testing

November 19, 2025

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import sys
from pathlib import Path
```

[2]:
```
warnings.filterwarnings('ignore')
sys.path.append('../src')
from config import *
from data_utils import *
from preprocessing import *
from eda_utils import *
```

[3]:
```
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', 100)
pd.set_option('display.float_format', '{:.2f}'.format)
```

[4]:
```
datasets = load_all_datasets(RAW_DATA_DIR)
```

```
2025-11-19 17:58:30 | INFO     |
data_utils:load_all_datasets - Loading datasets from

c:\Users\amine\Documents\ACADEMIC\DATA\notebooks\..\data\raw
2025-11-19 17:58:30 | INFO     |
data_utils:detect_encoding - Detected encoding for

AMDEC_clean.csv: ISO-8859-1 (confidence: 73.00%)
2025-11-19 17:58:30 | INFO     |
data_utils:load_csv - Loading AMDEC_clean.csv with

encoding: ISO-8859-1
2025-11-19 17:58:30 | INFO     |
data_utils:_fix_numeric_columns -   Converted Durée arrêt

(h) from comma decimal to numeric
2025-11-19 17:58:30 | INFO     |
data_utils:_fix_numeric_columns -   Converted [Pièce].Prix

total from comma decimal to numeric
```

```
2025-11-19 17:58:30 | SUCCESS  |
data_utils:load_csv - Successfully loaded

AMDEC_clean.csv - Shape: (1942, 13)
2025-11-19 17:58:30 | INFO     |
data_utils:detect_encoding - Detected encoding for

GMAO_integrator_clean.csv: ISO-8859-1 (confidence: 73.00%)
2025-11-19 17:58:30 | INFO     |
data_utils:load_csv - Loading GMAO_integrator_clean.csv

with encoding: ISO-8859-1
2025-11-19 17:58:31 | INFO     |
data_utils:_fix_numeric_columns -   Converted Durée arrêt

(h) from comma decimal to numeric
2025-11-19 17:58:31 | INFO     |
data_utils:_fix_numeric_columns -   Converted Coût

matériel from comma decimal to numeric
2025-11-19 17:58:31 | SUCCESS  |
data_utils:load_csv - Successfully loaded

GMAO_integrator_clean.csv - Shape: (9341, 9)
2025-11-19 17:58:31 | INFO     |
data_utils:detect_encoding - Detected encoding for

Workload_clean.csv: ISO-8859-1 (confidence: 73.00%)
2025-11-19 17:58:31 | INFO     |
data_utils:load_csv - Loading Workload_clean.csv with

encoding: ISO-8859-1
2025-11-19 17:58:31 | INFO     |
data_utils:_fix_numeric_columns -   Converted Durée arrêt

(h) from comma decimal to numeric
2025-11-19 17:58:31 | INFO     |
data_utils:_fix_numeric_columns -   Converted Nombre

d'heures MO from comma decimal to numeric
2025-11-19 17:58:31 | INFO     |
data_utils:_fix_numeric_columns -   Converted Coût total

intervention from comma decimal to numeric
2025-11-19 17:58:31 | INFO     |
data_utils:_fix_numeric_columns -   Converted [MO

interne].Nombre d'heures from comma decimal to numeric
2025-11-19 17:58:31 | SUCCESS  |
data_utils:load_csv - Successfully loaded

Workload_clean.csv - Shape: (3169, 10)
2025-11-19 17:58:31 | SUCCESS  |
```

```
data_utils:load_all_datasets - Loaded 3 datasets
```

[5]:
```
df_amdec = datasets.get('AMDEC')
df_gmao = datasets.get('GMAO')
df_workload = datasets.get('Workload')
```

[6]:
```
df_amdec
```

[6]:

	Type de panne	Durée arrêt (h)	\
0	Hydraulique	1.08	
1	Automate	5.00	
2	Electronique	0.58	
3	Hydraulique	0.75	
4	Mécanique	114.72	
…	…	…	
1937	Arrosage	0.33	
1938	Arrosage	1.50	
1939	Arrosage	0.17	
1940	Arrosage	0.92	
1941	Pneumatique	1.42	

	Résumé intervention	Date intervention	\
0	Verification de la pompe du lubrifiant / demon…	29/03/2025	
1	Activation de la machine	28/03/2025	
2	Verification de l'armoire electrique / redemar…	28/03/2025	
3	Amorcage de la pompe/ verifier les conduites	28/03/2025	
4	Controle axe C et B / modification des axes XY…	28/03/2025	
…	…	…	
1937	Nettoyage pompe d'arrosage normal	02/04/2024	
1938	Nettoyage clapét anti-retour +Nettoyage circui…	02/04/2024	
1939	Basculer à la position deux filtres d'arrosage…	01/04/2024	
1940	Nettoyage la pompe d'arrisage de convoyeur + N…	01/04/2024	
1941	Fixation Raccord d'air + Fixation tuyaux	01/04/2024	

	Désignation	Date demande	Cause	Coût matériel	\
0	machine1	29/03/2025 10:05	PROBLEME DE LUBRIFICATION	0	
1	machine2	28/03/2025 08:00	PROBLEME INFORMATIQUE	0	
2	machine3	28/03/2025 06:00	PROBLEME ELECTRONIQUE	0	
3	machine1	28/03/2025 02:20	PROBLEME DE LUBRIFICATION	0	
4	machine2	28/03/2025 06:47	DECALAGE POSITION AXE	0	
…	…	…	…	…	
1937	machine6	02/04/2024 06:00	PROBLEME D'ARROSAGE	0	
1938	machine6	02/04/2024 10:30	PROBLEME D'ARROSAGE	0	
1939	machine4	01/04/2024 08:15	PROBLEME D'ARROSAGE	0	
1940	machine6	01/04/2024 09:20	DEBORDEMENT	0	
1941	machine1	01/04/2024 07:00	FUITE D'AIR	0	

```
                        Organe [Pièce].Désignation [Pièce].Référence  \
0     05-Unité de lubrification                  NaN                NaN
1          07-poste de commande                  NaN                NaN
2         03-Armoire électrique                  NaN                NaN
3     05-Unité de lubrification                  NaN                NaN
4                       02-Axes                  NaN                NaN
…                           …                    …                  …
1937        04-Groupe d'arrosage                 NaN                NaN
1938        04-Groupe d'arrosage                 NaN                NaN
1939        04-Groupe d'arrosage                 NaN                NaN
1940        04-Groupe d'arrosage                 NaN                NaN
1941      11-Système pneumatique                 NaN                NaN

      [Pièce].Quantité  [Pièce].Prix total
0                  NaN                 NaN
1                  NaN                 NaN
2                  NaN                 NaN
3                  NaN                 NaN
4                  NaN                 NaN
…                  …                   …
1937               NaN                 NaN
1938               NaN                 NaN
1939               NaN                 NaN
1940               NaN                 NaN
1941               NaN                 NaN

[1942 rows x 13 columns]
```

[7]:
```
df_gmao
```

[7]:
```
     Type de panne  Durée arrêt (h) Résultat Date intervention Désignation  \
0        Mécanique             0.42      NaN        25/04/2025    machine1
1        Mécanique             1.33      NaN        24/04/2025    machine1
2        Mécanique             0.50      NaN        24/04/2025    machine2
3        Mécanique             0.33      NaN        24/04/2025    machine2
4        Mécanique             6.08      NaN        24/04/2025    machine3
…              …                …        …                 …           …
9336    Electrique              NaN      NaN        18/07/2016    machine6
9337      Arrosage              NaN      NaN        16/07/2016    machine6
9338    Electrique             4.00      NaN        16/07/2016    machine2
9339   Pneumatique              NaN      NaN        14/07/2016    machine1
9340    Electrique              NaN      NaN        01/07/2016    machine6

      Coût matériel [Pièce].Désignation [Pièce].Référence  [Pièce].Quantité
0              0.00                 NaN                NaN               NaN
1              0.00                 NaN                NaN               NaN
2              0.00                 NaN                NaN               NaN
```

```
3              0.00                  NaN               NaN               NaN
4              0.00                  NaN               NaN               NaN
…               …                     …                 …                 …
9336           0.00                  NaN               NaN               NaN
9337           0.00                  NaN               NaN               NaN
9338           0.00                  NaN               NaN               NaN
9339           0.00                  NaN               NaN               NaN
9340           0.00                  NaN               NaN               NaN

[9341 rows x 9 columns]
```

[8]:
```
df_workload
```

[8]:
```
     Type de panne  Durée arrêt (h) Date intervention Désignation  \
0      Hydraulique             1.08        29/03/2025    machine1
1         Automate             5.00        28/03/2025    machine2
2     Electronique             0.58        28/03/2025    machine3
3      Hydraulique             0.75        28/03/2025    machine1
4      Hydraulique             0.75        28/03/2025    machine1
…              …                …                 …          …
3164      Arrosage             0.17        01/04/2024    machine4
3165      Arrosage             0.92        01/04/2024    machine6
3166      Arrosage             0.92        01/04/2024    machine6
3167      Arrosage             0.92        01/04/2024    machine6
3168   Pneumatique             1.42        01/04/2024    machine1

     Catégorie de panne  Nombre d'heures MO  Coût total intervention  \
0           Hydraulique                1.00                    56.00
1              Automate                0.08                     0.69
2          Electronique                0.25                    22.17
3           Hydraulique                1.17                    42.72
4           Hydraulique                1.17                    42.72
…                   …                   …                        …
3164           Arrosage                0.17                     1.39
3165           Arrosage                2.25                    44.80
3166           Arrosage                2.25                    44.80
3167           Arrosage                2.25                    44.80
3168        Pneumatique                1.42                     0.00

     [MO interne].Nom [MO interne].Prénom  [MO interne].Nombre d'heures
0                tech                 one                          1.00
1                tech                 two                          0.08
2                tech               three                          0.25
3                tech                four                          0.58
4                tech                five                          0.58
…                …                    …                             …
3164             tech               seven                          0.17
```

```
3165            tech                two                        0.58
3166            tech                one                        0.83
3167            tech              three                        0.83
3168            tech              three                        1.42

[3169 rows x 10 columns]
```

[16]:
```
amdec_info = get_dataset_info(df_amdec, "AMDEC")
gmao_info = get_dataset_info(df_gmao, "GMAO")
workload_info = get_dataset_info(df_workload, "Workload")
```

```
2025-11-15 09:25:55 | INFO     |
data_utils:get_dataset_info - Generating info for

AMDEC
2025-11-15 09:25:55 | INFO     |
data_utils:get_dataset_info - Generating info for GMAO
2025-11-15 09:25:55 | INFO     |
data_utils:get_dataset_info - Generating info for

Workload
```

[17]:
```
amdec_info
```

[17]:
```
                 Column     Type  Non-Null Count  Null Count  Null %  \
0         Type de panne   object            1942           0    0.00
1       Durée arrêt (h)  float64            1942           0    0.00
2   Résumé intervention   object            1942           0    0.00
3     Date intervention   object            1942           0    0.00
4           Désignation   object            1942           0    0.00
5          Date demande   object            1939           3    0.15
6                 Cause   object            1942           0    0.00
7         Coût matériel   object            1942           0    0.00
8                Organe   object            1930          12    0.62
9   [Pièce].Désignation   object             166        1776   91.45
10    [Pièce].Référence   object             165        1777   91.50
11     [Pièce].Quantité  float64             166        1776   91.45
12   [Pièce].Prix total  float64             166        1776   91.45

    Unique Values                                  Sample Value
0              10                                    Hydraulique
1             323                                           1.08
2            1764  Verification de la pompe du lubrifiant / demon…
3             348                                     29/03/2025
4              10                                       machine1
5            1845                              29/03/2025 10:05
6              70                        PROBLEME DE LUBRIFICATION
7              75                                              0
8              68                       05-Unité de lubrification
```

```
9              96  Protecteur d'axe X droit ( SOUFFLET AXE X DROI…
10             96                                          M-PR0019
11              5                                              1.00
12            101                                           1700.00
```

[18]:
```
gmao_info
```

[18]:
```
                Column     Type  Non-Null Count  Null Count  Null %  \
0        Type de panne   object            9341           0    0.00
1      Durée arrêt (h)  float64            9285          56    0.60
2              Résultat   object              24        9317   99.74
3    Date intervention   object            9341           0    0.00
4           Désignation   object            9341           0    0.00
5        Coût matériel  float64            9341           0    0.00
6  [Pièce].Désignation   object             985        8356   89.46
7    [Pièce].Référence   object             984        8357   89.47
8     [Pièce].Quantité  float64             985        8356   89.46

   Unique Values                                    Sample Value
0             10                                       Mécanique
1            973                                            0.42
2             20                machine rendu sans arrosage centre
3           2368                                      25/04/2025
4             12                                        machine1
5            339                                            0.00
6            320  FILTER ELEMENT d'arrosage au centre DA-300  (Z…
7            316                                         M-F0040
8              8                                            1.00
```

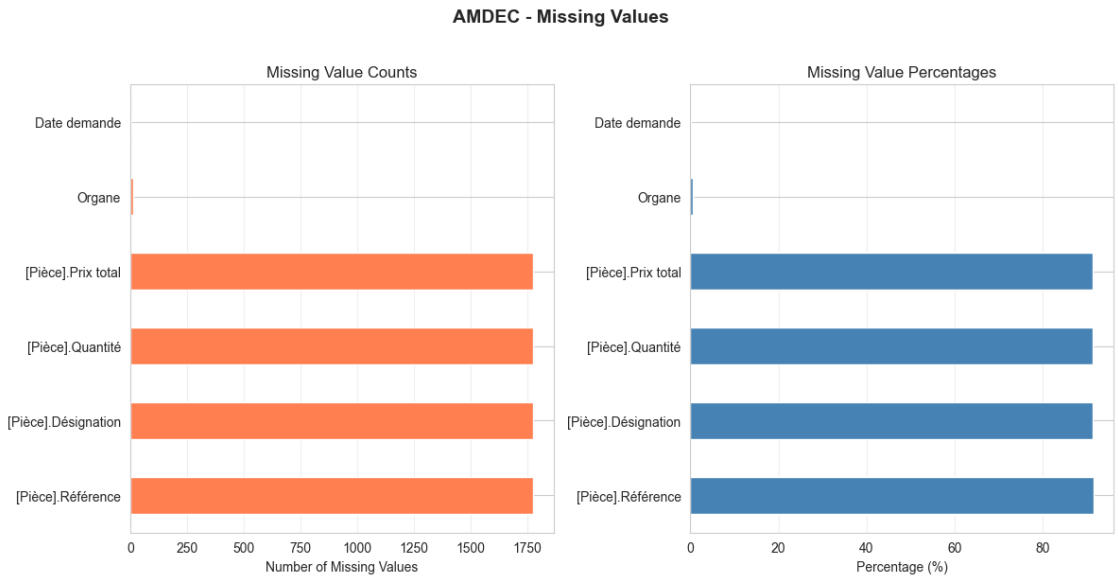
[19]:
```
workload_info
```

[19]:
```
                       Column     Type  Non-Null Count  Null Count  Null %  \
0                Type de panne   object            3169           0    0.00
1              Durée arrêt (h)  float64            3169           0    0.00
2           Date intervention   object            3169           0    0.00
3                 Désignation   object            3169           0    0.00
4          Catégorie de panne   object            3169           0    0.00
5          Nombre d'heures MO  float64            3167           2    0.06
6      Coût total intervention  float64            3169           0    0.00
7              [MO interne].Nom   object            3169           0    0.00
8          [MO interne].Prénom   object            3167           2    0.06
9  [MO interne].Nombre d'heures  float64            3167           2    0.06

   Unique Values Sample Value
0             10  Hydraulique
1            323         1.08
2            348   29/03/2025
```

```
3              10     machine1
4              10  Hydraulique
5             185         1.00
6             863        56.00
7               1         tech
8              10          one
9              94         1.00
```

[20]:
```
plot_missing_values(df_amdec, title="AMDEC - Missing Values",
                    save_path=FIGURES_DIR / "amdec_missing_before.png")
```

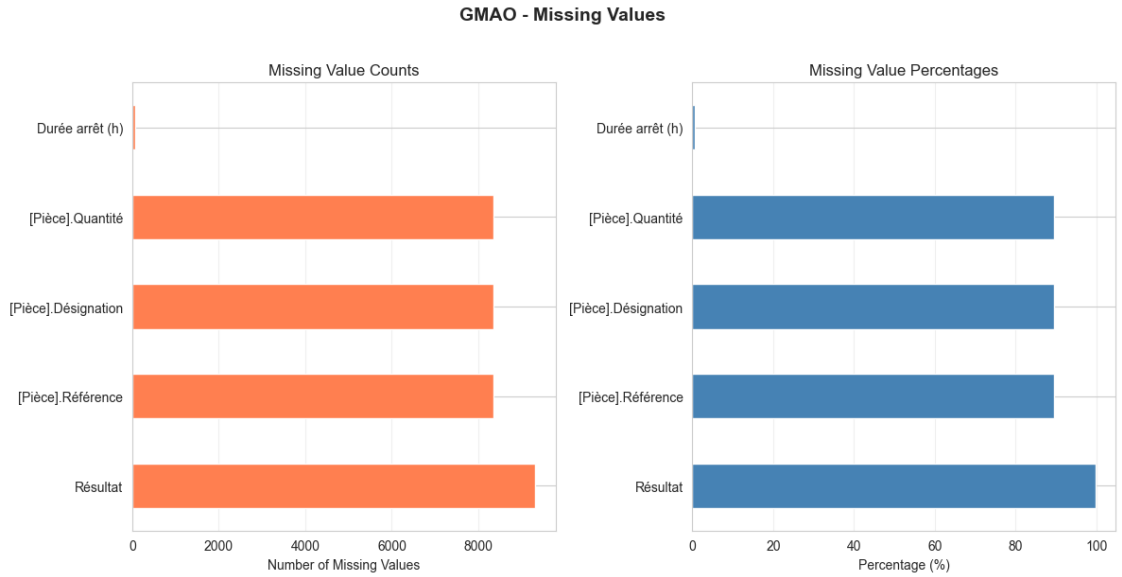
```
2025-11-15 09:26:51 | INFO     |
eda_utils:plot_missing_values - Generating missing values

plot
2025-11-15 09:26:51 | SUCCESS  |
eda_utils:plot_missing_values - Saved plot to c:\User

s\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\amdec_missing_befor

e.png
```



[21]:
```
plot_missing_values(df_gmao, title="GMAO - Missing Values",
                    save_path=FIGURES_DIR / "gmao_missing_before.png")
```

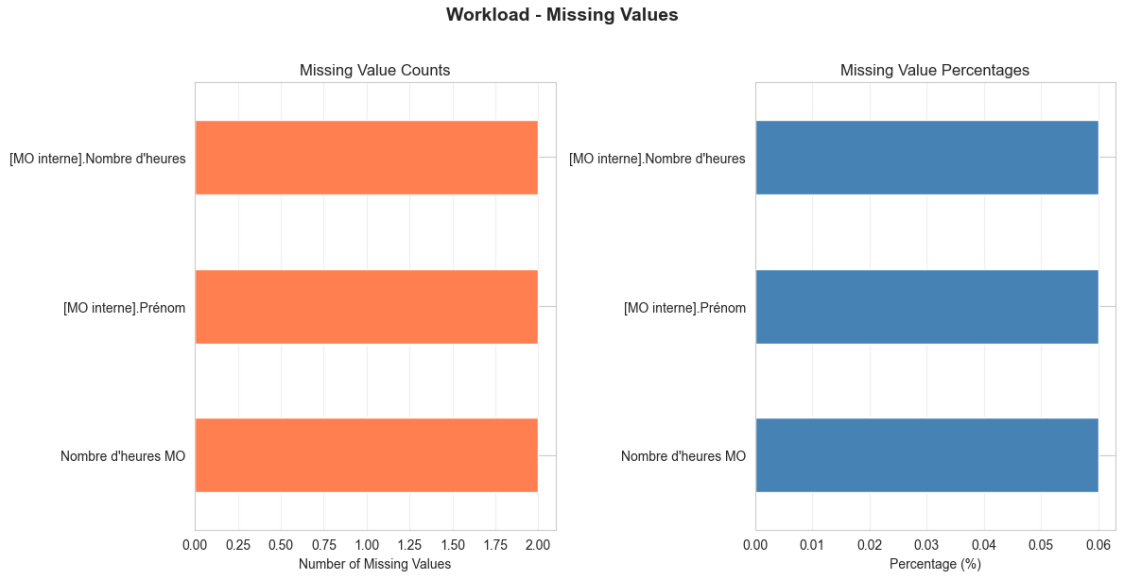
```
2025-11-15 09:27:01 | INFO     |
eda_utils:plot_missing_values - Generating missing values

plot
2025-11-15 09:27:02 | SUCCESS  |
```

eda_utils:plot_missing_values - Saved plot to c:\User
s\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\gmao_missing_before
.png



[22]:
```
plot_missing_values(df_workload, title="Workload - Missing Values",
                    save_path=FIGURES_DIR / "workload_missing_before.png")
```

2025-11-15 09:27:11 | INFO | eda_utils:plot_missing_values - Generating missing values plot
2025-11-15 09:27:12 | SUCCESS | eda_utils:plot_missing_values - Saved plot to c:\Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\workload_missing_before.png



[29]:
```
numeric_cols = df_workload.select_dtypes(include=[np.number]).columns.tolist()
df_workload_clean = handle_missing_values(df_workload, strategy='median',␣
↪columns=numeric_cols)
```

```
2025-11-15 09:31:06 | INFO     |
preprocessing:handle_missing_values - Handling missing
values - Strategy: median
2025-11-15 09:31:06 | INFO     |
preprocessing:handle_missing_values -   Filled 4 numeric
columns with median
2025-11-15 09:31:06 | SUCCESS  |
preprocessing:handle_missing_values -   Shape: (3169,
10) → (3169, 10) (Removed 0 rows, 0 columns)
```

[30]:
```
if 'Durée arrêt (h)' in df_amdec_clean.columns:
        outliers_amdec = detect_outliers(
            df_amdec_clean,
            columns=['Durée arrêt (h)'],
            method='zscore',
            threshold=OUTLIER_THRESHOLD
        )

numeric_workload_cols = df_workload_clean.select_dtypes(include=[np.number]).
↪columns.tolist()
if len(numeric_workload_cols) > 0:
    outliers_workload = detect_outliers(
```

```
            df_workload_clean,
            columns=numeric_workload_cols,
            method='zscore',
            threshold=OUTLIER_THRESHOLD
        )
```

```
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers - Detecting outliers using

zscore method
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers -   Durée arrêt (h): 21

outliers detected (1.09%)
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers - Detecting outliers using

zscore method
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers -   Durée arrêt (h): 42

outliers detected (1.33%)
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers -   Nombre d'heures MO: 103

outliers detected (3.25%)
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers -   Coût total intervention:

23 outliers detected (0.73%)
2025-11-15 09:31:08 | INFO     |
preprocessing:detect_outliers -   [MO interne].Nombre

d'heures: 147 outliers detected (4.64%)
```

[32]:
```
amdec_summary = generate_summary_statistics(df_amdec)
gmao_summary = generate_summary_statistics(df_gmao)
workload_summary = generate_summary_statistics(df_workload)
```

```
2025-11-15 09:33:02 | INFO     |
eda_utils:generate_summary_statistics - Generating summary

statistics
2025-11-15 09:33:02 | INFO     |
eda_utils:generate_summary_statistics - Generating summary

statistics
2025-11-15 09:33:02 | INFO     |
eda_utils:generate_summary_statistics - Generating summary

statistics
```

[33]:
```
amdec_summary
```

[33]:

	count	mean	std	min	25%	50%	75%	max
Durée arrêt (h)	1942.00	16.70	125.62	0.00	0.50	0.83	2.17	2558.50
[Pièce].Quantité	166.00	1.22	0.64	1.00	1.00	1.00	1.00	7.00
[Pièce].Prix total	166.00	1078.90	3308.85	0.00	46.30	216.37	607.76	26000.00

	variance	skewness	kurtosis	missing	missing_%
Durée arrêt (h)	15780.10	15.39	279.88	0	0.00
[Pièce].Quantité	0.41	5.45	41.38	1776	91.45
[Pièce].Prix total	10948480.59	5.08	28.38	1776	91.45

[34]:
```
gmao_summary
```

[34]:

	count	mean	std	min	25%	50%	75%	max
Durée arrêt (h)	9285.00	46.90	276.34	0.00	0.58	1.33	4.50	3855.00
Coût matériel	9341.00	834.31	5364.93	0.00	0.00	0.00	0.00	68736.30
[Pièce].Quantité	985.00	1.35	1.00	1.00	1.00	1.00	1.00	12.00

	variance	skewness	kurtosis	missing	missing_%
Durée arrêt (h)	76361.56	8.44	77.95	56	0.60
Coût matériel	28782494.36	9.39	103.20	0	0.00
[Pièce].Quantité	0.99	4.35	25.54	8356	89.46

[35]:
```
workload_summary
```

[35]:

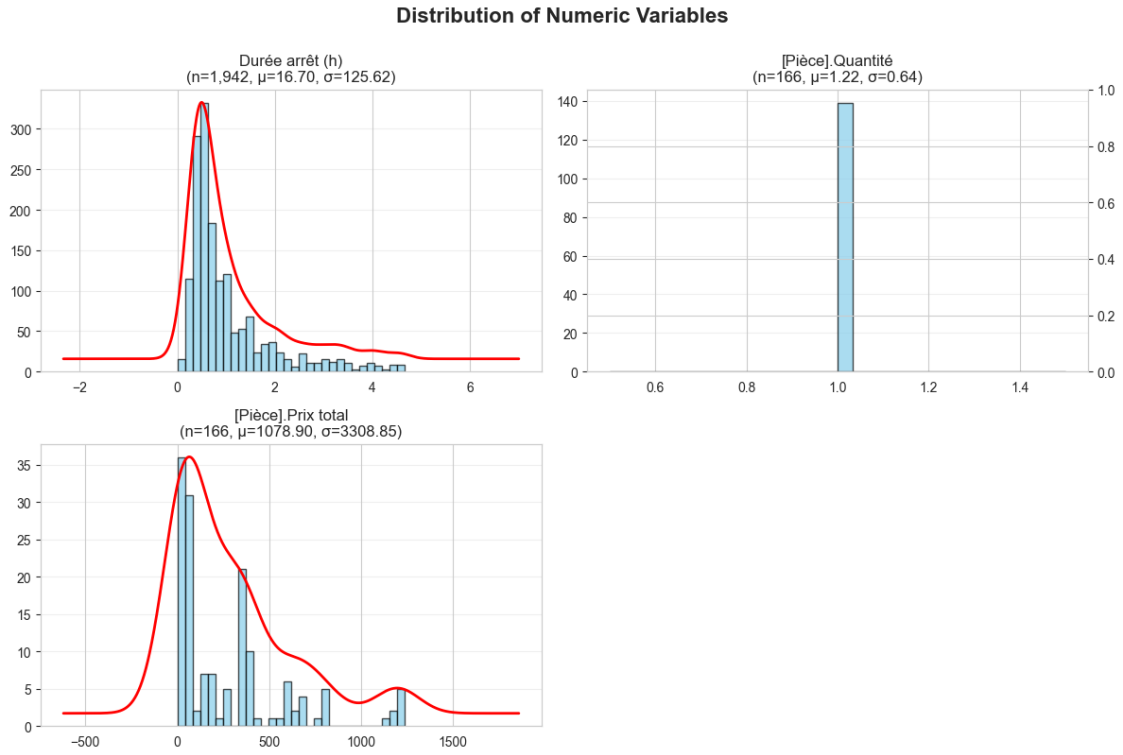
	count	mean	std	min	25%	50%	75%	max
Durée arrêt (h)	3169.00	49.91	232.73	0.00	0.55	1.17	6.00	2558.50
Nombre d'heures MO	3167.00	11.62	30.18	0.08	0.50	1.33	5.00	255.00
Coût total intervention	3169.00	1851.90	8488.08	0.00	12.50	36.89	189.32	99489.43
[MO interne].Nombre d'heures	3167.00	2.37	4.45	0.08	0.42	0.67	2.00	36.00

	variance	skewness	kurtosis	missing	missing_%
Durée arrêt (h)	54162.26	8.34	79.80	0	0.00
Nombre d'heures MO	910.67	4.04	18.99	2	0.06
Coût total intervention	72047430.84	9.10	97.76	0	0.00
[MO interne].Nombre d'heures	19.82	3.60	14.99	2	0.06

[38]:
```
plot_distributions(df_amdec, ncols=2, figsize=(12, 8),
                   save_path=FIGURES_DIR / "amdec_distributions.png")
```

```
2025-11-15 09:34:59 | INFO     |
```

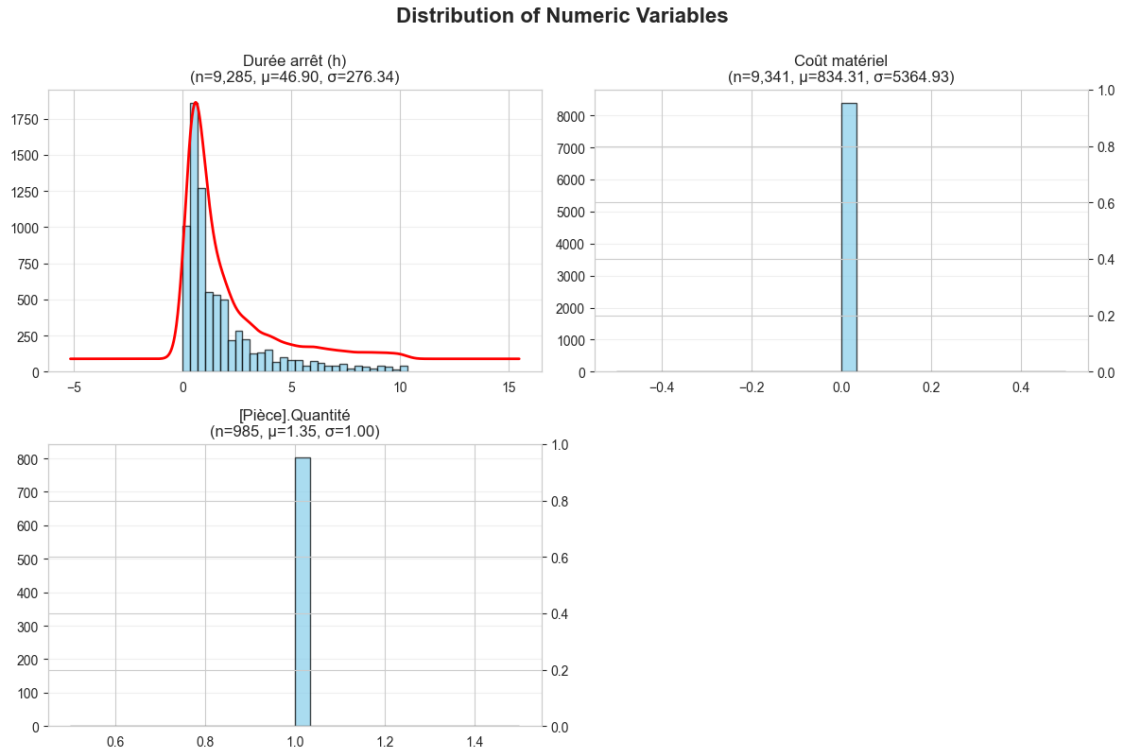
```
eda_utils:plot_distributions - Generating distribution
plots
2025-11-15 09:34:59 | WARNING  |
eda_utils:plot_distributions - Skipping KDE plot for
[Pièce].Quantité: Data is constant (std is 0).
2025-11-15 09:35:00 | SUCCESS  |
eda_utils:plot_distributions - Saved plot to c:\Users
\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\amdec_distributions.
png
```



[39]:
```
plot_distributions(df_gmao, ncols=2, figsize=(12, 8),
                   save_path=FIGURES_DIR / "gmao_distributions.png")
```

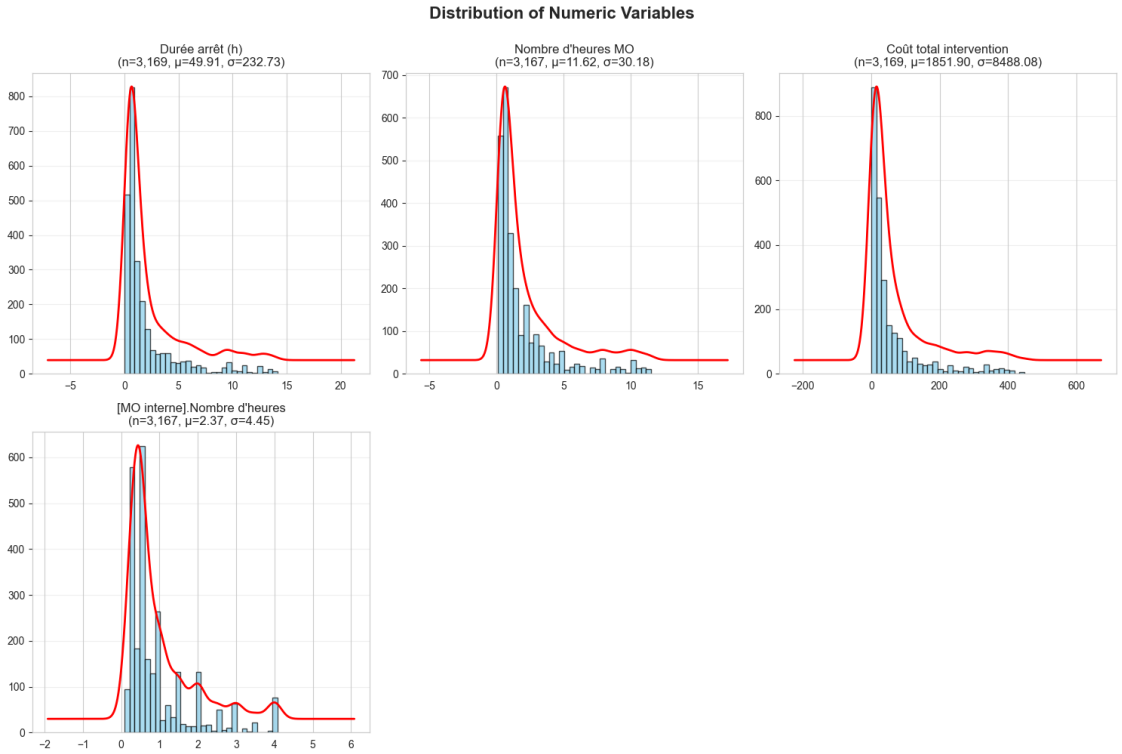
```
2025-11-15 09:35:08 | INFO     |
eda_utils:plot_distributions - Generating distribution
plots
2025-11-15 09:35:08 | WARNING  |
eda_utils:plot_distributions - Skipping KDE plot for
Coût matériel: Data is constant (std is 0).
2025-11-15 09:35:08 | WARNING  |
```

```
eda_utils:plot_distributions - Skipping KDE plot for
[Pièce].Quantité: Data is constant (std is 0).
2025-11-15 09:35:09 | SUCCESS  |
eda_utils:plot_distributions - Saved plot to c:\Users
\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\gmao_distributions.p
ng
```



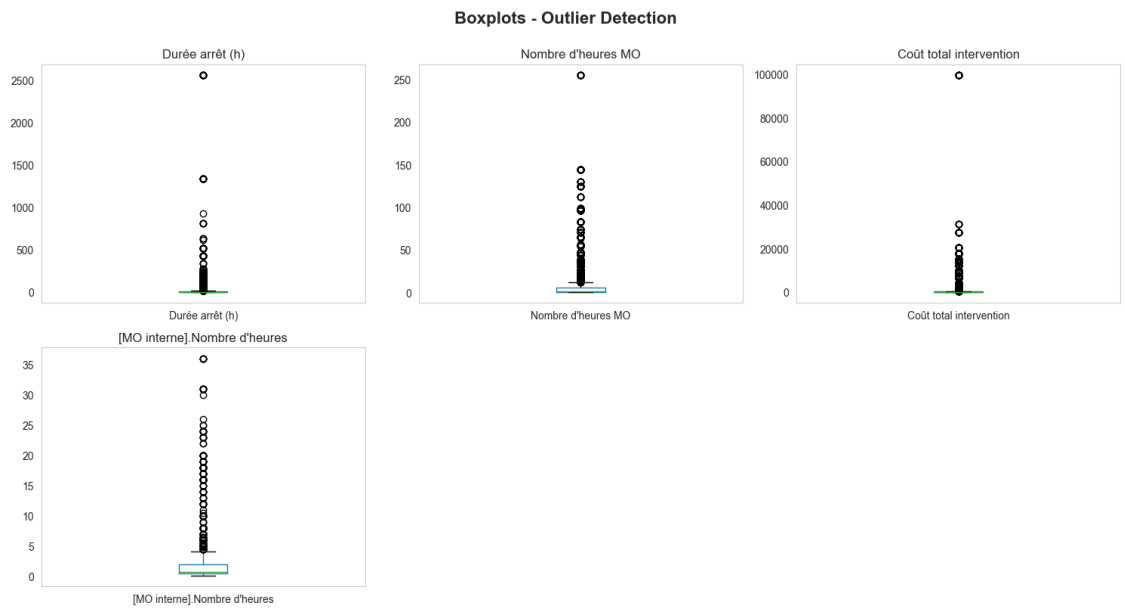
[41]:
```
plot_distributions(df_workload, ncols=3, figsize=(15, 10),
                   save_path=FIGURES_DIR / "workload_distributions.png")
```

```
2025-11-15 09:35:35 | INFO     |
eda_utils:plot_distributions - Generating distribution
plots
2025-11-15 09:35:37 | SUCCESS  |
eda_utils:plot_distributions - Saved plot to c:\Users
\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\workload_distributio
ns.png
```



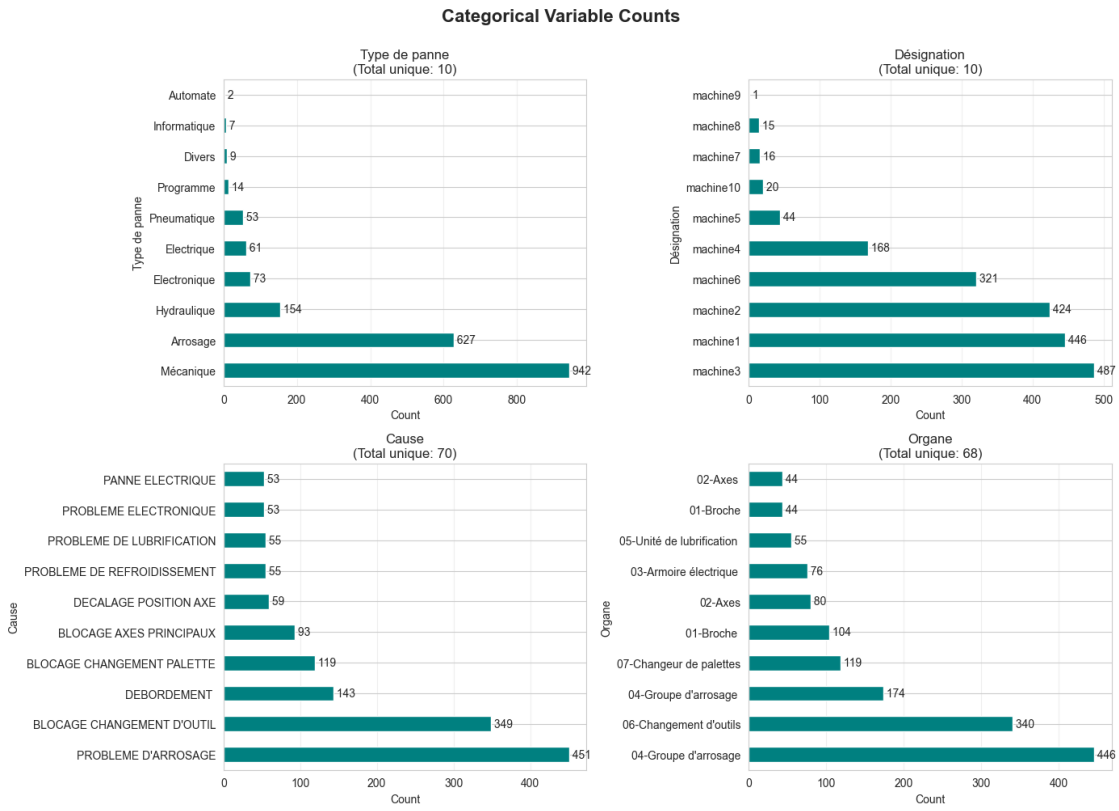
[42]:
```
plot_boxplots(df_workload, ncols=3, figsize=(15, 8),
                  save_path=FIGURES_DIR / "workload_boxplots.png")
```

```
2025-11-15 09:35:53 | INFO     |
eda_utils:plot_boxplots - Generating boxplots
2025-11-15 09:35:54 | SUCCESS  |
eda_utils:plot_boxplots - Saved plot to c:\Users\amin
e\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\workload_boxplots.png
```

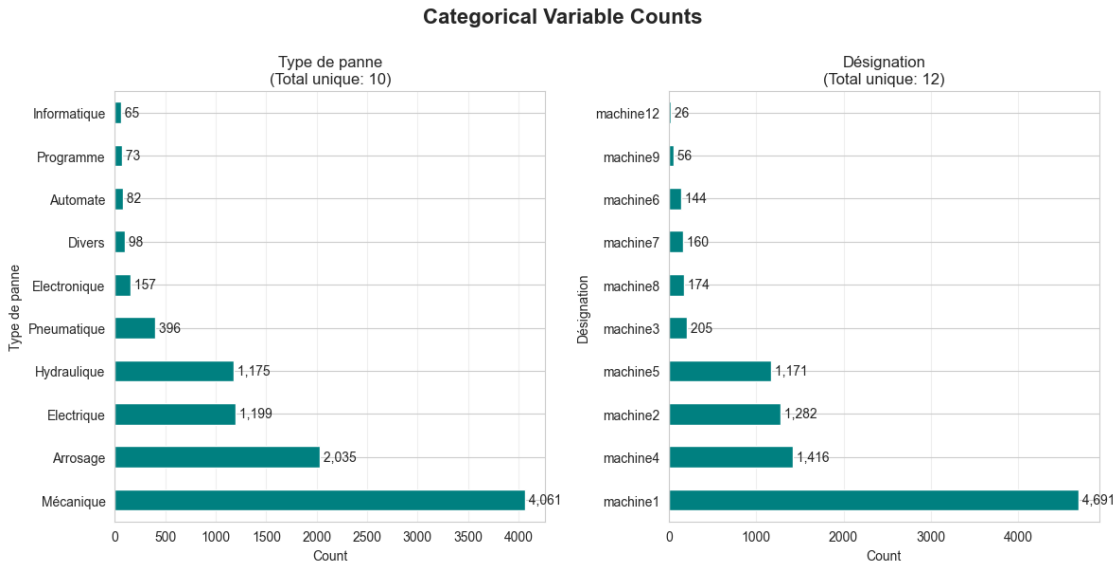


[43]:
```
categorical_cols_amdec = ['Type de panne', 'Désignation', 'Cause', 'Organe']
available_cats_amdec = [col for col in categorical_cols_amdec if col in␣
 ↪df_amdec.columns]
if available_cats_amdec:
    plot_categorical_counts(df_amdec, columns=available_cats_amdec, top_n=10,
                            ncols=2, figsize=(14, 10),
                            save_path=FIGURES_DIR / "amdec_categorical.png")

categorical_cols_gmao = ['Type de panne', 'Désignation']
available_cats_gmao = [col for col in categorical_cols_gmao if col in df_gmao.
 ↪columns]
if available_cats_gmao:
    plot_categorical_counts(df_gmao, columns=available_cats_gmao, top_n=10,
                            ncols=2, figsize=(12, 6),
                            save_path=FIGURES_DIR / "gmao_categorical.png")
```

```
2025-11-15 09:36:42 | INFO     |
eda_utils:plot_categorical_counts - Generating categorical

counts plots
2025-11-15 09:36:43 | SUCCESS  |
eda_utils:plot_categorical_counts - Saved plot to c:\

Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\amdec_categoric

al.png
```

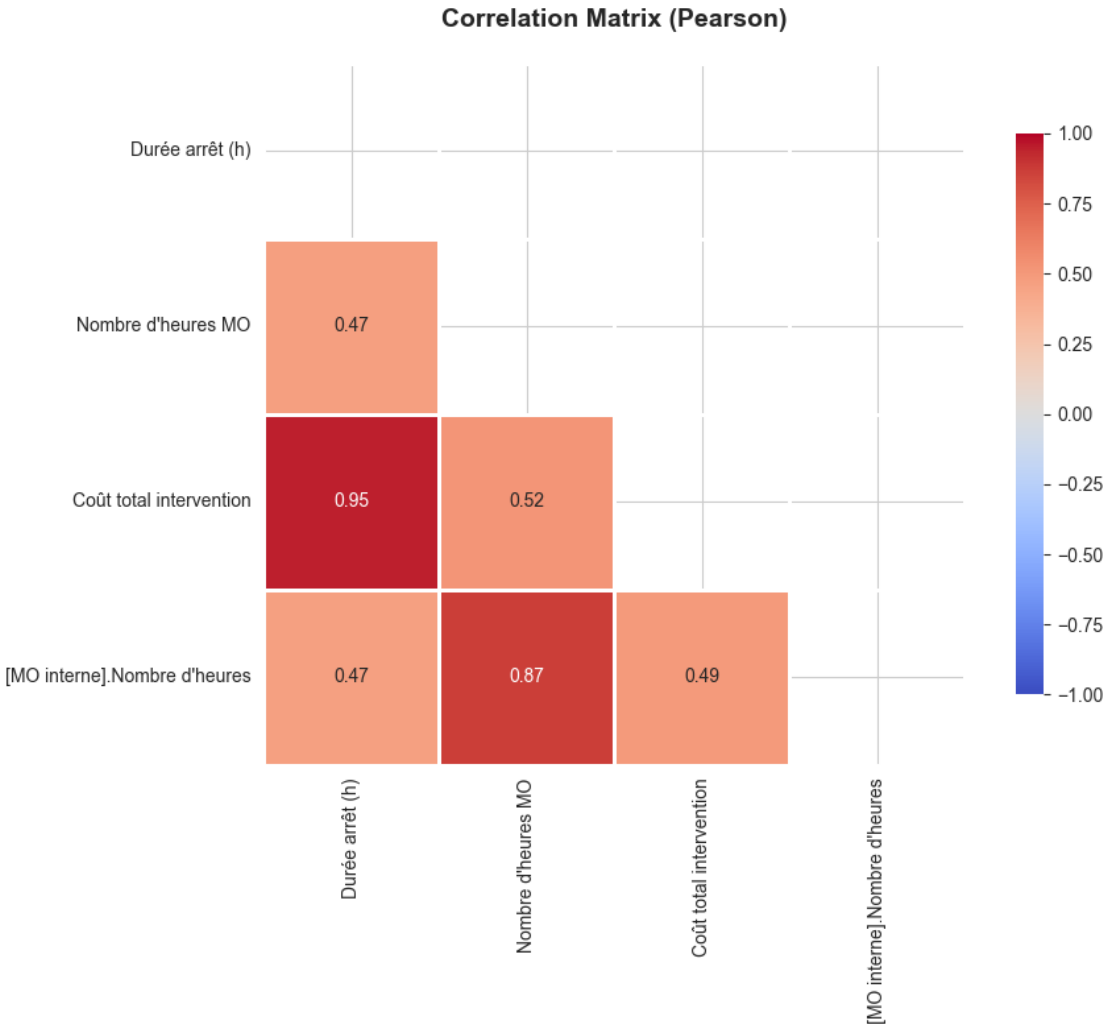



```
2025-11-15 09:36:43 | INFO     |
eda_utils:plot_categorical_counts - Generating categorical
counts plots
2025-11-15 09:36:44 | SUCCESS  |
eda_utils:plot_categorical_counts - Saved plot to c:\
Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\gmao_categorica
l.png
```



[44]:
```
workload_corr = plot_correlation_matrix(df_workload, method='pearson',
                                        figsize=(10, 8),
                                        save_path=FIGURES_DIR /␣
 ↪"workload_correlation.png")
if workload_corr is not None:
        print("\nTop Correlations (absolute value > 0.5):")
        corr_pairs = workload_corr.abs().unstack().sort_values(ascending=False)
        corr_pairs = corr_pairs[corr_pairs < 1.0]
        corr_pairs = corr_pairs[corr_pairs > 0.5]
        if len(corr_pairs) > 0:
            print(corr_pairs.head(10))
print("  Correlation matrix saved")
```

```
2025-11-15 09:37:28 | INFO     |
eda_utils:plot_correlation_matrix - Generating correlation

matrix (pearson)
2025-11-15 09:37:29 | SUCCESS  |
eda_utils:plot_correlation_matrix - Saved plot to c:\

Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures\workload_correl

ation.png
```



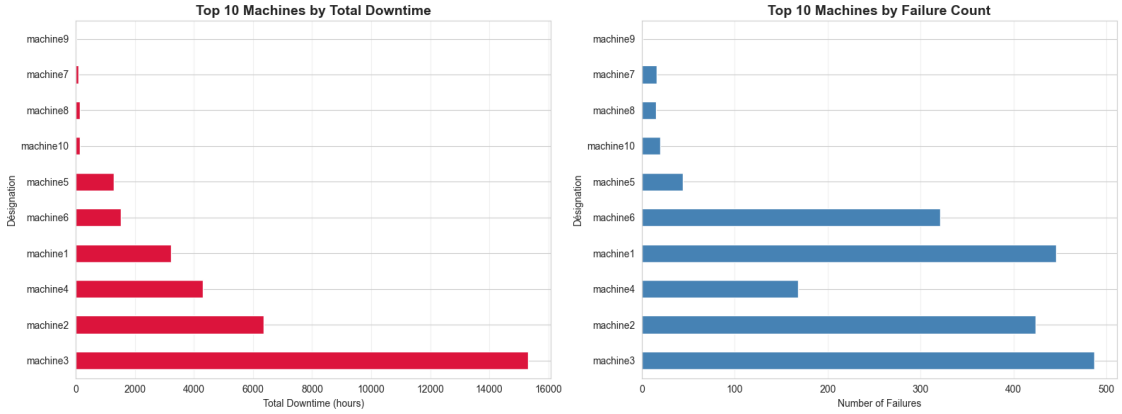

```
Top Correlations (absolute value > 0.5):
Durée arrêt (h)               Coût total intervention        0.95
Coût total intervention       Durée arrêt (h)                0.95
[MO interne].Nombre d'heures  Nombre d'heures MO             0.87
Nombre d'heures MO            [MO interne].Nombre d'heures   0.87
Coût total intervention       Nombre d'heures MO             0.52
Nombre d'heures MO            Coût total intervention        0.52
dtype: float64
  Correlation matrix saved
```

[48]:
```
if 'Désignation' in df_amdec.columns and 'Durée arrêt (h)' in df_amdec.columns:
        machine_stats = df_amdec.groupby('Désignation').agg({
            'Durée arrêt (h)': ['count', 'sum', 'mean', 'median'],
```

```
            'Type de panne': lambda x: x.mode()[0] if not x.mode().empty else␣
↪None
        }).round(2)

        machine_stats.columns = ['Failure_Count', 'Total_Downtime_h',␣
↪'Avg_Downtime_h',
                                 'Median_Downtime_h', 'Most_Common_Failure']
        machine_stats = machine_stats.sort_values('Total_Downtime_h',␣
↪ascending=False)

        print("\nTop 10 Machines by Total Downtime:")
        print(machine_stats.head(10))

        # Visualize
        fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 6))

        machine_stats['Total_Downtime_h'].head(10).plot(kind='barh', ax=ax1,␣
↪color='crimson')
        ax1.set_title('Top 10 Machines by Total Downtime', fontsize=14,␣
↪fontweight='bold')
        ax1.set_xlabel('Total Downtime (hours)')
        ax1.grid(axis='x', alpha=0.3)

        machine_stats['Failure_Count'].head(10).plot(kind='barh', ax=ax2,␣
↪color='steelblue')
        ax2.set_title('Top 10 Machines by Failure Count', fontsize=14,␣
↪fontweight='bold')
        ax2.set_xlabel('Number of Failures')
        ax2.grid(axis='x', alpha=0.3)

        plt.tight_layout()
        plt.savefig(FIGURES_DIR / "machine_analysis.png", dpi=300,␣
↪bbox_inches='tight')
        plt.show()
        plt.close()
```

```
Top 10 Machines by Total Downtime:
             Failure_Count  Total_Downtime_h  Avg_Downtime_h  \
Désignation
machine3               487          15326.08           31.47
machine2               424           6371.57           15.03
machine4               168           4306.03           25.63
machine1               446           3236.22            7.26
machine6               321           1524.92            4.75
machine5                44           1289.27           29.30
machine10               20            143.53            7.18
```

```
machine8                15          140.08           9.34
machine7                16           81.13           5.07
machine9                 1           19.00          19.00

             Median_Downtime_h Most_Common_Failure
Désignation
machine3                  1.03          Mécanique
machine2                  0.75           Arrosage
machine4                  0.67           Arrosage
machine1                  0.83          Mécanique
machine6                  0.67           Arrosage
machine5                  1.50          Mécanique
machine10                 1.33          Mécanique
machine8                  2.00        Electronique
machine7                  0.91         Hydraulique
machine9                 19.00          Mécanique
```

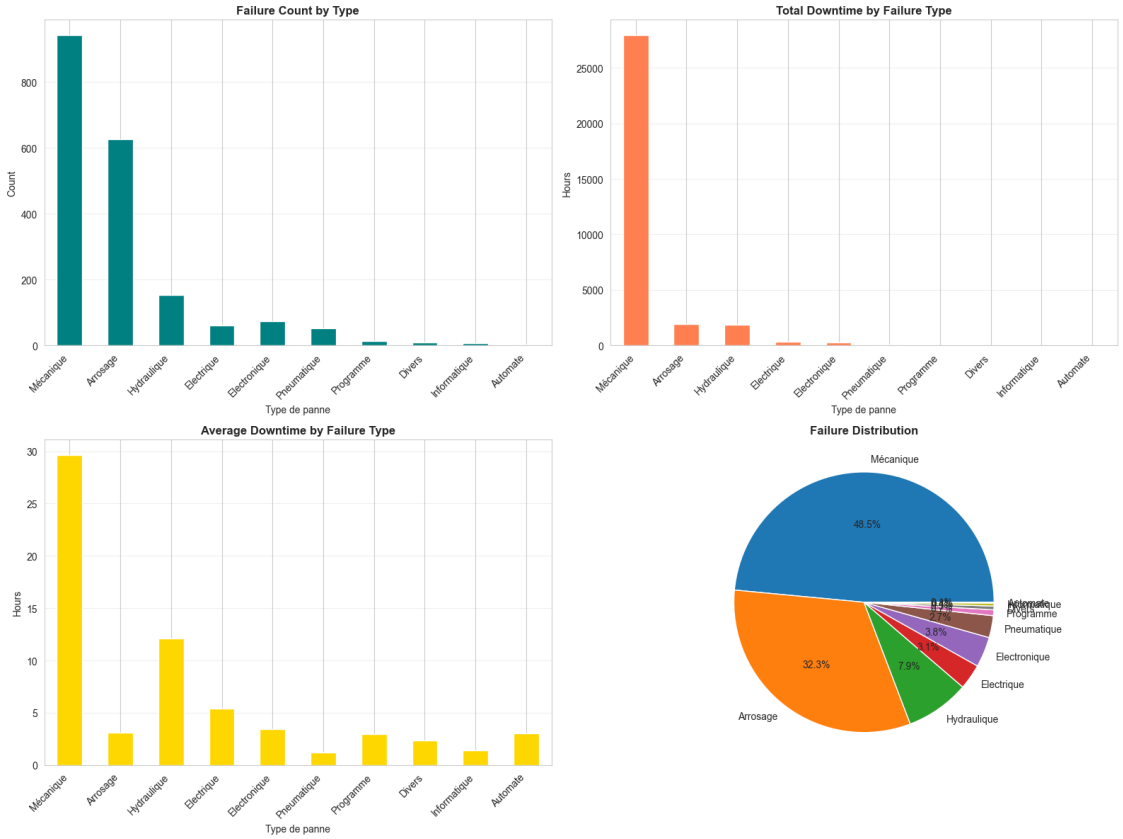


[53]:
```
if 'Type de panne' in df_amdec.columns and 'Durée arrêt (h)' in df_amdec.
↪columns:
        failure_stats = df_amdec.groupby('Type de panne').agg({
            'Durée arrêt (h)': ['count', 'sum', 'mean', 'std']
        }).round(2)

        failure_stats.columns = ['Count', 'Total_Downtime', 'Avg_Downtime',␣
↪'Std_Downtime']
        failure_stats = failure_stats.sort_values('Total_Downtime',␣
↪ascending=False)

        print("\nFailure Type Statistics:")
        print(failure_stats)

        # Visualize
```

```
        fig, axes = plt.subplots(2, 2, figsize=(16, 12))

        failure_stats['Count'].plot(kind='bar', ax=axes[0, 0], color='teal')
        axes[0, 0].set_title('Failure Count by Type', fontsize=12,␣
↪fontweight='bold')
        axes[0, 0].set_ylabel('Count')
        axes[0, 0].grid(axis='y', alpha=0.3)
        plt.setp(axes[0, 0].xaxis.get_majorticklabels(), rotation=45,␣
↪ha='right')

        failure_stats['Total_Downtime'].plot(kind='bar', ax=axes[0, 1],␣
↪color='coral')
        axes[0, 1].set_title('Total Downtime by Failure Type', fontsize=12,␣
↪fontweight='bold')
        axes[0, 1].set_ylabel('Hours')
        axes[0, 1].grid(axis='y', alpha=0.3)
        plt.setp(axes[0, 1].xaxis.get_majorticklabels(), rotation=45,␣
↪ha='right')

        failure_stats['Avg_Downtime'].plot(kind='bar', ax=axes[1, 0],␣
↪color='gold')
        axes[1, 0].set_title('Average Downtime by Failure Type', fontsize=12,␣
↪fontweight='bold')
        axes[1, 0].set_ylabel('Hours')
        axes[1, 0].grid(axis='y', alpha=0.3)
        plt.setp(axes[1, 0].xaxis.get_majorticklabels(), rotation=45,␣
↪ha='right')

        failure_stats['Count'].plot(kind='pie', ax=axes[1, 1], autopct='%1.
↪1f%%')
        axes[1, 1].set_title('Failure Distribution', fontsize=12,␣
↪fontweight='bold')
        axes[1, 1].set_ylabel('')

        plt.tight_layout()
        plt.savefig(FIGURES_DIR / "failure_type_analysis.png", dpi=300,␣
↪bbox_inches='tight')
        plt.show()
        plt.close()
```

```
Failure Type Statistics:
                Count  Total_Downtime  Avg_Downtime  Std_Downtime
Type de panne
Mécanique         942        27925.58         29.64        177.34
Arrosage          627         1922.55          3.07         16.76
Hydraulique       154         1865.05         12.11         58.34
```

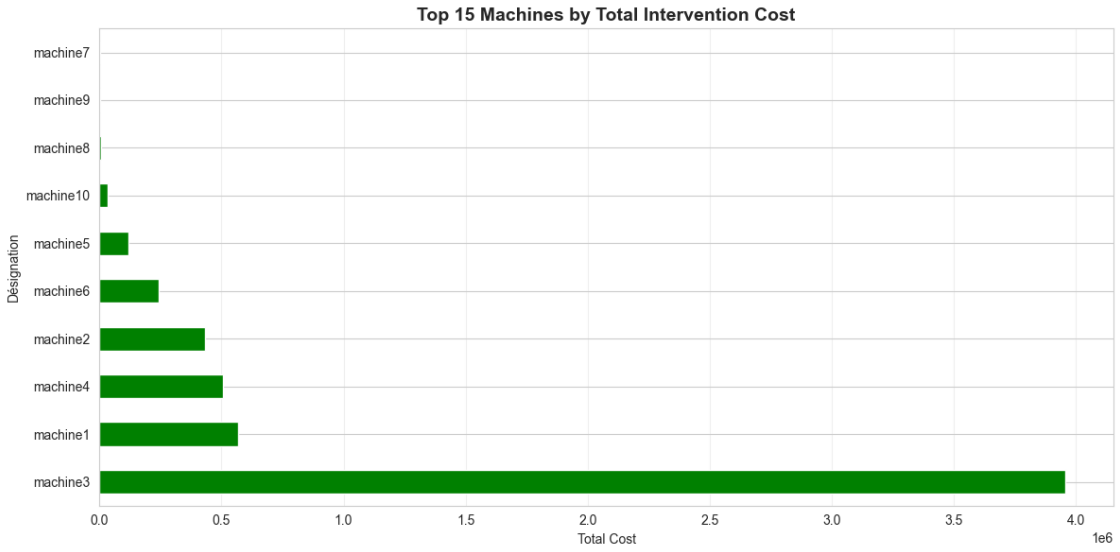
Electrique	61	331.52	5.43	13.30
Electronique	73	249.23	3.41	11.98
Pneumatique	53	64.78	1.22	2.36
Programme	14	41.75	2.98	4.57
Divers	9	21.28	2.36	3.05
Informatique	7	10.00	1.43	0.79
Automate	2	6.08	3.04	2.77



[52]:
```
if 'Coût total intervention' in df_workload.columns:
        total_cost = df_workload['Coût total intervention'].sum()
        avg_cost = df_workload['Coût total intervention'].mean()
        median_cost = df_workload['Coût total intervention'].median()

        print(f"\nTotal Intervention Cost: {total_cost:,.2f}")
        print(f"Average Cost per Intervention: {avg_cost:,.2f}")
        print(f"Median Cost per Intervention: {median_cost:,.2f}")

        if 'Désignation' in df_workload.columns:
            cost_by_machine = df_workload.groupby('Désignation')['Coût total␣
↪intervention'].agg(
                ['sum', 'mean', 'count']
```

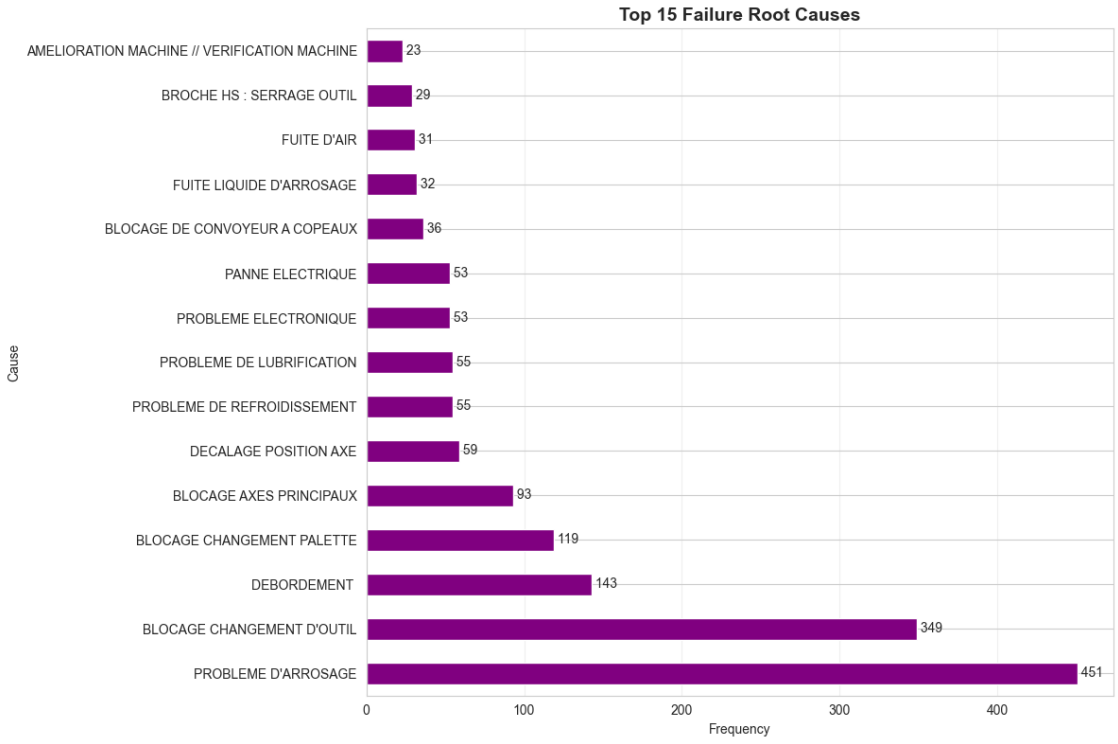
```
            ).round(2)
            cost_by_machine.columns = ['Total_Cost', 'Avg_Cost',␣
↪'Intervention_Count']
            cost_by_machine = cost_by_machine.sort_values('Total_Cost',␣
↪ascending=False)

            print("\nTop 10 Machines by Total Cost:")
            print(cost_by_machine.head(10))

            # Visualize
            fig, ax = plt.subplots(figsize=(12, 6))
            cost_by_machine['Total_Cost'].head(15).plot(kind='barh', ax=ax,␣
↪color='green')
            ax.set_title('Top 15 Machines by Total Intervention Cost',
                         fontsize=14, fontweight='bold')
            ax.set_xlabel('Total Cost')
            ax.grid(axis='x', alpha=0.3)
            plt.tight_layout()
            plt.savefig(FIGURES_DIR / "cost_analysis.png", dpi=300,␣
↪bbox_inches='tight')
            plt.show()
            plt.close()
```

```
Total Intervention Cost: 5,868,680.56
Average Cost per Intervention: 1,851.90
Median Cost per Intervention: 36.89

Top 10 Machines by Total Cost:
             Total_Cost  Avg_Cost  Intervention_Count
Désignation
machine3     3955805.34   4439.74                 891
machine1      567420.85    857.13                 662
machine4      505688.31   2022.75                 250
machine2      433546.18    611.49                 709
machine6      242524.05    536.56                 452
machine5      119414.80   1341.74                  89
machine10      32910.93    645.31                  51
machine8        8175.87    255.50                  32
machine9        2159.46    719.82                   3
machine7        1034.77     34.49                  30
```



[51]:
```
if 'Cause' in df_amdec.columns:
        cause_counts = df_amdec['Cause'].value_counts().head(15)

        print("\nTop 15 Failure Causes:")
        cause_df = pd.DataFrame({
            'Cause': cause_counts.index,
            'Count': cause_counts.values,
            'Percentage': (cause_counts.values / len(df_amdec) * 100).round(2)
        })
        print(cause_df)

        # Visualize
        fig, ax = plt.subplots(figsize=(12, 8))
        cause_counts.plot(kind='barh', ax=ax, color='purple')
        ax.set_title('Top 15 Failure Root Causes', fontsize=14,␣
↪fontweight='bold')
        ax.set_xlabel('Frequency')
        ax.grid(axis='x', alpha=0.3)

        for i, v in enumerate(cause_counts.values):
            ax.text(v, i, f' {v}', va='center')

        plt.tight_layout()
        plt.savefig(FIGURES_DIR / "root_cause_analysis.png", dpi=300,␣
↪bbox_inches='tight')
        plt.show()
        plt.close()
```

```
Top 15 Failure Causes:
                                    Cause  Count  Percentage
0                        PROBLEME D'ARROSAGE    451       23.22
1                  BLOCAGE CHANGEMENT D'OUTIL    349       17.97
2                                DEBORDEMENT     143        7.36
3                  BLOCAGE CHANGEMENT PALETTE    119        6.13
4                     BLOCAGE AXES PRINCIPAUX     93        4.79
5                       DECALAGE POSITION AXE     59        3.04
6                PROBLEME DE REFROIDISSEMENT     55        2.83
7                  PROBLEME DE LUBRIFICATION     55        2.83
8                      PROBLEME ELECTRONIQUE     53        2.73
9                           PANNE ELECTRIQUE     53        2.73
10             BLOCAGE DE CONVOYEUR A COPEAUX     36        1.85
11                   FUITE LIQUIDE D'ARROSAGE     32        1.65
12                              FUITE D'AIR     31        1.60
13                BROCHE HS : SERRAGE OUTIL     29        1.49
14  AMELIORATION MACHINE // VERIFICATION MACHINE     23        1.18
```



[55]:
```
print(f"   • AMDEC: {df_amdec.shape[0]:,}")
print(f"   • GMAO: {df_gmao.shape[0]:,}")
print(f"   • Workload: {df_workload.shape[0]:,}")
```

- AMDEC: 1,942

- GMAO: 9,341
- Workload: 3,169

[62]:
```
if 'Date intervention' in df_amdec.columns:
        df_amdec['Date intervention'] = pd.to_datetime(
            df_amdec['Date intervention'],
            dayfirst=True,
            errors='coerce'
        )
        min_date = df_amdec['Date intervention'].min()
        max_date = df_amdec['Date intervention'].max()
        print(f"\n TIME PERIOD:")
        print(f"   {min_date.strftime('%Y-%m-%d')} to {max_date.
 ↪strftime('%Y-%m-%d')}")
        print(f"   Duration: {(max_date - min_date).days} days")

if 'Durée arrêt (h)' in df_amdec.columns:
        total_downtime = df_amdec['Durée arrêt (h)'].sum()
        print(f"\n  TOTAL DOWNTIME:")
        print(f"   {total_downtime:,.2f} hours")
        print(f"   {total_downtime/24:,.2f} days")
        print(f"   {total_downtime/24/365:,.2f} years")

if 'Coût total intervention' in df_workload.columns:
        total_cost = df_workload['Coût total intervention'].sum()
        print(f"\n TOTAL INTERVENTION COST:")
        print(f"   {total_cost:,.2f}")

if 'Type de panne' in df_amdec.columns:
        most_common = df_amdec['Type de panne'].mode()
        if not most_common.empty:
            print(f"\n MOST COMMON FAILURE TYPE:")
            print(f"   {most_common[0]}")

if 'Désignation' in df_amdec.columns and 'Durée arrêt (h)' in df_amdec.columns:
        problem_machine = df_amdec.groupby('Désignation')['Durée arrêt (h)'].
 ↪sum().idxmax()
        problem_downtime = df_amdec.groupby('Désignation')['Durée arrêt (h)'].
 ↪sum().max()
        print(f"\n MACHINE WITH HIGHEST DOWNTIME:")
        print(f"   {problem_machine} ({problem_downtime:,.2f} hours)")
```

```
TIME PERIOD:
  2024-04-01 to 2025-03-29
  Duration: 362 days

 TOTAL DOWNTIME:
```

```
  32,437.83 hours
  1,351.58 days
  3.70 years

TOTAL INTERVENTION COST:
  5,868,680.56

MOST COMMON FAILURE TYPE:
  Mécanique

MACHINE WITH HIGHEST DOWNTIME:
  machine3 (15,326.08 hours)
```

[60]:
```
print(f"   • Cleaned datasets: {CLEANED_DATA_DIR}")
print(f"   • Visualizations: {FIGURES_DIR}")
print(f"   • Summary report: {OUTPUTS_DIR / 'GMAO_Analysis_Summary.xlsx'}")
```

```
   • Cleaned datasets:
c:\Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\cleaned_data
   • Visualizations:
c:\Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs\figures
   • Summary report: c:\Users\amine\Documents\ACADEMIC\DATA\notebooks\..\outputs
\GMAO_Analysis_Summary.xlsx
```