In [19]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
import warnings
```

In [20]:
```
warnings.filterwarnings('ignore')
plt.style.use('seaborn-v0_8-whitegrid')
RANDOM_STATE = 42
df = pd.read_excel('D:\Data\D07-parkinsons.data.xlsx')
print(f"✓ Kích thước dữ liệu: {df.shape[0]} hàng × {df.shape[1]} cột")
if 'name' in df.columns:
    df = df.drop(columns=['name'])
    print(" Đã loại bỏ cột 'name'.")
```

```
✓ Kích thước dữ liệu: 195 hàng × 24 cột
 Đã loại bỏ cột 'name'.
```

In [21]:
```
df.to_csv(csv_path, index=False)
print(f" Đã lưu file CSV: {csv_path}")
df = pd.read_csv(csv_path)
print("\n 5 dòng đầu tiên của dữ liệu:")
print(df.head(), "\n")
```

```
Đã lưu file CSV: data.csv

5 dòng đầu tiên của dữ liệu:
   MDVP:Fo(Hz)  MDVP:Fhi(Hz)  MDVP:Flo(Hz)  MDVP:Jitter(%)  MDVP:Jitter(Abs)  \
0      119.992       157.302        74.997         0.00784           0.00007
1      122.400       148.650       113.819         0.00968           0.00008
2      116.682       131.111       111.555         0.01050           0.00009
3      116.676       137.871       111.366         0.00997           0.00009
4      116.014       141.781       110.655         0.01284           0.00011

   MDVP:RAP  MDVP:PPQ  Jitter:DDP  MDVP:Shimmer  MDVP:Shimmer(dB)  ...  \
0   0.00370   0.00554     0.01109       0.04374             0.426  ...
1   0.00465   0.00696     0.01394       0.06134             0.626  ...
2   0.00544   0.00781     0.01633       0.05233             0.482  ...
3   0.00502   0.00698     0.01505       0.05492             0.517  ...
4   0.00655   0.00908     0.01966       0.06425             0.584  ...

   Shimmer:DDA      NHR     HNR  status      RPDE       DFA   spread1  \
0      0.06545  0.02211  21.033       1  0.414783  0.815285 -4.813031
1      0.09403  0.01929  19.085       1  0.458359  0.819521 -4.075192
2      0.08270  0.01309  20.651       1  0.429895  0.825288 -4.443179
3      0.08771  0.01353  20.644       1  0.434969  0.819235 -4.117501
4      0.10470  0.01767  19.649       1  0.417356  0.823484 -3.747787

    spread2        D2       PPE
0  0.266482  2.301442  0.284654
1  0.335590  2.486855  0.368674
2  0.311173  2.342259  0.332634
3  0.334147  2.405554  0.368975
4  0.234513  2.332180  0.410335

[5 rows x 23 columns]
```

In [22]:

```python
print("a) Kích thước dữ liệu:")
print(" - Số hàng, cột:", df.shape)
print(" - Số chiều:", df.ndim)
print(" - Tổng số ô:", df.size, "\n")
# Kiểu dữ liệu của các thuộc tính
print("b) Kiểu dữ liệu của các thuộc tính:")
print(df.dtypes, "\n")

# Số lượng thực thể theo nhãn
label_col = "status"
if label_col in df.columns:
    print("c) Số lượng thực thể theo nhãn:")
    print(df[label_col].value_counts())
    print("\nTỷ lệ phần trăm mỗi nhãn:")
    print(df[label_col].value_counts(normalize=True).mul(100).round(2))
    print()
else:
    print(" Không tìm thấy cột nhãn 'status'! Kiểm tra lại dữ liệu.\n")

# Thống kê min/max/mean của các cột số
numeric_cols = df.select_dtypes(include=[np.number]).columns
print(f"d) Các cột số ({len(numeric_cols)} cột):\n", list(numeric_cols), "\n")
```

```
stats = pd.DataFrame({
    "min": df[numeric_cols].min(),
    "max": df[numeric_cols].max(),
    "mean": df[numeric_cols].mean().round(4)
})
stats_path = "numeric_columns_stats.csv"
stats.to_csv(stats_path)
print(f" Đã lưu thống kê numeric vào {stats_path}")

print("\n 5 dòng đầu tiên của bảng thống kê numeric:")
print(stats.head())
```

```
a) Kích thước dữ liệu:
 - Số hàng, cột: (195, 23)
 - Số chiều: 2
 - Tổng số ô: 4485

b) Kiểu dữ liệu của các thuộc tính:
MDVP:Fo(Hz)         float64
MDVP:Fhi(Hz)        float64
MDVP:Flo(Hz)        float64
MDVP:Jitter(%)      float64
MDVP:Jitter(Abs)    float64
MDVP:RAP            float64
MDVP:PPQ            float64
Jitter:DDP          float64
MDVP:Shimmer        float64
MDVP:Shimmer(dB)    float64
Shimmer:APQ3        float64
Shimmer:APQ5        float64
MDVP:APQ            float64
Shimmer:DDA         float64
NHR                 float64
HNR                 float64
status                int64
RPDE                float64
DFA                 float64
spread1             float64
spread2             float64
D2                  float64
PPE                 float64
dtype: object

c) Số lượng thực thể theo nhãn:
status
1    147
0     48
Name: count, dtype: int64

Tỷ lệ phần trăm mỗi nhãn:
status
1    75.38
0    24.62
Name: proportion, dtype: float64

d) Các cột số (23 cột):
 ['MDVP:Fo(Hz)', 'MDVP:Fhi(Hz)', 'MDVP:Flo(Hz)', 'MDVP:Jitter(%)', 'MDVP:Jitter(Ab
s)', 'MDVP:RAP', 'MDVP:PPQ', 'Jitter:DDP', 'MDVP:Shimmer', 'MDVP:Shimmer(dB)', 'Shim
mer:APQ3', 'Shimmer:APQ5', 'MDVP:APQ', 'Shimmer:DDA', 'NHR', 'HNR', 'status', 'RPD
E', 'DFA', 'spread1', 'spread2', 'D2', 'PPE']

 Đã lưu thống kê numeric vào numeric_columns_stats.csv

 5 dòng đầu tiên của bảng thống kê numeric:
                         min        max      mean
MDVP:Fo(Hz)        88.333000  260.10500  154.2286
MDVP:Fhi(Hz)      102.145000  592.03000  197.1049
MDVP:Flo(Hz)       65.476000  239.17000  116.3246
```

```
MDVP:Jitter(%)      0.001680    0.03316    0.0062
MDVP:Jitter(Abs)    0.000007    0.00026    0.0000
```

In [23]:
```
label_col = "status"
continuous_cols = df.select_dtypes(include=[np.number]).columns.tolist()
if label_col in continuous_cols:
    continuous_cols.remove(label_col)
X = df[continuous_cols].copy()
y = df[label_col].copy() if label_col in df.columns else None
```

In [24]:
```
if y is not None:
    counts = y.value_counts()
missing_count = X.isnull().sum().sum()
if missing_count > 0:
    print(f" Phát hiện {missing_count} giá trị thiếu")
    imputer = SimpleImputer(strategy='median')
    X_imputed = imputer.fit_transform(X)
else:
    X_imputed = X.values
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_imputed)
```

In [25]:
```
pca = PCA(n_components=2, random_state=RANDOM_STATE)
X_pca = pca.fit_transform(X_scaled)
```

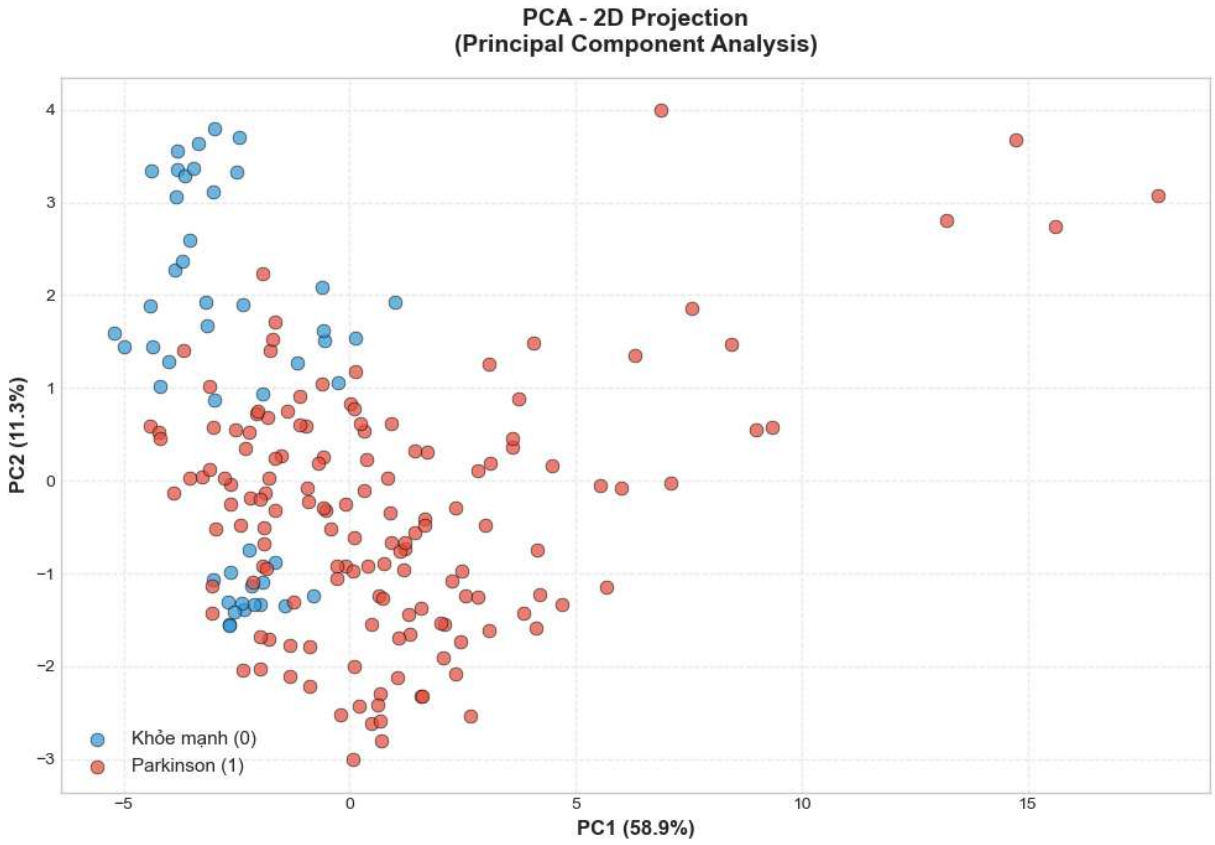
In [26]:
```
fig = plt.figure(figsize=(10, 7))
ax = plt.subplot(1, 1, 1)

if y is not None:
    colors = ['#3498db', '#e74c3c']
    labels_text = ['Khỏe mạnh (0)', 'Parkinson (1)']
    for status, color, label in zip([0, 1], colors, labels_text):
        mask = y == status
        ax.scatter(X_pca[mask, 0], X_pca[mask, 1],
                   c=color, label=label, s=60, alpha=0.7,
                   edgecolors='black', linewidth=0.5)
    ax.legend(loc='best', fontsize=11, framealpha=0.9)
else:
    ax.scatter(X_pca[:, 0], X_pca[:, 1], s=60, alpha=0.7)

ax.set_xlabel(f'PC1 ({pca.explained_variance_ratio_[0]*100:.1f}%)',
              fontsize=12, fontweight='bold')
ax.set_ylabel(f'PC2 ({pca.explained_variance_ratio_[1]*100:.1f}%)',
              fontsize=12, fontweight='bold')
ax.set_title('PCA - 2D Projection\n(Principal Component Analysis)',
             fontsize=14, fontweight='bold', pad=15)
ax.grid(True, alpha=0.3, linestyle='--')

plt.tight_layout()
output_file = 'pca_2d_projection.png'
plt.savefig(output_file, dpi=300, bbox_inches='tight', facecolor='white')
print(f"Đã lưu biểu đồ: '{output_file}'")
plt.show()
```

```
Đã lưu biểu đồ: 'pca_2d_projection.png'
```



In [27]:
```
report = f"""
I. THÔNG TIN DỮ LIỆU
   Tổng số mẫu: {len(df)}
   Số features sử dụng: {len(continuous_cols)}
   Phân bố nhãn:
    - Bệnh Parkinson (1): {counts[1]} mẫu ({counts[1]/len(y)*100:.1f}%)
    - Khỏe mạnh (0): {counts[0]} mẫu ({counts[0]/len(y)*100:.1f}%)
II. DANH SÁCH CÁC CỘT SỬ DỤNG
"""
cols_per_row = 3
for i in range(0, len(continuous_cols), cols_per_row):
    row_cols = continuous_cols[i:i+cols_per_row]
    report += "   " + " | ".join(f"{j+i+1:2d}. {col:20s}"
                                 for j, col in enumerate(row_cols)) + "\n"
report += f"""
III. KẾT QUẢ GIẢM CHIỀU - PCA
   Số thành phần chính: 2
   Variance Explained:
    - PC1 (Thành phần chính 1): {pca.explained_variance_ratio_[0]*100:.2f}%
    - PC2 (Thành phần chính 2): {pca.explained_variance_ratio_[1]*100:.2f}%
    - Tổng cộng: {sum(pca.explained_variance_ratio_)*100:.2f}%
"""
print(report)
with open('bao_cao_quan_sat.txt', 'w', encoding='utf-8') as f:
    f.write("BÁO CÁO QUAN SÁT - PHÂN TÍCH DỮ LIỆU PARKINSON\n")
    f.write(report)
print("\nĐã lưu báo cáo chi tiết: 'bao_cao_quan_sat.txt'")
```

```
I. THÔNG TIN DỮ LIỆU
   Tổng số mẫu: 195
   Số features sử dụng: 22
   Phân bố nhãn:
    - Bệnh Parkinson (1): 147 mẫu (75.4%)
    - Khỏe mạnh (0): 48 mẫu (24.6%)
II. DANH SÁCH CÁC CỘT SỬ DỤNG
   1. MDVP:Fo(Hz)          |  2. MDVP:Fhi(Hz)         |  3. MDVP:Flo(Hz)
   4. MDVP:Jitter(%)       |  5. MDVP:Jitter(Abs)     |  6. MDVP:RAP
   7. MDVP:PPQ             |  8. Jitter:DDP           |  9. MDVP:Shimmer
  10. MDVP:Shimmer(dB)     | 11. Shimmer:APQ3         | 12. Shimmer:APQ5
  13. MDVP:APQ             | 14. Shimmer:DDA          | 15. NHR
  16. HNR                  | 17. RPDE                 | 18. DFA
  19. spread1              | 20. spread2              | 21. D2
  22. PPE

III. KẾT QUẢ GIẢM CHIỀU - PCA
   Số thành phần chính: 2
   Variance Explained:
    - PC1 (Thành phần chính 1): 58.90%
    - PC2 (Thành phần chính 2): 11.30%
    - Tổng cộng: 70.20%


Đã lưu báo cáo chi tiết: 'bao_cao_quan_sat.txt'
```