In [1]:
```
#Nama : Ni Nyoman Vika Andini
#NIM : 2305551125
#Kelas : Aljabar Linier (C)
```

In [6]:
```
import os
os.environ["OMP_NUM_THREADS"] = "1"
# Menetapkan variabel lingkungan untuk menghindari peringatan kebocoran memori

import warnings
warnings.filterwarnings("ignore", message="KMeans is known to have a memory leak on Wi
# Mengabaikan peringatan tentang kebocoran memori pada KMeans di Windows dengan MKL

# Mengimpor modul pandas untuk manipulasi data
import pandas as pd
# Mengimpor modul numpy untuk operasi numerik
import numpy as np
# Mengimpor modul matplotlib.pyplot untuk visualisasi data
import matplotlib.pyplot as plt
# Mengimpor seaborn untuk visualisasi data yang lebih baik
import seaborn as sns
# Mengimpor KMeans dari scikit-learn untuk algoritma clustering
from sklearn.cluster import KMeans
# Mengimpor StandardScaler dari scikit-learn untuk normalisasi fitur
from sklearn.preprocessing import StandardScaler
# Mengimpor silhouette_score dari scikit-learn untuk mengevaluasi hasil clustering
from sklearn.metrics import silhouette_score

# Memuat dataset dari file CSV
df = pd.read_csv(r'C:\clustering\Mall_Customers.csv')

# Mengeksplorasi data - menampilkan 5 baris pertama dari dataset
print(df.head())
# Menampilkan informasi umum tentang dataset seperti tipe data dan jumlah non-null val
print(df.info())
```

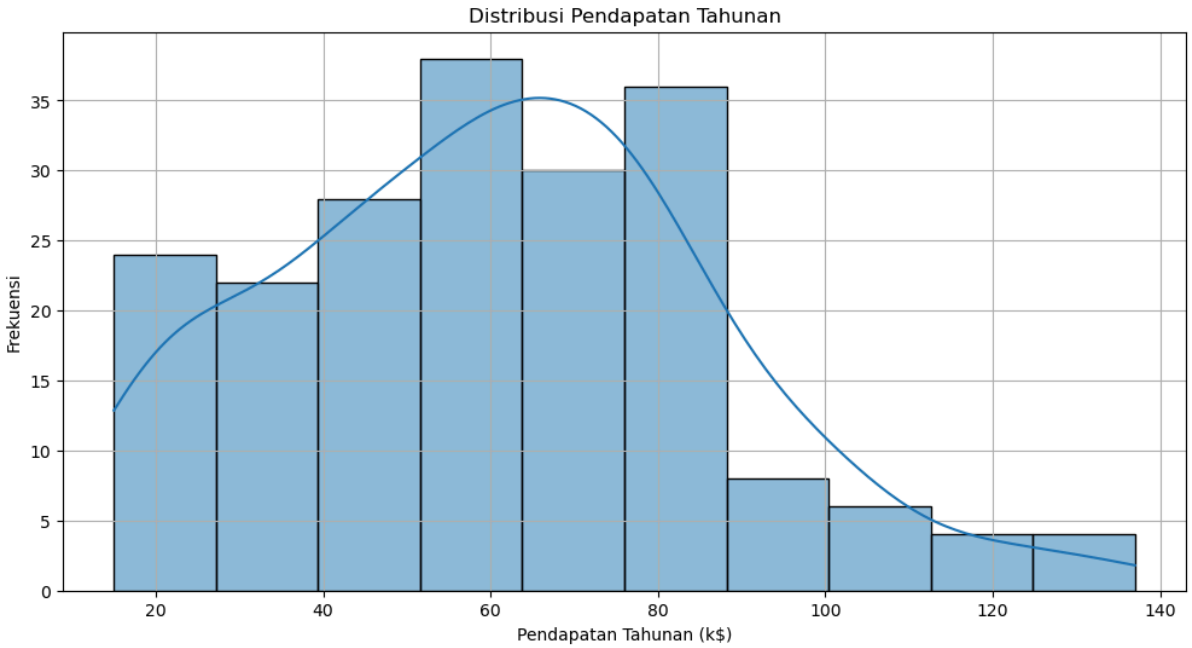
```
   CustomerID  Gender  Age  Annual Income (k$)  Spending Score (1-100)
0           1    Male   19                  15                      39
1           2    Male   21                  15                      81
2           3  Female   20                  16                       6
3           4  Female   23                  16                      77
4           5  Female   31                  17                      40
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   CustomerID              200 non-null    int64
 1   Gender                  200 non-null    object
 2   Age                     200 non-null    int64
 3   Annual Income (k$)      200 non-null    int64
 4   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
None
```

In [5]:
```
# Statistik deskriptif - memberikan ringkasan statistik untuk
# setiap kolom numerik dalam dataset
print(df.describe())
```

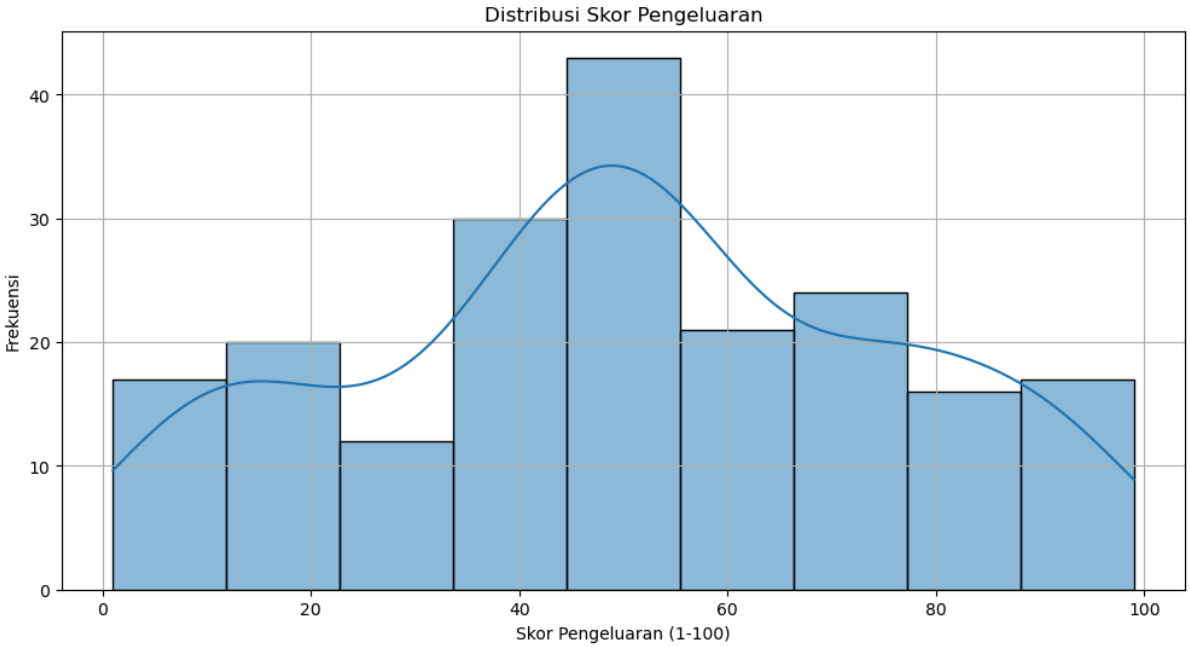
```
       CustomerID         Age  Annual Income (k$)  Spending Score (1-100)
count  200.000000  200.000000          200.000000              200.000000
mean   100.500000   38.850000           60.560000               50.200000
std     57.879185   13.969007           26.264721               25.823522
min      1.000000   18.000000           15.000000                1.000000
25%     50.750000   28.750000           41.500000               34.750000
50%    100.500000   36.000000           61.500000               50.000000
75%    150.250000   49.000000           78.000000               73.000000
max    200.000000   70.000000          137.000000               99.000000
```

In [7]:
```
# Distribusi data Pendapatan Tahunan
plt.figure(figsize=(12, 6))  # Mengatur ukuran gambar menjadi 12x6 inci
sns.histplot(df['Annual Income (k$)'], kde=True)
# Membuat histogram dari kolom 'Annual Income (k$)' dengan estimasi kepadatan kernel (

plt.title('Distribusi Pendapatan Tahunan')  # Menambahkan judul grafik
plt.xlabel('Pendapatan Tahunan (k$)')  # Menambahkan label sumbu x
plt.ylabel('Frekuensi')  # Menambahkan label sumbu y
plt.grid(True)  # Menambahkan grid pada grafik untuk memudahkan pembacaan
plt.show()  # Menampilkan grafik

# Distribusi data Skor Pengeluaran
plt.figure(figsize=(12, 6))  # Mengatur ukuran gambar menjadi 12x6 inci
sns.histplot(df['Spending Score (1-100)'], kde=True)
# Membuat histogram dari kolom 'Spending Score (1-100)' dengan estimasi kepadatan kern

plt.title('Distribusi Skor Pengeluaran')  # Menambahkan judul grafik
plt.xlabel('Skor Pengeluaran (1-100)')  # Menambahkan label sumbu x
plt.ylabel('Frekuensi')  # Menambahkan label sumbu y
plt.grid(True)  # Menambahkan grid pada grafik untuk memudahkan pembacaan
plt.show()  # Menampilkan grafik
```

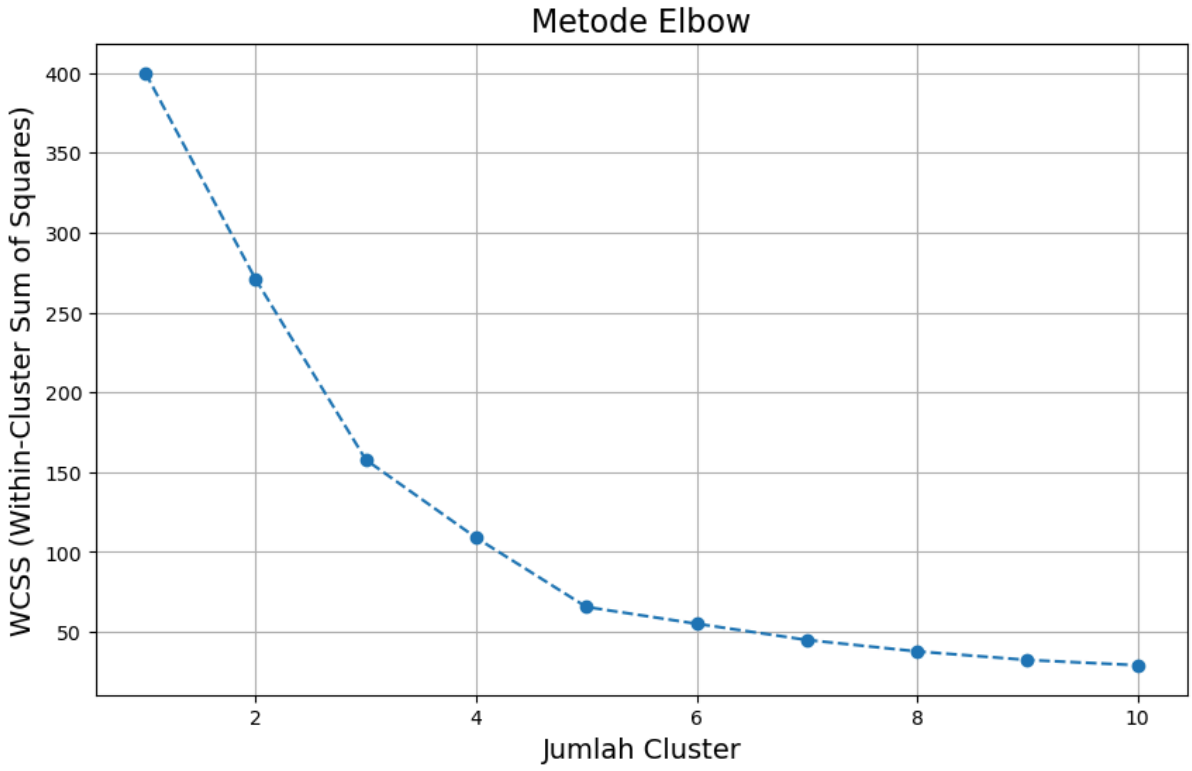
```
C:\Users\ASUS\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: us
e_inf_as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



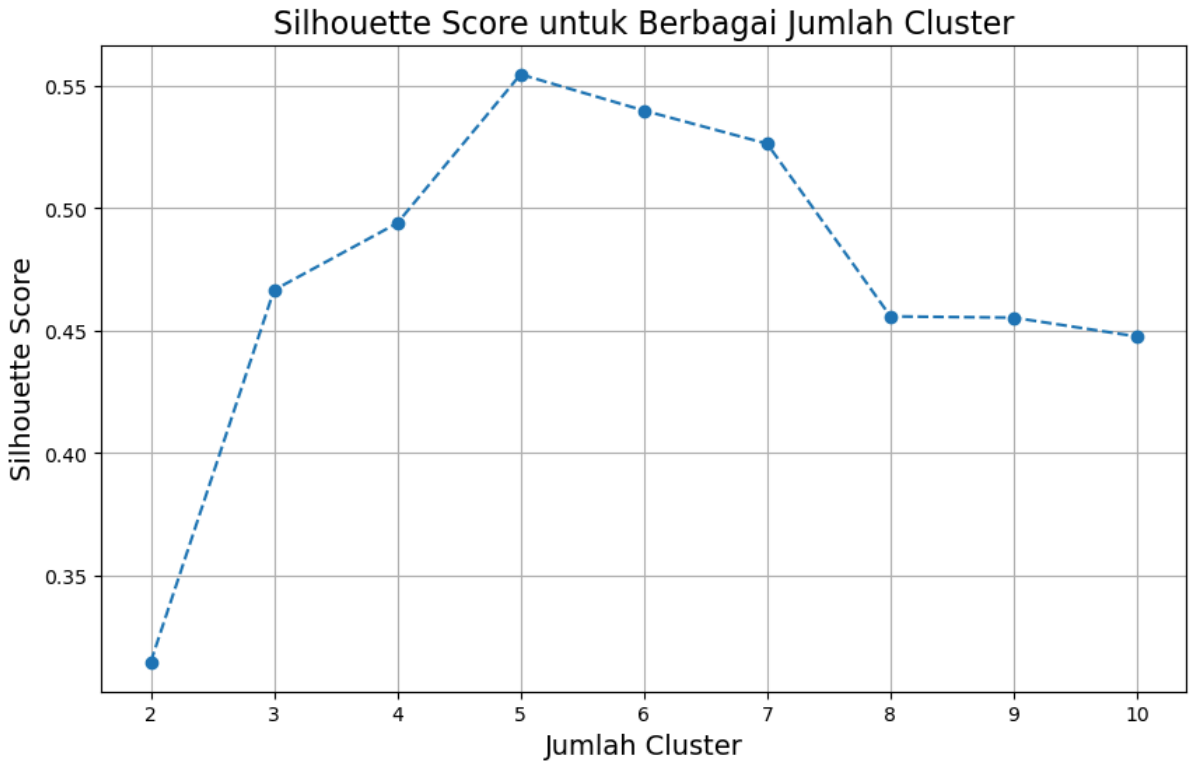

```
C:\Users\ASUS\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: us
e_inf_as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```


Distribusi Skor Pengeluaran
Frekuensi
Skor Pengeluaran (1-100)
0
10
20
30
40
0
20
40
60
80
100

In [8]:
```
# Praproses data (memilih fitur yang relevan dan melakukan skala)
X = df[['Annual Income (k$)', 'Spending Score (1-100)']]  # Memilih dua fitur yang rel
scaler = StandardScaler()  # Menggunakan StandardScaler untuk normalisasi fitur
X_scaled = scaler.fit_transform(X)  # Menerapkan scaler pada data untuk mendapatkan fi

# Menentukan jumlah cluster optimal menggunakan Metode Elbow
wcss = []  # Menyimpan nilai Within-Cluster Sum of Squares (WCSS) untuk berbagai jumla
for i in range(1, 11):  # Menguji jumlah cluster dari 1 hingga 10
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42, n_init=10)  # Men
    kmeans.fit(X_scaled)  # Melatih model pada data yang diskalakan
    wcss.append(kmeans.inertia_)  # Menyimpan nilai inertia (WCSS) untuk jumlah cluste

# Menampilkan grafik Metode Elbow
plt.figure(figsize=(10, 6))  # Mengatur ukuran gambar menjadi 10x6 inci
plt.plot(range(1, 11), wcss, marker='o', linestyle='--')  # Membuat plot WCSS terhadap
plt.xlabel('Jumlah Cluster', fontsize=14)  # Menambahkan label sumbu x dengan ukuran f
plt.ylabel('WCSS (Within-Cluster Sum of Squares)', fontsize=14)  # Menambahkan label s
plt.title('Metode Elbow', fontsize=16)  # Menambahkan judul grafik dengan ukuran font
plt.grid(True)  # Menambahkan grid pada grafik untuk memudahkan pembacaan
plt.show()  # Menampilkan grafik
```



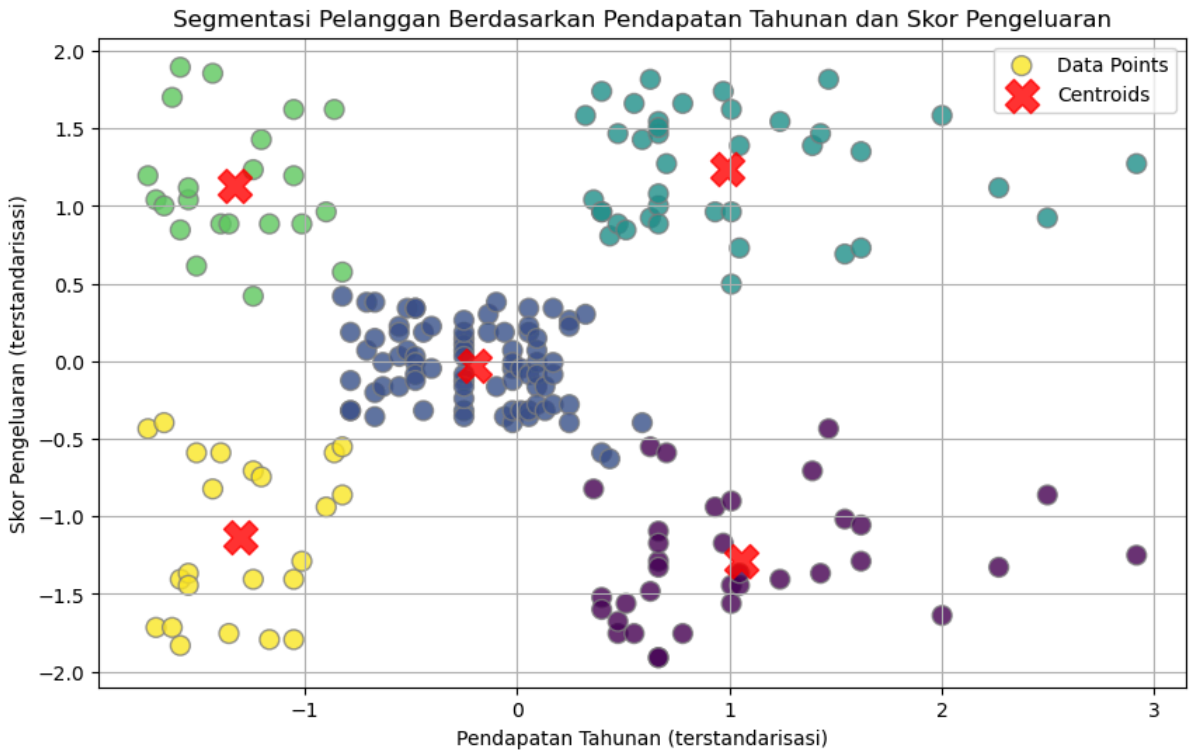
In [9]:
```
# Evaluasi menggunakan Silhouette Score
silhouette_scores = []  # Menyimpan nilai Silhouette Score untuk berbagai jumlah clust
for i in range(2, 11):
    # Menguji jumlah cluster dari 2 hingga 10 (karena Silhouette Score tidak dapat dih

    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42, n_init=10)  # Men
    kmeans.fit(X_scaled)  # Melatih model pada data yang diskalakan
    score = silhouette_score(X_scaled, kmeans.labels_)  # Menghitung Silhouette Score
    silhouette_scores.append(score)  # Menyimpan nilai Silhouette Score dalam list

# Menampilkan grafik Silhouette Score untuk berbagai jumlah cluster
plt.figure(figsize=(10, 6))  # Mengatur ukuran gambar menjadi 10x6 inci
plt.plot(range(2, 11), silhouette_scores, marker='o', linestyle='--')
# Membuat plot Silhouette Score terhadap jumlah cluster

plt.xlabel('Jumlah Cluster', fontsize=14)  # Menambahkan label sumbu x dengan ukuran f
plt.ylabel('Silhouette Score', fontsize=14)  # Menambahkan label sumbu y dengan ukuran
plt.title('Silhouette Score untuk Berbagai Jumlah Cluster', fontsize=16)
# Menambahkan judul grafik dengan ukuran font 16

plt.grid(True)  # Menambahkan grid pada grafik untuk memudahkan pembacaan
plt.show()  # Menampilkan grafik
```



In [10]:
```
# Berdasarkan Metode Elbow dan Silhouette Score, kita pilih jumlah cluster optimal (mi
jumlah_cluster_optimal = 5  # Menentukan jumlah cluster optimal berdasarkan analisis s

# Menerapkan clustering K-Means dengan n_init yang ditentukan untuk menghindari Future
kmeans = KMeans(n_clusters=jumlah_cluster_optimal, random_state=0, n_init=10)
# Menginisialisasi model KMeans dengan jumlah cluster optimal

kmeans.fit(X_scaled)  # Melatih model pada data yang telah diskalakan
labels = kmeans.labels_  # Mendapatkan label cluster untuk setiap data point

# Menambahkan label cluster ke dataframe asli
df['Cluster'] = labels  # Menambahkan kolom baru pada dataframe untuk menyimpan label

# Menampilkan rincian setiap cluster
print(df.groupby('Cluster').mean(numeric_only=True))
# Menghitung rata-rata fitur numerik untuk setiap cluster dan menampilkannya
```

```
         CustomerID        Age  Annual Income (k$)  Spending Score (1-100)
Cluster
0        164.371429  41.114286           88.200000               17.114286
1         86.320988  42.716049           55.296296               49.518519
2        162.000000  32.692308           86.538462               82.128205
3         23.090909  25.272727           25.727273               79.363636
4         23.000000  45.217391           26.304348               20.913043
```

In [11]:
```
# Memvisualisasikan hasil clustering
plt.figure(figsize=(10, 6))  # Mengatur ukuran gambar menjadi 10x6 inci

# Plot scatter dari data points
plt.scatter(X_scaled[:, 0],
            X_scaled[:, 1], c=labels, cmap='viridis', edgecolors='grey', alpha=0.8, s=
# Membuat scatter plot dari data points dengan warna berdasarkan label cluster, menggu
# edgecolors 'grey' untuk memberikan garis tepi abu-abu pada setiap titik,
# alpha 0.8 untuk membuat titik sedikit transparan
# dan ukuran titik (s) 100 untuk memperbesar titik

# Plotting pusat cluster
centers = kmeans.cluster_centers_  # Mendapatkan pusat cluster dari model KMeans
plt.scatter(centers[:, 0], centers[:, 1], c='red', s=300, alpha=0.8, marker='X', label
# Membuat scatter plot dari pusat cluster dengan warna merah, ukuran 300, alpha 0.8 un
# dan menggunakan marker 'X' untuk menandai pusat cluster

# Label dan judul
plt.xlabel('Pendapatan Tahunan (terstandarisasi)')  # Menambahkan label sumbu x
plt.ylabel('Skor Pengeluaran (terstandarisasi)')  # Menambahkan label sumbu y
plt.title('Segmentasi Pelanggan Berdasarkan Pendapatan Tahunan dan Skor Pengeluaran')
plt.legend()  # Menambahkan legenda untuk menjelaskan plot

plt.grid(True)  # Menambahkan grid pada grafik untuk memudahkan pembacaan
plt.show()  # Menampilkan grafik
```



In [12]:
```
# Analisis Cluster
for i in range(jumlah_cluster_optimal):  # Loop melalui setiap cluster
    print(f"\nCluster {i+1} Rincian:")
    # Menampilkan informasi cluster yang sedang dianalisis

    print(df[df['Cluster'] == i].describe())
    # Menampilkan statistik deskriptif untuk data yang termasuk dalam cluster ini
```

```
Cluster 1 Rincian:
       CustomerID        Age  Annual Income (k$)  Spending Score (1-100)  \
count   35.000000  35.000000           35.000000               35.000000
mean   164.371429  41.114286           88.200000               17.114286
std     21.457325  11.341676           16.399067                9.952154
min    125.000000  19.000000           70.000000                1.000000
25%    148.000000  34.000000           77.500000               10.000000
50%    165.000000  42.000000           85.000000               16.000000
75%    182.000000  47.500000           97.500000               23.500000
max    199.000000  59.000000          137.000000               39.000000

       Cluster
count     35.0
mean       0.0
std        0.0
min        0.0
25%        0.0
50%        0.0
75%        0.0
max        0.0

Cluster 2 Rincian:
       CustomerID        Age  Annual Income (k$)  Spending Score (1-100)  \
count   81.000000  81.000000           81.000000               81.000000
mean    86.320988  42.716049           55.296296               49.518519
std     24.240889  16.447822            8.988109                6.530909
min     44.000000  18.000000           39.000000               34.000000
25%     66.000000  27.000000           48.000000               44.000000
50%     86.000000  46.000000           54.000000               50.000000
75%    106.000000  54.000000           62.000000               55.000000
max    143.000000  70.000000           76.000000               61.000000

       Cluster
count     81.0
mean       1.0
std        0.0
min        1.0
25%        1.0
50%        1.0
75%        1.0
max        1.0

Cluster 3 Rincian:
       CustomerID        Age  Annual Income (k$)  Spending Score (1-100)  \
count   39.000000  39.000000           39.000000               39.000000
mean   162.000000  32.692308           86.538462               82.128205
std     22.803509   3.728650           16.312485                9.364489
min    124.000000  27.000000           69.000000               63.000000
25%    143.000000  30.000000           75.500000               74.500000
50%    162.000000  32.000000           79.000000               83.000000
75%    181.000000  35.500000           95.000000               90.000000
max    200.000000  40.000000          137.000000               97.000000

       Cluster
count     39.0
mean       2.0
std        0.0
min        2.0
25%        2.0
50%        2.0
75%        2.0
max        2.0

Cluster 4 Rincian:
       CustomerID        Age  Annual Income (k$)  Spending Score (1-100)  \
```

count	22.000000	22.000000	22.000000	22.000000
mean	23.090909	25.272727	25.727273	79.363636
std	13.147185	5.257030	7.566731	10.504174
min	2.000000	18.000000	15.000000	61.000000
25%	12.500000	21.250000	19.250000	73.000000
50%	23.000000	23.500000	24.500000	77.000000
75%	33.500000	29.750000	32.250000	85.750000
max	46.000000	35.000000	39.000000	99.000000

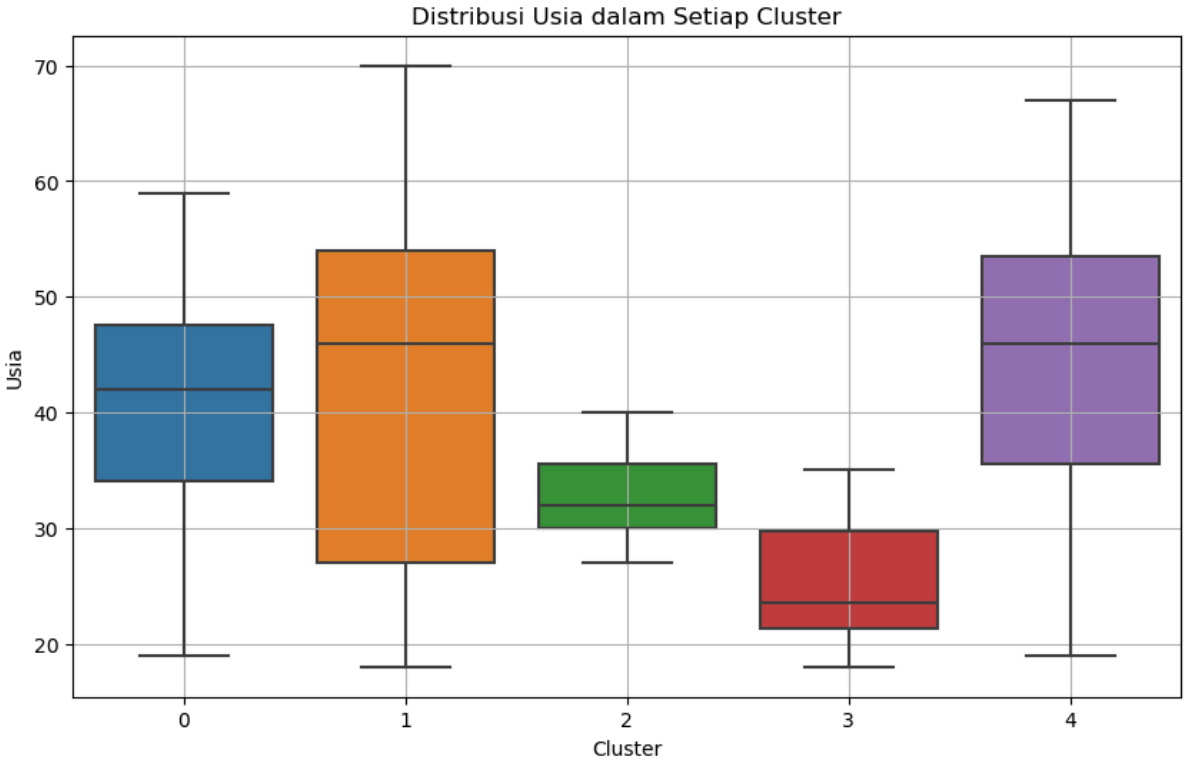
	Cluster
count	22.0
mean	3.0
std	0.0
min	3.0
25%	3.0
50%	3.0
75%	3.0
max	3.0

Cluster 5 Rincian:

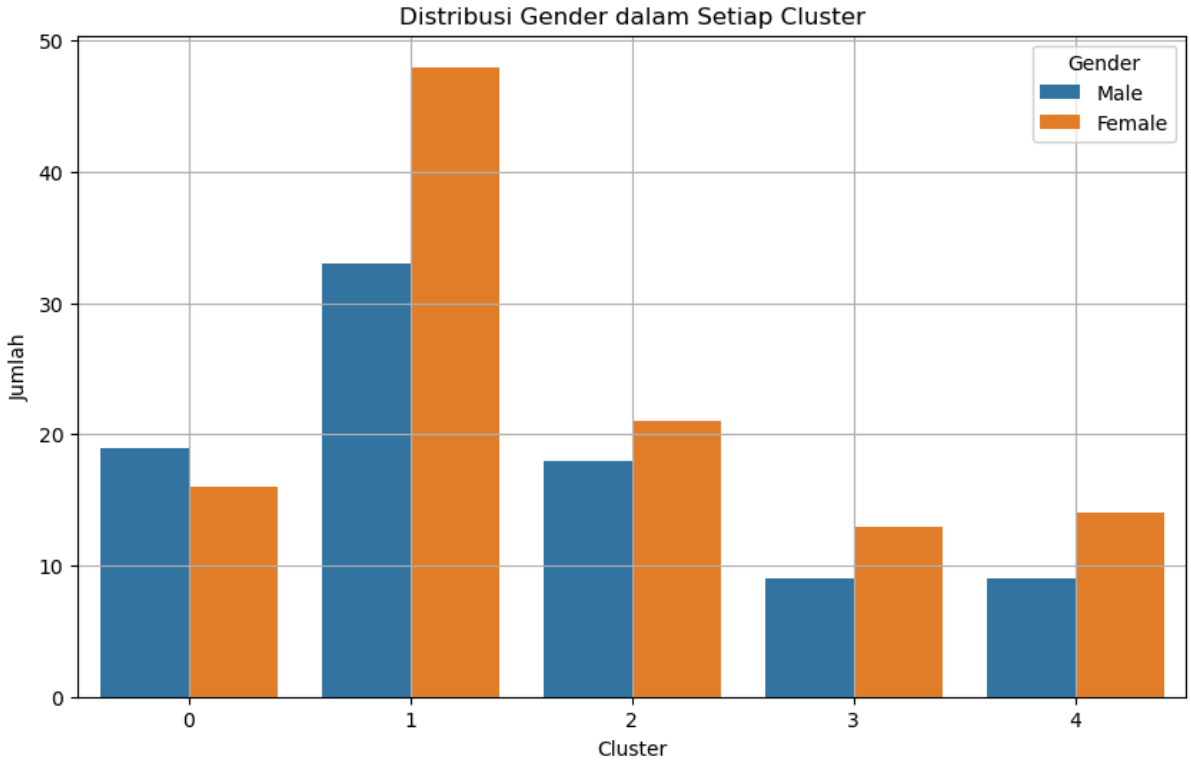
	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)	\
count	23.00000	23.000000	23.000000	23.000000	
mean	23.00000	45.217391	26.304348	20.913043	
std	13.56466	13.228607	7.893811	13.017167	
min	1.00000	19.000000	15.000000	3.000000	
25%	12.00000	35.500000	19.500000	9.500000	
50%	23.00000	46.000000	25.000000	17.000000	
75%	34.00000	53.500000	33.000000	33.500000	
max	45.00000	67.000000	39.000000	40.000000	

	Cluster
count	23.0
mean	4.0
std	0.0
min	4.0
25%	4.0
50%	4.0
75%	4.0
max	4.0

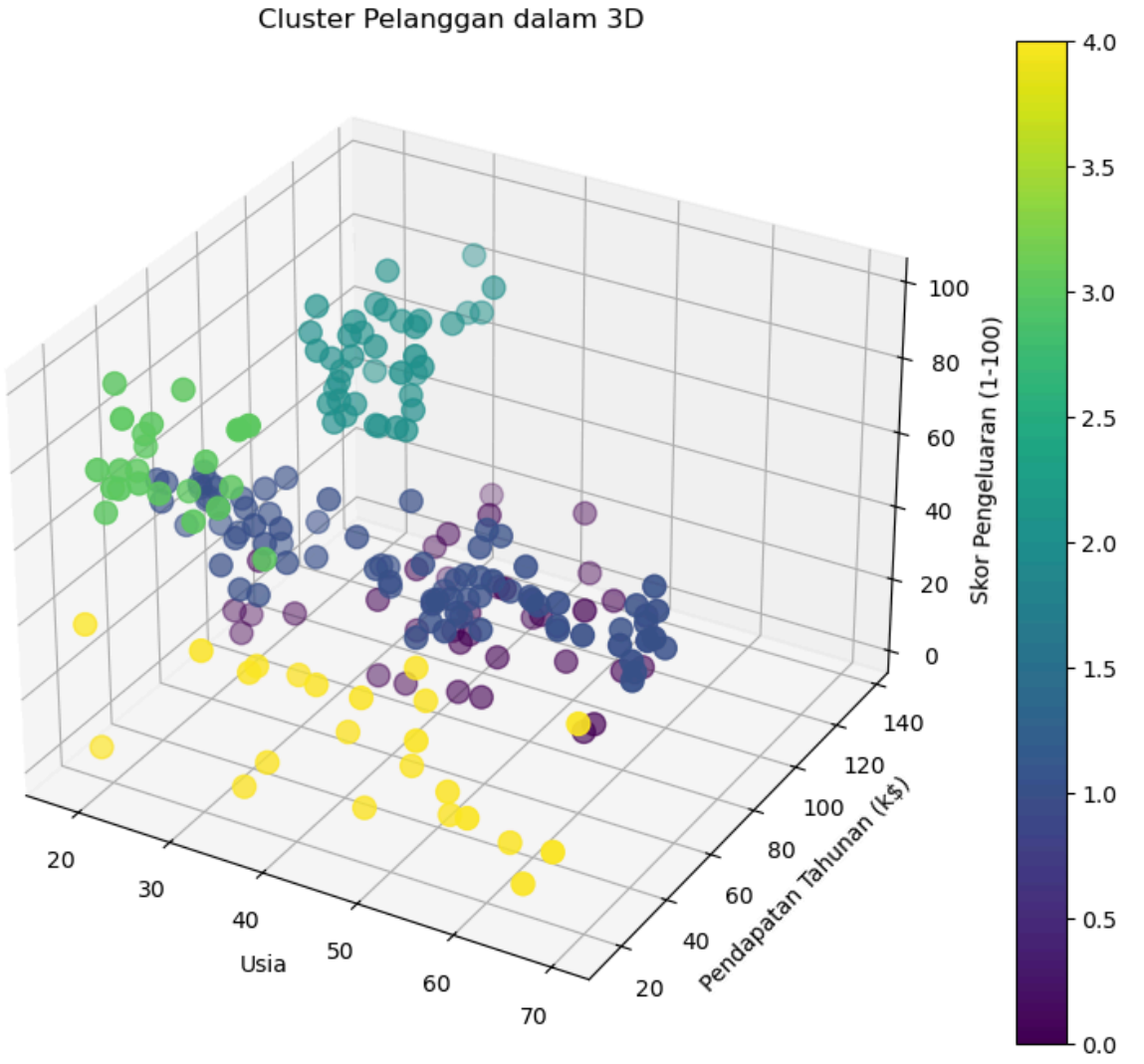
In [13]:
```
# Visualisasi distribusi usia pelanggan dalam setiap cluster
plt.figure(figsize=(10, 6))  # Mengatur ukuran gambar menjadi 10x6 inci

# Menggunakan boxplot untuk menampilkan distribusi usia dalam setiap cluster
sns.boxplot(x='Cluster', y='Age', data=df)
# x='Cluster': Menunjukkan kolom yang akan digunakan untuk membagi data menjadi grup (
# y='Age': Menunjukkan kolom yang akan digunakan untuk membangun boxplot (usia)
# data=df: Menunjukkan sumber data yang akan digunakan (dataframe df)

plt.title('Distribusi Usia dalam Setiap Cluster')  # Menambahkan judul plot
plt.xlabel('Cluster')  # Menambahkan label sumbu x (cluster)
plt.ylabel('Usia')  # Menambahkan label sumbu y (usia)
plt.grid(True)  # Menambahkan grid pada plot untuk memudahkan pembacaan
plt.show()  # Menampilkan plot
```



In [14]:
```
# Visualisasi distribusi gender dalam setiap cluster
plt.figure(figsize=(10, 6))  # Mengatur ukuran gambar menjadi 10x6 inci

# Menggunakan countplot untuk menampilkan distribusi gender dalam setiap cluster
sns.countplot(x='Cluster', hue='Gender', data=df)
# x='Cluster': Menunjukkan kolom yang akan digunakan untuk membagi data menjadi grup (
# hue='Gender': Menunjukkan kolom yang akan digunakan untuk membedakan data berdasarka
# data=df: Menunjukkan sumber data yang akan digunakan (dataframe df)

plt.title('Distribusi Gender dalam Setiap Cluster')  # Menambahkan judul plot
plt.xlabel('Cluster')  # Menambahkan label sumbu x (cluster)
plt.ylabel('Jumlah')  # Menambahkan label sumbu y (jumlah)
plt.grid(True)  # Menambahkan grid pada plot untuk memudahkan pembacaan
plt.show()  # Menampilkan plot
```



In [15]:
```
# Visualisasi cluster dalam 3D berdasarkan Usia, Pendapatan Tahunan, dan Skor Pengelua
from mpl_toolkits.mplot3d import Axes3D  # Mengimpor Axes3D untuk plot 3D

fig = plt.figure(figsize=(10, 8))  # Membuat objek gambar dengan ukuran 10x8 inci
ax = fig.add_subplot(111, projection='3d')  # Menambahkan subplot 3D ke dalam objek ga

# Membuat scatter plot 3D dari data
sc = ax.scatter(df['Age'], df['Annual Income (k$)'], df['Spending Score (1-100)'],
                c=labels, cmap='viridis', s=100)
# Menyertakan label cluster dengan warna yang berbeda untuk setiap cluster

ax.set_xlabel('Usia')  # Menyertakan label sumbu x (Usia)
ax.set_ylabel('Pendapatan Tahunan (k$)')  # Menyertakan label sumbu y (Pendapatan Tahu
ax.set_zlabel('Skor Pengeluaran (1-100)')  # Menyertakan label sumbu z (Skor Pengeluar
plt.title('Cluster Pelanggan dalam 3D')  # Menyertakan judul plot
plt.colorbar(sc)  # Menambahkan colorbar untuk menjelaskan nilai-nilai warna pada plot
plt.show()  # Menampilkan plot
```



In [ ]:
```
```