ClaimServices_Services_VF

October 2, 2020

[1]:
```
# import necessaries modules:
import numpy as np
import datetime
import pandas as pd
import gc
```

0.1 Step 1: Reading data related to tblClaimService and tblService

0.1.1 Step 1.1: Reading the data related to tblClaimServices

[2]:
```
# csv file related to the tblClaimServices:
filename = 'openIMIS csv/claim_services2020.csv'

# selection of columns (the entire table has 30 columns)
cols = ['ClaimServiceID', 'ClaimID', 'ServiceID','ProdID','PolicyID',\
        'ClaimServiceStatus','RejectionReason',\
       ␣
 ↪'QtyProvided','QtyApproved','PriceAsked','PriceApproved','PriceValuated',\
       'Explanation','Justification',
       'ValidityFromReview','AuditUserIDReview']

# read the csv file
df_claim_services_raw = pd.read_csv(filename,low_memory=False,usecols=cols,\
                                    parse_dates = ['ValidityFromReview'])
df_claim_services_raw = df_claim_services_raw.iloc[:-2,:]

# add a column 'ItemServiceType' with all values equal to 'Activity␣
 ↪definition'-
# in order to make a difference between the medication and activity definitions
# entities:
df_claim_services_raw['ItemServiceType']='Activity definition'

# rename the columns in order to have similar name as the items related dataset
df_claim_services_raw.rename(columns = {'ClaimServiceID': 'ClaimItemID',\
                                        'ClaimServiceStatus':␣
 ↪'ClaimItemStatus',\
                                        'ServiceID': 'ItemID'}, inplace = True)
```

```
memStats_claim_servs = (df_claim_services_raw.memory_usage()/1024/1024).sum()
shape_claim_servs = df_claim_services_raw.shape
```

0.1.2 Step 1.2: Reading data related to tblServices

[3]:
```
# csv file related to the tblServices:
filename = 'openIMIS csv/services2020.csv'

# selection of columns
cols = ['ServiceID', 'ServCode', 'ServName', 'ServType','ServLevel', \
        'ServPrice', 'ServCareType', 'ServFrequency', 'ServPatCat',\
        'ServiceUUID','ValidityFrom','ValidityTo']

# read the csv file
df_services_raw = pd.read_csv(filename,low_memory=False,usecols=cols,
                              parse_dates = ['ValidityFrom','ValidityTo'])
df_services_raw = df_services_raw.iloc[:-2,:]

# rename the columns in order to have similar name as the items related dataset
df_services_raw.rename(columns = {'ServiceID': 'ItemID',
                                  'ServCode': 'ItemCode',
                                  'ServName': 'ItemName',
                                  'ServType': 'ItemType',
                                  'ServLevel': 'ItemLevel',
                                  'ServPrice': 'ItemPrice',
                                  'ServCareType':'ItemCareType',
                                  'ServFrequency': 'ItemFrequency',
                                  'ServPatCat': 'ItemPatCat',
                                  'ServiceUUID': 'ItemUUID',
                                  'ValidityFrom': 'ItemValidityFrom',
                                  'ValidityTo': 'ItemValidityTo'}, inplace = True)
df_services_raw['ItemID'] = df_services_raw['ItemID'].astype(float)

memStats_services = (df_services_raw.memory_usage()/1024/1024).sum()
shape_services = df_services_raw.shape
```

0.1.3 Step 1.3: Concatention of the tblClaimServices with tblServices (based on ItemID column)

[4]:
```
df_claim_services_raw = pd.
 ↪merge(df_claim_services_raw,df_services_raw,on='ItemID')

memStats_claim_servs_c = (df_claim_services_raw.memory_usage()/1024/1024).sum()
shape_claim_servs_c = df_claim_services_raw.shape
```

[5]:
```
# deleting dataframes no longer necessary
del [df_services_raw]
df_services_raw=pd.DataFrame()
gc.collect()
```

[5]: 40

0.2 Step 2: Reading the claim related file

- this is an already concatenated file with all the fields related to other tables (tblClaimAdmins, tblHF, tblLocations, tblInsuree, tblFamilies, tblDiagnosis)

[6]:
```
# open the concatenated file related to claims
df_claims_raw=pd.read_pickle('openIMIS csv/ClaimsPlus2020_sel.pkl')

memStats_claims = (df_claims_raw.memory_usage()/1024/1024).sum()
shape_claims = df_claims_raw.shape
```

0.3 Step 3: Merge all dataframes:

[7]:
```
# merge all the previous tables
df_citems_sel = pd.merge(df_claim_services_raw,df_claims_raw,on='ClaimID')
```

[8]:
```
# Verify if item valid
cond1 = (df_citems_sel['DateFrom']>df_citems_sel['ItemValidityFrom'])&\
(df_citems_sel['ItemValidityTo'].isnull())
cond2 = (df_citems_sel['DateFrom']>df_citems_sel['ItemValidityFrom'])&\
(df_citems_sel['ItemValidityTo'].notnull())&\
(df_citems_sel['DateTo']>df_citems_sel['ItemValidityTo'])

# create a list of Valid/Not Valid items
validcond = cond1|cond2

# create a new column
df_citems_sel.loc[validcond,('ValidItem')] = 1
df_citems_sel.loc[~validcond,('ValidItem')] = 0
```

[9]:
```
# Verify if ClaimAdmin valid
cond1 = (df_citems_sel['DateFrom']>df_citems_sel['ClaimAdminValidityFrom'])&\
(df_citems_sel['ClaimAdminValidityTo'].isnull())
cond2 = (df_citems_sel['DateFrom']>df_citems_sel['ClaimAdminValidityFrom'])&\
(df_citems_sel['ClaimAdminValidityTo'].notnull())&\
(df_citems_sel['DateTo']>df_citems_sel['ClaimAdminValidityTo'])

# create a list of Valid/Not Valid items
validcond = cond1|cond2

# create a new column
```

```
df_citems_sel.loc[validcond,('ValidClaimAdmin')] = 1
df_citems_sel.loc[~validcond,('ValidClaimAdmin')] = 0
```

[10]:
```
# Verify if Insuree valid
cond1 = (df_citems_sel['DateFrom']>df_citems_sel['InsureeValidityFrom'])&\
(df_citems_sel['InsureeValidityTo'].isnull())
cond2 = (df_citems_sel['DateFrom']>df_citems_sel['ClaimAdminValidityFrom'])&\
(df_citems_sel['InsureeValidityTo'].notnull())&\
(df_citems_sel['DateTo']>df_citems_sel['InsureeValidityTo'])

# create a list of Valid/Not Valid items
validcond = cond1|cond2

# create a new column
df_citems_sel.loc[validcond,('ValidInsuree')] = 1
df_citems_sel.loc[~validcond,('ValidInsuree')] = 0
```

[11]:
```
# Verify if Family valid
cond1 = (df_citems_sel['DateFrom']>df_citems_sel['FamilyValidityFrom'])&\
(df_citems_sel['FamilyValidityTo'].isnull())
cond2 = (df_citems_sel['DateFrom']>df_citems_sel['FamilyValidityFrom'])&\
(df_citems_sel['FamilyValidityTo'].notnull())&\
(df_citems_sel['DateTo']>df_citems_sel['FamilyValidityTo'])

# create a list of Valid/Not Valid items
validitem = cond1|cond2

# create a new column
df_citems_sel.loc[validitem,('ValidFamily')] = 1
df_citems_sel.loc[~validitem,('ValidFamily')] = 0
```

[12]:
```
# Verify if Location valid
cond1 = (df_citems_sel['DateFrom']>df_citems_sel['LocationValidityFrom'])&\
(df_citems_sel['LocationValidityTo'].isnull())
cond2 = (df_citems_sel['DateFrom']>df_citems_sel['LocationValidityFrom'])&\
(df_citems_sel['LocationValidityTo'].notnull())&\
(df_citems_sel['DateTo']>df_citems_sel['LocationValidityTo'])

# create a list of Valid/Not Valid items
validitem = cond1|cond2

# create a new column
df_citems_sel.loc[validitem,('ValidLocation')] = 1
df_citems_sel.loc[~validitem,('ValidLocation')] = 0
```

[13]:
```
# Verify if ICD valid
cond1 = (df_citems_sel['DateFrom']>df_citems_sel['ICDValidityFrom'])&\
(df_citems_sel['ICDValidityTo'].isnull())
cond2 = (df_citems_sel['DateFrom']>df_citems_sel['ICDValidityFrom'])&\
```

```
(df_citems_sel['ICDValidityTo'].notnull())&\
(df_citems_sel['DateTo']>df_citems_sel['ICDValidityTo'])

# create a list of Valid/Not Valid items
validitem = cond1|cond2

# create a new column
df_citems_sel.loc[validitem,('ValidICD')] = 1
df_citems_sel.loc[~validitem,('ValidICD')] = 0
```

[14]:
```
df_citems_sel.drop(['ClaimAdminValidityFrom','ClaimAdminValidityTo',\
                    'InsureeValidityFrom','InsureeValidityTo',\
                    'FamilyValidityFrom','FamilyValidityTo',\
                    'LocationValidityFrom','LocationValidityTo',\
                    'ICDValidityFrom','ICDValidityTo'], axis=1, inplace=True)
```

[17]:
```
df_citems_sel.drop(['ItemValidityFrom','ItemValidityTo'], axis=1, inplace=True)
```

[18]:
```
memStats = (df_citems_sel.memory_usage()/1024/1024).sum()
shape_ciems_sel = df_citems_sel.shape
```

[19]:
```
# save the results:
df_citems_sel.to_pickle('openIMIS csv/ClaimServices_Services2020_sel.pkl')
#df_claim_services_raw.to_csv('openIMIS csv/ClaimServices_Services2020_sel.
 ↪csv')
```

0.4 Summary

[20]:
```
# Summary concerning the concatenation
print(f'''Summary of the concatenation process:
- tblClaimServices has : {shape_claim_servs[0]} rows; {shape_claim_servs[1]}␣
 ↪columns; {round(memStats_claim_servs,2)} MB memory consumption;
- tblServices has : {shape_services[0]} rows; {shape_services[1]} columns;␣
 ↪{round(memStats_services,2)} MB memory consumption;
- Concatenation of tblClaimServices and tblServices has :␣
 ↪{shape_claim_servs_c[0]} rows;\
{shape_claim_servs_c[1]} columns; {round(memStats_claim_servs_c,2)} MB memory␣
 ↪consumption;
- tblClaims has : {shape_claims[0]} rows; {shape_claims[1]} columns;␣
 ↪{round(memStats_claims,2)} MB memory consumption;
- Concatenation of previous tables has : {shape_ciems_sel[0]} rows;\
{shape_ciems_sel[1]} columns; {round(memStats,2)} MB memory consumption;
''')
```

```
Summary of the concatenation process:
- tblClaimServices has : 16655364 rows; 17 columns; 2160.2 MB memory
consumption;
- tblServices has : 1868 rows; 12 columns; 0.17 MB memory consumption;
```

- Concatenation of tblClaimServices and tblServices has : 16655364 rows;28 columns; 3685.04 MB memory consumption;
- tblClaims has : 5953640 rows; 61 columns; 2816.21 MB memory consumption;
- Concatenation of previous tables has : 16600273 rows;82 columns; 10511.95 MB memory consumption;