

DATA CLEANING AUTOMATION AND MERGING

1

Cleaning Automation of VAERS Databases (2020-2025)

```
1. # Author: Miguel Atarés
2. # coding: utf-8

3. import pandas as pd

4. def limpiar_vaers_data(path_csv: str, anio: int, guardar_csv=False,
5. ruta_guardado=None):
6. """
7. Limpia un archivo VAERS DATA según los pasos realizados en el análisis de 2021.

8. Parámetros:
9.     - path_csv: ruta del archivo CSV original (string)
10.    - anio: año del archivo (por ahora no afecta, pero puede servir para
11.      adaptar filtros en el futuro)
12.    - guardar_csv: True si quieres guardar el archivo limpio en disco
13.    - ruta_guardado: ruta donde guardar el archivo limpio (si guardar_csv es
14.      True)

15. Devuelve:
16.     - DataFrame limpio (pandas)
17. """

18. # Leer el CSV
19. df = pd.read_csv(path_csv, encoding="ISO-8859-1", delimiter=";",
20. low_memory=False)

21. print("\n📊 Información INICIAL del dataset:")
22. print(f"Forma (filas, columnas): {df.shape}")
23. df.info()
24. print("\nDescribe (solo columnas numéricas):")
25. print(df.describe())
26. print("\nPorcentaje de nulos por columna:")
27. print(df.isnull().mean().sort_values(ascending=False) * 100)

28. # Columnas que deben mantenerse
29. columnas_a_conservar = [
30.     "VAERS_ID", "RECVDATE", "VAX_DATE", "ONSET_DATE", "NUMDAYS",
31.     "DATEDIED",
32.     "STATE", "AGE_YRS", "SEX",
33.     "DIED", "L_THREAT", "ER_VISIT", "HOSPITAL", "HOSPDAYS",
34.     "DISABLE", "RECOVD", "BIRTH_DEFECT"
35. ]
```

```

22. columnas_presentes = set(df.columns)
23. faltantes = set(columnas_a_conservar) - columnas_presentes
24. if faltantes:
    - raise ValueError(f"FALTAN columnas obligatorias en el archivo:
      {faltantes}")

25. df = df[columnas_a_conservar].copy()

26. df.drop_duplicates(inplace=True)

27. fechas = ["RECVDATE", "VAX_DATE", "ONSET_DATE", "DATEDIED"]
28. for col in fechas:
    - df[col] = pd.to_datetime(df[col], errors="coerce")

29. # Reemplazar valores problemáticos
30. df["RECOVD"] = df["RECOVD"].replace("U", pd.NA)
31. df["STATE"] = df["STATE"].str.upper()

32. estados_validos = [
    - 'AL', 'AK', 'AZ', 'AR', 'CA', 'CO', 'CT', 'DE', 'FL', 'GA', 'HI', 'ID', 'IL', 'IN', '
      IA', 'KS', 'KY', 'LA', 'ME', 'MD', 'MA', 'MI',
    - 'MN', 'MS', 'MO', 'MT', 'NE', 'NV', 'NH', 'NJ', 'NM', 'NY', 'NC', 'ND', 'OH', 'OK', '
      OR', 'PA', 'RI', 'SC', 'SD', 'TN', 'TX', 'UT',
    - 'VT', 'VA', 'WA', 'WV', 'WI', 'WY', 'DC', 'PR', 'VI', 'GU', 'AS', 'MP', 'MH', 'FM'
33. ]
34. df.loc[~df["STATE"].isin(estados_validos), "STATE"] = pd.NA
35. df.dropna(subset=["STATE"], inplace=True)

36. columnas_binarias = ["DIED", "L_THREAT", "HOSPITAL", "DISABLE", "BIRTH_DEFECT",
    "ER_VISIT"]
37. for col in columnas_binarias:
    - df[col] = df[col].fillna("N")

38. # Fechas válidas (ajustables)
39. fecha_min = pd.to_datetime(f"{anio}-01-01")
40. fecha_max = pd.to_datetime(f"{anio + 1}-01-15")
41. df = df[
    - (df["VAX_DATE"].isna() | ((df["VAX_DATE"] >= fecha_min) &
      (df["VAX_DATE"] <= fecha_max))) &
    - (df["ONSET_DATE"].isna() | ((df["ONSET_DATE"] >= fecha_min) &
      (df["ONSET_DATE"] <= fecha_max))) &
    - (df["DATEDIED"].isna() | ((df["DATEDIED"] >= fecha_min) &
      (df["DATEDIED"] <= fecha_max)))
42. ]

43. # HOSPDAYS máximo de 1 año
44. df = df[(df["HOSPDAYS"].isna()) | (df["HOSPDAYS"] <= 365)]

```

```

45. print("\n📊 Información FINAL del dataset:")
46. print(f"Forma (filas, columnas): {df.shape}")
47. df.info()
48. print("\nDescribe (solo columnas numéricas):")
49. print(df.describe())
50. print("\nPorcentaje de nulos por columna:")
51. print(df.isnull().mean().sort_values(ascending=False) * 100)

52. # Guardar si se solicita
53. if guardar_csv and ruta_guardado:
    - df.to_csv(ruta_guardado, index=False)
    - print(f"✅ Archivo limpio guardado en: {ruta_guardado}")

54. return df

55. df_2022_Data = limpiar_vaers_data(
56. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2022VAERSData\2022VAERSDATA.csv",
57. anio=2022,
58. guardar_csv=True,
59. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2022Data_limpio_para_powerbi.csv"
60. )

61. def limpiar_vaers_symptoms(path_csv: str, guardar_csv=False,
    ruta_guardado=None):
62. """
63. Limpia un archivo VAERS SYMPTOMS eliminando columnas irrelevantes y duplicados.

64. Parámetros:
    - path_csv: ruta del archivo original
    - guardar_csv: si True, guarda el resultado
    - ruta_guardado: ruta donde guardar el archivo limpio

65. Devuelve:
    - DataFrame limpio
66. """

67. # 📖 Leer el CSV original
68. df = pd.read_csv(path_csv, encoding="ISO-8859-1", delimiter=",")

69. # 📊 Información inicial
70. print("\n📊 Información INICIAL del dataset:")
71. print(f"Forma (filas, columnas): {df.shape}")
72. df.info()
73. print("\nDescribe (solo columnas numéricas):")
74. print(df.describe())
75. print("\nPorcentaje de nulos por columna:")
76. print(df.isnull().mean().sort_values(ascending=False) * 100)

```

```

77. # Columnas a eliminar
78. columnas_a_eliminar = [
    - "SYMPTOMVERSION1", "SYMPTOMVERSION2", "SYMPTOMVERSION3",
    - "SYMPTOMVERSION4", "SYMPTOMVERSION5", "ORDER"
79. ]
80. df = df.drop(columns=[col for col in columnas_a_eliminar if col in df.columns])

81. # Eliminar duplicados
82. df.drop_duplicates(inplace=True)

83. # 📊 Información final
84. print("\n📊 Información FINAL del dataset:")
85. print(f"Forma (filas, columnas): {df.shape}")
86. df.info()
87. print("\nDescribe (solo columnas numéricas):")
88. print(df.describe())
89. print("\nPorcentaje de nulos por columna:")
90. print(df.isnull().mean().sort_values(ascending=False) * 100)

91. # Guardar si se solicita
92. if guardar_csv and ruta_guardado:
    - df.to_csv(ruta_guardado, index=False)
    - print(f"\n✅ Archivo limpio guardado en: {ruta_guardado}")

93. return df

94. df_2022_Symptoms = limpiar_vaers_symptoms(
95. path_csv=r"C:\Users\miguel\Desktop\COVID-19
    Project\Databases\2022VAERSData\2022VAERSSYMPTOMS.csv",
96. guardar_csv=True,
97. ruta_guardado=r"C:\Users\miguel\Desktop\COVID-19
    Project\dfVAERS2022Symptoms_limpio_para_powerbi.csv"
98. )

99. def limpiar_vaers_vax(path_csv: str, guardar_csv=False, ruta_guardado=None):
100. """
101. Limpia un archivo VAERS VAX eliminando columnas irrelevantes,
102. filtrando vacunas COVID y mostrando análisis de 'UNK' en dosis.

103. Parámetros:
    - path_csv: ruta del archivo original
    - guardar_csv: si True, guarda el resultado
    - ruta_guardado: ruta donde guardar el archivo limpio

104. Devuelve:
    - DataFrame limpio
105. """

```

```

106. # Leer CSV
107. df = pd.read_csv(path_csv, encoding="ISO-8859-1", delimiter=",")

108. # 📊 Información inicial
109. print("\n📊 Información INICIAL del dataset:")
110. print(f"Forma (filas, columnas): {df.shape}")
111. df.info()
112. print("\nDescribe (solo columnas numéricas):")
113. print(df.describe())
114. print("\nPorcentaje de nulos por columna:")
115. print(df.isnull().mean().sort_values(ascending=False) * 100)

116. # Columnas a eliminar
117. columnas_a_eliminar = ["VAX_LOT", "VAX_ROUTE", "VAX_SITE", "VAX_NAME", "ORDER"]
118. df = df.drop(columns=[col for col in columnas_a_eliminar if col in df.columns])

119. # Filtrar solo vacunas COVID19
120. df = df[df["VAX_TYPE"].str.contains("COVID19", case=False, na=False)]

121. # Eliminar columna redundante VAX_TYPE
122. df = df.drop(columns=["VAX_TYPE"])

123. # Eliminar duplicados
124. df.drop_duplicates(inplace=True)

125. # 📊 Información final
126. print("\n📊 Información FINAL del dataset:")
127. print(f"Forma (filas, columnas): {df.shape}")
128. df.info()
129. print("\nDescribe (solo columnas numéricas):")
130. print(df.describe())
131. print("\nPorcentaje de nulos por columna:")
132. print(df.isnull().mean().sort_values(ascending=False) * 100)

133. # Porcentaje de UNK en VAX_DOSE_SERIES
134. if "VAX_DOSE_SERIES" in df.columns:
    - total = len(df)
    - unk_count = (df["VAX_DOSE_SERIES"] == "UNK").sum()
    - unk_pct = round((unk_count / total) * 100, 2)
    - print(f"\n🔗 'UNK' en VAX_DOSE_SERIES: {unk_count} filas ({unk_pct}%)")

135. # Guardar si se solicita
136. if guardar_csv and ruta_guardado:
    - df.to_csv(ruta_guardado, index=False)
    - print(f"\n✅ Archivo limpio guardado en: {ruta_guardado}")

137. return df

```

```

138. df_vax_2022 = limpiar_vaers_vax(
139. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2022VAERSData\2022VAERSVAX.csv",
140. guardar_csv=True,
141. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2022VAX_limpio_para_powerbi.csv"
142. )
143. # Ahora que ya hemos hecho la prueba, ejecutamos los scripts de limpieza para
    el resto de años
144. df_2023_Data = limpiar_vaers_data(
145. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2023VAERSData\2023VAERSDATA.csv",
146. anio=2023,
147. guardar_csv=True,
148. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2023Data_limpio_para_powerbi.csv"
149. )

150. df_2024_Data = limpiar_vaers_data(
151. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2024VAERSData\2024VAERSDATA.csv",
152. anio=2024,
153. guardar_csv=True,
154. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2024Data_limpio_para_powerbi.csv"
155. )

156. df_2025_Data = limpiar_vaers_data(
157. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2025VAERSData\2025VAERSDATA.csv",
158. anio=2025,
159. guardar_csv=True,
160. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2025Data_limpio_para_powerbi.csv"
161. )

162. df_2020_Data = limpiar_vaers_data(
163. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2020VAERSData\2020VAERSDATA.csv",
164. anio=2020,
165. guardar_csv=True,
166. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2020Data_limpio_para_powerbi.csv"
167. )

168. df_2023_Symptoms = limpiar_vaers_symptoms(
169. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2023VAERSData\2023VAERSSYMPTOMS.csv",
170. guardar_csv=True,

```

```
171. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2023Symptoms_limpio_para_powerbi.csv"
172. )

173. df_2024_Symptoms = limpiar_vaers_symptoms(
174. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2024VAERSData\2024VAERSSYMPTOMS.csv",
175. guardar_csv=True,
176. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2024Symptoms_limpio_para_powerbi.csv"
177. )

178. df_2025_Symptoms = limpiar_vaers_symptoms(
179. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2025VAERSData\2025VAERSSYMPTOMS.csv",
180. guardar_csv=True,
181. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2025Symptoms_limpio_para_powerbi.csv"
182. )

183. df_2020_Symptoms = limpiar_vaers_symptoms(
184. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2020VAERSData\2020VAERSSYMPTOMS.csv",
185. guardar_csv=True,
186. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2020Symptoms_limpio_para_powerbi.csv"
187. )

188. df_vax_2023 = limpiar_vaers_vax(
189. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2023VAERSData\2023VAERSVAX.csv",
190. guardar_csv=True,
191. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2023VAX_limpio_para_powerbi.csv"
192. )

193. df_vax_2024 = limpiar_vaers_vax(
194. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2024VAERSData\2024VAERSVAX.csv",
195. guardar_csv=True,
196. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2024VAX_limpio_para_powerbi.csv"
197. )

198. df_vax_2025 = limpiar_vaers_vax(
199. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2025VAERSData\2025VAERSVAX.csv",
200. guardar_csv=True,
```

```

201. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2025VAX_limpio_para_powerbi.csv"
202. )

203. df_vax_2020 = limpiar_vaers_vax(
204. path_csv=r"C:\Users\migue\Desktop\COVID-19
    Project\Databases\2020VAERSData\2020VAERSVAX.csv",
205. guardar_csv=True,
206. ruta_guardado=r"C:\Users\migue\Desktop\COVID-19
    Project\dfVAERS2020VAX_limpio_para_powerbi.csv"
207. )

```

2

Merge of all VAERS Databases (2020-2025)

```

1. # Author: Miguel Atarés
2. # coding: utf-8

3. import pandas as pd

4. # Leemos los archivos individualmente
5. df_2020 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VARs\2020\dfVAERS2020Data_limpio_para_powerbi.csv")
6. df_2020["YEAR"] = 2020

7. df_2021 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VARs\2021\dfVAERS2021Data_limpio_para_powerbi.csv")
8. df_2021["YEAR"] = 2021

9. df_2022 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VARs\2022\dfVAERS2022Data_limpio_para_powerbi.csv")
10. df_2022["YEAR"] = 2022

11. df_2023 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VARs\2023\dfVAERS2023Data_limpio_para_powerbi.csv")
12. df_2023["YEAR"] = 2023

13. df_2024 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VARs\2024\dfVAERS2024Data_limpio_para_powerbi.csv")
14. df_2024["YEAR"] = 2024

15. df_2025 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VARs\2025\dfVAERS2025Data_limpio_para_powerbi.csv")
16. df_2025["YEAR"] = 2025

17. # Unimos todos
18. df_merged = pd.concat([df_2020, df_2021, df_2022, df_2023, df_2024, df_2025],
    ignore_index=True)

```

```

19. # Guardamos el resultado
20. df_merged.to_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VAERS_Data_2020_2025.csv", index=False)

21. print("✅ Merge completado para VAERS Data.")

22. # Leer archivos SYMPTOMS
23. df_symptoms_2020 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases
    Limpias\VARs\2020\dfVAERS2020Symptoms_limpio_para_powerbi.csv")
24. df_symptoms_2020["YEAR"] = 2020

25. df_symptoms_2021 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases
    Limpias\VARs\2021\dfVAERS2021Symptoms_limpio_para_powerbi.csv")
26. df_symptoms_2021["YEAR"] = 2021

27. df_symptoms_2022 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases
    Limpias\VARs\2022\dfVAERS2022Symptoms_limpio_para_powerbi.csv")
28. df_symptoms_2022["YEAR"] = 2022

29. df_symptoms_2023 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases
    Limpias\VARs\2023\dfVAERS2023Symptoms_limpio_para_powerbi.csv")
30. df_symptoms_2023["YEAR"] = 2023

31. df_symptoms_2024 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases
    Limpias\VARs\2024\dfVAERS2024Symptoms_limpio_para_powerbi.csv")
32. df_symptoms_2024["YEAR"] = 2024

33. df_symptoms_2025 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases
    Limpias\VARs\2025\dfVAERS2025Symptoms_limpio_para_powerbi.csv")
34. df_symptoms_2025["YEAR"] = 2025

35. # Concatenar
36. df_symptoms_merged = pd.concat([
37. df_symptoms_2020, df_symptoms_2021, df_symptoms_2022,
38. df_symptoms_2023, df_symptoms_2024, df_symptoms_2025
39. ], ignore_index=True)

40. # Guardar
41. df_symptoms_merged.to_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VAERS_Symptoms_2020_2025.csv", index=False)

42. print("✅ Merge completado para VAERS Symptoms.")

```

```

43. # Leer archivos VAX
44. df_vax_2020 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VARs\2020\dfVAERS2020VAX_limpio_para_powerbi.csv")
45. df_vax_2020["YEAR"] = 2020

46. df_vax_2021 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VARs\2021\dfVAERS2021VAX_limpio_para_powerbi.csv")
47. df_vax_2021["YEAR"] = 2021

48. df_vax_2022 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VARs\2022\dfVAERS2022VAX_limpio_para_powerbi.csv")
49. df_vax_2022["YEAR"] = 2022

50. df_vax_2023 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VARs\2023\dfVAERS2023VAX_limpio_para_powerbi.csv")
51. df_vax_2023["YEAR"] = 2023

52. df_vax_2024 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VARs\2024\dfVAERS2024VAX_limpio_para_powerbi.csv")
53. df_vax_2024["YEAR"] = 2024

54. df_vax_2025 = pd.read_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1.
    COVID-19\Databases Limpias\VARs\2025\dfVAERS2025VAX_limpio_para_powerbi.csv")
55. df_vax_2025["YEAR"] = 2025

56. # Concatenar
57. df_vax_merged = pd.concat([
58. df_vax_2020, df_vax_2021, df_vax_2022,
59. df_vax_2023, df_vax_2024, df_vax_2025
60. ], ignore_index=True)

61. # Guardar
62. df_vax_merged.to_csv(r"G:\Mi unidad\PROGRAMACIÓN\Portfolio Projetcs\1. COVID-
    19\Databases Limpias\VAERS_VAX_2020_2025.csv", index=False)

63. print("✅ Merge completado para VAERS VAX.")

64. # Calcular totales individuales
65. total_vax = len(df_vax_2020) + len(df_vax_2021) + len(df_vax_2022) +
    len(df_vax_2023) + len(df_vax_2024) + len(df_vax_2025)
66. total_symptoms = len(df_symptoms_2020) + len(df_symptoms_2021) +
    len(df_symptoms_2022) + len(df_symptoms_2023) + len(df_symptoms_2024) +
    len(df_symptoms_2025)
67. total_data = len(df_2020) + len(df_2021) + len(df_2022) + len(df_2023) +
    len(df_2024) + len(df_2025)

68. # Prints comparativos
69. print("VAX → Individual:", total_vax, "| Mergeado:", len(df_vax_merged), "|",
    "✅ OK" if total_vax == len(df_vax_merged) else "❌ Mismatch")

```

```
70. print("SYMPTOMS → Individual:", total_symptoms, "| Mergeado:",  
      len(df_symptoms_merged), "|", "✅ OK" if total_symptoms ==  
      len(df_symptoms_merged) else "❌ Mismatch")  
71. print("DATA → Individual:", total_data, "| Mergeado:", len(df_merged), "|", "✅  
      OK" if total_data == len(df_merged) else "❌ Mismatch")
```