01-revisando_datos_regionales

July 22, 2020

1 Revisión de consistencia de datos

1.1 Configuración inicial del ambiente

Ejecuto el primer script fuente (`src/00-setup.py`) para obtener y/o refrescar los datos.

[1]:
```
%run -i src/00-setup.py
```

```
[ INF - 2020-07-22 22:06:42 (jtapia) ] Configurando el entorno.
[ INF - 2020-07-22 22:06:42 (jtapia) ] Obteniendo data del origen remoto…
[ INF - 2020-07-22 22:06:42 (jtapia) ] Clonando el repositorio de Git con la
data original…
[ ERR - 2020-07-22 22:06:42 (jtapia) ] El directorio ya existe.
[ INF - 2020-07-22 22:06:42 (jtapia) ] Refrescando los datos…
[ INF - 2020-07-22 22:06:44 (jtapia) ] Datos refrescados.
```

1.2 Configuración del notebook

Importo las librerias de Python que voy a usar: `os`, `numpy` y `pandas`.

[2]:
```
import os
import numpy as np
import pandas as pd
```

Limpieza

[3]:
```
del Repo, current_wd, git_file, git_url, new_wd
```

1.3 Definición de funciones y variables

Defino alguna variables para manejar de manera más fácil el entorno, para ir directo al resultado de los productos de datos. También defino alguna funciones auxiliares para labores repetitivas.

[4]:
```
folder = os.path.join(os.path.abspath(""), "data", "output")
prod1_folder = os.path.join(folder, "producto1")
prod2_folder = os.path.join(folder, "producto2")
prod4_folder = os.path.join(folder, "producto4")
prod5_folder = os.path.join(folder, "producto5")
prod6_folder = os.path.join(folder, "producto6", "bulk")
```

```
prod11_folder = os.path.join(folder, "producto11", "bulk")
```

[5]:
```
def get_sample_comp_1(input: pd.DataFrame, prod_num: int, sum: bool) -> None:
    query_str = "base_dt == '2020-07-10' and (cod_region == 15)"
    columns_select = ["base_dt", "cod_region", "casos_total"]
    print_str = "Datos del producto {0}"
    if sum == True:
        tmp_df = input.query(query_str)[columns_select].
↪groupby("base_dt")["casos_total"].sum()
    else:
        tmp_df = input.query(query_str)[columns_select]
    print(print_str.format(prod_num), "\n---")
    print(tmp_df, "\n\n")
    del tmp_df
    return None
```

[6]:
```
def get_sample_comp_2(input: pd.DataFrame, prod_num: int, sum: bool) -> None:
    if prod_num == 4:
        query_str = "base_dt == '2020-07-10' and cod_region > 0"
    else:
        query_str = "base_dt == '2020-07-10'"
    columns_select = ["base_dt", "casos_total"]
    print_str = "Datos del producto {0}"
    if sum == True:
        tmp_df = input.query(query_str)[columns_select].
↪groupby("base_dt")["casos_total"].sum()
    else:
        tmp_df = input.query(query_str)[columns_select]
    print(print_str.format(prod_num), "\n---")
    print(tmp_df, "\n\n")
    del tmp_df
    return None
```

[7]:
```
def parse_numbers_to_str(value: str) -> float:
    if value == "-":
        return None
    else:
        return float(value)
```

[8]:
```
def accum_data(input_str: str, accum_df: pd.DataFrame,
               base_folder: str, column_change: dict) -> pd.DataFrame:
    df_tmp = pd.read_csv(os.path.join(base_folder, input_str))
    date_file = input_str[:10]
    df_tmp["base_dt"] = date_file
    df_tmp.rename(columns = column_change, inplace = True)
    accum_df = pd.concat([accum_df, df_tmp])
    del df_tmp
```

```
    return accum_df
```

1.4 Datos del producto 1 (casos totales por comuna, incremental)

El producto 1 consta de sólo un archvo, por lo que se lee el CSV y se le cambian los nombres a las columnas para obtener consistencia.

[9]:
```
column_rename = {"Region": "region",
                 "Codigo region": "cod_region",
                 "Comuna": "comuna",
                 "Codigo comuna": "cod_comuna",
                 "Poblacion": "poblacion",
                 "Fecha": "base_dt",
                 "Casos confirmados": "casos_total"}

df_prod1 = pd.read_csv(os.path.join(prod1_folder, "Covid-19_std.csv"))
df_prod1.rename(columns = column_rename, inplace = True)
```

[10]:
```
df_prod1.head()
```

[10]:

	region	cod_region	comuna	cod_comuna	poblacion	base_dt	casos_total
0	Arica y Parinacota	15	Arica	15101.0	247552.0	2020-03-30	6.0
1	Arica y Parinacota	15	Camarones	15102.0	1233.0	2020-03-30	0.0
2	Arica y Parinacota	15	General Lagos	15202.0	810.0	2020-03-30	0.0
3	Arica y Parinacota	15	Putre	15201.0	2515.0	2020-03-30	0.0
4	Arica y Parinacota	15	Desconocido Arica y Parinacota	NaN	NaN	2020-03-30	NaN

[11]:
```
df_prod1.shape
```

[11]:
```
(12670, 7)
```

[12]:
```
df_prod1.describe()
```

[12]:

	cod_region	cod_comuna	poblacion	casos_total
count	12670.000000	12110.000000	12110.000000	12251.000000
mean	8.784530	9034.997110	56237.890173	335.377030
std	3.879263	3812.783311	88821.006783	1339.836198
min	1.000000	1101.000000	137.000000	0.000000
25%	6.000000	6109.000000	9546.000000	1.000000
50%	8.000000	8313.500000	19770.000000	15.000000

```
75%        13.000000  13103.000000   56058.000000     87.000000
max        16.000000  16305.000000  645909.000000  21568.000000
```

[13]:
```
df_prod1.base_dt.nunique()
```

[13]:
```
35
```

1.5 Datos del producto 2 (casos totales por comuna)

Los datos del producto 2 están separados en varios archivos (uno por fecha), por lo que se crea un DataFrame vacío, se obtiene una lista con todos los archivos CSV que existen en la carpeta, se leen todos, se les cambia el nombre de las columnas, y finalmente se concatenan al acumulador. A los datos intermedios se les agrega una columna nueva, `base_dt`, que refleja la fecha del informe leído.

[14]:
```
column_names = ["region", "cod_region", "comuna", "cod_comuna", "poblacion",␣
 ↪"casos_total", "base_dt"]
column_rename = {"Region": "region",
                 "Codigo region": "cod_region",
                 "Comuna": "comuna",
                 "Codigo comuna": "cod_comuna",
                 "Poblacion": "poblacion",
                 "Casos Confirmados": "casos_total"}

df_prod2 = pd.DataFrame(columns = column_names)
prod2_files = [csv for csv in os.listdir(prod2_folder) if "csv" in csv]

for csv in prod2_files:
    df_prod2 = accum_data(input_str = csv,
                          accum_df = df_prod2,
                          base_folder = prod2_folder,
                          column_change = column_rename)
```

[15]:
```
df_prod2.head()
```

[15]:

	region	cod_region	comuna	cod_comuna	poblacion	casos_total	base_dt
0	Arica y Parinacota	15	Arica	15101.0	247552.0	1046.0	2020-06-12
1	Arica y Parinacota	15	Camarones	15102.0	1233.0	0.0	2020-06-12
2	Arica y Parinacota	15	General Lagos	15202.0	810.0	0.0	2020-06-12
3	Arica y Parinacota	15	Putre	15201.0	2515.0	2.0	2020-06-12
4	Arica y Parinacota	15	Desconocido Arica y Parinacota	NaN	NaN	NaN	2020-06-12

[16]:
```
df_prod2.shape
```

[16]:
```
(12670, 7)
```

[17]:
```
df_prod2.describe()
```

[17]:

	cod_comuna	poblacion	casos_total
count	12110.000000	12110.000000	12251.000000
mean	9034.997110	56237.890173	335.377030
std	3812.783311	88821.006783	1339.836198
min	1101.000000	137.000000	0.000000
25%	6109.000000	9546.000000	1.000000
50%	8313.500000	19770.000000	15.000000
75%	13103.000000	56058.000000	87.000000
max	16305.000000	645909.000000	21568.000000

[18]:
```
df_prod2.base_dt.nunique()
```

[18]:
```
35
```

1.6 Datos del producto 4 (casos totales por región)

Los datos del producto 4 están separados en varios archivos (uno por fecha), por lo que se crea un DataFrame vacío, se obtiene una lista con todos los archivos CSV que existen en la carpeta, se leen todos, se les cambia el nombre de las columnas, y finalmente se concatenan al acumulador. A los datos intermedios se les agregan columnas nuevas: `base_dt`, que refleja la fecha del informe leído, `cod_region`, que mapea el nombre de la región a su número (para evitar errores de manipulación).

[19]:
```
column_names = ["region", "casos_nuevo", "casos_nuevo_total",␣
↪"casos_nuevo_sintomas", "casos_nuevo_nosintomas",
                "casos_nuevo_nonotif", "casos_activo_confirm",␣
↪"casos_activo_prob", "casos_probables_accum",
                "casos_confirm_recup", "casos_total", "casos_recuperado",␣
↪"muertes", "incremento", "tpm",
                "perc_total", "cod_region"]
column_rename = {'Region': "region",
                    "Region": "region",
                    'Región': "region",
                    'Casos  nuevos': "casos_nuevo",
                    'Casos nuevos': "casos_nuevo",
                    ' Casos nuevos': "casos_nuevo",
                    ' Casos nuevos': "casos_nuevo",
                    'Casos nuevos totales': "casos_nuevo_total",
                    "Casos nuevos totales": "casos_nuevo_total",
                    'Casos  nuevos  totales': "casos_nuevo_total",
                    'Casos nuevos totales ': "casos_nuevo_total",
                    'Casos nuevos con sintomas': "casos_nuevo_sintomas",
```

```
                    "Casos nuevos con sintomas": "casos_nuevo_sintomas",
                    'Casos  nuevos  con  sintomas': "casos_nuevo_sintomas",
                    'Casos nuevos con sintomas ': "casos_nuevo_sintomas",
                    'Casos  nuevos  sin  sintomas': "casos_nuevo_nosintomas",
                    'Casos nuevos sin sintomas* ': "caso_nuevo_nosintomas",
                    'Casos nuevos sin sintomas*': "caso_nuevo_nosintomas",
                    "Casos nuevos sin sintomas*": "caso_nuevo_nosintomas",
                    'Casos  nuevos  sin  sintomas*': "caso_nuevo_nosintomas",
                    'Casos nuevos sin notificar': "caso_nuevo_nonotif",
                    "Casos nuevos sin notificar": "caso_nuevo_nonotif",
                    'Casos activos confirmados': "casos_activo_confirm",
                    "Casos activos confirmados": "casos_activo_confirm",
                    'Casos activos probables': "casos_activo_prob",
                    "Casos activos probables": "casos_activo_prob",
                    'Casos activos probables ': "casos_activo_prob",
                    'Casos probables acumulados ': "casos_probables_accum",
                    "Casos probables acumulados ": "casos_probables_accum",
                    'Casos confirmados recuperados': "casos_confirm_recup",
                    "Casos confirmados recuperados": "casos_confirm_recup",
                    'Casos  totales': "casos_total",
                    'Casos totales': "casos_total",
                    ' Casos totales': "casos_total",
                    'Casos totales acumulados': "casos_total",
                    'Casos  totales  acumulados': "casos_total",
                    "Casos totales acumulados": "casos_total",
                    'Casos totales acumulados ': "casos_total",
                    ' Casos recuperados': "casos_recuperado",
                    'Fallecidos totales': "muertes",
                    "Fallecidos totales": "muertes",
                    'Fallecidos': "muertes",
                    'Fallecidos totales ': "muertes",
                    ' Casos fallecidos': "muertes",
                    'Incremento  diario': "incremento",
                    'Tasa  *100000': "tpm",
                    '%  Casos  totales**': "perc_total",
                    '% Total': "perc_total",
                    '%  Total': "perc_total",
                    '% Total ': "perc_total"}
dict_region = {'Arica y Parinacota': 15,
                'Tarapacá': 1,
                'Antofagasta': 2,
                'Atacama': 3,
                'Coquimbo': 4,
                'Valparaíso': 5,
                'Metropolitana': 13,
                'O'Higgins': 6,
                'Maule': 7,
```

```
                'Ñuble': 16,
                'Biobío': 8,
                'Araucanía': 9,
                'Los Ríos': 14,
                'Los Lagos': 10,
                'Aysén': 11,
                'Magallanes': 12,
                'Arica y Parinacota': 15,
                'Tarapacá': 1,
                'Antofagasta': 2,
                'Atacama': 3,
                'Coquimbo': 4,
                'Valparaíso': 5,
                'Metropolitana': 13,
                "O'Higgins": 6,
                'Maule': 7,
                'Ñuble': 16,
                'Biobío': 8,
                'Araucania': 9,
                'Los Ríos': 14,
                'Los Lagos': 10,
                'Aysén': 11,
                'Magallanes': 12,
                'Total': 0,
                'Arica  y  Parinacota': 15,
                'Tarapaca': 1,
                'Valparaiso': 5,
                'Nuble': 16,
                'Biobio': 8,
                'Los  Rios': 14,
                'Los  Lagos': 10,
                'Aysen': 11,
                'Se desconoce región de origen': None,
                'O’Higgins': 6,
                'Araucanía': 9,
                'Arica y Paricota': 15,
                'Metropolita': 13,
                'Los Rios': 14,
                'Los  Ríos': 14}

df_prod4 = pd.DataFrame(columns = column_names)
prod4_files = [csv for csv in os.listdir(prod4_folder) if "csv" in csv]

for csv in prod4_files:
    df_prod4 = accum_data(input_str = csv,
                          accum_df = df_prod4,
                          base_folder = prod4_folder,
```

```
                              column_change = column_rename)

df_prod4["cod_region"] = df_prod4.apply(lambda fila: dict_region.get(fila.
 ↪region), axis = 1)
```

[20]:
```
df_prod4.head()
```

[20]:

	region	casos_nuevo	casos_nuevo_total	casos_nuevo_sintomas	casos_nuevo_nosintomas	casos_nuevo_nonotif	casos_activo_confirm	casos_activo_prob	casos_probables_accum	casos_confirm_recup	casos_total	casos_recuperado	muertes	incremento	tpm	perc_total	cod_region	caso_nuevo_nosintomas	caso_nuevo_nonotif	base_dt
0	Arica y Parinacota	NaN	56	52	NaN	NaN	NaN	NaN	NaN	NaN	1295	NaN	12	NaN	NaN	0.56%	15.0	4.0	0.0	2020-06-19
1	Tarapacá	NaN	126	108	NaN	NaN	NaN	NaN	NaN	NaN	4995	NaN	71	NaN	NaN	2.16%	1.0	14.0	4.0	2020-06-19
2	Antofagasta	NaN	300	256	NaN	NaN	NaN	NaN	NaN	NaN	5828	NaN	93	NaN	NaN	2.52%	2.0	33.0	11.0	2020-06-19
3	Atacama	NaN	37	25	NaN	NaN	NaN	NaN	NaN	NaN	623	NaN	0	NaN	NaN	0.27%	3.0	12.0	0.0	2020-06-19
4	Coquimbo	NaN	119	99	NaN	NaN	NaN	NaN	NaN	NaN	2195	NaN	16	NaN	NaN	0.95%	4.0	9.0	11.0	2020-06-19

[21]:
```
df_prod4.shape
```

[21]:
```
(2438, 20)
```

[22]:
```
df_prod4.describe()
```

[22]:
```
         cod_region  caso_nuevo_nosintomas  caso_nuevo_nonotif
count  2414.000000            1450.000000          617.000000
mean      8.000000              41.994483           34.534846
std       4.899995             102.123040          101.153184
min       0.000000               0.000000            0.000000
25%       4.000000               2.000000            0.000000
50%       8.000000               7.000000            2.000000
75%      12.000000              19.000000           11.000000
max      16.000000             645.000000          832.000000
```

[23]:
```
df_prod4.base_dt.nunique()
```

[23]:
```
142
```

1.7 Datos del producto 5 (totales nacionales diarios)

El producto 5 consta de sólo un archvo, por lo que se lee el CSV y se le cambian los nombres a las columnas para obtener consistencia.

[24]:
```
column_rename = {"Fecha": "base_dt",
                    "Casos totales": "casos_total",
                    "Casos nuevos con sintomas": "casos_nuevo_sintomas",
                    "Casos nuevos sin sintomas": "casos_nuevo_sinsintomas",
                    "Casos nuevos totales": "casos_nuevo_total",
                    "Casos nuevos sin notificar": "casos_nuevo_sinnotif",
                    "Casos activos": "casos_activo",
                    "Casos activos por FD": "casos_activo_fd",
                    "Casos activos por FIS": "casos_activo_fis",
                    "Casos activos cofirmados": "casos_activo_confirmado",
                    "Casos activos probables": "casos_activo_probable",
                    "Casos recuperados": "casos_recuperado",
                    "Casos recuperados por FIS": "casos_recuperado_fis",
                    "Casos recuperados por FD": "casos_recuperado_fd",
                    "Casos confirmados recuperados":␣
 ↪"casos_confirmado_recuperado",
                    "Casos probables acumulados": "casos_probable_accum",
                    "Fallecidos": "muertes"}

df_prod5 = pd.read_csv(os.path.join(prod5_folder, "TotalesNacionales_T.csv"))
df_prod5.rename(columns = column_rename, inplace = True)
```

[25]:
```
df_prod5.head()
```

[25]:
```
      base_dt  casos_nuevo_sintomas  casos_total  casos_recuperado  muertes  \
0  2020-03-02                   1.0          1.0               0.0      0.0
1  2020-03-03                   0.0          1.0               0.0      0.0
2  2020-03-04                   2.0          3.0               0.0      0.0
```

```
3  2020-03-05                  1.0          4.0               0.0      0.0
4  2020-03-06                  1.0          5.0               0.0      0.0

   casos_activo  casos_nuevo_sinsintomas  casos_nuevo_total  casos_activo_fd  \
0           1.0                      NaN                1.0              1.0
1           1.0                      NaN                0.0              1.0
2           3.0                      NaN                2.0              3.0
3           4.0                      NaN                1.0              4.0
4           5.0                      NaN                1.0              5.0

   casos_activo_fis  casos_recuperado_fis  casos_recuperado_fd  \
0               NaN                   NaN                  0.0
1               NaN                   NaN                  0.0
2               NaN                   NaN                  0.0
3               NaN                   NaN                  0.0
4               NaN                   NaN                  0.0

   casos_confirmado_recuperado  Casos activos confirmados  \
0                          NaN                        NaN
1                          NaN                        NaN
2                          NaN                        NaN
3                          NaN                        NaN
4                          NaN                        NaN

   casos_probable_accum  casos_activo_probable  casos_nuevo_sinnotif
0                   NaN                    NaN                   NaN
1                   NaN                    NaN                   NaN
2                   NaN                    NaN                   NaN
3                   NaN                    NaN                   NaN
4                   NaN                    NaN                   NaN
```

[26]:
```
df_prod5.shape
```

[26]:
```
(143, 17)
```

[27]:
```
df_prod5.describe()
```

[27]:
```
       casos_nuevo_sintomas    casos_total  casos_recuperado      muertes  \
count            143.000000     143.000000         92.000000   143.000000
mean            1842.734266  101006.664336       8647.413043  1943.783217
std             1725.688901  118919.689709      11519.219916  2676.204634
min                0.000000       1.000000          0.000000     0.000000
25%              329.000000    4965.500000         20.750000    40.000000
50%             1391.000000   31721.000000       3460.000000   335.000000
75%             3290.000000  202538.500000      12778.250000  3499.000000
max             6368.000000  336402.000000      44946.000000  8722.000000
```

```
       casos_activo  casos_nuevo_sinsintomas  casos_nuevo_total  \
count    143.000000                85.000000         143.000000
mean   17155.531469               361.129412        2131.895105
std    14796.346265               125.556835        1972.572281
min        1.000000                56.000000           0.000000
25%     4112.500000               265.000000         329.000000
50%    17261.000000               373.000000        1656.000000
75%    27112.000000               432.000000        3781.000000
max    59100.000000               645.000000        6938.000000

       casos_activo_fd  casos_activo_fis  casos_recuperado_fis  \
count       143.000000         51.000000             51.000000
mean      31339.762238      27183.941176         214062.529412
std       32823.542727       5576.058889          71156.904835
min           1.000000      18439.000000          86173.000000
25%        4112.500000      22751.000000         146248.000000
50%       17261.000000      25000.000000         228055.000000
75%       53303.500000      33369.500000         276487.500000
max      109003.000000      37307.000000         309241.000000

       casos_recuperado_fd  casos_confirmado_recuperado  \
count           143.000000                    32.000000
mean          67723.118881                261907.250000
std           93898.411326                 32992.395211
min               0.000000                200569.000000
25%             813.000000                235064.500000
50%           14125.000000                266308.000000
75%          109877.000000                289936.250000
max          294361.000000                309241.000000

       Casos activos confirmados  casos_probable_accum  casos_activo_probable  \
count                  27.000000             32.000000               32.00000
mean                28997.074074          33859.937500             7013.15625
std                  5270.685176           5477.848647             2346.73129
min                 21107.000000          21897.000000             1853.00000
25%                 24303.000000          30204.750000             5750.25000
50%                 28210.000000          35017.500000             6950.00000
75%                 34317.000000          38751.250000             9340.75000
max                 37307.000000          40056.000000            10482.00000

       casos_nuevo_sinnotif
count              5.000000
mean             130.400000
std               42.388678
min               84.000000
25%              108.000000
50%              112.000000
```

```
75%              160.000000
max              188.000000
```

[28]:
```
df_prod5.base_dt.nunique()
```

[28]:
```
143
```

1.8 Datos del producto 6 (enriquecimiento del producto 2)

El producto 6 contiene el producto 4 en un solo archivo. Las fechas deben ser manipuladas para dejarlas en el mismo formato que el resto de los datos, y los valores de casos totales son manipulados para convertirlos en números decimales.

[29]:
```
column_rename = {"Poblacion": "poblacion",
                     "Casos Confirmados": "casos_total",
                     "Fecha": "base_dt",
                     "Region ID": "cod_region",
                     "Region": "region",
                     "Provincia ID": "cod_provincia",
                     "Provincia": "provincia",
                     "Comuna ID": "cod_comuna",
                     "Comuna": "comuna",
                     "Tasa": "tasa"}

df_prod6 = pd.read_csv(os.path.join(prod6_folder, "data.csv"))
df_prod6.rename(columns = column_rename, inplace = True)
df_prod6["base_dt"] = df_prod6.apply(lambda fila: str(fila.base_dt).replace("/
 ↪", "-"), axis = 1)
df_prod6["casos_total"] = df_prod6.apply(lambda fila: parse_numbers_to_str(fila.
 ↪casos_total), axis = 1)
```

[30]:
```
df_prod6.head()
```

[30]:

	poblacion	casos_total	base_dt	cod_region	region	cod_provincia	provincia	cod_comuna	comuna	tasa
0	247552.0	525.0	2020-05-25	15.0	Arica y Parinacota	151.0	Arica	15101.0	Arica	212.07665460186143
1	247552.0	806.0	2020-06-05	15.0	Arica y Parinacota	151.0	Arica	15101.0	Arica	325.58815925542916
2	247552.0	2721.0	2020-07-13	15.0	Arica y Parinacota	151.0	Arica	15101.0	Arica	1099.1630041365047
3	247552.0	87.0	2020-04-10	15.0	Arica y Parinacota	151.0	Arica	15101.0	Arica	35.14413133402275
4	247552.0	1887.0	2020-07-01	15.0	Arica y Parinacota	151.0	Arica	15101.0	Arica	762.2640899689762

[31]:
```
df_prod6.shape
```

[31]:
```
(12671, 10)
```

[32]:
```
df_prod6.describe()
```

[32]:

	poblacion	casos_total	cod_region	cod_provincia	cod_comuna
count	12110.000000	12251.000000	12076.000000	12076.000000	12076.000000
mean	56237.890173	335.377030	8.791487	90.230788	9028.898725
std	88821.006783	1339.836198	3.845235	38.153331	3816.412242
min	137.000000	0.000000	1.000000	11.000000	1101.000000
25%	9546.000000	1.000000	6.000000	61.000000	6109.000000
50%	19770.000000	15.000000	8.000000	83.000000	8313.000000
75%	56058.000000	87.000000	13.000000	131.000000	13103.000000
max	645909.000000	21568.000000	16.000000	163.000000	16305.000000

[33]:
```
df_prod6.base_dt.nunique()
```

[33]:
```
36
```

1.9 Datos del producto 11 (enriquecimiento del producto 4)

El producto 11 contiene los datos del producto 4 en un solo archivo. Las fechas deben ser manipuladas para dejarlas en el mismo formato que el resto de los datos, y se crea una nueva columna con el código de la región.

[34]:
```
column_rename = {"Region": "region",
                     "Casostotalesacumulados": "casos_total",
                     "Casosnuevostotales": "casos_nuevo_total",
                     "Casosnuevosconsintomas": "casos_nuevo_sintomas",
                     "Casosnuevossinsintomas*": "casos_nuevo_nosintomas",
                     "Fallecidos": "muertes",
                     "%Total": "perc_total",
                     "Fecha": "base_dt",
                     "Fallecidostotales": "muertes_total",
                     "Nuevos Casos": "casos_nuevo",
                     "Casos Confirmados": "casos_confirmado",
                     "Casosnuevossinsintomas": "casos_nuevo_nosintomas2",
                     "Tasa*100000": "tpm",
                     "Incrementodiario": "incremento",
                     "Region ID": "cod_region",
                     "Poblacion": "poblacion",
                     "Tasa": "tasa"}
dict_region = {'Arica y Parinacota': 15,
                'Tarapacá': 1,
                'Antofagasta': 2,
                'Atacama': 3,
```

```
                'Coquimbo': 4,
                'Valparaíso': 5,
                'Metropolitana': 13,
                'O'Higgins': 6,
                'Maule': 7,
                'Ñuble': 16,
                'Biobío': 8,
                'Araucanía': 9,
                'Los Ríos': 14,
                'Los Lagos': 10,
                'Aysén': 11,
                'Magallanes': 12}

df_prod11 = pd.read_csv(os.path.join(prod11_folder, "producto4.csv"))
df_prod11.rename(columns = column_rename, inplace = True)
df_prod11["base_dt"] = df_prod11.apply(lambda fila: str(fila.base_dt).replace("/
 ↪", "-"), axis = 1)
df_prod11["cod_region"] = df_prod11.apply(lambda fila: dict_region.get(fila.
 ↪region), axis = 1)
```

[35]:
```
df_prod11.head()
```

[35]:

	region	casos_total	casos_nuevo_total	casos_nuevo_sintomas	casos_nuevo_nosintomas	muertes	perc_total	base_dt	muertes_total	casos_nuevo	casos_confirmado	casos_nuevo_nosintomas2	tpm	incremento	cod_region	poblacion	tasa
0	Arica y Parinacota	378.0	1.0	1.0	0.0	7	0.86%	2020-05-17	NaN	NaN	NaN	NaN	NaN	NaN	15	252110	NaN
1	Tarapacá	975.0	130.0	108.0	22.0	4	2.23%	2020-05-17	NaN	NaN	NaN	NaN	NaN	NaN	1	382773	NaN
2	Antofagasta	1505.0	76.0	60.0	16.0	13	3.44%	2020-05-17	NaN	NaN	NaN	NaN	NaN	NaN	2	691854	NaN
3	Atacama	149.0	3.0	2.0	1.0	0	0.34%	2020-05-17	NaN	NaN	NaN	NaN	NaN	NaN			
4	Coquimbo	194.0	7.0	1.0	6.0	2	0.44%	2020-05-17	NaN	NaN	NaN	NaN	NaN	NaN			

```
3           3     314709   NaN
4           4     836096   NaN
```

[36]:
```
df_prod11.shape
```

[36]:
```
(1552, 17)
```

[37]:
```
df_prod11.describe()
```

[37]:
```
         casos_total  casos_nuevo_total  casos_nuevo_sintomas  \
count     640.000000         640.000000            640.000000
mean     3681.695312         187.164062            167.212500
std     13320.003662         684.582458            625.346853
min         7.000000           0.000000              0.000000
25%       308.000000           6.000000              4.000000
50%       775.000000          18.000000             14.000000
75%      1362.750000          52.250000             43.000000
max    108462.000000        5268.000000           4797.000000

       casos_nuevo_nosintomas      muertes  muertes_total  casos_nuevo  \
count              624.000000  1552.000000      96.000000   912.000000
mean                20.062500    13.059923      87.968750    15.751096
std                 62.362691    58.355651     259.091955    47.118606
min                  0.000000     0.000000       0.000000     0.000000
25%                  1.000000     0.000000       8.000000     0.000000
50%                  4.000000     1.000000      19.000000     2.000000
75%                  9.000000     8.000000      32.500000    12.000000
max                471.000000   824.000000    1273.000000   442.000000

       casos_confirmado  casos_nuevo_nosintomas2         tpm   cod_region  \
count        912.000000                   16.000   16.000000  1552.000000
mean         275.662281                   15.625   78.364375     8.500000
std          847.994171                   62.500   96.521866     4.611258
min            0.000000                    0.000    6.520000     1.000000
25%            1.000000                    0.000   23.460000     4.750000
50%           31.500000                    0.000   46.330000     8.500000
75%          214.500000                    0.000  106.720000    12.250000
max         8300.000000                  250.000  400.310000    16.000000

          poblacion        tasa
count  1.552000e+03  912.000000
mean   1.216144e+06   24.399794
std    1.854756e+06   50.856275
min    1.072970e+05    0.000000
25%    3.657570e+05    0.289078
50%    7.639750e+05    5.305173
75%    1.043742e+06   27.282167
```

```
max    8.125072e+06  388.535675
```

[38]:
```
df_prod11.base_dt.nunique()
```

[38]:
```
97
```

Limpieza

[39]:
```
del column_names, column_rename, dict_region, folder, prod1_folder,␣
 ↪prod2_folder, prod2_files
del prod4_files, prod4_folder, prod5_folder, prod6_folder, prod11_folder
```

1.10 Comparación de totales por región (productos 1, 2, 4, 6, 11)

Para obtener una primera imagen de consistencia, se toma como fecha base `base_dt` = 2020-07-10 y se revisarán los datos de la región de Arica y Parinacota (`cod_region` = 15).

[40]:
```
get_sample_comp_1(df_prod1, 1, True)
get_sample_comp_1(df_prod2, 2, True)
get_sample_comp_1(df_prod4, 4, True)
get_sample_comp_1(df_prod6, 6, True)
get_sample_comp_1(df_prod11, 11, True)
```

```
Datos del producto 1
---
base_dt
2020-07-10    2493.0
Name: casos_total, dtype: float64


Datos del producto 2
---
base_dt
2020-07-10    2493.0
Name: casos_total, dtype: float64


Datos del producto 4
---
base_dt
2020-07-10    2322
Name: casos_total, dtype: int64


Datos del producto 6
---
base_dt
2020-07-10    2490.0
```

```
Name: casos_total, dtype: float64


Datos del producto 11
---
Series([], Name: casos_total, dtype: float64)
```

[41]:
```
get_sample_comp_1(df_prod1, 1, False)
get_sample_comp_1(df_prod2, 2, False)
get_sample_comp_1(df_prod4, 4, False)
get_sample_comp_1(df_prod6, 6, False)
get_sample_comp_1(df_prod11, 11, False)
```

```
Datos del producto 1
---
          base_dt  cod_region  casos_total
11222  2020-07-10          15       2464.0
11223  2020-07-10          15          0.0
11224  2020-07-10          15          0.0
11225  2020-07-10          15         26.0
11226  2020-07-10          15          3.0


Datos del producto 2
---
      base_dt cod_region  casos_total
0  2020-07-10         15       2464.0
1  2020-07-10         15          0.0
2  2020-07-10         15          0.0
3  2020-07-10         15         26.0
4  2020-07-10         15          3.0


Datos del producto 4
---
      base_dt  cod_region casos_total
0  2020-07-10        15.0        2322


Datos del producto 6
---
        base_dt  cod_region  casos_total
11   2020-07-10        15.0       2464.0
46   2020-07-10        15.0          0.0
81   2020-07-10        15.0          0.0
116  2020-07-10        15.0         26.0
```

```
Datos del producto 11
---
Empty DataFrame
Columns: [base_dt, cod_region, casos_total]
Index: []
```

Se observan las primeras discrepancias: * El producto 11 **no está actualizado** a la fecha más reciente (un breve estudio de los datos muestra que se dejó de actualizar a mediados de junio de 2020). * El producto 6 **no incluye los datos** de la región seleccionada con comuna desconocida. Faltaría validar que esos valores se consideren en una región distinta (es decir, validar los totales nacionales), pero una comparación por región no es posible. * El producto 1 y el producto 2 entregan los mismos valores para las condiciones elegidas. * El producto 4 **no coincide** con el producto 1/producto 2. Esto es grave, puesto que el producto 4 es el total por región, y los productos 1 y 2 son el total por región desagregado a comuna.

A continuación, se armonizarán los datos a nivel de regiones entre los productos 1, 2 y 4, para identificar las diferencias que puedan existir.

[42]:
```
grpby = ["cod_region", "base_dt"]
df_prod1_agg = df_prod1.groupby(grpby)["casos_total"].sum().reset_index()
df_prod2_agg = df_prod2.groupby(grpby)["casos_total"].sum().reset_index()
df_prod4_agg = df_prod4.groupby(grpby)["casos_total"].sum().reset_index()
```

[43]:
```
df_prod1_agg.head()
```

[43]:

	cod_region	base_dt	casos_total
0	1	2020-03-30	5.0
1	1	2020-04-01	9.0
2	1	2020-04-03	10.0
3	1	2020-04-06	18.0
4	1	2020-04-08	22.0

[44]:
```
df_prod2_agg.head()
```

[44]:

	cod_region	base_dt	casos_total
0	1	2020-03-30	5.0
1	1	2020-04-01	9.0
2	1	2020-04-03	10.0
3	1	2020-04-06	18.0
4	1	2020-04-08	22.0

[45]:
```
df_prod4_agg.head()
```

[45]:

	cod_region	base_dt	casos_total
0	0.0	2020-03-03	1.0
1	0.0	2020-03-04	3.0
2	0.0	2020-03-05	4.0
3	0.0	2020-03-06	5.0
4	0.0	2020-03-07	7.0

[46]:
```
column_rename = {"casos_total_x": "casos_prod1",
                    "casos_total_y": "casos_prod2",
                    "casos_total": "casos_prod4"}

df_armonizacion = pd.merge(left = df_prod1_agg, right = df_prod2_agg, how =␣
 ↪"outer", on = ["cod_region", "base_dt"])
df_armonizacion = pd.merge(left = df_armonizacion, right = df_prod4_agg, how =␣
 ↪"outer", on = ["cod_region", "base_dt"])
df_armonizacion.rename(columns = column_rename, inplace = True)
df_armonizacion["dif_1_2"] = df_armonizacion.casos_prod1 - df_armonizacion.
 ↪casos_prod2
df_armonizacion["dif_1_4"] = df_armonizacion.casos_prod1 - df_armonizacion.
 ↪casos_prod4
df_armonizacion["dif_2_4"] = df_armonizacion.casos_prod2 - df_armonizacion.
 ↪casos_prod4
```

[47]:
```
df_armonizacion.head()
```

[47]:

	cod_region	base_dt	casos_prod1	casos_prod2	casos_prod4	dif_1_2	\
0	1	2020-03-30	5.0	5.0	8.0	0.0	
1	1	2020-04-01	9.0	9.0	10.0	0.0	
2	1	2020-04-03	10.0	10.0	13.0	0.0	
3	1	2020-04-06	18.0	18.0	21.0	0.0	
4	1	2020-04-08	22.0	22.0	26.0	0.0	

	dif_1_4	dif_2_4
0	-3.0	-3.0
1	-1.0	-1.0
2	-3.0	-3.0
3	-3.0	-3.0
4	-4.0	-4.0

[48]:
```
df_armonizacion.describe()
```

[48]:

	cod_region	casos_prod1	casos_prod2	casos_prod4	dif_1_2	\
count	2414.000000	560.000000	560.000000	2414.000000	560.0	
mean	8.000000	7336.971429	7336.971429	11966.611433	0.0	
std	4.899995	33281.998142	33281.998142	46495.438225	0.0	
min	0.000000	0.000000	0.000000	0.000000	0.0	
25%	4.000000	167.500000	167.500000	56.000000	0.0	

```
50%       8.000000     743.500000     743.500000     652.000000       0.0
75%      12.000000    2424.750000    2424.750000    2573.000000       0.0
max      16.000000  275391.000000  275391.000000  336402.000000       0.0
```

```
            dif_1_4       dif_2_4
count    560.000000    560.000000
mean     518.419643    518.419643
std     3141.737768   3141.737768
min     -129.000000   -129.000000
25%       -6.000000     -6.000000
50%        0.000000      0.000000
75%       28.250000     28.250000
max    28731.000000  28731.000000
```

Se aprecia en la tabla superior que: * Los productos 1 y 2 son idénticos. * `count(casos_prod_1) = count(casos_prod_2)` * `min(dif_1_2) = 0`; `max(dif_1_2) = 0` * Los prductos 1 y 2 tienen menos registros que el producto 4 * `count(casos_prod_1) < count(casos_prod_4)` * Esto se debe a que los productos 1/2 contienen los informes emitidos por el MINSAL, mientras que el producto 4 tiene los datos diarios. * Entre los productos 1/2 y 4, se aprecian diferencias importantes, con una diferencia mínima de 129 casos totales, y una diferencia máxima de 28.731 casos. * `min(dif_2_4) = -129` * `max(dif_2_4) = 28731` * `median(dif_2_4) = 0`

Limpieza

[49]:
```
del column_rename, grpby, df_armonizacion, df_prod1_agg, df_prod2_agg,␣
↪df_prod4_agg
```

1.11 Comparación de totales nacionales (productos 1/2, 4, 5)

Para obtener una primera imagen de consistencia, se toma como fecha base `base_dt` = 2020-07-10.

[50]:
```
get_sample_comp_2(df_prod1, 1, True)
get_sample_comp_2(df_prod2, 2, True)
get_sample_comp_2(df_prod4, 4, True)
get_sample_comp_2(df_prod5, 5, True)
```

```
Datos del producto 1
---
base_dt
2020-07-10    346601.0
Name: casos_total, dtype: float64


Datos del producto 2
---
base_dt
2020-07-10    346601.0
Name: casos_total, dtype: float64
```

```
Datos del producto 4
---
base_dt
2020-07-10    309274
Name: casos_total, dtype: int64


Datos del producto 5
---
base_dt
2020-07-10    309274.0
Name: casos_total, dtype: float64
```

[51]:
```
get_sample_comp_2(df_prod1, 1, False)
get_sample_comp_2(df_prod2, 2, False)
get_sample_comp_2(df_prod4, 4, False)
get_sample_comp_2(df_prod5, 5, False)
```

```
Datos del producto 1
---
          base_dt  casos_total
11222  2020-07-10       2464.0
11223  2020-07-10          0.0
11224  2020-07-10          0.0
11225  2020-07-10         26.0
11226  2020-07-10          3.0
…            …           …
11579  2020-07-10          0.0
11580  2020-07-10          2.0
11581  2020-07-10          0.0
11582  2020-07-10          1.0
11583  2020-07-10         70.0

[362 rows x 2 columns]


Datos del producto 2
---
        base_dt  casos_total
0    2020-07-10       2464.0
1    2020-07-10          0.0
2    2020-07-10          0.0
3    2020-07-10         26.0
4    2020-07-10          3.0
```

```
..          …          …
357  2020-07-10          0.0
358  2020-07-10          2.0
359  2020-07-10          0.0
360  2020-07-10          1.0
361  2020-07-10         70.0

[362 rows x 2 columns]


Datos del producto 4
---
       base_dt casos_total
0   2020-07-10        2322
1   2020-07-10        6961
2   2020-07-10       11267
3   2020-07-10        1467
4   2020-07-10        3918
5   2020-07-10       14053
6   2020-07-10      235370
7   2020-07-10        8069
8   2020-07-10        6833
9   2020-07-10        2838
10  2020-07-10        8151
11  2020-07-10        3538
12  2020-07-10         758
13  2020-07-10        2146
14  2020-07-10          50
15  2020-07-10        1533


Datos del producto 5
---
        base_dt  casos_total
130  2020-07-10     309274.0
```

Se observan las discrepancias esperadas: * Productos 1 y 2 discrepan de los resultados de los productos 4 y 5 (totales regionales y total nacional). * Productos 1 y 2 son equivalentes en su composición. Se observan, también, los siguientes resultados: * Los productos 4 y 5 entregan el mismo resultado para la fecha analizada, aún cuando sus composiciones son diferentes.

A continuación, se armonizarán los valores en el total nacional, para analizar las diferencias que puedan existir. Sólo se usará el conjunto de datos del producto 2, entendiendo que los productos 1 y 2 son idénticos en su nivel agregado.

[52]:
```
grpby = ["base_dt"]
df_prod2_agg = df_prod2.groupby(grpby)["casos_total"].sum().reset_index()
df_prod4_agg = df_prod4.groupby(grpby)["casos_total"].sum().reset_index()
```

[53]:
```
df_prod2_agg.head()
```

[53]:

	base_dt	casos_total
0	2020-03-30	1937.0
1	2020-04-01	2817.0
2	2020-04-03	3515.0
3	2020-04-06	4586.0
4	2020-04-08	5294.0

[54]:
```
df_prod4_agg.head()
```

[54]:

	base_dt	casos_total
0	2020-03-03	2.0
1	2020-03-04	6.0
2	2020-03-05	8.0
3	2020-03-06	10.0
4	2020-03-07	14.0

[55]:
```
columns_select = ["base_dt", "casos_total"]
df_prod5_agg = df_prod5[columns_select]
```

[56]:
```
df_prod5_agg.head()
```

[56]:

	base_dt	casos_total
0	2020-03-02	1.0
1	2020-03-03	1.0
2	2020-03-04	3.0
3	2020-03-05	4.0
4	2020-03-06	5.0

[57]:
```
column_rename = {"casos_total_x": "casos_prod2",
                 "casos_total_y": "casos_prod4",
                 "casos_total": "casos_prod5"}

df_armonizacion = pd.merge(left = df_prod2_agg, right = df_prod4_agg, how =␣
 ↪"outer", on = ["base_dt"])
df_armonizacion = pd.merge(left = df_armonizacion, right = df_prod5_agg, how =␣
 ↪"outer", on = ["base_dt"])
df_armonizacion.rename(columns = column_rename, inplace = True)
df_armonizacion["dif_2_4"] = df_armonizacion.casos_prod2 - df_armonizacion.
 ↪casos_prod4
df_armonizacion["dif_2_5"] = df_armonizacion.casos_prod2 - df_armonizacion.
 ↪casos_prod5
```

```
df_armonizacion["dif_4_5"] = df_armonizacion.casos_prod4 - df_armonizacion.
↪casos_prod5
```

[58]:
```
df_armonizacion.head()
```

[58]:
```
      base_dt  casos_prod2  casos_prod4  casos_prod5  dif_2_4  dif_2_5  \
0  2020-03-30       1937.0       4898.0       2449.0  -2961.0   -512.0
1  2020-04-01       2817.0       6062.0       3031.0  -3245.0   -214.0
2  2020-04-03       3515.0       7474.0       3737.0  -3959.0   -222.0
3  2020-04-06       4586.0       9630.0       4815.0  -5044.0   -229.0
4  2020-04-08       5294.0      11092.0       5546.0  -5798.0   -252.0

   dif_4_5
0   2449.0
1   3031.0
2   3737.0
3   4815.0
4   5546.0
```

[59]:
```
df_armonizacion.describe()
```

[59]:
```
          casos_prod2    casos_prod4    casos_prod5        dif_2_4  \
count       35.000000     142.000000     143.000000      35.000000
mean    117391.542857  203435.936620  101006.664336 -100810.485714
std     132939.069865  238069.943565  118919.689709  105597.194732
min       1937.000000       2.000000       1.000000 -294809.000000
25%       9835.000000   10447.000000    4965.500000 -195740.000000
50%      46022.000000   66102.000000   31721.000000  -46096.000000
75%     215089.000000  423166.500000  202538.500000   -9924.000000
max     371249.000000  672804.000000  336402.000000   -2961.000000

            dif_2_5        dif_4_5
count     35.000000     142.000000
mean    8290.514286  101717.964789
std    14882.008017  119034.971680
min     -512.000000       1.000000
25%      -97.500000    5223.500000
50%      -38.000000   33051.000000
75%     9680.000000  211583.250000
max    39984.000000  336402.000000
```

Se aprecia en la tabla superior que: * No existen dos productos iguales en este caso. * El producto 4 tiene, al momento de analizar los datos, un registro menos que el producto 5. * Las medias de casos tienen grandes diferencias: * `mean(casos_prod2) = 117.391,54` * `mean(casos_prod4) = 238.069,94` * `mean(casos_prod5) = 118.919,69` * Las medianas de casos tienen grandes diferencias: * `median(casos_prod2) = 46.022` * `median(casos_prod4) = 66.102` * `median(casos_prod5) = 31.721` * Entre el producto 4 y 5, la mínima diferencia de casos es 1

caso, mientras que el máximo es de 336.402. No sólo ambos productos son diferentes, sino el producto 4 siempre contiene más casos.

Teniendo estos resultados, llama la atención la discrepancia en los números, teniendo las mismas fuentes. Un control de calidad en las agregaciones es imperativa y urgente.