```
import polars as pl
import pandas as pd
from tableone import TableOne
```

# 1. Table 1 descriptive des caractéristiques de la population étudiée à l'inclusion (baseline)

## 1.1 Importation des dataset et concatination avec polars (Je pense que je prefere polars par rapport a pandas)

```
## DM : Demographie
# Variable demo :
    # GENDER (form DM )
  # AGE (form DM )
  # ARM (form DM )
  # MARTIAL ( form SC )
  # EDUCYRS ( form SC )
  # EMPLOYMENT (form SC )
  # BMI (form VS)
dm1 = pl.read_csv("ascii-data-files-nida-ctn-0001-20251024/dm.csv")   # Démographie
dm2 =pl.read_csv("ascii-data-files-nida-ctn-0002-20251027/dm.csv")

sc1=pl.read_csv("ascii-data-files-nida-ctn-0001-20251024/sc.csv")
sc2=pl.read_csv("ascii-data-files-nida-ctn-0002-20251027/sc.csv")

#------------------------------------------------------------------------------------------

# Variable signe vitauxx :
  # DBP : Pression diastolique (mmHg)
  # SBP : Pression systolique (mmHg)
  # HEIGHT : taille
  # WEIGHT : poids
  # PULSE : fréquence cardiaque (heart rate)
  # RESP : Frequence respiratoire (/min)
  # TEMP : Temperature corporelle (F)

vs1 = pl.read_csv("ascii-data-files-nida-ctn-0001-20251024/vs.csv")   # Signe vitaux
vs2 =pl.read_csv("ascii-data-files-nida-ctn-0002-20251027/vs.csv")

#------------------------------------------------------------------------------------------

# Variable antecedant medicaux :
  # MHTERM : MEDICAL CONDITION
  # MHOCCUR : PAST HISTORY ( YES / NO )

am1 = pl.read_csv("ascii-data-files-nida-ctn-0001-20251024/mh.csv")   # Antecedent médicaux
am2 =pl.read_csv("ascii-data-files-nida-ctn-0002-20251027/mh.csv")

#------------------------------------------------------------------------------------------

# Variable laboratoire :



lb1 = pl.read_csv("ascii-data-files-nida-ctn-0001-20251024/lb.csv")   # Laboratory
lb2 =pl.read_csv("ascii-data-files-nida-ctn-0002-20251027/lb.csv")
```

```
# Concaténation demographie , signe vitaux , laboratoire , historique medical
DM = pl.concat([dm1, dm2], how="vertical")
SC = pl.concat([sc1, sc2], how="vertical")
VS = pl.concat([vs1, vs2], how="vertical")
LB = pl.concat([lb1, lb2], how="vertical")
AM = pl.concat([am1, am2], how="vertical")
```

## 1.2 Variable Demographique

```
DM.columns
```

```
['STUDYID',
 'DOMAIN',
 'USUBJID',
 'EPOCH',
 'VISIT',
 'VISITNUM',
 'RFSTDTC',
 'RFENDTC',
 'SITEID',
 'BRTHDTC',
 'AGE',
 'AGEU',
 'SEX',
 'RACE',
 'ETHNIC',
 'ARMCD',
 'ARM',
 'COUNTRY',
 'DMDTC',
 'DMDY']
```

```
# Liste des variables representatif des la démographie et identifiant du patient
dm_variables=["USUBJID","ARMCD","AGE","SEX","RACE"]
DM0=DM.select(pl.col(dm_variables))
DM0.columns
```

```
['USUBJID', 'ARMCD', 'AGE', 'SEX', 'RACE']
```

```
DM0.shape
```

```
(411, 5)
```

image.png

## Nous avions bien 411 sujet dans l'etude au baseline (Voir code flowchart)

```
DM0.sample(n=3)
```

shape: (3, 5)

| USUBJID | ARMCD | AGE | SEX | RACE |
|---|---|---|---|---|
| str | str | str | str | str |
| "02_021976" | "BUPNAL" | "50.726899384" | "F" | "BLACK, AFRICAN AMERICAN, OR NE… |
| "02_056168" | "BUPNAL" | "42.420260096" | "F" | "SPANISH, HISPANIC, OR LATINO" |
| "01_005053" | "SCRFAIL" | " " | "M" | "SPANISH, HISPANIC, OR LATINO" |

## Recuperer le information d education , martial et emploie depuis la table SC et l ajouter à notre table DM0 (Demographique)

```
variableSC_DM = ["EDUCYRS", "MARITAL", "EMPLOY30"]
SC.select("SCTESTCD").unique().sort("SCTESTCD")
```

shape: (5, 1)

| SCTESTCD |
|---|
| str |
| "ALLINCL" |
| "EDUCYRS" |
| "EMPLOY30" |
| "EMPLOY3Y" |
| "MARITAL" |

```
SC=SC.filter(pl.col("SCTESTCD").is_in(variableSC_DM))
```

```
SC
```

shape: (1_214, 11)

| STUDYID<br>str | DOMAIN<br>str | USUBJID<br>str | SCSEQ<br>i64 | SCTESTCD<br>str | SCTEST<br>str | SCORRES<br>str | SCSTRESC<br>str | SCSTRESN<br>str | SCDTC<br>i64 | SCDY<br>str |
|---:|---:|---:|---:|---:|---:|---:|---:|---:|---:|---:|
| "NIDA-CTN-0001" | "SC" | "01_000579" | 9 | "EDUCYRS" | "EDUCATION COMPLETED" | "14" | "14" | "14" | 2001 | "-1" |
| "NIDA-CTN-0001" | "SC" | "01_000579" | 11 | "EMPLOY30" | "USUAL EMPLOYMENT PATTERN: PAST… | "UNEMPLOYED" | "UNEMPLOYED" | " " | 2001 | "-1" |
| "NIDA-CTN-0001" | "SC" | "01_000579" | 12 | "MARITAL" | "MARITAL STATUS" | "DIVORCED" | "DIVORCED" | " " | 2001 | "-1" |
| "NIDA-CTN-0001" | "SC" | "01_001362" | 9 | "EDUCYRS" | "EDUCATION COMPLETED" | "13" | "13" | "13" | 2001 | "1" |
| "NIDA-CTN-0001" | "SC" | "01_001362" | 11 | "EMPLOY30" | "USUAL EMPLOYMENT PATTERN: PAST… | "UNEMPLOYED" | "UNEMPLOYED" | " " | 2001 | "1" |
| … | … | … | … | … | … | … | … | … | … | … |
| "NIDA-CTN-0002" | "SC" | "02_099368" | 12 | "EMPLOY30" | "USUAL EMPLOYMENT PATTERN: PAST… | "UNEMPLOYED" | "UNEMPLOYED" | " " | 2001 | "1" |
| "NIDA-CTN-0002" | "SC" | "02_099368" | 13 | "MARITAL" | "MARITAL STATUS" | "NEVER MARRIED" | "NEVER MARRIED" | " " | 2001 | "1" |
| "NIDA-CTN- | "SC" | "02_099926" | 9 | "EDUCYRS" | "EDUCATION | "13" | "13" | "13" | 2001 | "-1" |

```
SC=SC.select(pl.col("USUBJID"),
             pl.col('SCTESTCD'),
             pl.col("SCORRES"))
```

```
SC
```

shape: (1_214, 3)

| USUBJID<br>str | SCTESTCD<br>str | SCORRES<br>str |
|---:|---:|---:|
| "01_000579" | "EDUCYRS" | "14" |
| "01_000579" | "EMPLOY30" | "UNEMPLOYED" |
| "01_000579" | "MARITAL" | "DIVORCED" |
| "01_001362" | "EDUCYRS" | "13" |
| "01_001362" | "EMPLOY30" | "UNEMPLOYED" |
| … | … | … |
| "02_099368" | "EMPLOY30" | "UNEMPLOYED" |
| "02_099368" | "MARITAL" | "NEVER MARRIED" |
| "02_099926" | "EDUCYRS" | "13" |
| "02_099926" | "EMPLOY30" | "FULL TIME (35+ HRS/WK)" |
| "02_099926" | "MARITAL" | "NEVER MARRIED" |

```
SC.select(pl.col("SCTESTCD")).unique()
```

shape: (3, 1)

| SCTESTCD<br>str |
|---:|
| "EDUCYRS" |
| "MARITAL" |
| "EMPLOY30" |

```
# Pivot : long -> wide (1 ligne par patient)
SC0=SC.pivot(
    index="USUBJID",
    columns="SCTESTCD",
    values="SCORRES",
)
```

```
C:\Users\Abdo\AppData\Local\Temp\ipykernel_8024\3014803618.py:2: DeprecationWarning: the argument `columns` for `DataFrame.p
  SC0=SC.pivot(
```

```
SC0
```

shape: (406, 4)

| USUBJID | EDUCYRS | EMPLOY30 | MARITAL |
|---|---|---|---|
| str | str | str | str |
| "01_000579" | "14" | "UNEMPLOYED" | "DIVORCED" |
| "01_001362" | "13" | "UNEMPLOYED" | "NEVER MARRIED" |
| "01_001490" | "14" | "UNEMPLOYED" | "NEVER MARRIED" |
| "01_002199" | "13" | "PART TIME (REGULAR HOURS)" | "NEVER MARRIED" |
| "01_002844" | "11" | "STUDENT" | "NEVER MARRIED" |
| … | … | … | … |
| "02_098425" | "12" | "FULL TIME (35+ HRS/WK)" | "DIVORCED" |
| "02_098689" | "12" | "UNEMPLOYED" | "LIVING WITH PARTNER/COHABITATI… |
| "02_099053" | "11" | "PART TIME (IRREGULAR DAYWORK)" | "NEVER MARRIED" |
| "02_099368" | "11" | "UNEMPLOYED" | "NEVER MARRIED" |
| "02_099926" | "13" | "FULL TIME (35+ HRS/WK)" | "NEVER MARRIED" |

```
DM0= DM0.join(SC0, on="USUBJID", how="left")
DM0.sample(n=3)
```

shape: (3, 8)

| USUBJID | ARMCD | AGE | SEX | RACE | EDUCYRS | EMPLOY30 | MARITAL |
|---|---|---|---|---|---|---|---|
| str | str | str | str | str | str | str | str |
| "02_055918" | "CLON" | "41.94934976" | "F" | "OTHER" | "9" | "FULL TIME (35+ HRS/WK)" | "WIDOWED" |
| "02_029611" | "BUPNAL" | "32.700889802" | "M" | "SPANISH, HISPANIC, OR LATINO" | "13" | "FULL TIME (35+ HRS/WK)" | "NEVER MARRIED" |

```
DM0 = DM0.with_columns([
    # Nettoyer AGE
    pl.col("AGE")
        .str.strip_chars()                        # enlever les espaces
        .replace("", None)                    # remplacer vide par null
        .cast(pl.Float64, strict=False)       # convertir, sans planter
        .alias("AGE"),

    # Nettoyer EDUCYSR
    pl.col("EDUCYRS")
        .str.strip_chars()
        .replace("", None)
        .cast(pl.Float64, strict=False)
        .alias("EDUCYRS")
])
```

```
DM0.sample(n=3)
```

shape: (3, 8)

| USUBJID | ARMCD | AGE | SEX | RACE | EDUCYRS | EMPLOY30 | MARITAL |
|---|---|---|---|---|---|---|---|
| str | str | f64 | str | str | f64 | str | str |
| "01_006180" | "BUPNAL" | 27.811088 | "M" | "SPANISH, HISPANIC, OR LATINO" | 11.0 | "UNEMPLOYED" | "LEGALLY MARRIED" |
| "02_059257" | "BUPNAL" | 37.125257 | "M" | "WHITE" | 13.0 | "UNEMPLOYED" | "DIVORCED" |
| "02_018880" | "CLON" | 23.192334 | "M" | "SPANISH, HISPANIC, OR LATINO" | 8.0 | "UNEMPLOYED" | "NEVER MARRIED" |

## Recuperer le poids et la taille dans la table VS pour creer le BMI et l ajouter à notre table DMO (Demographique)

```
mesure_anthro=["HEIGHT","WEIGHT"]
VS_BMI=VS.filter(pl.col("VSTESTCD").is_in(mesure_anthro))
```

```
VS_BMI.sample(n=3)
```

shape: (3, 19)

| STUDYID | DOMAIN | USUBJID | EPOCH | VSSEQ | VSTESTCD | VSTEST | VSCAT | VSPOS | VSORRES | VSORRESU | VSSTRESC | VSS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| str | str | str | str | i64 | str | str | str | str | f64 | str | f64 | |
| "NIDA-CTN-0002" | "VS" | "02_076025" | "SCREENING" | 2 | "WEIGHT" | "WEIGHT" | "PHYSICAL EXAMINATION FORM" | null | 178.0 | null | null | |
| "NIDA-CTN-0002" | "VS" | "02_036130" | "SCREENING" | 6 | "HEIGHT" | "HEIGHT" | "PHYSICAL EXAMINATION FORM" | null | 61.0 | "INCHES" | 61.0 | |
| "NIDA-CTN-0001" | "VS" | "01_041193" | "SCREENING" | 1 | "HEIGHT" | "HEIGHT" | "PHYSICAL EXAMINATION FORM" | null | 67.0 | "INCHES" | 67.0 | |

```
VS_BMI.select(pl.col("VSTESTCD")).unique()
```

shape: (2, 1)

| VSTESTCD |
|---|
| str |
| "HEIGHT" |
| "WEIGHT" |

```
VS_BMI=VS_BMI.select(pl.col("USUBJID"),
             pl.col('VSTESTCD'),
             pl.col("VSORRES"))
```

```
VS_BMI.sample(n=3)
```

shape: (3, 3)

| USUBJID | VSTESTCD | VSORRES |
|---|---|---|
| str | str | f64 |
| "02_091644" | "WEIGHT" | 208.0 |
| "01_081605" | "WEIGHT" | 130.0 |
| "01_018881" | "WEIGHT" | 139.0 |

```
# Pivot : long -> wide (1 ligne par patient)
VS_BMI=VS_BMI.pivot(
    index="USUBJID",
    columns="VSTESTCD",
    values="VSORRES",
    aggregate_function="mean"

)
```

```
C:\Users\Abdo\AppData\Local\Temp\ipykernel_8024\1188560740.py:2: DeprecationWarning: the argument `columns` for `DataFrame.p
  VS_BMI=VS_BMI.pivot(
```

```
VS_BMI
```

```
shape: (381, 3)
  USUBJID     HEIGHT  WEIGHT
      str        f64     f64
"01_000579"     67.0   130.0
"01_001362"     58.0   179.0
"01_001490"     72.0   155.0
"01_002844"     60.0   116.0
"01_003330"     70.0   175.0
          …        …       …
"02_098074"     62.0   132.0
"02_098425"     71.0   250.0
"02_099053"     67.0   135.0
"02_099368"     71.0   174.0
"02_099926"     69.0   165.0
```

```
VS_BMI = VS_BMI.with_columns([
    # Conversion vers kg et mètres
    (pl.col("WEIGHT") * 0.453592).alias("weight_kg"),
    (pl.col("HEIGHT") * 0.0254).alias("height_m")
])

VS_BMI = VS_BMI.with_columns([
    (pl.col("weight_kg") / (pl.col("height_m") ** 2)).alias("BMI")
])
```

```
VS_BMI
```

```
shape: (381, 6)
  USUBJID     HEIGHT  WEIGHT  weight_kg  height_m        BMI
      str        f64     f64        f64       f64        f64
"01_000579"     67.0   130.0   58.96696    1.7018  20.360653
"01_001362"     58.0   179.0  81.192968    1.4732  37.410628
"01_001490"     72.0   155.0   70.30676    1.8288  21.021546
"01_002844"     60.0   116.0  52.616672     1.524  22.654446
"01_003330"     70.0   175.0    79.3786     1.778  25.109607
          …        …       …          …         …          …
"02_098074"     62.0   132.0  59.874144    1.5748  24.142848
"02_098425"     71.0   250.0    113.398    1.8034  34.867537
"02_099053"     67.0   135.0   61.23492    1.7018  21.143755
"02_099368"     71.0   174.0  78.925008    1.8034  24.267805
"02_099926"     69.0   165.0   74.84268    1.7526  24.365971
```

```
VS_BMI = VS_BMI.drop(["HEIGHT", "WEIGHT","weight_kg","height_m"])
```

```
DM_SC= DM0.join(VS_BMI, on="USUBJID", how="left")
DM_SC.sample(n=3)
```

```
shape: (3, 9)
  USUBJID       ARMCD        AGE  SEX                    RACE  EDUCYRS              EMPLOY30     MARITAL        BMI
      str         str        f64  str                     str      f64                   str         str        f64
"02_003614"  "BUPNAL"  30.110883  "F"                 "WHITE"     12.0  "PART TIME (REGULAR      "NEVER  20.982317
                                                                                      HOURS)"    MARRIED"
"02_080565"    "CLON"  31.624914  "F"                 "WHITE"     18.0       "FULL TIME (35+      "NEVER  22.112134
                                                                                      HRS/WK)"    MARRIED"
"02_037401"    "CLON"  46.562628  "M"       "BLACK, AFRICAN       12.0       "FULL TIME (35+      "NEVER  24.432783
                                       AMERICAN, OR NE…                               HRS/WK)"    MARRIED"
```

## 1.3 Variables VS ( Signe vitaux )

```
VS.sample(n=1)
```

shape: (1, 19)

| STUDYID | DOMAIN | USUBJID | EPOCH | VSSEQ | VSTESTCD | VSTEST | VSCAT | VSPOS | VSORRES | VSORRESU | VSSTRESC | VSST |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| str | str | str | str | i64 | str | str | str | str | f64 | str | f64 | |
| "NIDA-CTN-0002" | "VS" | "02_070407" | "ACTIVE" | 27 | "TEMP" | "TEMPERATURE" | "VITAL SIGNS FORM" | "SITTING" | 99.8 | "F" | 99.8 | |

```
# Voir les moments de VISIT
VS.select(pl.col("VISITNUM")).unique()
```

shape: (18, 1)

| VISITNUM |
|---|
| i64 |
| 6 |
| 9 |
| 0 |
| 12 |
| 3 |
| … |
| 2 |
| 8 |
| 14 |
| 5 |
| 11 |

```
# Filtrer le VS pour ne rester que sur le VISIT = 0 qui represente le Baseline
VS=VS.filter(pl.col("VISITNUM")==0)
```

```
VS.shape
```

```
(3150, 19)
```

```
VS.sample(2)
```

shape: (2, 19)

| STUDYID | DOMAIN | USUBJID | EPOCH | VSSEQ | VSTESTCD | VSTEST | VSCAT | VSPOS | VSORRES | VSORRESU | VSSTF |
|---|---|---|---|---|---|---|---|---|---|---|---|
| str | str | str | str | i64 | str | str | str | str | f64 | str | |
| "NIDA-CTN-0002" | "VS" | "02_033861" | "SCREENING" | 6 | "PULSE" | "PULSE" | "CLINICAL OPIATE WITHDRAWAL SCA… | "SITTING" | 84.0 | "BEATS/MINUTE" | |
| "NIDA-CTN-0002" | "VS" | "02_038844" | "SCREENING" | 8 | "PULSE" | "PULSE" | "CLINICAL OPIATE WITHDRAWAL SCA… | "SITTING" | 80.0 | "BEATS/MINUTE" | |

```
VS.null_count()
```

shape: (1, 19)

| STUDYID | DOMAIN | USUBJID | EPOCH | VSSEQ | VSTESTCD | VSTEST | VSCAT | VSPOS | VSORRES | VSORRESU | VSSTRESC | VSSTRESN | VSSTRESU | VSBL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u32 | u |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 772 | 0 | 7 | 7 | 7 | 6 | |

```
# Voir les valeurs representatifs des signe vitaux dans VSTESTCD
VS.select("VSTESTCD").unique().sort("VSTESTCD")
```

```
shape: (7, 1)
 VSTESTCD
      str
 --------
    "DBP"
 "HEIGHT"
  "PULSE"
   "RESP"
    "SBP"
   "TEMP"
 "WEIGHT"
```

```
# Liste des variables representatif des signes vitaux
vs_variables=["DBP","SBP","PULSE","RESP","TEMP"]
```

```
VS0=VS.filter(pl.col("VSTESTCD").is_in(vs_variables))

# Pivot : long -> wide (1 ligne par patient)
VS0_large=VS0.pivot(
    index="USUBJID",
    columns="VSTESTCD",
    values="VSORRES",

)
```

```
C:\Users\Abdo\AppData\Local\Temp\ipykernel_8024\3187526914.py:4: DeprecationWarning: the argument `columns` for `DataFrame.p
  VS0_large=VS0.pivot(
---------------------------------------------------------------------------
ComputeError                              Traceback (most recent call last)
Cell In[114], line 4
      1 VS0=VS.filter(pl.col("VSTESTCD").is_in(vs_variables))
      3 # Pivot : long -> wide (1 ligne par patient)
----> 4 VS0_large=VS0.pivot(
      5     index="USUBJID",
      6     columns="VSTESTCD",
      7     values="VSORRES",
      8 
      9 )

File c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\polars\_utils\deprecation.py:128, in
deprecate_renamed_parameter.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    123 @wraps(function)
    124 def wrapper(*args: P.args, **kwargs: P.kwargs) -> T:
    125     _rename_keyword_argument(
    126         old_name, new_name, kwargs, function.__qualname__, version
    127     )
--> 128     return function(*args, **kwargs)

File c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\polars\dataframe\frame.py:9472, in
DataFrame.pivot(self, on, index, values, aggregate_function, maintain_order, sort_columns, separator)
   9468 else:
   9469     aggregate_expr = aggregate_function._pyexpr
   9471 return self._from_pydf(
-> 9472     self._df.pivot_expr(
   9473         on,
   9474         index,
   9475         values,
   9476         maintain_order,
   9477         sort_columns,
   9478         aggregate_expr,
   9479         separator,
   9480     )
   9481 )

ComputeError: found multiple elements in the same group, please specify an aggregation function
```

```
# Verifier qu il y a plus 2 valeur pour tout les patients
doublons_global = (
    VS0
    .group_by(["USUBJID", "VSTESTCD"])
    .len()
    .filter(pl.col("len") > 1)
)

doublons_global
```

```
shape: (414, 3)
   USUBJID  VSTESTCD  len
       str       str  u32
"01_033479"  "PULSE"    2
"01_048738"  "PULSE"    2
"01_046633"  "PULSE"    2
"02_072701"  "PULSE"    2
"01_067888"  "PULSE"    2
         …         …    …
"02_078138"  "PULSE"    2
"02_092341"  "PULSE"    2
"02_045563"    "SBP"    2
"01_033941"  "PULSE"    2
"02_046788"  "PULSE"    2
```

## Remarque

- Chaque patient doit avoir au max 5 paramètres vitaux au baseline : DBP, SBP, PULSE, RESP, TEMP.
- Des doublons persistent pour certains paramètres vitaux .
- Trois sources expliquent ces doublons :
  - VSPOS : un même test peut être mesuré en SITTING et STANDING au baseline.
  - VSCAT : la catégorie “CLINICAL OPIATE WITHDRAWAL SCALE FORM” (formulaire COWS) fournit aussi une mesure de PULSE, créant une duplication.
  - Mesures répétées dans un même formulaire, même position qui nous donne plusieurs valeurs valides pour un meme patient
- La solution consiste à :
  - filtrer les bons formulaires (VITAL SIGNS / PHYSICAL EXAM),
  - ne garder qu’une position (ex. SITTING),
  - puis regrouper par (USUBJID, VSTESTCD) et conserver la moyenne pour obtenir une valeur unique : ex (Pour 1 patient ,2 valeurs de PULSE = 75, 78 on fait la moyenne de ces test identique : VSORRES = (75 + 78) / 2 = 76.5)

```
# Voir les categorie de VSCAT
VS.select("VSCAT").unique()
```

```
shape: (3, 1)
                              VSCAT
                                str
                "VITAL SIGNS FORM"
"CLINICAL OPIATE WITHDRAWAL SCA…
        "PHYSICAL EXAMINATION FORM"
```

```
VS.select("VSPOS").unique()
```

```
shape: (3, 1)
      VSPOS
        str
  "SITTING"
       null
 "STANDING"
```

```
# Formulaires pertinents
vs_forms = ["VITAL SIGNS FORM", "PHYSICAL EXAMINATION FORM"]
# Séparation vitaux vs anthropométrie
m_vitaux = ["DBP", "SBP", "PULSE", "RESP", "TEMP"]
# Filtrer signes vitaux classiques (inclut VSPOS)
# HEIGHT et WEIGHT n’ont PAS de VSPOS ce qui pourrais les supprimer si on applique un filtre VSPOS
# Comme on a deja utiliser HEIGHT et  WEIGHT pour calculer le BMI , nous allons appliquer le filtre VSPOS.
```

```
VS_vitals = VS0.filter(
    pl.col("VSTESTCD").is_in(m_vitaux)
    & pl.col("VSCAT").is_in(vs_forms)
    & (pl.col("VSPOS") == "SITTING")
)

# Corriger les doublons une mesure par patient (Ex : 1 SEULE PULSE )
VS0_filtrage = (
    VS_vitals
    .group_by(["USUBJID", "VSTESTCD"])
    .agg(pl.col("VSORRES").cast(pl.Float64).mean())
)

# Pivot final : Construit notre tables
VS0_pivotage = VS0_filtrage.pivot(
    index="USUBJID",
    columns="VSTESTCD",
    values="VSORRES"
)

VS0_final = VS0_pivotage.select([
    pl.col("USUBJID"),
    pl.col("DBP").alias("Pression_diastolique_mmHg"),
    pl.col("SBP").alias("Pression_systolique_mmHg"),
    pl.col("PULSE").alias("battement_minute"),
    pl.col("RESP").alias("cycle_respiratoire_minute"),
    ((pl.col("TEMP") - 32) * 5/9).alias("TEMP_C")
])




VS0_final
```

```
C:\Users\Abdo\AppData\Local\Temp\ipykernel_8024\1652366950.py:23: DeprecationWarning: the argument `columns` for `DataFrame.
  VS0_pivotage = VS0_filtrage.pivot(
```

shape: (386, 6)

| USUBJID | Pression_diastolique_mmHg | Pression_systolique_mmHg | battement_minute | cycle_respiratoire_minute | TEMP_C |
|---|---|---|---|---|---|
| str | f64 | f64 | f64 | f64 | f64 |
| "01_096723" | 75.0 | 100.0 | 58.0 | 23.0 | 36.277778 |
| "02_015027" | 80.0 | 132.0 | 68.0 | 18.0 | 37.111111 |
| "02_049710" | 84.0 | 123.0 | 55.0 | 12.0 | 37.0 |
| "01_003330" | 78.0 | 112.0 | 116.0 | 20.0 | 36.555556 |
| "01_089078" | 58.0 | 114.0 | 96.0 | 20.0 | 36.111111 |
| … | … | … | … | … | … |
| "01_018415" | 85.0 | 120.0 | 102.0 | 20.0 | 37.0 |
| "02_037401" | 68.0 | 98.0 | 56.0 | 16.0 | 37.333333 |
| "01_003653" | 70.0 | 100.0 | 92.0 | 18.0 | 37.666667 |
| "02_011526" | 90.0 | 112.0 | 76.0 | 16.0 | 36.666667 |
| "01_002199" | 77.0 | 125.0 | null | null | null |

```
# Verifier qu il y a plus 2 valeur pour tout les patients
VS0_final.group_by("USUBJID").count().filter(pl.col("count") > 1)
```

```
C:\Users\Abdo\AppData\Local\Temp\ipykernel_8024\3072057161.py:2: DeprecationWarning: `GroupBy.count` was renamed; use `Group
  VS0_final.group_by("USUBJID").count().filter(pl.col("count") > 1)
```

shape: (0, 2)

| USUBJID | count |
|---|---|
| str | u32 |

```
#Voir valeurs manquante apres filtrage
print(VS0_filtrage.null_count())
#Il y avais 0 valeurs manquante des parametre vitaux contenue dans VSORRES apres le filtrage
```

```
shape: (1, 3)
┌─────────┬──────────┬─────────┐
│ USUBJID ┆ VSTESTCD ┆ VSORRES │
│ ---     ┆ ---      ┆ ---     │
│ u32     ┆ u32      ┆ u32     │
```

```
┆ 0        ┆ 0         ┆ 0         │
└──────────┴───────────┴───────────┘
```

```
#Voir valeurs manquante apres pivotage
print(VS0_pivotage.null_count())
#On observe des  valeurs manquante des certaines  parametre vitaux apres pivotage
```

```
shape: (1, 6)
┌─────────┬──────┬─────┬─────┬──────┬───────┐
│ USUBJID ┆ TEMP ┆ DBP ┆ SBP ┆ RESP ┆ PULSE │
│ ---     ┆ ---  ┆ --- ┆ --- ┆ ---  ┆ ---   │
│ u32     ┆ u32  ┆ u32 ┆ u32 ┆ u32  ┆ u32   │
╞═════════╪══════╪═════╪═════╪══════╪═══════╡
│ 0       ┆ 8    ┆ 0   ┆ 0   ┆ 7    ┆ 1     │
└─────────┴──────┴─────┴─────┴──────┴───────┘
```

```
#### Apres pivotage on observe des Valeurs Manquante alors qu au debut y avais pas .
# Le pivotage a creer des valeurs manquante , la cause est que chaque patient n’avais pas forcément toutes les 5 mesures.
print(VS0_final.null_count())
```

```
shape: (1, 6)
┌─────────┬────────────────────┬────────────────────┬─────────────────┬────────────────────┬────────┐
│ USUBJID ┆ Pression_diastoliq ┆ Pression_systoliq  ┆ battement_minute┆ cycle_respiratoir  ┆ TEMP_C │
│ ---     ┆ ue_mmHg            ┆ ue_mmHg            ┆ ---             ┆ e_minute           ┆ ---    │
│ u32     ┆ ---                ┆ ---                ┆ u32             ┆ ---                ┆ u32    │
│         ┆ u32                ┆ u32                ┆                 ┆ u32                ┆        │
╞═════════╪════════════════════╪════════════════════╪═════════════════╪════════════════════╪════════╡
│ 0       ┆ 0                  ┆ 0                  ┆ 1               ┆ 7                  ┆ 8      │
└─────────┴────────────────────┴────────────────────┴─────────────────┴────────────────────┴────────┘
```

```
### Nous allons maintenant joindre la table VS0_final avec celui de la demographie

DM_SC_VS= DM_SC.join(VS0_final, on="USUBJID", how="left")
DM_SC_VS.sample(n=3)
```

shape: (3, 14)

| USUBJID | ARMCD | AGE | SEX | RACE | EDUCYRS | EMPLOY30 | MARITAL | BMI | Pression_diastolique_mmHg |
|---|---|---|---|---|---|---|---|---|---|
| str | str | f64 | str | str | f64 | str | str | f64 | f64 |
| "02_053677" | "SCRFAIL" | null | "M" | "SPANISH, HISPANIC, OR LATINO" | 11.0 | "UNEMPLOYED" | "NEVER MARRIED" | null | null |
| "01_013173" | "BUPNAL" | 39.608487 | "M" | "WHITE" | 12.0 | "UNEMPLOYED" | "DIVORCED" | 36.618177 | 78.0 |
| "01_045827" | "SCRFAIL" | null | "M" | "WHITE" | 13.0 | "FULL TIME (35+ HRS/WK)" | "LEGALLY MARRIED" | 18.470569 | 70.0 |

```
# Apres avoir joint la dataframe demographique avec celui de signes vitaux , nous avons toujours les 411 sujets , ce qui ve
DM_SC_VS.shape
```

```
(411, 14)
```

## 1.4 Table 1 provisoire

```
# On Converti DM0 en pandas , car table one n accepte pas polars
DM_SC_VS = DM_SC_VS.to_pandas()
# On definis les colonnes categorielle et continue
categorical = ['SEX', 'RACE', 'MARITAL',"EMPLOY30"]
continuous = ['AGE', 'EDUCYRS',"BMI","Pression_diastolique_mmHg","Pression_systolique_mmHg","battement_minute","cycle_resp
# On creer la  TableOne object
table = TableOne(DM_SC_VS, columns=categorical+continuous, groupby='ARMCD')
table
```

```
c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\numpy\lib\_nanfunctions_impl.py:2015: Runti
  var = nanvar(a, axis=axis, dtype=dtype, out=out, ddof=ddof,
c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\numpy\lib\_nanfunctions_impl.py:1406: Runti
  return _nanquantile_unchecked(
c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\numpy\lib\_nanfunctions_impl.py:1406: Runti
  return _nanquantile_unchecked(
c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\tableone\tableone.py:596: RuntimeWarning: M
  return f.format(np.nanmean(x.values), self.statistics._std(x, self._ddof))  # type: ignore
c:\Users\Abdo\OneDrive\Desktop\M1\Data Challenge 1\challenge\Lib\site-packages\numpy\lib\_nanfunctions_impl.py:2015: Runti
  var = nanvar(a, axis=axis, dtype=dtype, out=out, ddof=ddof,
```

| | | Grouped by ARMCD | | | | |
|---|---|---|---|---|---|---|
| | | Missing | Overall | BUPNAL | CLON | SCRFAIL |
| **n** | | | 411 | 233 | 110 | 68 |
| **SEX, n (%)** | **F** | | 128 (31.1) | 72 (30.9) | 38 (34.5) | 18 (26.5) |
| | **M** | | 281 (68.4) | 161 (69.1) | 72 (65.5) | 48 (70.6) |
| | **U** | | 2 (0.5) | 0 (0.0) | 0 (0.0) | 2 (2.9) |
| **RACE, n (%)** | **BLACK, AFRICAN AMERICAN, OR NEGRO** | | 120 (29.2) | 71 (30.5) | 35 (31.8) | 14 (20.6) |
| | **None** | | 2 (0.5) | 0 (0.0) | 0 (0.0) | 2 (2.9) |
| | **OTHER** | | 24 (5.8) | 12 (5.2) | 6 (5.5) | 6 (8.8) |
| | **SPANISH, HISPANIC, OR LATINO** | | 87 (21.2) | 45 (19.3) | 19 (17.3) | 23 (33.8) |
| | **WHITE** | | 178 (43.3) | 105 (45.1) | 50 (45.5) | 23 (33.8) |
| **MARITAL, n (%)** | **DIVORCED** | | 63 (15.3) | 38 (16.3) | 17 (15.5) | 8 (11.8) |
| | **LEGALLY MARRIED** | | 74 (18.0) | 47 (20.2) | 15 (13.6) | 12 (17.6) |
| | **LIVING WITH PARTNER/COHABITATING** | | 38 (9.2) | 20 (8.6) | 12 (10.9) | 6 (8.8) |
| | **NEVER MARRIED** | | 188 (45.7) | 109 (46.8) | 51 (46.4) | 28 (41.2) |
| | **None** | | 6 (1.5) | 0 (0.0) | 0 (0.0) | 6 (8.8) |
| | **SEPARATED** | | 32 (7.8) | 14 (6.0) | 12 (10.9) | 6 (8.8) |
| | **WIDOWED** | | 10 (2.4) | 5 (2.1) | 3 (2.7) | 2 (2.9) |
| **EMPLOY30, n (%)** | **FULL TIME (35+ HRS/WK)** | | 133 (32.4) | 73 (31.3) | 44 (40.0) | 16 (23.5) |
| | **HOMEMAKER** | | 14 (3.4) | 8 (3.4) | 4 (3.6) | 2 (2.9) |
| | **IN CONTROLLED ENVIRONMENT** | | 1 (0.2) | 1 (0.4) | 0 (0.0) | 0 (0.0) |
| | **None** | | 7 (1.7) | 0 (0.0) | 0 (0.0) | 7 (10.3) |
| | **PART TIME (IRREGULAR DAYWORK)** | | 45 (10.9) | 28 (12.0) | 9 (8.2) | 8 (11.8) |
| | **PART TIME (REGULAR HOURS)** | | 17 (4.1) | 11 (4.7) | 4 (3.6) | 2 (2.9) |
| | **RETIRED/DISABILITY** | | 11 (2.7) | 3 (1.3) | 6 (5.5) | 2 (2.9) |
| | **STUDENT** | | 8 (1.9) | 4 (1.7) | 2 (1.8) | 2 (2.9) |
| | **UNEMPLOYED** | | 175 (42.6) | 105 (45.1) | 41 (37.3) | 29 (42.6) |
| **AGE, mean (SD)** | | 68 | 38.0 (10.1) | 37.4 (10.5) | 39.2 (9.3) | nan (nan) |
| **EDUCYRS, mean (SD)** | | 6 | 12.5 (2.1) | 12.6 (2.0) | 12.7 (2.3) | 11.8 (1.9) |
| **BMI, mean (SD)** | | 34 | 25.1 (4.9) | 24.8 (5.0) | 25.6 (4.6) | 25.7 (5.3) |
| **Pression_diastolique_mmHg, mean (SD)** | | 28 | 79.9 (34.0) | 80.9 (43.0) | 80.0 (10.8) | 74.9 (11.9) |
| **Pression_systolique_mmHg, mean (SD)** | | 28 | 121.1 (16.5) | 120.5 (15.9) | 124.4 (17.1) | 116.9 (16.8) |
| **battement_minute, mean (SD)** | | 29 | 75.7 (12.1) | 76.3 (11.8) | 75.2 (12.7) | 74.0 (12.0) |

```
# Valeurs manquante dans la table 1
DM_SC_VS.groupby("ARMCD").agg(lambda x: x.isna().sum())
```