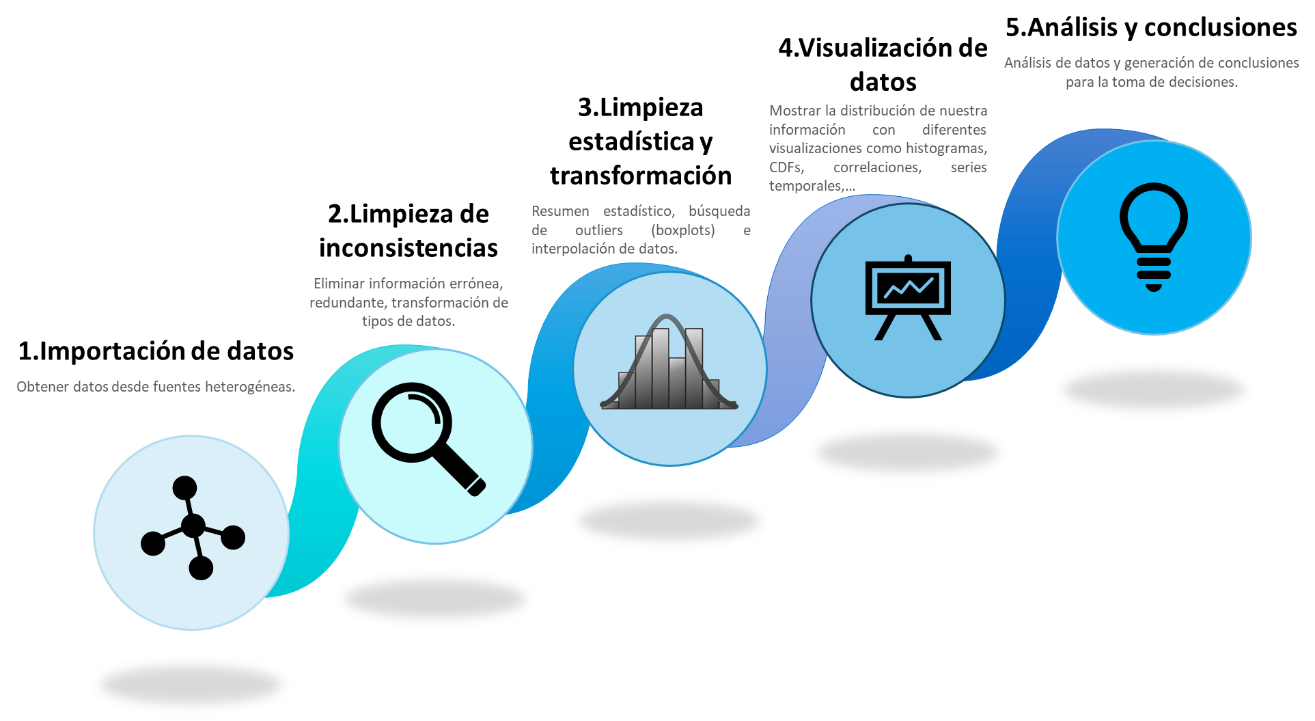
* Renta per cápita:

Opción sencilla: Utilice la columna "GDP per cápita" del dataframe importado anteriormente PIB vs País, por tanto así no tendría que realizar ninguna importación adicional.

Opción avanzada (precisa de limpieza avanzada con el método split):

<https://en.wikipedia.org/wiki/List_of_countries_by_average_wage>

En base a estas fuentes de información, como Data Scientist debe seguir el siguiente flujograma que ha aprendido en este curso y con ello dar respuesta a las preguntas que se plantean posteriormente:



***\*NOTA: Si en algún paso se queda bloqueado, no se preocupe, puede consultar la solución proporcionada y el script de solución para poder avanzar.***

**PASO 1: Importación de datos**

Debe importar las 6 fuentes de datos en diferentes dataframes nombrándolos como:

* Ventas 🡪 df\_ventas
* Clientes 🡪 df\_clientes
* Productos 🡪 df\_productos
* Población / PIB 🡪 df\_pob\_pib
* Renta per cápita 🡪 df\_rpc

**PREGUNTA 1**: ¿Cuáles son las dimensiones (registros x columnas) de cada dataframe?

Resultado:

**PREGUNTA 2**: ¿Existen valores NaN en nuestras fuentes?

Resultado:

**PASO 2: Limpieza de inconsistencias**

Ahora debe realizar la limpieza de aquéllos registros inconsistentes, como norma en su empresa se declara un registro como inconsistente si no existe una fecha de compra (está vacía).

Realice la limpieza en base a la suposición anterior (Ayuda: Puede aplicar el método isnull() sobre la columna de interés para filtrar la información y verificar si hay registros vacíos, para eliminarlos puede aplicar el método notnull() para quedarse con los registros no vacíos).

**PREGUNTA 3**: ¿Cuántos registros se han eliminado de df\_ventas?

Resultado:

**PREGUNTA 4**: ¿Existen registros duplicados en nuestros datos? Si es así limpie la información de esta información redundante.

Resultado:

Varios campos de nuestros dataframes contienen datos numéricos, pero en algunos casos, puede que se hayan interpretado como strings debido al separador decimal “,” en lugar de “.” o bien contienen el símbolo “$” (utilice el método info() para ver el formato de cada columna).

**PREGUNTA 5**: Transforme las columnas implicadas para que pueda modificar el tipo de datos de los campos que deben ser numéricos (Recordatorio: para reemplazar puede usar el método str.replace, para convertir a numérico el método pd.to\_numeric). ¿Qué campos ha tenido que modificar?

Resultado:

**PASO 3: Limpieza estadística**

Es el momento de analizar nuestros datos desde el punto de vista estadístico, en primer lugar, visualice los diferentes dataframes para obtener los principales datos estadísticos (Recordatorio: método describe).

**PREGUNTA 6**: Cree una columna en df\_ventas (Importe) para calcular el importe total de venta de cada pedido (Precio unitario x cantidad). ¿Cuál es la mediana del importe de nuestras ventas?

Resultado

**PREGUNTA 7**: ¿Qué cantidad de unidades de productos no se superan en el 75% de los pedidos de sus clientes?

Resultado:

Nuestra empresa ha decidido que la cantidad máxima de unidades de pedido que puede tramitar a un cliente es de 13 unidades, por tanto, si en nuestros datos existen registros con número de unidades mayor, es un outlier y es un dato no válido para los análisis posteriores.

**PREGUNTA 8**: ¿Cuántos pedidos se han realizado con un número de unidades mayor que 13?

Resultado:

**PREGUNTA 9**: Realiza la limpieza del dataframe df\_ventas de tal manera que descartemos los registros con unidades > 13. ¿Cuáles son las nuevas dimensiones del dataframe?

Resultado:

**PREGUNTA 10**: En algunos pedidos, no está registrada la “Fecha de envío”, por tanto se deben completar esos registros como la “Fecha compra” + mediana de la diferencia entre “Fecha de envio” - “Fecha compra” del resto de pedidos (debe crear una columna auxiliar “Dif\_env\_comp” con la diferencia para obtener la mediana, después utilice el método fillna() sobre “Fecha envío” de tal manera que complete los vacíos como “Fecha compra” + Mediana\_dif)

Una vez tenga completada la “Fecha de envío”, recalcule la columna “Dif\_env\_comp”.

Resultado:

**PASO 4: Visualización de datos**

Este paso se explicará en detalle en los siguientes bloques, pero no forma parte de este caso práctico.

**PASO 5: Análisis y conclusiones**

En base a la información consolidada de la que ya disponemos, como data scientist utilizando Pandas nos gustaría tener una tabla maestra que agregara las diferentes tablas y realizar sus análisis.

En primer lugar, se deben combinar los dataframes df\_ventas, df\_prod y df\_clientes para obtener un único dataframe global denominado df\_global (utilice el método inner, puede usar merge para combinar dataframes de manera anidada o bien paso a paso, primero combinar 2 dataframes y con el resultado combinarlo con el 3er dataframe).

**PREGUNTA 11**: ¿Qué país representa el mayor volumen de nuestras ventas? (puedes usar el método .nlargest(x) tras su sentencia groupby donde x es el número de elementos a devolver en orden descendente)

Resultado:

**PREGUNTA 12**: ¿En qué año se han obtenido un mayor número de ventas? (puede insertar una función lambda dentro del argumento de groupby para obtener el año de cada fecha: df\_global["Fecha compra"].map(lambda x: x.year) )

Resultado:

**PREGUNTA 13**: Teniendo en cuenta el precio de venta unitario, las unidades vendidas y el coste unitario, nos solicitan obtener el beneficio obtenido en cada uno de los pedidos. Para ello debe crear una nueva columna “Beneficio”. ¿Cuál es la categoría de productos que supone un mayor beneficio?

Resultado:

**PREGUNTA 14**: ¿Existe correlacion entre el “Beneficio” total obtenido en un país y su renta per cápita? (debemos combinar de nuevo con df\_rpc renombrando previamente la columna “Country” para que coincida con “País” de df\_global. Se recomienda que el método de combinación sea “left” para mantener todos los registros de df\_global. La correlacion se puede obtener con el método corr())

**NOTA:** Tenga en cuenta que debe aplicar un merge con el dataframe que contenga los datos de "Renta per cápita", si está utilizando la columna "AverageMonthlyWage" y aparece "NaN" es porque en ocasiones los dataframes tienen alguna columna cuyos datos tienen algún espacio en blanco al inicio o al final que es complicado de detectar.

Todo tiene solución, si ejecutaa esta sentencia para crear una columna que verifique la longitud de la columna País, verá que es diferente la longitud de esa columna en ambos dataframes:

df\_rpc['long']=df\_rpc['País'].str.len() --> Ejemplo, la longitud de "United States" te dice que es 14, pero si haces lo mismo en df\_global, te dice que es 13, esto pasa porque en df\_rpc tienes un carácter vacío a la izquierda, para quitarlo, puedes usar el siguiente comando:

df\_rpc['País']=df\_rpc['País'].str.lstrip()

Y ya le hará el merge adecuadamente.

Resultado:

**PREGUNTA 15**: ¿Existe correlacion entre el Importe total de nuestros pedidos y la cantidad de producto que solicita el cliente?

Resultado:

**PREGUNTA 16**: Intentando analizar a la población de los diferentes países, existe alguna correlación entre el PIB de un país, su población y la renta per cápita? (debe renombrar la columna “Country” de df\_pob\_pib antes del merge)

Resultado:

**PREGUNTA 17**: La empresa se está planteando abrir nuevas plantas de producción en aquéllas ubicaciones donde el promedio de la diferencia entre “Fecha envío” y “Fecha compra” es más elevada para reducir los tiempos. ¿En qué países elegiría construir estas nuevas plantas de producción? (Para que pueda calcular ese ese promedio con groupby, se precisa una columna numérica en lugar de tipo “Date” como es actualmente “Dif\_env\_comp”, puede crear una columna auxiliar que calcule los días en valor numérico de esa diferencia con el método to\_numeric, tenga en cuenta que retornada el valor en ns de esa diferencia temporal, por lo tanto para obtener los días tendrá que dividir por (1000000000\*3600\*24).

Resultado:

**PREGUNTA 18**: Como resultado final, el CEO de su compañía necesita disponer de un reporte en formato Excel con el beneficio de cada categoría en cada uno de los países. Exporta el resultado a un fichero con nombre “Reporte Beneficios” y pestaña “Beneficios\_cat\_pais”. (Puede convertir un objeto grouped a dataframe con el método to\_frame()

Resultado: