PROJECT FILE

Data Loading

- Importing the necessary libraries such as pandas,numpy etc.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import networkx as nx
import matplotlib.colors as mcolors
import plotly.graph_objects as go
import plotly.express as px
from scipy.stats import ttest_ind
```

- Reading the excel file from the directory and creating the dataframe out of it and print it

In [2]:
```
df=pd.read_excel("Coursera Dataset copy.xlsx")
```

In [3]:
```
df
```

Out[3]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Lea
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	NaN	NaN	NaN	
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	NaN	NaN	NaN	
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	NaN	NaN	NaN	
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	NaN	NaN	NaN	
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	
6630	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-02-15T11:48:06.000Z	2023-03-17T06:11:14.000Z	
6631	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	2023-04-02T14:10:21.000Z	NaN	
6632	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	NaN	NaN	NaN	
6633	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-02-12T12:04:42.000Z	2023-04-02T14:01:56.000Z	
6634	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	NaN	NaN	NaN	

6635 rows × 13 columns

- Columns - 13
- Rows - 6635

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Name', 'Division', 'Group', 'Department', 'Program Name',
       'Course Name', 'Duration(in hrs)', 'Completed', 'Enrollment Time',
       'Completion Time', 'Learning Hours Spent', 'Course Grade',
       'Skills Learned'],
      dtype='object')
```

- In this given dataset we have the following data columns basically namely:

0. Name (object)
1. Division (object)
2. Group (object)
3. Department (object)
4. Program Name (object)
5. Course Name (object)
6. Duration(in hrs) (float64)
7. Completed (object)
8. Enrollment Time (object)
9. Completion Time (object)
10. Learning Hours Spent (float64)
11. Course Grade (float64)
12. Skills Learned (object)

In [5]:
```
df.describe()
```

Out[5]:

	Duration(in hrs)	Learning Hours Spent	Course Grade
count	6635.000000	1738.000000	1738.000000
mean	7.788832	2.968953	41.338423
std	3.425742	3.892452	41.900973
min	1.600000	0.000000	0.000000
25%	5.800000	0.210000	0.000000
50%	6.600000	1.590000	22.500000
75%	8.900000	4.377500	87.890000
max	49.000000	40.940000	100.000000

- Describe command is basically calculating the metrics of the numerical data

We have three columns for the new numerical dataset namely:-

1. Duration(in hrs)
2. Learning Hours Spent(in Hrs)
3. Course Grade

In [6]:
```
df.head()
```

Out[6]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Course Grade	Ski Learn
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	NaN	NaN	NaN	NaN	NaN	Trigonomet Integral; Le Six Sign Six S
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	NaN	NaN	NaN	NaN	NaN	Six Sign Trainir Cultu Analys Leade
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	NaN	NaN	NaN	NaN	NaN	Proce Analys Proce: Measureme Genera
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	NaN	NaN	NaN	NaN	NaN	Trigonomet Integral; Sign Leadership
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	NaN	NaN	NaN	NaN	NaN	Proje Conf Manageme Conf Resolu

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6635 entries, 0 to 6634
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Name                  6635 non-null   object 
 1   Division              6635 non-null   object 
 2   Group                 6635 non-null   object 
 3   Department            6635 non-null   object 
 4   Program Name          6635 non-null   object 
 5   Course Name           6635 non-null   object 
 6   Duration(in hrs)      6635 non-null   float64
 7   Completed             1738 non-null   object 
 8   Enrollment Time       1738 non-null   object 
 9   Completion Time       675 non-null    object 
 10  Learning Hours Spent  1738 non-null   float64
 11  Course Grade          1738 non-null   float64
 12  Skills Learned        6635 non-null   object 
dtypes: float64(3), object(10)
memory usage: 674.0+ KB
```

- Giving the details of the total number of entries in each particular columns which are empty

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
Name                     0
Division                 0
Group                    0
Department               0
Program Name             0
Course Name              0
Duration(in hrs)         0
Completed             4897
Enrollment Time       4897
Completion Time       5960
Learning Hours Spent  4897
Course Grade          4897
Skills Learned           0
dtype: int64
```

Data Cleaning

In [9]:
```
df['Completed']
```

Out[9]:
```
0       NaN
1       NaN
2       NaN
3       NaN
4       NaN
       ...
6630    Yes
6631     No
6632    NaN
6633    Yes
6634    NaN
Name: Completed, Length: 6635, dtype: object
```

In [10]:
```
df['Completed'].fillna('Not Started Yet',inplace=True)
```

In [11]:
```
df['Completed']
```

Out[11]:
```
0       Not Started Yet
1       Not Started Yet
2       Not Started Yet
3       Not Started Yet
4       Not Started Yet
             ...
6630                Yes
6631                 No
6632    Not Started Yet
6633                Yes
6634    Not Started Yet
Name: Completed, Length: 6635, dtype: object
```

In [12]:
```
df['Completed'].fillna('Not Started Yet',inplace=True)
```

In [13]:
```
df['Completed']
```

Out[13]:
```
0       Not Started Yet
1       Not Started Yet
2       Not Started Yet
3       Not Started Yet
4       Not Started Yet
             ...
6630                Yes
6631                 No
6632    Not Started Yet
6633                Yes
6634    Not Started Yet
Name: Completed, Length: 6635, dtype: object
```

- In the "Completed" column for the missing values thodse who have not started the course yet have the value "Not Started Yet"

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
Name                     0
Division                 0
Group                    0
Department               0
Program Name             0
Course Name              0
Duration(in hrs)         0
Completed                0
Enrollment Time       4897
Completion Time       5960
Learning Hours Spent  4897
Course Grade          4897
Skills Learned           0
dtype: int64
```

In [15]:
```
type(df['Enrollment Time'])
```

Out[15]:
```
pandas.core.series.Series
```

In [16]:
```
type(df['Completion Time'])
```

Out[16]:
```
pandas.core.series.Series
```

In [17]:
```
type(df['Learning Hours Spent'])
```

Out[17]:
```
pandas.core.series.Series
```

In [18]:
```
type(df['Course Grade'])
```

Out[18]:
```
pandas.core.series.Series
```

- Creating a temporary dataframe df_test so that in case lost the the datframe can anytime be recovered

In [19]:
```
type(df['Completion Time'])
```

Out[19]:
```
pandas.core.series.Series
```

In [20]:
```
df_test=df
```

In [21]:
```
df_test['Completion Time'] = pd.to_datetime(df_test['Completion Time'], errors='coerce', infer_datetime_format=True)
```

```
C:\Users\Siddharth\AppData\Local\Temp\ipykernel_14528\3463797058.py:1: UserWarning: The argument 'infer_datetime_form
at' is deprecated and will be removed in a future version. A strict version of it is now the default, see https://pan
das.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. (https://pandas.pydata.org/pdeps/0004-consistent-to-da
tetime-parsing.html.) You can safely remove this argument.
  df_test['Completion Time'] = pd.to_datetime(df_test['Completion Time'], errors='coerce', infer_datetime_format=Tru
e)
```

In [22]:
```
df_test['Completion Time']
```

Out[22]:
```
0                           NaT
1                           NaT
2                           NaT
3                           NaT
4                           NaT
                  ...
6630   2023-03-17 06:11:14+00:00
6631                        NaT
6632                        NaT
6633   2023-04-02 14:01:56+00:00
6634                        NaT
Name: Completion Time, Length: 6635, dtype: datetime64[ns, UTC]
```

In [23]:
```
df_test['Completion Time'] = pd.to_datetime(df_test['Completion Time'], errors='coerce')
# Converting to the desired format
df_test['Formatted Completion Time'] = df_test['Completion Time'].dt.strftime('%d-%m-%y %H:%M:%S')
df_test[['Completion Time', 'Formatted Completion Time']]
```

Out[23]:

	Completion Time	Formatted Completion Time
0	NaT	NaN
1	NaT	NaN
2	NaT	NaN
3	NaT	NaN
4	NaT	NaN
...	...	...
6630	2023-03-17 06:11:14+00:00	17-03-23 06:11:14
6631	NaT	NaN
6632	NaT	NaN
6633	2023-04-02 14:01:56+00:00	02-04-23 14:01:56
6634	NaT	NaN

6635 rows × 2 columns

In [24]:
```
df_test['Formatted Completion Time']
```

Out[24]:
```
0                    NaN
1                    NaN
2                    NaN
3                    NaN
4                    NaN
               ...
6630    17-03-23 06:11:14
6631                 NaN
6632                 NaN
6633    02-04-23 14:01:56
6634                 NaN
Name: Formatted Completion Time, Length: 6635, dtype: object
```

In [25]:
```
type(df_test['Formatted Completion Time'])
```

Out[25]:
```
pandas.core.series.Series
```

In [26]:
```
df_test['Completion Time']
```

Out[26]:
```
0                           NaT
1                           NaT
2                           NaT
3                           NaT
4                           NaT
                  ...
6630   2023-03-17 06:11:14+00:00
6631                        NaT
6632                        NaT
6633   2023-04-02 14:01:56+00:00
6634                        NaT
Name: Completion Time, Length: 6635, dtype: datetime64[ns, UTC]
```

In [27]:
```
df['Completion Time'] = pd.to_datetime(df['Completion Time'], errors='coerce')
df['Completion Time'] = df['Completion Time'].dt.strftime('%d-%m-%y %H:%M:%S')
```

In [28]:
```
df['Completion Time']
```

Out[28]:
```
0                    NaN
1                    NaN
2                    NaN
3                    NaN
4                    NaN
               ...
6630    17-03-23 06:11:14
6631                 NaN
6632                 NaN
6633    02-04-23 14:01:56
6634                 NaN
Name: Completion Time, Length: 6635, dtype: object
```

In [29]:
```
df_test['Enrollment Time']
```

Out[29]:
```
0                           NaN
1                           NaN
2                           NaN
3                           NaN
4                           NaN
                  ...
6630    2023-02-15T11:48:06.000Z
6631    2023-04-02T14:10:21.000Z
6632                        NaN
6633    2023-02-12T12:04:42.000Z
6634                        NaN
Name: Enrollment Time, Length: 6635, dtype: object
```

In [30]:
```
df['Enrollment Time'] = pd.to_datetime(df['Enrollment Time'], errors='coerce')
df['Enrollment Time'] = df['Enrollment Time'].dt.strftime('%d-%m-%y %H:%M:%S')
```

In [31]:
```
df['Enrollment Time']
```

Out[31]:
```
0                    NaN
1                    NaN
2                    NaN
3                    NaN
4                    NaN
               ...
6630    15-02-23 11:48:06
6631    02-04-23 14:10:21
6632                 NaN
6633    12-02-23 12:04:42
6634                 NaN
Name: Enrollment Time, Length: 6635, dtype: object
```

In [32]:
```
df_test['Completed']
```

Out[32]:
```
0       Not Started Yet
1       Not Started Yet
2       Not Started Yet
3       Not Started Yet
4       Not Started Yet
             ...
6630               Yes
6631                No
6632    Not Started Yet
6633               Yes
6634    Not Started Yet
Name: Completed, Length: 6635, dtype: object
```

In [33]:
```
df['Completed']
```

Out[33]:
```
0       Not Started Yet
1       Not Started Yet
2       Not Started Yet
3       Not Started Yet
4       Not Started Yet
             ...
6630               Yes
6631                No
6632    Not Started Yet
6633               Yes
6634    Not Started Yet
Name: Completed, Length: 6635, dtype: object
```

In [34]:
```
df_test['Learning Hours Spent']
```

Out[34]:
```
0        NaN
1        NaN
2        NaN
3        NaN
4        NaN
        ...
6630    7.31
6631    2.09
6632     NaN
6633    6.34
6634     NaN
Name: Learning Hours Spent, Length: 6635, dtype: float64
```

In [35]:
```
df_test['Learning Hours Spent'].isna().sum()
```

Out[35]:
```
4897
```

In [36]:
```
df['Learning Hours Spent']
```

Out[36]:
```
0        NaN
1        NaN
2        NaN
3        NaN
4        NaN
        ...
6630    7.31
6631    2.09
6632     NaN
6633    6.34
6634     NaN
Name: Learning Hours Spent, Length: 6635, dtype: float64
```

In [37]:
```
df['Course Grade']
```

Out[37]:
```
0         NaN
1         NaN
2         NaN
3         NaN
4         NaN
        ...
6630    90.00
6631    33.23
6632      NaN
6633    90.00
6634      NaN
Name: Course Grade, Length: 6635, dtype: float64
```

In [38]:
```
df.isnull().sum()
```

Out[38]:
```
Name                          0
Division                      0
Group                         0
Department                    0
Program Name                  0
Course Name                   0
Duration(in hrs)              0
Completed                     0
Enrollment Time            4897
Completion Time            5960
Learning Hours Spent       4897
Course Grade               4897
Skills Learned                0
Formatted Completion Time  5960
dtype: int64
```

In [39]:
```
dummy_datetime = pd.to_datetime('1900-01-01 00:00:00', format='%Y-%m-%d %H:%M:%S')
df_test['Enrollment Time'].fillna(dummy_datetime, inplace=True)
```

In [40]:
```
df_test['Enrollment Time']
```

Out[40]:
```
0       1900-01-01 00:00:00
1       1900-01-01 00:00:00
2       1900-01-01 00:00:00
3       1900-01-01 00:00:00
4       1900-01-01 00:00:00
               ...
6630      15-02-23 11:48:06
6631      02-04-23 14:10:21
6632    1900-01-01 00:00:00
6633      12-02-23 12:04:42
6634    1900-01-01 00:00:00
Name: Enrollment Time, Length: 6635, dtype: object
```

- Filling all the NaN values with the dummy values of Enrollment time by filling them with the time 1 January 1900 and time will be midnight timming

In [41]:
```
df['Completion Time']
```

Out[41]:
```
0                    NaN
1                    NaN
2                    NaN
3                    NaN
4                    NaN
               ...
6630    17-03-23 06:11:14
6631                 NaN
6632                 NaN
6633    02-04-23 14:01:56
6634                 NaN
Name: Completion Time, Length: 6635, dtype: object
```

In [42]:
```
dummy_datetime = pd.to_datetime('1900-01-01 00:00:00', format='%Y-%m-%d %H:%M:%S')
df['Completion Time'].fillna(dummy_datetime, inplace=True)
```

- Filling all the NaN values with the dummy values of Complement time by filling them with the time 1 January 1900 and time will be midnight timming

In [43]:
```
df.isnull().sum()
```

Out[43]:
```
Name                          0
Division                      0
Group                         0
Department                    0
Program Name                  0
Course Name                   0
Duration(in hrs)              0
Completed                     0
Enrollment Time               0
Completion Time               0
Learning Hours Spent       4897
Course Grade               4897
Skills Learned                0
Formatted Completion Time  5960
dtype: int64
```

In [44]:
```
df['Completion Time']
```

Out[44]:
```
0       1900-01-01 00:00:00
1       1900-01-01 00:00:00
2       1900-01-01 00:00:00
3       1900-01-01 00:00:00
4       1900-01-01 00:00:00
               ...
6630      17-03-23 06:11:14
6631    1900-01-01 00:00:00
6632    1900-01-01 00:00:00
6633      02-04-23 14:01:56
6634    1900-01-01 00:00:00
Name: Completion Time, Length: 6635, dtype: object
```

In [45]:
```
df['Learning Hours Spent']
```

Out[45]:
```
0        NaN
1        NaN
2        NaN
3        NaN
4        NaN
        ...
6630    7.31
6631    2.09
6632     NaN
6633    6.34
6634     NaN
Name: Learning Hours Spent, Length: 6635, dtype: float64
```

In [46]:
```
df_test['Learning Hours Spent'].fillna(0.0, inplace=True)
```

In [47]:
```
df_test['Learning Hours Spent']
```

Out[47]:
```
0       0.00
1       0.00
2       0.00
3       0.00
4       0.00
        ...
6630    7.31
6631    2.09
6632    0.00
6633    6.34
6634    0.00
Name: Learning Hours Spent, Length: 6635, dtype: float64
```

In [48]:
```
df['Learning Hours Spent'].fillna(0.0, inplace=True)
```

- Filled all the not availible values with 0.00 in the learning hours column

In [49]:
```
df['Learning Hours Spent']
```

Out[49]:
```
0       0.00
1       0.00
2       0.00
3       0.00
4       0.00
        ...
6630    7.31
6631    2.09
6632    0.00
6633    6.34
6634    0.00
Name: Learning Hours Spent, Length: 6635, dtype: float64
```

In [50]:
```
df['Course Grade']
```

Out[50]:
```
0         NaN
1         NaN
2         NaN
3         NaN
4         NaN
        ...
6630    90.00
6631    33.23
6632      NaN
6633    90.00
6634      NaN
Name: Course Grade, Length: 6635, dtype: float64
```

In [51]:
```
df['Course Grade'].fillna(0.0, inplace=True)
```

- Filled all the not availible values with 0.00 in the Course Grade column

In [52]:
```
df['Course Grade']
```

Out[52]:
```
0        0.00
1        0.00
2        0.00
3        0.00
4        0.00
        ...
6630    90.00
6631    33.23
6632     0.00
6633    90.00
6634     0.00
Name: Course Grade, Length: 6635, dtype: float64
```

In [53]:
```
df.isnull().sum()
```

Out[53]:
```
Name                           0
Division                       0
Group                          0
Department                     0
Program Name                   0
Course Name                    0
Duration(in hrs)               0
Completed                      0
Enrollment Time                0
Completion Time                0
Learning Hours Spent           0
Course Grade                   0
Skills Learned                 0
Formatted Completion Time    5960
dtype: int64
```

In [54]:
```
df['Completion Time']
```

Out[54]:
```
0       1900-01-01 00:00:00
1       1900-01-01 00:00:00
2       1900-01-01 00:00:00
3       1900-01-01 00:00:00
4       1900-01-01 00:00:00
                  ...
6630      17-03-23 06:11:14
6631    1900-01-01 00:00:00
6632    1900-01-01 00:00:00
6633      02-04-23 14:01:56
6634    1900-01-01 00:00:00
Name: Completion Time, Length: 6635, dtype: object
```

In [55]:
```
df['Formatted Completion Time']
```

Out[55]:
```
0                    NaN
1                    NaN
2                    NaN
3                    NaN
4                    NaN
                ...
6630    17-03-23 06:11:14
6631                 NaN
6632                 NaN
6633    02-04-23 14:01:56
6634                 NaN
Name: Formatted Completion Time, Length: 6635, dtype: object
```

In [56]:
```
df.drop(columns=['Formatted Completion Time'], inplace=True)
```

- Dropped the unnecessary 'Formatted Completion Time' Column from the dataframe

In [57]:
```
df
```

Out[57]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Cou Gra
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
...	...	...	...	...	...	...	...	...	...	...	...	
6630	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	15-02-23 11:48:06	17-03-23 06:11:14	7.31	90
6631	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	02-04-23 14:10:21	1900-01-01 00:00:00	2.09	33
6632	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
6633	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	12-02-23 12:04:42	02-04-23 14:01:56	6.34	90
6634	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0

6635 rows × 13 columns

In [58]:
```
df.isnull().sum()
```

Out[58]:
```
Name                  0
Division              0
Group                 0
Department            0
Program Name          0
Course Name           0
Duration(in hrs)      0
Completed             0
Enrollment Time       0
Completion Time       0
Learning Hours Spent  0
Course Grade          0
Skills Learned        0
dtype: int64
```

- Cleaning the # values present in the dataframe

In [59]:
```
hash_counts_per_column = df.apply(lambda x: x.eq('#').sum())
hash_counts_per_column
```

Out[59]:
```
Name                     0
Division                 0
Group                 2080
Department               4
Program Name             0
Course Name              0
Duration(in hrs)         0
Completed                0
Enrollment Time          0
Completion Time          0
Learning Hours Spent     0
Course Grade             0
Skills Learned           0
dtype: int64
```

In [60]:
```
df_test=df
```

In [61]:
```
df['Group'] = df['Group'].replace('#', 'Not Available')
```

In [62]:
```
hash_counts_per_column = df.apply(lambda x: x.eq('#').sum())
hash_counts_per_column
```

Out[62]:
```
Name                   0
Division               0
Group                  0
Department             4
Program Name           0
Course Name            0
Duration(in hrs)       0
Completed              0
Enrollment Time        0
Completion Time        0
Learning Hours Spent   0
Course Grade           0
Skills Learned         0
dtype: int64
```

In [63]:
```
df['Department'] = df['Department'].replace('#', 'Not Available')
```

In [64]:
```
hash_counts_per_column = df.apply(lambda x: x.eq('#').sum())
hash_counts_per_column
```

Out[64]:
```
Name                   0
Division               0
Group                  0
Department             0
Program Name           0
Course Name            0
Duration(in hrs)       0
Completed              0
Enrollment Time        0
Completion Time        0
Learning Hours Spent   0
Course Grade           0
Skills Learned         0
dtype: int64
```

In [65]:
```
df_test=df
```

Data Processing And Evaluation

In [66]:
```
unique_divisions = df['Division'].unique()
df_unique_divisions = pd.DataFrame({'Unique Divisions': unique_divisions})
df_unique_divisions
```

Out[66]:

	Unique Divisions
0	Operations - TSK
1	VP One SC
2	TSM
3	M&S(LP)
4	ED & CFO
5	Raw Materials
6	M&S(FP)
7	One IT
8	TQM and E&P
9	VP Corp. Finance, Treasury & Risk Mgmt.
10	VP - GSP
11	HRM
12	Iron Making
13	Steel Manufacturing
14	Shared Services
15	CEO & MD
16	SHS
17	Tech&NMB
18	VP Financial Oprns.& Corporate Reporting
19	Tubes Division
20	S&S
21	Corporate Services

- We have 22 different divisions

In [67]:
```
unique_persons = df['Name'].unique()
df_unique_persons = pd.DataFrame({'Unique Persons': unique_persons})
df_unique_persons
```

Out[67]:

	Unique Persons
0	Person_1
1	Person_2
2	Person_3
3	Person_4
4	Person_5
...	...
1404	Person_1405
1405	Person_1406
1406	Person_1407
1407	Person_1408
1408	Person_1409

1409 rows × 1 columns

- We have 1409 different candidates / persons in the whole dataset

In [68]:
```
unique_groups = df['Group'].unique()

df_unique_groups = pd.DataFrame({'Unique Groups': unique_groups})
df_unique_groups
```

Out[68]:

	Unique Groups
0	Steel & Mills - TSK
1	Integrated Planning & Service
2	Quality
3	Chief Commercial Officer (Long Product)
4	Portfolio Management & FFI
...	...
58	Financial Planning & Group Reporting
59	Vice President Operations TSM
60	Power Systems
61	Security & Brand Protection
62	Finance & Accounts

63 rows × 1 columns

- There are 63 unique groups in the dataframe

In [69]:
```
unique_Departments = df['Department'].unique()
df_unique_Departments = pd.DataFrame({'Unique Departments': unique_Departments})
df_unique_Departments
```

Out[69]:

	Unique Departments
0	Steel Melting Shop & LCP TSK
1	Master Production Planning & Margin Mgmt
2	Scientific Services -Iron Making
3	LP-M&S (TISCON Retail),
4	Tax Centre of Excellence- Indirect Tax
...	...
284	Office of Chief Commercial
285	Not Available
286	CRM Downstream
287	Office of Chief Khondbond
288	Mills Mechanical Maintenance

289 rows × 1 columns

- There are 289 different departments

In [70]:
```
unique_Programs = df['Program Name'].unique()
df_unique_Programs = pd.DataFrame({'Unique Programs': unique_Programs})
df_unique_Programs
```

Out[70]:

	Unique Programs
0	Six Sigma Black Belt Learning Program
1	School of Analytics-Basic
2	Supply Chain Planning Learning Program
3	School of Visualization ONE IT
4	Teamwork Skills Learning Program
...	...
129	Google Data Analytics Learning Program
130	Communication Roles Learning Program
131	Solve Business Problems- AI & ML ONE IT
132	Leading Positive Change Learning Program
133	Business Strategy Learning Program

134 rows × 1 columns

- There are 134 differnt programs in the dataframe

In [71]:
```
unique_Courses = df['Course Name'].unique()
df_unique_Courses = pd.DataFrame({'Unique Courses': unique_Courses})
df_unique_Courses
```

Out[71]:

	Unique Courses
0	The Define Phase for the 6 σ Black Belt
1	The Analyze Phase for the 6 σ Black Belt
2	The Measure Phase for the 6 σ Black Belt
3	DFSS for the 6 σ Black Belt
4	Team Management for the 6 σ Black Belt
...	...
286	Advanced Business Strategy
287	Foundations of Business Strategy
288	Strategic Planning and Execution
289	Business Strategy in Practice (Project-centere...
290	Business Growth Strategy

291 rows × 1 columns

- There are 291 different courses in the dataframe .

In [72]:
```
df['Skills Learned']
```

Out[72]:
```
0       Trigonometric Integral; Lean Six Sigma; Six Si...
1       Six Sigma; Training; Culture; Analysis; Leader...
2       Process Analysis; Process; Measurement; Genera...
3       Trigonometric Integral; Six Sigma; Leadership ...
4       Project; Conflict Management; Conflict Resolut...
                              ...
6630    Data Visualization; Data Analysis; Analysis; P...
6631    Supply Chain; Finance; Chaining; Leadership an...
6632    Supply Chain; Trigonometric Integral; Lean Six...
6633    Data Visualization; Data Analysis; Analysis; P...
6634    General Statistics; Statistical Analysis; Prob...
Name: Skills Learned, Length: 6635, dtype: object
```

In [73]:
```
skills_counts = df_test['Skills Learned'].str.split('; ').explode().value_counts()
df_skills_counts = pd.DataFrame({'Skill': skills_counts.index, 'Count': skills_counts.values})
df_skills_counts
```

Out[73]:

	Skill	Count
0	Analysis	3160
1	Leadership and Management	1607
2	Data Visualization	1488
3	Data Analysis	1412
4	General Statistics	1325
...	...	...
352	Search Algorithm	1
353	Graphs	1
354	Apache Hadoop	1
355	Financial Risk	1
356	Personal Advertisement	1

357 rows × 2 columns

- There are 357 different skills spanning across all the courses

In [74]:
```
df['Learning Hours Spent']
```

Out[74]:
```
0       0.00
1       0.00
2       0.00
3       0.00
4       0.00
        ...
6630    7.31
6631    2.09
6632    0.00
6633    6.34
6634    0.00
Name: Learning Hours Spent, Length: 6635, dtype: float64
```

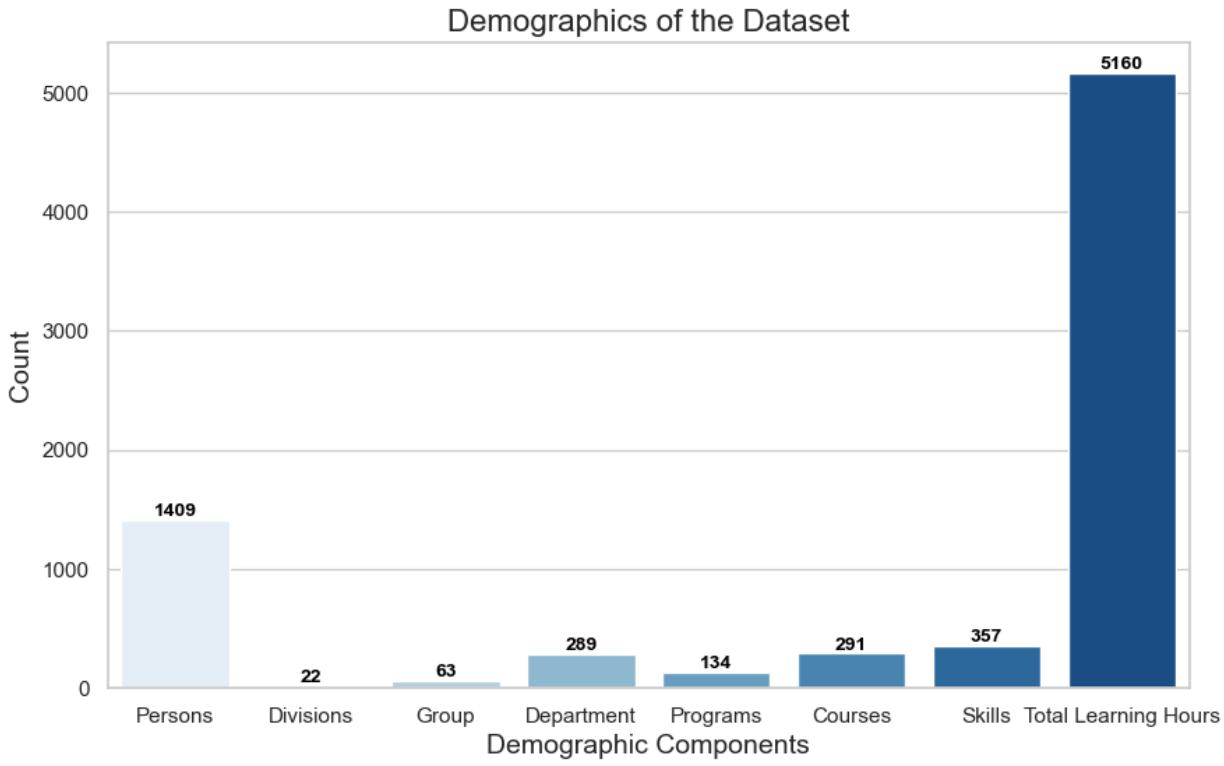
In [75]:
```
total_learning_hours = df['Learning Hours Spent'].sum()
print("Total Learning Hours Spent:", total_learning_hours)
```

```
Total Learning Hours Spent: 5160.04
```

Demographics Of The Dataset

- Persons - 1409
- Divisions - 22
- Group - 63
- Department- 289
- Programs - 134
- Courses - 291
- Skills- 357
- Total Learning Hours - 5160

In [76]:
```
demographics = {
    'Persons': 1409,
    'Divisions': 22,
    'Group': 63,
    'Department': 289,
    'Programs': 134,
    'Courses': 291,
    'Skills': 357,
    'Total Learning Hours': 5160
}
# Converting demographics dictionary to a DataFrame
demographics_df = pd.DataFrame(list(demographics.items()), columns=['Demographic Components', 'Count'])

# Setting the style of seaborn
sns.set(style="whitegrid")

# Plotting with seaborn
plt.figure(figsize=(10, 6))
bar_plot = sns.barplot(x='Demographic Components', y='Count', data=demographics_df, palette='Blues')

# Adding data labels on top of the bars
for p in bar_plot.patches:
    bar_plot.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2, p.get_height()),
                      ha='center', va='bottom', color='black', fontsize=10, fontweight='bold')

# Adding title and labels
plt.title('Demographics of the Dataset', fontsize=16)
plt.xlabel('Demographic Components', fontsize=14)
plt.ylabel('Count', fontsize=14)

# Show the plot
plt.show()
```



In [77]:
```
# Group by 'Division', 'Group', and 'Department' and get the count of learners
distribution_counts_division_group_Department = df.groupby(['Division', 'Group', 'Department']).size().reset_index(nam
distribution_counts_division_group_Department
distribution_counts_division_group_Department_sorted = distribution_counts_division_group_Department.sort_values(by='L
distribution_counts_division_group_Department_sorted
```

Out[77]:

	Division	Group	Department	Learner Count
68	One IT	Not Available	Automation	329
235	Tech&NMB	Not Available	Scientific Services	240
213	TSM	Khopoli Operations	Maintenance	238
30	HRM	Not Available	Learning & Development	159
202	TSM	Angul Operation - IM	Blast Furnaces	156
...	...	...	...	...
267	VP Financial Oprns.& Corporate Reporting	Office of VP FO&CR	Sust. Reporting & Disclosure	1
266	VP Financial Oprns.& Corporate Reporting	Office of VP FO&CR	Financial Controller S&S	1
264	VP Financial Oprns.& Corporate Reporting	Financial Planning & Group Reporting	Business Analysis Group	1
262	VP Financial Oprns.& Corporate Reporting	Financial Planning & Corporate Reporting	Capital Planning	1
66	M&S(LP)	Not Available	Steel Recycling Business	1

291 rows × 4 columns

- The ONE IT Division and the Automation Department has maximum number of learners enrolled - 329
- The M&S(LP) Division and the Steel Recycling Business Department has minimum number of learners enrolled - 1

- The distribution_counts_division_group_Department dataframe will help us track the learners from the three combnination as a whole

In [78]:
```
distribution_counts_division_group = df.groupby(['Division', 'Group']).size().reset_index(name='Learner Count')
distribution_counts_division_group
distribution_counts_division_group_sorted = distribution_counts_division_group.sort_values(by='Learner Count', ascendi
distribution_counts_division_group_sorted
```

Out[78]:

	Division	Group	Learner Count
21	One IT	Not Available	861
57	TSM	Khopoli Operations	469
17	M&S(FP)	Not Available	313
43	TQM and E&P	Design & Engineering	292
52	TSM	Angul Operation - IM	288
...	...	...	...
74	VP Financial Oprns.& Corporate Reporting	Financial Planning & Corporate Reporting	3
50	TQM and E&P	Projects & Construction TSM	3
4	Steel Manufacturing	Not Available	3
77	VP Financial Oprns.& Corporate Reporting	Office of VP FO&CR	2
75	VP Financial Oprns.& Corporate Reporting	Financial Planning & Group Reporting	1

82 rows × 3 columns

- The distribution_counts_division_group will help us to learners with the division and the group
- The One IT Division has the maximum number of learners enrolled in it with the number being 861
- The VP Financial Oprns.& Corporate Reporting division and the Financial Planning & Group Reporting has the minimum number of enrolled learners i.e is 1.

In [79]:
```
distribution_counts_division_Department = df.groupby(['Division', 'Department']).size().reset_index(name='Learner Coun
distribution_counts_division_Department
distribution_counts_division_Department_sorted = distribution_counts_division_Department.sort_values(by='Learner Count
distribution_counts_division_Department_sorted
```

Out[79]:

	Division	Department	Learner Count
68	One IT	Automation	329
238	Tech&NMB	Scientific Services	240
214	TSM	Maintenance	238
29	HRM	Learning & Development	159
203	TSM	Blast Furnaces	156
...	...	...	...
256	VP Financial Oprns.& Corporate Reporting	Business Analysis Group	1
262	VP Financial Oprns.& Corporate Reporting	Financial Controller S&S	1
259	VP Financial Oprns.& Corporate Reporting	Capital Planning	1
267	VP Financial Oprns.& Corporate Reporting	Sust. Reporting & Disclosure	1
65	M&S(LP)	Steel Recycling Business	1

291 rows × 3 columns

- This distribution_counts_division_Department dataframe will help us to track the division and department concurrently
- The maximum number of learners enrolled are in the one it division of the automation department - 329
- The minimum number of learners enrolled are in the M&S(LP) division of the Steel Recycling Business department - 1

In [80]:
```
distribution_counts_group_Department = df.groupby(['Group', 'Department']).size().reset_index(name='Learner Count')
distribution_counts_group_Department
distribution_counts_group_Department_sorted = distribution_counts_group_Department.sort_values(by='Learner Count', asc
distribution_counts_group_Department_sorted
```

Out[80]:

	Group	Department	Learner Count
158	Not Available	Automation	329
193	Not Available	Scientific Services	240
121	Khopoli Operations	Maintenance	238
179	Not Available	Learning & Development	159
7	Angul Operation - IM	Blast Furnaces	156
...	...	...	...
194	Not Available	Steel Recycling Business	1
218	Office of VP FO&CR	Sust. Reporting & Disclosure	1
217	Office of VP FO&CR	Financial Controller S&S	1
81	Financial Planning & Corporate Reporting	Capital Planning	1
83	Financial Planning & Group Reporting	Business Analysis Group	1

291 rows × 3 columns

- This distribution_counts_division_Department dataframe will help us to track the group and department concurrently

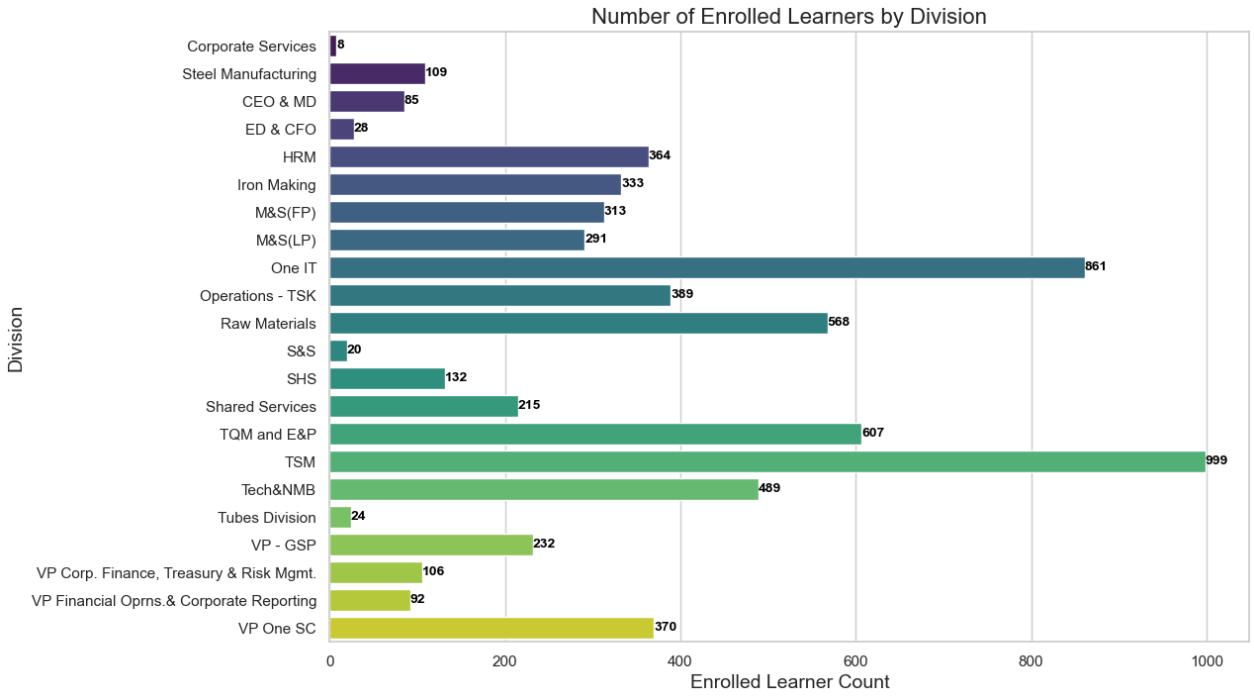
- Enrolled learners demographics

In [81]:
```
# Filtering for the enrolled learners
enrolled_learners = df

# Group by 'Division' and get the count of enrolled learners
enrolled_by_division = enrolled_learners.groupby('Division').size().reset_index(name='Enrolled Learner Count')
enrolled_by_division
```

Out[81]:

	Division	Enrolled Learner Count
0	Corporate Services	8
1	Steel Manufacturing	109
2	CEO & MD	85
3	ED & CFO	28
4	HRM	364
5	Iron Making	333
6	M&S(FP)	313
7	M&S(LP)	291
8	One IT	861
9	Operations - TSK	389
10	Raw Materials	568
11	S&S	20
12	SHS	132
13	Shared Services	215
14	TQM and E&P	607
15	TSM	999
16	Tech&NMB	489
17	Tubes Division	24
18	VP - GSP	232
19	VP Corp. Finance, Treasury & Risk Mgmt.	106
20	VP Financial Oprns.& Corporate Reporting	92
21	VP One SC	370

In [82]:
```
sns.set(style="whitegrid")

# Plotting the enrolled learners by division
plt.figure(figsize=(12, 8))
bar_plot = sns.barplot(x='Enrolled Learner Count', y='Division', data=enrolled_by_division, palette='viridis')

# Adding data labels on the bars
for p in bar_plot.patches:
    bar_plot.annotate(f'{int(p.get_width())}', (p.get_width(), p.get_y() + p.get_height() / 2),
                      ha='left', va='center', color='black', fontsize=10, fontweight='bold')

# Adding title and labels
plt.title('Number of Enrolled Learners by Division', fontsize=16)
plt.xlabel('Enrolled Learner Count', fontsize=14)
plt.ylabel('Division', fontsize=14)

# Show the plot
plt.show()
```



- The TSM Division is basically having the maximum number of enrolled learners - 999
- The Corporate Services Division is basically having the maximum number of enrolled learners - 999

In [83]:
```
df
```

Out[83]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Cou Gra
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
...	...	...	...	...	...	...	...	...	...	...	...	
6630	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	15-02-23 11:48:06	17-03-23 06:11:14	7.31	90
6631	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	02-04-23 14:10:21	1900-01-01 00:00:00	2.09	33
6632	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
6633	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	12-02-23 12:04:42	02-04-23 14:01:56	6.34	90
6634	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0

6635 rows × 13 columns

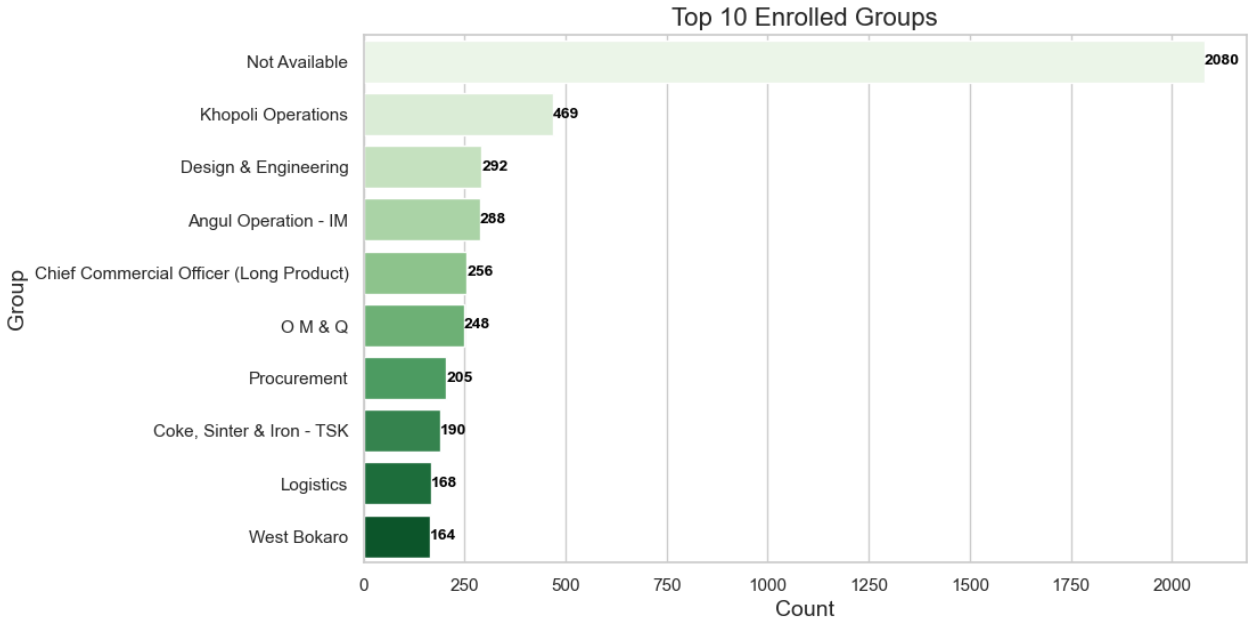
In [84]:
```
# Group by 'Group' and get the count of enrolled learners
enrolled_by_group = enrolled_learners.groupby('Group').size().reset_index(name='Enrolled Learner Count')
enrolled_by_group
```

Out[84]:

	Group	Enrolled Learner Count
0	Agglomerates	44
1	Angul Operation - IM	288
2	Angul Operation - SMU	43
3	Blast Furnaces	132
4	Chief Commercial Officer (Long Product)	256
...	...	...
58	Steel & Mills - TSK	79
59	Tata Growth Shop	21
60	Transformation	10
61	Vice President Operations TSM	6
62	West Bokaro	164

63 rows × 2 columns

In [85]:
```
# Getting the top 10 groups with the highest enrolled learner count
top_groups = enrolled_by_group.nlargest(10, 'Enrolled Learner Count')

# Set the style of seaborn
sns.set(style="whitegrid")

# Plotting with seaborn
plt.figure(figsize=(10, 6))
bar_plot = sns.barplot(x='Enrolled Learner Count', y='Group', data=top_groups, palette='Greens')

# Adding data labels on the right side of the bars
for p in bar_plot.patches:
    bar_plot.annotate(f'{int(p.get_width())}', (p.get_width(), p.get_y() + p.get_height() / 2),
                      ha='left', va='center', color='black', fontsize=10, fontweight='bold')

# Adding title and labels
plt.title('Top 10 Enrolled Groups', fontsize=16)
plt.xlabel('Count', fontsize=14)
plt.ylabel('Group', fontsize=14)

# Show the plot
plt.show()
```



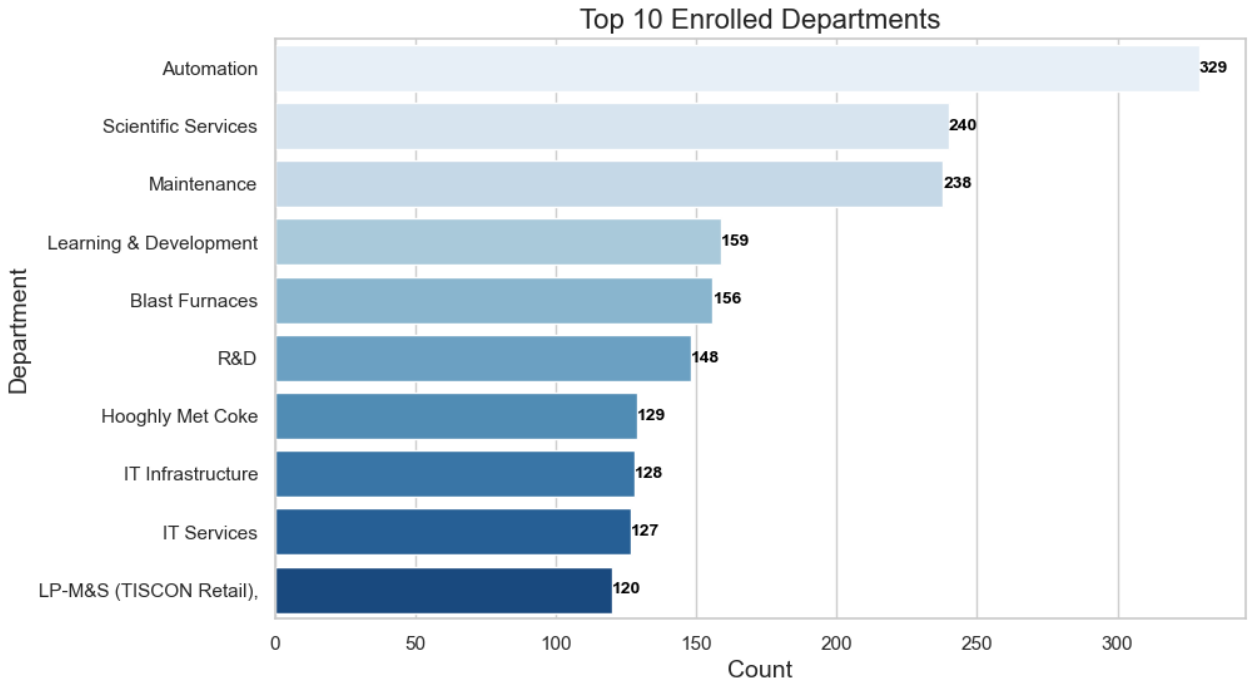
In [86]:
```
enrolled_by_department = enrolled_learners.groupby('Department').size().reset_index(name='Enrolled Learner Count')
enrolled_by_department
```

Out[86]:

	Department	Enrolled Learner Count
0	'MPPI	6
1	A-F Blast Furnaces	50
2	Administration	20
3	Administration Jharia	6
4	Administration OMQ	6
...	...	...
284	Vigilance	30
285	Visualisation	54
286	WCRM & NCRM	79
287	Washery II	6
288	Washery III	12

289 rows × 2 columns

In [228]:
```
# Getting the top 10 department with the highest enrolled learner count
top_departments = enrolled_by_department.nlargest(10, 'Enrolled Learner Count')

# Set the style of seaborn
sns.set(style="whitegrid")
# Plotting with seaborn
plt.figure(figsize=(10, 6))
bar_plot = sns.barplot(x='Enrolled Learner Count', y='Department', data=top_departments, palette='Blues')
# Adding data labels on the right side of the bars
for p in bar_plot.patches:
    bar_plot.annotate(f'{int(p.get_width())}', (p.get_width(), p.get_y() + p.get_height() / 2),
                      ha='left', va='center', color='black', fontsize=10, fontweight='bold')

# Adding title and labels
plt.title('Top 10 Enrolled Departments', fontsize=16)
plt.xlabel('Count', fontsize=14)
plt.ylabel('Department', fontsize=14)
plt.show()
```



In [88]:
```
# Sorting the DataFrame by 'Enrolled Learner Count' in descending order
enrolled_by_group_sorted = enrolled_by_group.sort_values(by='Enrolled Learner Count', ascending=False)

# Getting the group with the maximum count
max_group = enrolled_by_group_sorted.iloc[0]['Group']
max_count = enrolled_by_group_sorted.iloc[0]['Enrolled Learner Count']

# Getting the group with the second maximum count
second_max_group = enrolled_by_group_sorted.iloc[1]['Group']
second_max_count = enrolled_by_group_sorted.iloc[1]['Enrolled Learner Count']

# Getting the group with the minimum count
min_group = enrolled_by_group_sorted.iloc[-1]['Group']
min_count = enrolled_by_group_sorted.iloc[-1]['Enrolled Learner Count']

# Getting the group with the second minimum count
second_min_group = enrolled_by_group_sorted.iloc[-2]['Group']
second_min_count = enrolled_by_group_sorted.iloc[-2]['Enrolled Learner Count']

print("Group with Maximum Count:", max_group, "Count:", max_count)
print("Group with Second Maximum Count:", second_max_group, "Count:", second_max_count)
print("Group with Minimum Count:", min_group, "Count:", min_count)
print("Group with Second Minimum Count:", second_min_group, "Count:", second_min_count)
```

```
Group with Maximum Count: Not Available Count: 2080
Group with Second Maximum Count: Khopoli Operations Count: 469
Group with Minimum Count: Financial Planning & Group Reporting Count: 1
Group with Second Minimum Count: Office of VP FO&CR Count: 2
```

In [89]:
```
enrolled_by_group_sorted
```

Out[89]:

	Group	Enrolled Learner Count
36	Not Available	2080
29	Khopoli Operations	469
12	Design & Engineering	292
1	Angul Operation - IM	288
4	Chief Commercial Officer (Long Product)	256
...	...	...
54	Security & Brand Protection	4
20	Financial Planning & Corporate Reporting	3
49	Projects & Construction TSM	3
38	Office of VP FO&CR	2
21	Financial Planning & Group Reporting	1

63 rows × 2 columns

In [90]:
```
# Sorting the DataFrame by 'Enrolled Learner Count' in descending order
enrolled_by_division_sorted = enrolled_by_division.sort_values(by='Enrolled Learner Count', ascending=False)

# Getting the group with the maximum count
max_division = enrolled_by_division_sorted.iloc[0]['Division']
max_count = enrolled_by_division_sorted.iloc[0]['Enrolled Learner Count']

# Getting the group with the second maximum count
second_max_division = enrolled_by_division_sorted.iloc[1]['Division']
second_max_count = enrolled_by_division_sorted.iloc[1]['Enrolled Learner Count']

# Getting the group with the minimum count
min_division = enrolled_by_division_sorted.iloc[-1]['Division']
min_count = enrolled_by_division_sorted.iloc[-1]['Enrolled Learner Count']

# Getting the group with the second minimum count
second_min_division = enrolled_by_division_sorted.iloc[-2]['Division']
second_min_count = enrolled_by_division_sorted.iloc[-2]['Enrolled Learner Count']

print("Division with Maximum Count:", max_division, "Count:", max_count)
print("Division with Second Maximum Count:", second_max_division, "Count:", second_max_count)
print("Division with Minimum Count:", min_division, "Count:", min_count)
print("Division with Second Minimum Count:", second_min_division, "Count:", second_min_count)
```

```
Division with Maximum Count: TSM Count: 999
Division with Second Maximum Count: One IT Count: 861
Division with Minimum Count:  Corporate Services Count: 8
Division with Second Minimum Count: S&S Count: 20
```

In [91]:
```
enrolled_by_division_sorted
```

Out[91]:

	Division	Enrolled Learner Count
15	TSM	999
8	One IT	861
14	TQM and E&P	607
10	Raw Materials	568
16	Tech&NMB	489
9	Operations - TSK	389
21	VP One SC	370
4	HRM	364
5	Iron Making	333
6	M&S(FP)	313
7	M&S(LP)	291
18	VP - GSP	232
13	Shared Services	215
12	SHS	132
1	Steel Manufacturing	109
19	VP Corp. Finance, Treasury & Risk Mgmt.	106
20	VP Financial Oprns.& Corporate Reporting	92
2	CEO & MD	85
3	ED & CFO	28
17	Tubes Division	24
11	S&S	20
0	Corporate Services	8

In [92]:
```
program_counts = df.groupby('Program Name').size().reset_index(name='Learner Count')
# Sort the DataFrame by 'Learner Count' in descending order
most_popular_programs = program_counts.sort_values(by='Learner Count', ascending=False)
most_popular_programs
```

Out[92]:

	Program Name	Learner Count
104	School of Analytics-Basic	2026
105	School of Blockchain	588
96	Project Mgmt Learning Program-Generic	408
13	Business Communication Learning Program	384
111	Six Sigma Black Belt Learning Program	360
...	...	...
75	MATLAB Learning Program ONE IT	1
71	Leading Positive Change Learning Program	1
48	Finance Learning Program ONE IT	1
66	Innovation and emerging technology	1
76	ML & AI on GCP Learning Program	1

134 rows × 2 columns

- The most popular programs which can be clearly seen as are "School Of Analytics-Basic" , "School Of Blockchain", "Project Mgmt Learning Program-Generic" followed by "Business Communication Learning Program"

In [93]:
```
# Calculating the completion rates for each course
course_completion_rates = df.groupby('Course Name')['Completed'].apply(lambda x: x.eq('Yes').sum() / x.notna().sum()).
course_completion_rates
```

Out[93]:

	Course Name	Completion Rate
0	AI Capstone Project with Deep Learning	0.034483
1	AI For Everyone	0.000000
2	AI Workflow: AI in Production	0.000000
3	AI Workflow: Business Priorities and Data Inge...	0.000000
4	AI Workflow: Data Analysis and Hypothesis Testing	0.000000
...	...	...
286	Transacting on the Blockchain	0.013605
287	User Experience & Interaction Design for AR/VR...	0.000000
288	Visual Analytics with Tableau	0.014286
289	What is Data Science?	0.083333
290	Work Smarter Not Harder: Time Management for P...	0.200000

291 rows × 2 columns

In [94]:
```
enrolled_learners = df[df['Enrollment Time'] != 'Not Started Yet']

# Grouping by the 'Program Name' and 'Course Name'
program_completion_summary = enrolled_learners.groupby(['Program Name', 'Course Name'])['Completed'].agg(
    TotalLearners='count',
    CompletedLearners=lambda x: x.eq('Yes').sum(),
    CompletionRate=lambda x: x.eq('Yes').sum() / x.notna().sum(),
    CompletionRatePercentage=lambda x: (x.eq('Yes').sum() / x.notna().sum()) * 100
).reset_index()
program_completion_summary
```

Out[94]:

	Program Name	Course Name	TotalLearners	CompletedLearners	CompletionRate	CompletionRatePercentage
0	AI Engineering Learning Program ONE IT	AI Capstone Project with Deep Learning	29	1	0.034483	3.448276
1	AI Engineering Learning Program ONE IT	Building Deep Learning Models with TensorFlow	29	2	0.068966	6.896552
2	AI Engineering Learning Program ONE IT	Deep Neural Networks with PyTorch	29	2	0.068966	6.896552
3	AI Engineering Learning Program ONE IT	Introduction to Deep Learning & Neural Network...	29	7	0.241379	24.137931
4	AI Engineering Learning Program ONE IT	Machine Learning with Python	29	12	0.413793	41.379310
...	...	...	...	...	...	...
321	Visualization Learning Program	Python Data Visualization	1	0	0.000000	0.000000
322	Visualization Learning Program	Visual Analytics with Tableau	1	0	0.000000	0.000000
323	Web Development Learning Program ONE IT	Advanced React	5	0	0.000000	0.000000
324	Web Development Learning Program ONE IT	React Basics	5	0	0.000000	0.000000
325	Web Development Learning Program ONE IT	Responsive Website Basics: Code with HTML CSS ...	5	0	0.000000	0.000000

326 rows × 6 columns

In [95]:
```
program_completion_summary_sorted = program_completion_summary.sort_values(by='CompletionRatePercentage', ascending=Fa
program_completion_summary_sorted
```

Out[95]:

	Program Name	Course Name	TotalLearners	CompletedLearners	CompletionRate	CompletionRatePercentage
55	Computer Vision with Tensor Flow ONE IT	Advanced Computer Vision with TensorFlow	1	1	1.0	100.0
276	Software Defined Networking ONE IT	Intel® Network Academy - Network Transformatio...	1	1	1.0	100.0
254	Security Engineering Learn Prog ONE IT	Introduction to Cybersecurity Tools & Cyber At...	1	1	1.0	100.0
253	Security Engineering Learn Prog ONE IT	Cybersecurity Compliance Framework & System Ad...	1	1	1.0	100.0
246	Scrum Master Learning Program	Introduction to Scrum Master Training	5	5	1.0	100.0
...	...	...	...	...	...	...
186	Marketing Strategy Learning Program	Market Research and Consumer Behavior	9	0	0.0	0.0
187	Marketing Strategy Learning Program	Marketing Mix Fundamentals	9	0	0.0	0.0
188	Marketing Strategy Learning Program	Marketing Strategy Capstone Project	9	0	0.0	0.0
189	Marketing Strategy Learning Program	Positioning: What you need for a successful Ma...	9	0	0.0	0.0
325	Web Development Learning Program ONE IT	Responsive Website Basics: Code with HTML CSS ...	5	0	0.0	0.0

326 rows × 6 columns

In [96]:
```
unique_completion_rates = program_completion_summary_sorted['CompletionRatePercentage'].nunique()
print(f'Number of unique completion rate percentages: {unique_completion_rates}')
```

```
Number of unique completion rate percentages: 58
```

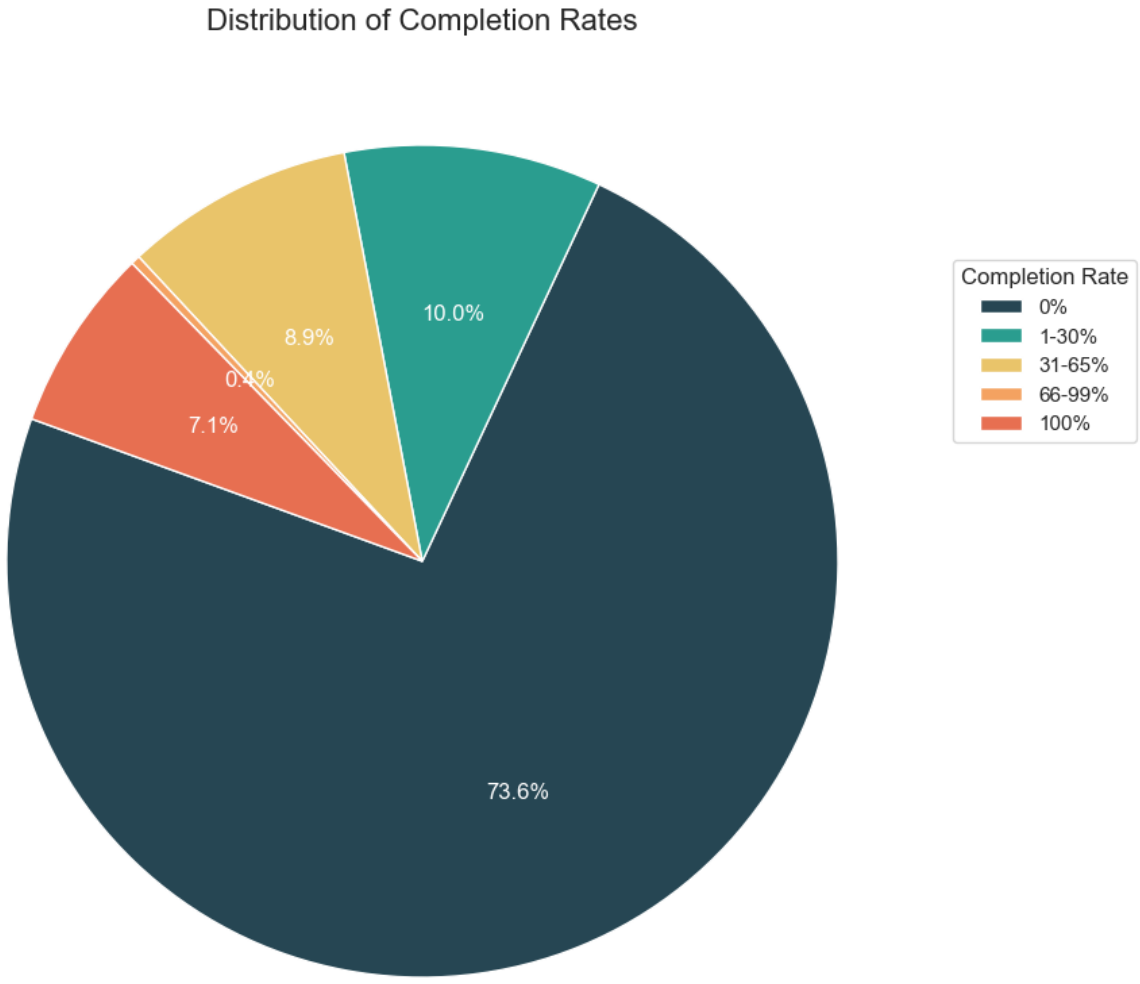
In [97]:
```
completion_rate_counts = program_completion_summary_sorted['CompletionRatePercentage'].round(1).value_counts().reset_i
completion_rate_counts.columns = ['CompletionRatePercentage', 'Frequency']

# Convert to dictionary
completion_rate_dict = dict(zip(completion_rate_counts['CompletionRatePercentage'], completion_rate_counts['Frequency'
completion_rate_dict
```

Out[97]:
```
{0.0: 206,
 100.0: 20,
 33.3: 10,
 50.0: 7,
 60.0: 5,
 16.7: 5,
 14.3: 4,
 1.4: 3,
 2.2: 3,
 4.4: 3,
 6.9: 3,
 5.4: 2,
 5.7: 2,
 6.2: 2,
 6.7: 2,
 6.8: 2,
 8.3: 2,
 10.0: 2,
 12.5: 2,
 12.7: 2,
 25.0: 2,
 24.1: 2,
 2.1: 1,
 40.0: 1,
 2.7: 1,
 3.1: 1,
 3.4: 1,
 4.2: 1,
 24.5: 1,
 30.8: 1,
 4.5: 1,
 4.9: 1,
 5.0: 1,
 26.7: 1,
 5.6: 1,
 25.7: 1,
 41.4: 1,
 4.3: 1,
 16.9: 1,
 18.8: 1,
 7.5: 1,
 16.4: 1,
 15.6: 1,
 15.4: 1,
 18.9: 1,
 13.7: 1,
 13.5: 1,
 13.3: 1,
 80.0: 1,
 20.0: 1,
 11.1: 1,
 10.5: 1,
 22.2: 1,
 8.5: 1,
 23.6: 1,
 8.1: 1,
 24.4: 1}
```

In [99]:
```
rounded_completion_rate_counts = completion_rate_counts.copy()
rounded_completion_rate_counts['CompletionRatePercentage'] = rounded_completion_rate_counts['CompletionRatePercentage'
rounded_completion_rate_dict = dict(zip(rounded_completion_rate_counts['CompletionRatePercentage'], rounded_completion_

# Display the updated dictionary
rounded_completion_rate_dict
```

Out[99]:
```
{0: 206,
 100: 20,
 33: 10,
 50: 7,
 60: 5,
 17: 1,
 14: 1,
 1: 3,
 2: 1,
 4: 1,
 7: 2,
 5: 1,
 6: 1,
 8: 1,
 10: 1,
 12: 2,
 13: 1,
 25: 2,
 24: 1,
 40: 1,
 3: 1,
 31: 1,
 27: 1,
 26: 1,
 41: 1,
 19: 1,
 16: 1,
 15: 1,
 80: 1,
 20: 1,
 11: 1,
 22: 1}
```

In [102]:
```
ranges = [
    {'label': '0%', 'start': -0.1, 'end': 0},
    {'label': '1-30%', 'start': 0.1, 'end': 30},
    {'label': '31-65%', 'start': 30.1, 'end': 65},
    {'label': '66-99%', 'start': 65.1, 'end': 99.9},
    {'label': '100%', 'start': 100, 'end': 100}
]

# Categorize completion rates
rounded_completion_rate_df['Category'] = pd.cut(
    rounded_completion_rate_df['CompletionRatePercentage'],
    bins=[range['start'] for range in ranges] + [float('inf')],
    labels=[range['label'] for range in ranges],
    right=False
)

# Group by category and sum the frequencies
category_counts = rounded_completion_rate_df.groupby('Category')['Frequency'].sum().reset_index()

# Set a custom color palette
custom_palette = ["#264653", "#2a9d8f", "#e9c46a", "#f4a261", "#e76f51"]

# Set a larger figure size
plt.figure(figsize=(12, 10))

# Plotting the pie chart
plt.pie(category_counts['Frequency'], labels=category_counts['Category'],
        autopct='%1.1f%%', startangle=160, wedgeprops=dict(width=1.0), textprops=dict(color="w"))

# Adding a legend
plt.legend(title="Completion Rate", loc="upper left", bbox_to_anchor=(1, 0.8))

# Adding a title
plt.title("Distribution of Completion Rates", fontsize=16)

# Show the plot
plt.show()
```



In [103]:
```
enrolled_learners = df[df['Enrollment Time'] != 'Not Started Yet']

# Group by 'Course Name' and calculate completion rates
course_completion_rates = enrolled_learners.groupby('Course Name')['Completed'].agg(
    TotalLearners='count',
    CompletedLearners=lambda x: x.eq('Yes').sum(),
    CompletionRate=lambda x: x.eq('Yes').sum() / x.notna().sum()
).reset_index()

# Set your threshold for significantly higher or lower completion rates (adjust as needed)
threshold_higher = 0.8
threshold_lower = 0.04

# Identify courses with significantly higher completion rates
higher_completion_courses = course_completion_rates[course_completion_rates['CompletionRate'] > threshold_higher]

# Identify courses with significantly lower completion rates
lower_completion_courses = course_completion_rates[course_completion_rates['CompletionRate'] < threshold_lower]
```

In [104]:
```
higher_completion_courses
```

Out[104]:

	Course Name	TotalLearners	CompletedLearners	CompletionRate
13	Advanced Computer Vision with TensorFlow	1	1	1.000000
21	Artificial Intelligence in Marketing	1	1	1.000000
34	Build personal resilience	1	1	1.000000
39	Business English: Planning & Negotiating	1	1	1.000000
46	Calculus for Machine Learning and Data Science	1	1	1.000000
49	Cloud Computing Basics (Cloud 101)	2	2	1.000000
53	Computer Vision with Embedded Machine Learning	1	1	1.000000
102	Engineering Project Management: Initiating and...	1	1	1.000000
103	Engineering Project Management: Risk, Quality,...	1	1	1.000000
104	Engineering Project Management: Scope, Time an...	1	1	1.000000
105	Enterprise and Infrastructure Security	1	1	1.000000
122	From Idea to Startup	1	1	1.000000
154	Intel® Network Academy - Network Transformatio...	1	1	1.000000
155	Intel® Network Academy - Network Transformatio...	1	1	1.000000
169	Introduction to Scrum Master Training	6	5	0.833333
176	Leading transformations: Manage change	1	1	1.000000
178	Linear Algebra for Machine Learning and Data S...	1	1	1.000000
251	Smart Analytics Machine Learning and AI on Goo...	1	1	1.000000

In [105]:
```
lower_completion_courses
```

Out[105]:

	Course Name	TotalLearners	CompletedLearners	CompletionRate
0	AI Capstone Project with Deep Learning	29	1	0.034483
1	AI For Everyone	3	0	0.000000
2	AI Workflow: AI in Production	2	0	0.000000
3	AI Workflow: Business Priorities and Data Inge...	2	0	0.000000
4	AI Workflow: Data Analysis and Hypothesis Testing	2	0	0.000000
...	...	...	...	...
284	Tools for Data Science	3	0	0.000000
285	Training and Learning Online	1	0	0.000000
286	Transacting on the Blockchain	147	2	0.013605
287	User Experience & Interaction Design for AR/VR...	1	0	0.000000
288	Visual Analytics with Tableau	70	1	0.014286

185 rows × 4 columns

- This higher_completion_courses and the lower_completion_courses will show the list of records having completion rate higher than 80% and lower completion will show records having the records having completion rates lower than 4%

In [106]:
```
courses_with_zero_completion_rate = course_completion_rates[course_completion_rates['CompletionRate'] == 0.00]
courses_with_zero_completion_rate
```

Out[106]:

	Course Name	TotalLearners	CompletedLearners	CompletionRate
1	AI For Everyone	3	0	0.0
2	AI Workflow: AI in Production	2	0	0.0
3	AI Workflow: Business Priorities and Data Inge...	2	0	0.0
4	AI Workflow: Data Analysis and Hypothesis Testing	2	0	0.0
5	AI Workflow: Enterprise Model Deployment	2	0	0.0
...	...	...	...	...
280	The Marketing Plan	9	0	0.0
283	Think Outside the Inbox: Email Marketing	14	0	0.0
284	Tools for Data Science	3	0	0.0
285	Training and Learning Online	1	0	0.0
287	User Experience & Interaction Design for AR/VR...	1	0	0.0

175 rows × 4 columns

In [107]:
```
completed_learners = df[(df['Completed'] == 'Yes') & df['Learning Hours Spent'].notna()]
average_learning_hours = completed_learners['Learning Hours Spent'].mean()
average_learning_hours
```

Out[107]: 5.479644444444444

In [108]:
```
completed_learners = df[(df['Completed'] == 'Yes') & df['Learning Hours Spent'].notna()]
average_learning_hours_per_course = completed_learners.groupby('Course Name')['Learning Hours Spent'].mean().reset_ind
average_learning_hours_per_course
```

Out[108]:

	Course Name	Average Learning Hours
0	AI Capstone Project with Deep Learning	4.45000
1	Advanced Computer Vision with TensorFlow	3.17000
2	Applied Data Science Capstone	0.98500
3	Artificial Intelligence in Marketing	0.81000
4	Basic Data Descriptors Statistical Distributio...	5.92066
...	...	...
111	The Nuts and Bolts of Public Relations	5.21000
112	Transacting on the Blockchain	4.21500
113	Visual Analytics with Tableau	0.82000
114	What is Data Science?	1.67000
115	Work Smarter Not Harder: Time Management for P...	4.19000

116 rows × 2 columns

- This is only for those learners data who have completed the course atleast once

In [110]:
```
completed_learners_all = df[df['Learning Hours Spent'].notna()]
average_learning_hours_all = completed_learners_all['Learning Hours Spent'].mean()
average_learning_hours_all
```

Out[110]: 0.7777000753579503

In [111]:
```
average_learning_hours_per_course_all = completed_learners_all.groupby('Course Name')['Learning Hours Spent'].mean().r
average_learning_hours_per_course_all
```

Out[111]:

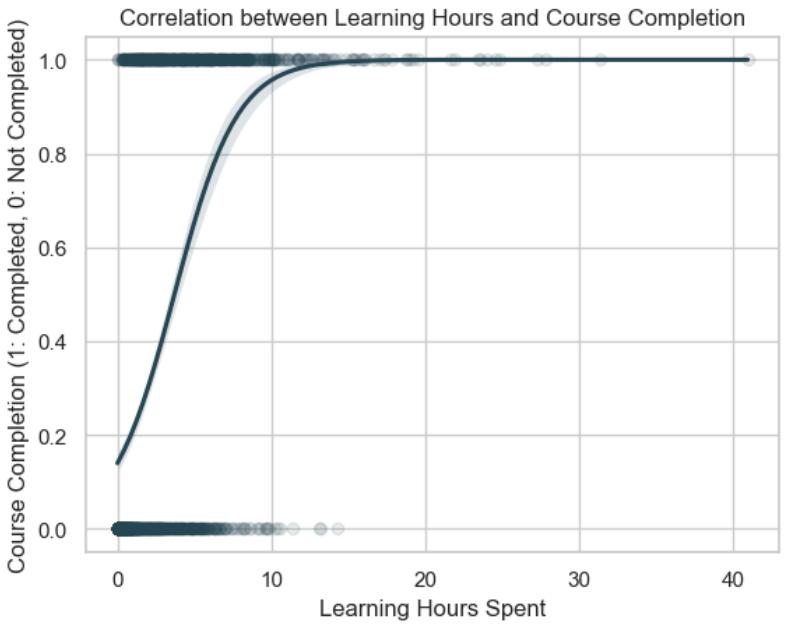
	Course Name	Average Learning Hours
0	AI Capstone Project with Deep Learning	0.16069
1	AI For Everyone	0.00000
2	AI Workflow: AI in Production	0.00000
3	AI Workflow: Business Priorities and Data Inge...	0.00000
4	AI Workflow: Data Analysis and Hypothesis Testing	0.00000
...	...	...
286	Transacting on the Blockchain	0.16551
287	User Experience & Interaction Design for AR/VR...	0.00000
288	Visual Analytics with Tableau	0.04400
289	What is Data Science?	0.23500
290	Work Smarter Not Harder: Time Management for P...	0.83800

291 rows × 2 columns

- This is the average learning hours for all the courses

In [115]:
```
learners_atleast_started_the_course = df[(df['Completed'] != 'Not Started Yet')]
learners_atleast_started_the_course['Completed'] = learners_atleast_started_the_course['Completed'].map({'Yes': 1, 'No

# Plot a scatter plot with logistic regression line
sns.regplot(x='Learning Hours Spent', y='Completed', data=learners_atleast_started_the_course, logistic=True, scatter_

# Show the plot
plt.title('Correlation between Learning Hours and Course Completion')
plt.xlabel('Learning Hours Spent')
plt.ylabel('Course Completion (1: Completed, 0: Not Completed)')
plt.show()
```

```
C:\Users\Siddharth\AppData\Local\Temp\ipykernel_14528\3795878626.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning
-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy)
  learners_atleast_started_the_course['Completed'] = learners_atleast_started_the_course['Completed'].map({'Yes': 1,
'No': 0})
```



- Scatter Plot:
- Each point on the scatter plot represents a data point with the learning hours spent on the x-axis and the completion status on the y-axis.
- Points with a value of 1 on the y-axis indicate completed courses, and points with a value of 0 indicate not completed courses.
- Logistic Regression Line:

- The logistic regression line is a curve that represents the best fit to the relationship between learning hours and the probability of course completion.
- It shows how the probability of course completion changes as the learning hours increase.
- Interpretation:
- The logistic regression line slopes upward, it suggests a positive correlation, indicating that higher learning hours are associated with a higher probability of course completion.
- Alpha Parameter:
- The scatter_kws={'alpha': 0.1} parameter is used to control the transparency of the scatter plot points. A lower alpha value (closer to 0)

In [116]:
```
correlation = learners_atleast_started_the_course['Learning Hours Spent'].corr(learners_atleast_started_the_course['Co
correlation
```

Out[116]:
```
0.5776350953280784
```

In [117]:
```
course_wise_correlation = learners_atleast_started_the_course.groupby('Course Name').apply(lambda group: group['Learni

course_wise_correlation
```

```
C:\ProgramData\anaconda3\Lib\site-packages\numpy\lib\function_base.py:2846: RuntimeWarning: Degrees of freedom <= 0 f
or slice
  c = cov(x, y, rowvar, dtype=dtype)
C:\ProgramData\anaconda3\Lib\site-packages\numpy\lib\function_base.py:2705: RuntimeWarning: divide by zero encountere
d in divide
  c *= np.true_divide(1, fact)
```

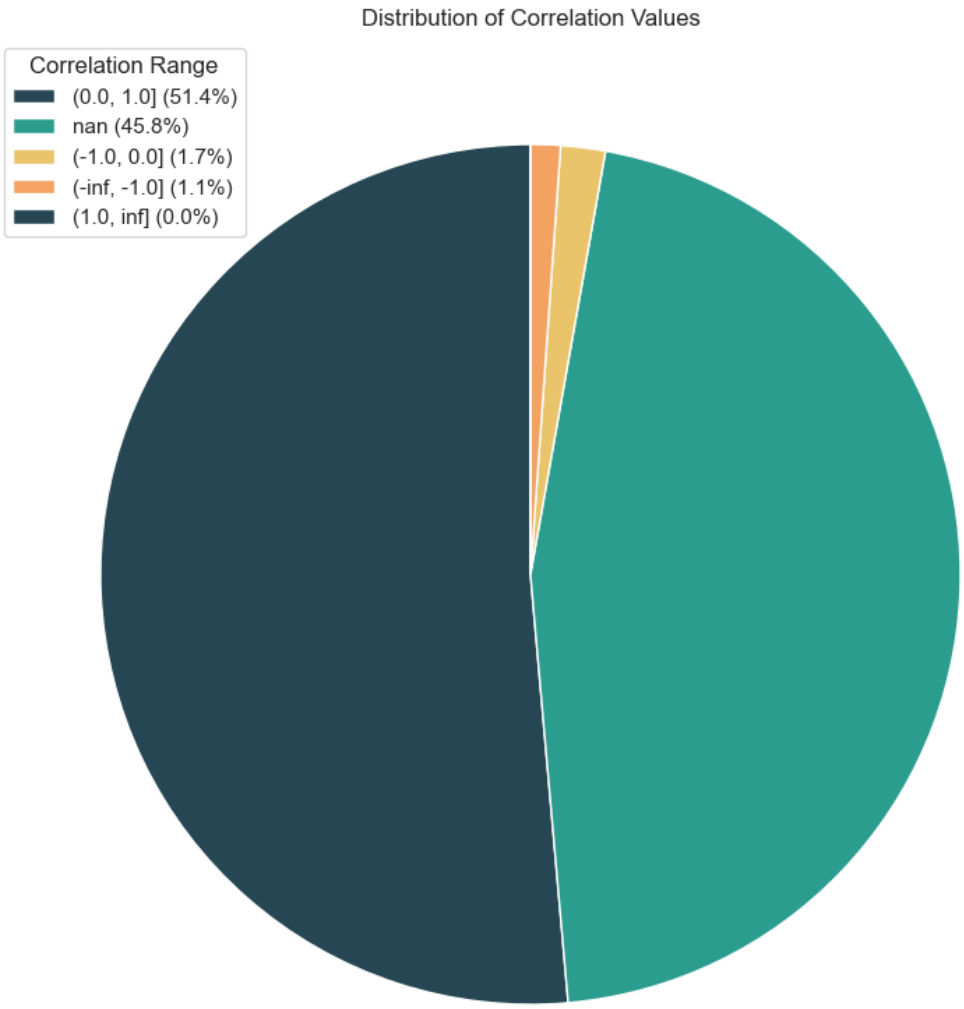
Out[117]:

	Course Name	Correlation
0	AI Capstone Project with Deep Learning	1.000000
1	Accounting: Principles of Financial Accounting	NaN
2	Advanced Computer Vision with TensorFlow	NaN
3	Advanced React	NaN
4	Algorithms Part I	NaN
...	...	...
174	Tools for Data Science	NaN
175	Transacting on the Blockchain	0.729776
176	Visual Analytics with Tableau	-1.000000
177	What is Data Science?	1.000000
178	Work Smarter Not Harder: Time Management for P...	NaN

179 rows × 2 columns

In [130]:
```
learners_atleast_started_the_course['Completed'] = learners_atleast_started_the_course['Completed'].map({'Yes': 1, 'No

course_wise_correlation = learners_atleast_started_the_course.groupby('Course Name').apply(lambda group: group['Learni

# Define the bins for categorization
bins = [-np.inf, -1, 0, 1, np.inf]

# Categorize the correlation values into bins and calculate value counts
correlation_bins = pd.cut(course_wise_correlation['Correlation'], bins=bins)
correlation_counts = correlation_bins.value_counts(dropna=False)

# Plotting a pie chart using Seaborn
custom_palette = ["#264653", "#2a9d8f", "#e9c46a", "#f4a261"]
plt.figure(figsize=(10, 10))
sns.set_palette(custom_palette)  # Set a color palette

# Plot a pie chart using correlation_counts
plt.pie(correlation_counts, labels=[''] * len(correlation_counts), autopct='', startangle=90)

# Create a legend
legend_labels = [f'{label} ({percentage:.1f}%)' for label, percentage in zip(correlation_counts.index, correlation_cou
plt.legend(title='Correlation Range', labels=legend_labels, loc='upper left')

# Show the plot
plt.title('Distribution of Correlation Values')
plt.show()
```

```
C:\Users\Siddharth\AppData\Local\Temp\ipykernel_14528\3407983466.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning
-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-
a-copy)
  learners_atleast_started_the_course['Completed'] = learners_atleast_started_the_course['Completed'].map({'Yes': 1,
'No': 0})
C:\ProgramData\anaconda3\Lib\site-packages\numpy\lib\function_base.py:2846: RuntimeWarning: Degrees of freedom <= 0 f
or slice
  c = cov(x, y, rowvar, dtype=dtype)
C:\ProgramData\anaconda3\Lib\site-packages\numpy\lib\function_base.py:2705: RuntimeWarning: divide by zero encountere
d in divide
  c *= np.true_divide(1, fact)
```

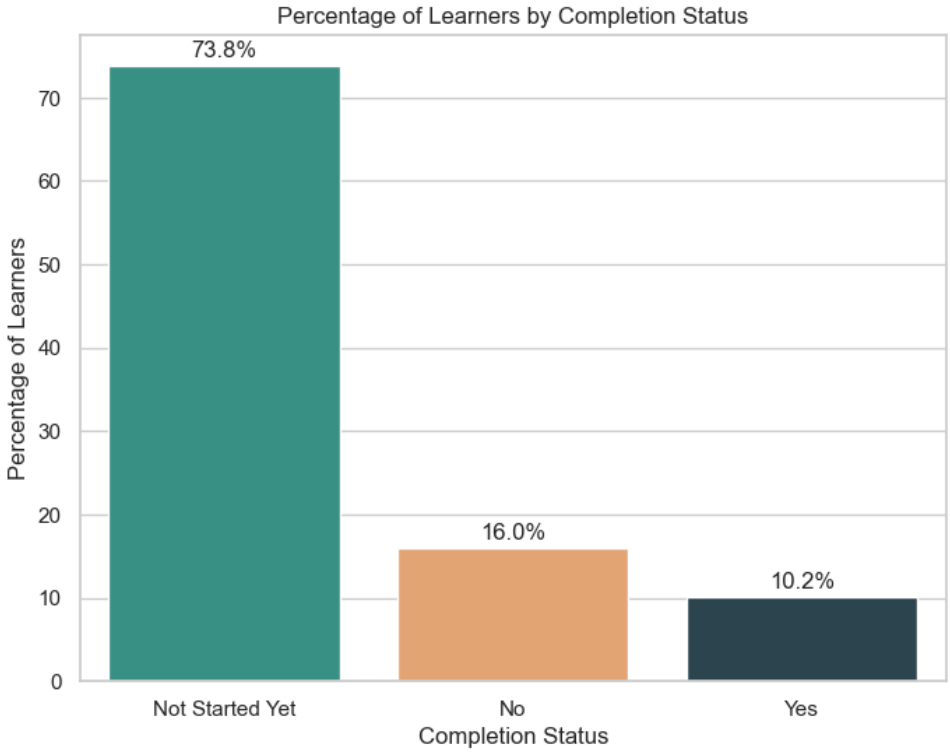


- 51 percent of the enrolled learners have positive correlation and the 45 percent of the users have NaN correlation meaning they havent started the course also yet and the remaining 4 percent learners have neagtive correlation between the course grade anf the learning hours

In [132]:
```
df['Completed']
```

Out[132]:
```
0       Not Started Yet
1       Not Started Yet
2       Not Started Yet
3       Not Started Yet
4       Not Started Yet
             ...
6630               Yes
6631                No
6632    Not Started Yet
6633               Yes
6634    Not Started Yet
Name: Completed, Length: 6635, dtype: object
```

In [135]:
```
percentage_counts = df['Completed'].value_counts(normalize=True) * 100

# Create a bar plot using Seaborn
custom_palette = ["#2a9d8f", "#f4a261", "#264653"]
plt.figure(figsize=(8, 6))
sns.set_palette(custom_palette)

# Use the 'barplot' function to create the bar graph
sns.barplot(x=percentage_counts.index, y=percentage_counts.values)

# Add labels and title
plt.xlabel('Completion Status')
plt.ylabel('Percentage of Learners')
plt.title('Percentage of Learners by Completion Status')

# Display percentages on top of each bar
for index, value in enumerate(percentage_counts.values):
    plt.text(index, value + 0.5, f'{value:.1f}%', ha='center', va='bottom')

plt.show()
```



In [136]:
```
df
```

Out[136]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Course Grade
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00
..	...	...	...	...	...	...	...	...	...	...	...	...
0	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	15-02-23 11:48:06	17-03-23 06:11:14	7.31	90.00
1	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	02-04-23 14:10:21	1900-01-01 00:00:00	2.09	33.23
2	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00
3	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	12-02-23 12:04:42	02-04-23 14:01:56	6.34	90.00
4	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0.00

5 rows × 13 columns

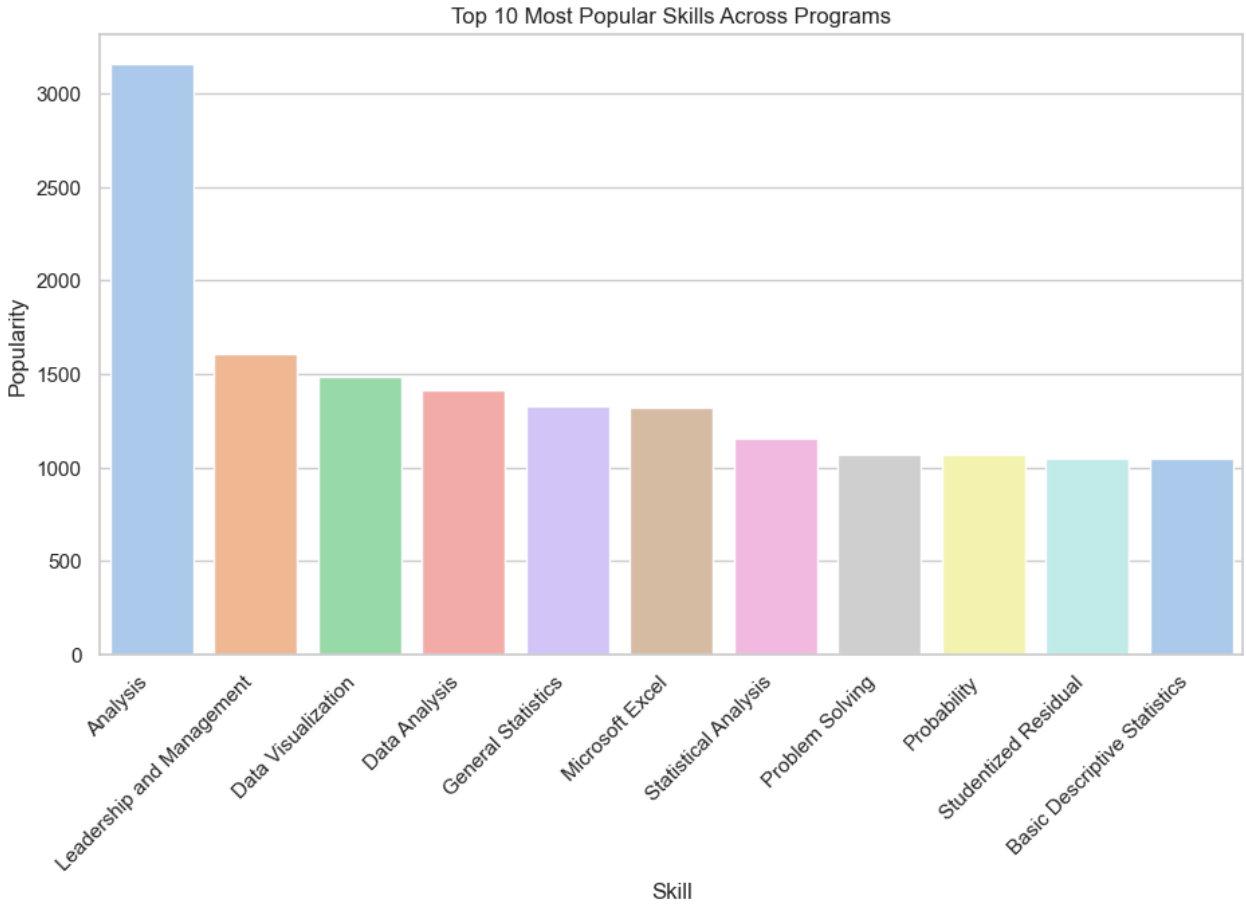
In [138]:
```
skills_counts = df_test['Skills Learned'].str.lower().str.split(';').explode().str.strip().value_counts()
df_skills_counts = pd.DataFrame({'Skill': skills_counts.index, 'Count': skills_counts.values})
df_skills_counts
```

Out[138]:

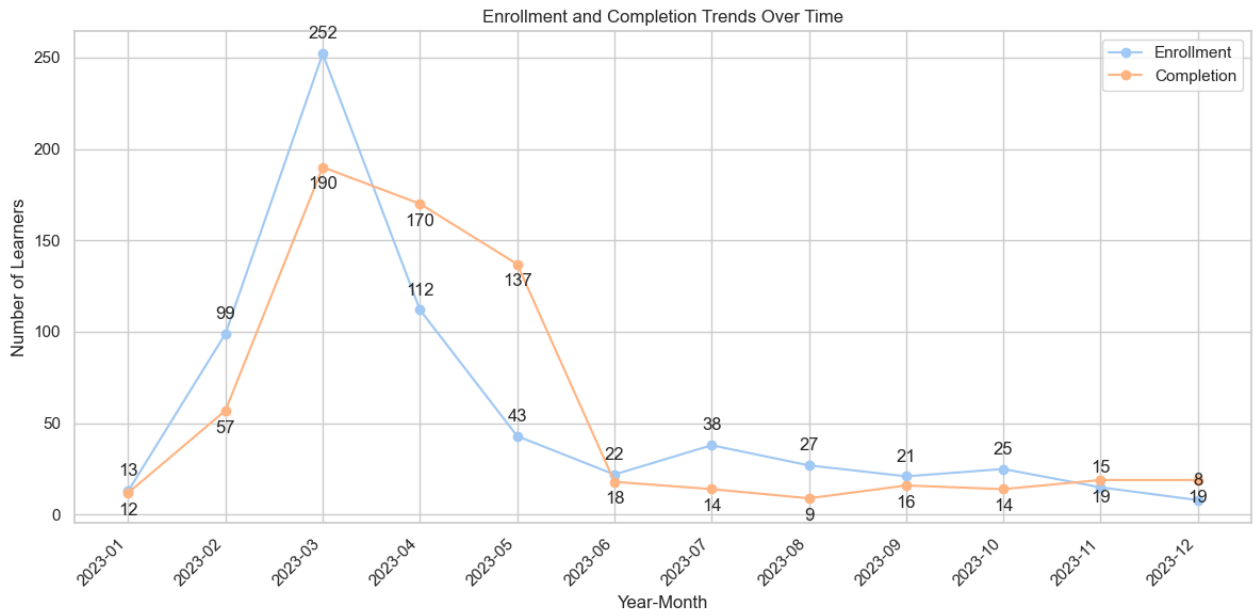
	Skill	Count
0	analysis	3160
1	leadership and management	1607
2	data visualization	1488
3	data analysis	1412
4	general statistics	1325
...	...	...
352	search algorithm	1
353	graphs	1
354	apache hadoop	1
355	financial risk	1
356	personal advertisement	1

357 rows × 2 columns

In [139]:
```
df_cleaned = df.dropna(subset=['Skills Learned'])

# Creating a dictionary to store the count of programs for each skill present in the dataset
skill_popularity = {}

# Iterate over rows and populate the dictionary
for index, row in df_cleaned.iterrows():
    skills = set(row['Skills Learned'].split('; '))

    for skill in skills:
        if skill not in skill_popularity:
            skill_popularity[skill] = 1
        else:
            skill_popularity[skill] += 1

# Converting the dictionary to a DataFrame
popularity_df = pd.DataFrame(list(skill_popularity.items()), columns=['Skill', 'Popularity'])

# Sort the DataFrame by popularity in descending order
popularity_df = popularity_df.sort_values(by='Popularity', ascending=False)

# Display the top skills
top_skills = popularity_df.head(11)

# Set a Seaborn color palette
sns.set_palette("pastel")

# Plot a bar graph using Seaborn
plt.figure(figsize=(11, 6))
sns.barplot(x='Skill', y='Popularity', data=top_skills, palette="pastel")
plt.xlabel('Skill')
plt.ylabel('Popularity')
plt.title('Top 10 Most Popular Skills Across Programs')
plt.xticks(rotation=45, ha='right')
plt.show()
```



In [141]:
```
df['Enrollment Time'] = pd.to_datetime(df['Enrollment Time'], errors='coerce')
df['Completion Time'] = pd.to_datetime(df['Completion Time'], errors='coerce')

# Extract year and month from 'Enrollment Time' and 'Completion Time' columns
df['Enrollment YearMonth'] = df['Enrollment Time'].dt.to_period('M')
df['Completion YearMonth'] = df['Completion Time'].dt.to_period('M')

# Filter out records with '1900-01-01' in either 'Enrollment Time' or 'Completion Time'
df_filtered = df[(df['Enrollment YearMonth'].dt.to_timestamp() != '1900-01-01') &
                 (df['Completion YearMonth'].dt.to_timestamp() != '1900-01-01')]

# Group by enrollment and completion time and count the number of records
enrollment_counts = df_filtered.groupby('Enrollment YearMonth').size()
completion_counts = df_filtered.groupby('Completion YearMonth').size()

# Create a DataFrame for plotting
plot_data = pd.DataFrame({'Enrollment': enrollment_counts, 'Completion': completion_counts})

# Plotting
plt.figure(figsize=(12, 6))
plt.plot(plot_data.index.astype(str), plot_data['Enrollment'], marker='o', label='Enrollment')
plt.plot(plot_data.index.astype(str), plot_data['Completion'], marker='o', label='Completion')

# Annotate each marker with the number of learners
for x, y_enroll, y_complete in zip(plot_data.index.astype(str), plot_data['Enrollment'], plot_data['Completion']):
    plt.annotate(f'{y_enroll}', (x, y_enroll), textcoords="offset points", xytext=(0,10), ha='center')
    plt.annotate(f'{y_complete}', (x, y_complete), textcoords="offset points", xytext=(0,-15), ha='center')

plt.xlabel('Year-Month')
plt.ylabel('Number of Learners')
plt.title('Enrollment and Completion Trends Over Time')
plt.legend()
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



- This basically shows the trebnds in the enrollemnt and the completion time of the it in the month of march

In [142]:
```
df['Enrollment Time']
```

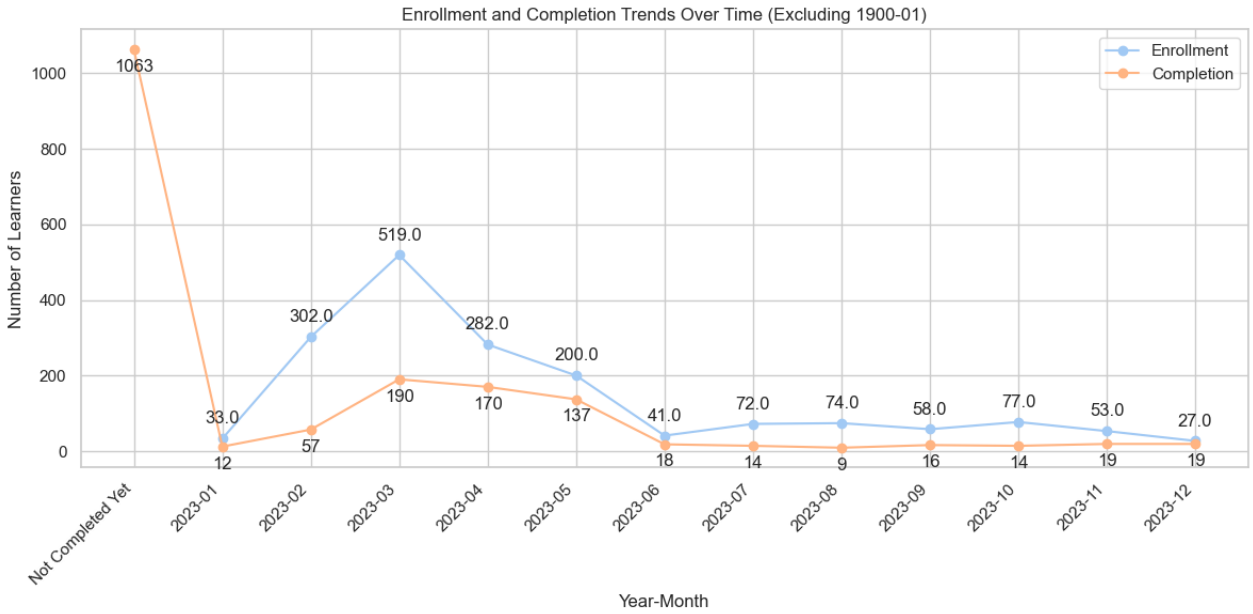
Out[142]:
```
0      1900-01-01 00:00:00
1      1900-01-01 00:00:00
2      1900-01-01 00:00:00
3      1900-01-01 00:00:00
4      1900-01-01 00:00:00
               ...
6630   2023-02-15 11:48:06
6631   2023-02-04 14:10:21
6632   1900-01-01 00:00:00
6633   2023-12-02 12:04:42
6634   1900-01-01 00:00:00
Name: Enrollment Time, Length: 6635, dtype: datetime64[ns]
```

In [143]:
```
df['Completion Time']
```

Out[143]:
```
0      1900-01-01 00:00:00
1      1900-01-01 00:00:00
2      1900-01-01 00:00:00
3      1900-01-01 00:00:00
4      1900-01-01 00:00:00
               ...
6630   2023-03-17 06:11:14
6631   1900-01-01 00:00:00
6632   1900-01-01 00:00:00
6633   2023-02-04 14:01:56
6634   1900-01-01 00:00:00
Name: Completion Time, Length: 6635, dtype: datetime64[ns]
```

In [147]:
```
df['Enrollment Time'] = pd.to_datetime(df['Enrollment Time'], errors='coerce')
df['Completion Time'] = pd.to_datetime(df['Completion Time'], errors='coerce')

# Filter out records with both enrollment and completion times as '1900-01-01 00:00:00'
df_filtered = df[~((df['Enrollment Time'] == '1900-01-01 00:00:00') & (df['Completion Time'] == '1900-01-01 00:00:00')

# Group by enrollment and completion time and count the number of records
enrollment_counts = df_filtered.groupby('Enrollment YearMonth').size()
completion_counts = df_filtered.groupby('Completion YearMonth').size()

# Create a DataFrame for plotting
plot_data = pd.DataFrame({'Enrollment': enrollment_counts, 'Completion': completion_counts})

# Plotting
plt.figure(figsize=(12, 6))
plt.plot(plot_data.index.astype(str), plot_data['Enrollment'], marker='o', label='Enrollment')
plt.plot(plot_data.index.astype(str), plot_data['Completion'], marker='o', label='Completion')

# Annotate each marker with the number of learners
for x, y_enroll, y_complete in zip(plot_data.index.astype(str), plot_data['Enrollment'], plot_data['Completion']):
    plt.annotate(f'{y_enroll}', (x, y_enroll), textcoords="offset points", xytext=(0,10), ha='center')
    plt.annotate(f'{y_complete}', (x, y_complete), textcoords="offset points", xytext=(0,-15), ha='center')

plt.xlabel('Year-Month')
plt.ylabel('Number of Learners')
plt.title('Enrollment and Completion Trends Over Time (Excluding 1900-01)')
plt.legend()
plt.xticks(rotation=45, ha='right')

# Change the label on the x-axis tick corresponding to '1900-01'
labels = [item.get_text() if item.get_text() != '1900-01' else 'Not Completed Yet' for item in plt.gca().get_xticklabe
plt.gca().set_xticklabels(labels)

plt.tight_layout()
plt.show()
```

```
C:\Users\Siddharth\AppData\Local\Temp\ipykernel_14528\1735051928.py:32: UserWarning: FixedFormatter should only be us
ed together with FixedLocator
  plt.gca().set_xticklabels(labels)
```



In [148]:
```
df_not_completed_status = df[(df['Completed'] == 'No') & (df['Course Grade'] != 0)].copy()
correlation_not_completed = df_not_completed_status['Learning Hours Spent'].corr(df_not_completed_status['Course Grade
print(f"Correlation for Not Completed Courses: {correlation_not_completed}")
```

```
Correlation for Not Completed Courses: 0.3816889520760214
```

In [149]:
```
df_completed_status = df[(df['Completed'] == 'Yes') & (df['Course Grade'] != 0)].copy()
correlation_completed = df_completed_status['Learning Hours Spent'].corr(df_completed_status['Course Grade'])
print(f"Correlation for Completed Courses: {correlation_completed}")
```

```
Correlation for Completed Courses: 0.006028130526509626
```

In [150]:
```
df_all_courses = df[(df['Completed']!='Not Started Yet') & (df['Course Grade'] != 0)].copy()
correlation_all_courses = df_all_courses['Learning Hours Spent'].corr(df_all_courses['Course Grade'])
print(f"Correlation for All Courses: {correlation_all_courses}")
```

```
Correlation for All Courses: 0.27332228984922036
```

In [151]:
```
df
```

Out[151]:

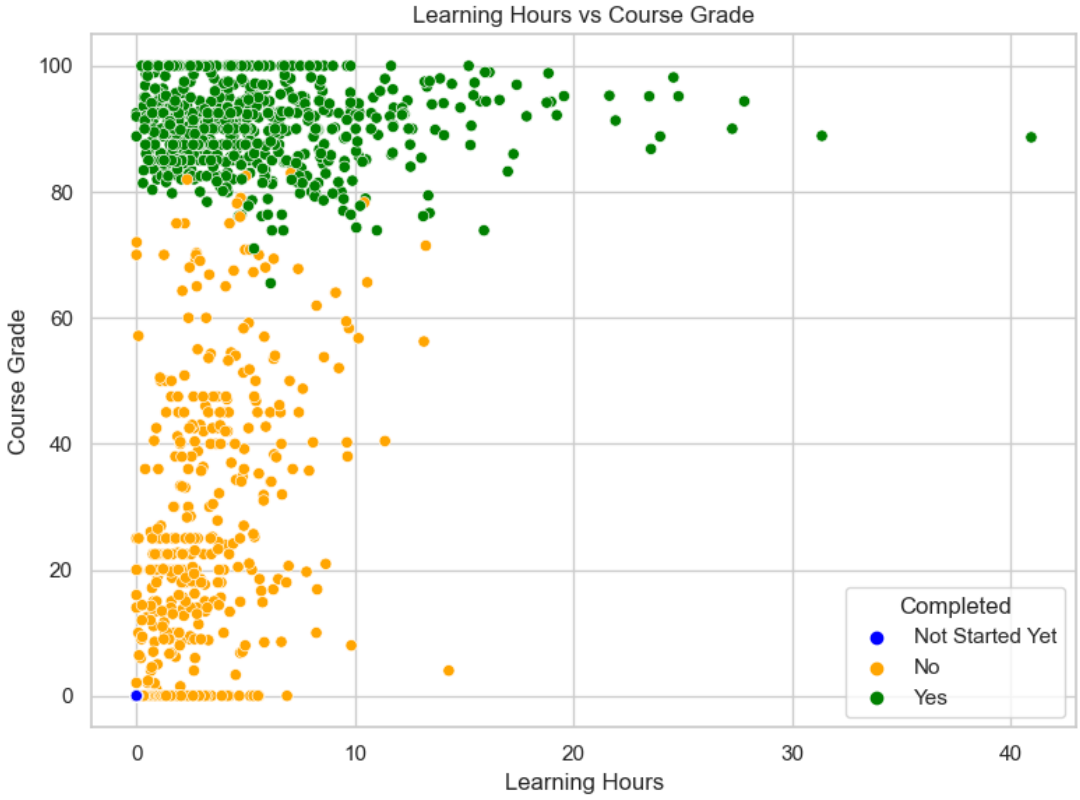
	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Cou Gra
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
...	...	...	...	...	...	...	...	...	...	...	...	
6630	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-02-15 11:48:06	2023-03-17 06:11:14	7.31	90
6631	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	2023-02-04 14:10:21	1900-01-01 00:00:00	2.09	33
6632	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
6633	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-12-02 12:04:42	2023-02-04 14:01:56	6.34	90
6634	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0

6635 rows × 15 columns

- Correlation for Not Completed Courses: 0.3817
- Interpretation: There is a moderate positive correlation between learning hours and course grades for learners who have not completed their courses. As learning hours increase, there is a tendency for grades to improve.
- Correlation for Completed Courses: 0.0060
- Interpretation: There is a very weak positive correlation between learning hours and course grades for learners who have completed their courses. The relationship between learning hours and grades for completed courses is nearly negligible.
- Correlation for All Courses: 0.2733
- Interpretation: When considering all courses, including both completed and not completed, there is a moderate positive correlation between learning hours and course grades. The overall relationship suggests that, on average, more learning hours are associated with higher grades.

- Comparison:
- The highest correlation is observed for not completed courses, indicating a relatively stronger relationship between learning hours and grades for learners still in progress.
- The correlation for completed courses is very low, suggesting that the time spent learning may not be a significant predictor of grades for those who have already completed the course.

In [156]:
```
plt.figure(figsize=(8, 6))

# Scatter plot for learning hours and course grade
sns.scatterplot(x='Learning Hours Spent', y='Course Grade', hue='Completed', data=df, palette=['blue', 'orange', 'gree
plt.title('Learning Hours vs Course Grade')
plt.xlabel('Learning Hours')
plt.ylabel('Course Grade')

plt.tight_layout()
plt.show()
```



In [157]:
```
#The criteria for being a top performer in the dataset
score_threshold = 85
courses_completed_threshold = 3

# Filtering the DataFrame on the basis of course grade here
high_performers_df = df[(df['Course Grade'] > score_threshold) & (df['Completed'] == 'Yes')]

# Filtering the DataFrame on the basis of no of skills in that course here
high_performers_df = high_performers_df.groupby('Name').filter(lambda x: len(x) >= courses_completed_threshold)
high_performers_df
```

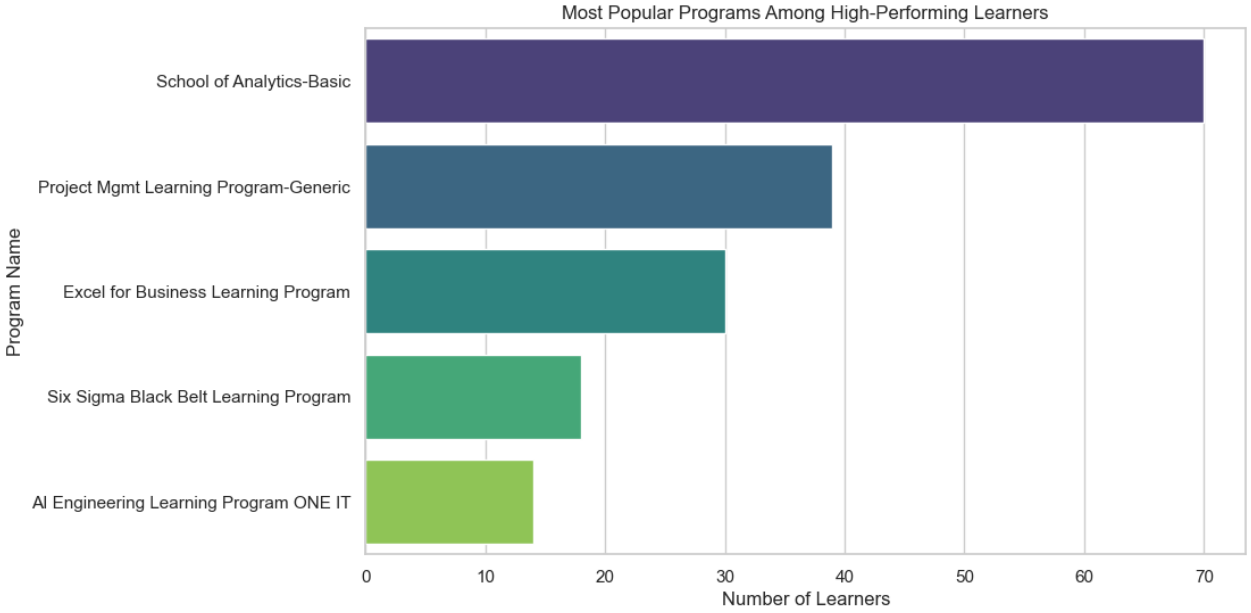
Out[157]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent
107	Person_25	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Yes	2023-07-02 10:31:36	2023-02-28 12:03:05	7.37
108	Person_25	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-07-02 10:27:38	2023-02-28 06:39:02	7.70
111	Person_25	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Project Mgmt Learning Program-Generic	Initiating and Planning Projects	4.8	Yes	2023-07-02 10:34:42	2023-07-05 09:10:08	3.68
183	Person_42	Shared Services	Electrical Maintenance	Iron Making Electrical Maintenance	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Yes	2023-09-04 07:41:14	2023-12-04 03:30:03	2.78
184	Person_42	Shared Services	Electrical Maintenance	Iron Making Electrical Maintenance	IT Security Learning Program	IT Security: Defense against the digital dark ...	13.4	Yes	2023-03-17 06:21:05	2023-09-04 07:35:21	13.29
...	...	...	...	...	...	...	...	...	...	...	...
6342	Person_1347	Steel Manufacturing	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Supply Market Analysis	2.7	Yes	2023-05-13 07:15:40	2023-05-14 14:22:09	0.52
6343	Person_1347	Steel Manufacturing	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Procurement & Sourcing Introduction	1.7	Yes	2023-09-05 10:16:48	2023-09-05 13:11:15	1.01
6344	Person_1347	Steel Manufacturing	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Strategic Sourcing	3.0	Yes	2023-10-05 04:18:59	2023-10-05 13:45:01	0.41
6346	Person_1347	Steel Manufacturing	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Procurement Basics	3.9	Yes	2023-09-05 13:28:29	2023-10-05 09:47:24	1.49
6347	Person_1347	Steel Manufacturing	Manufacturing Flat Products	CRC - West	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-03-13 14:17:21	2023-03-13 16:03:40	1.72

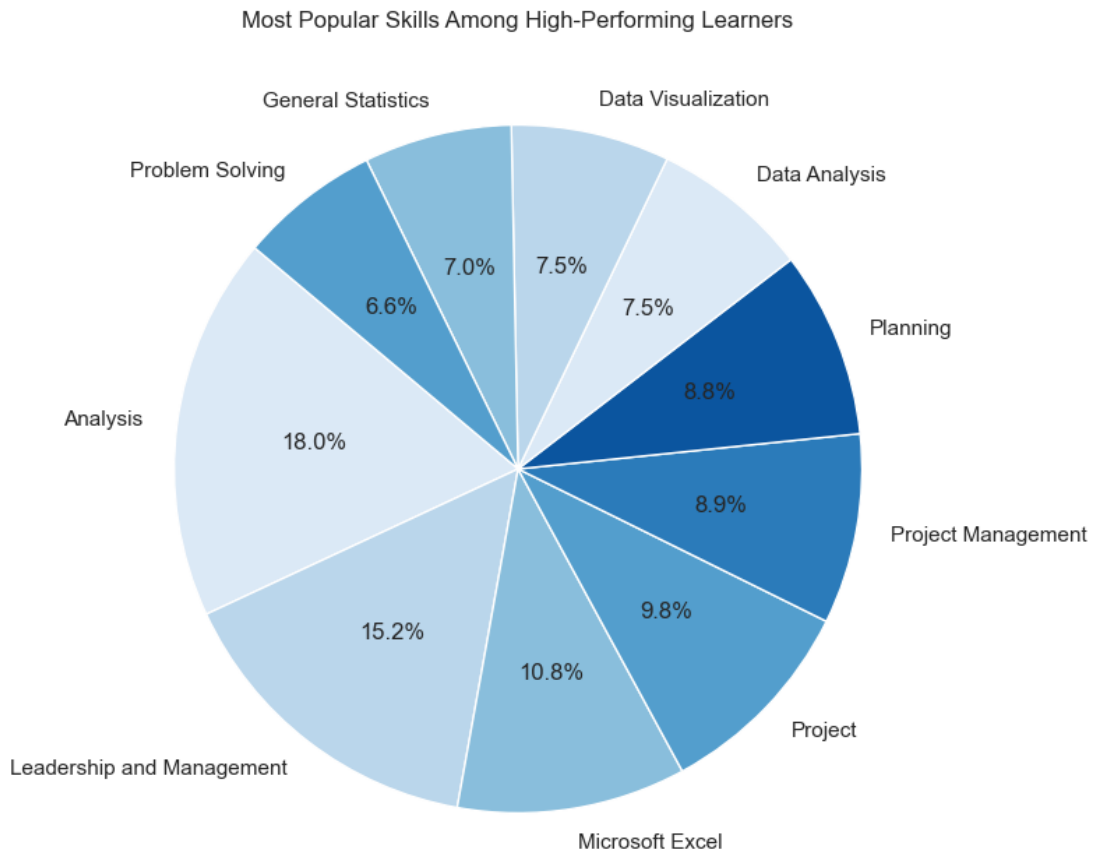
265 rows × 15 columns

- So there are 256 people who are in this top performers list out of the the 6635 peoples records and "high_performers_df" will contain the following record

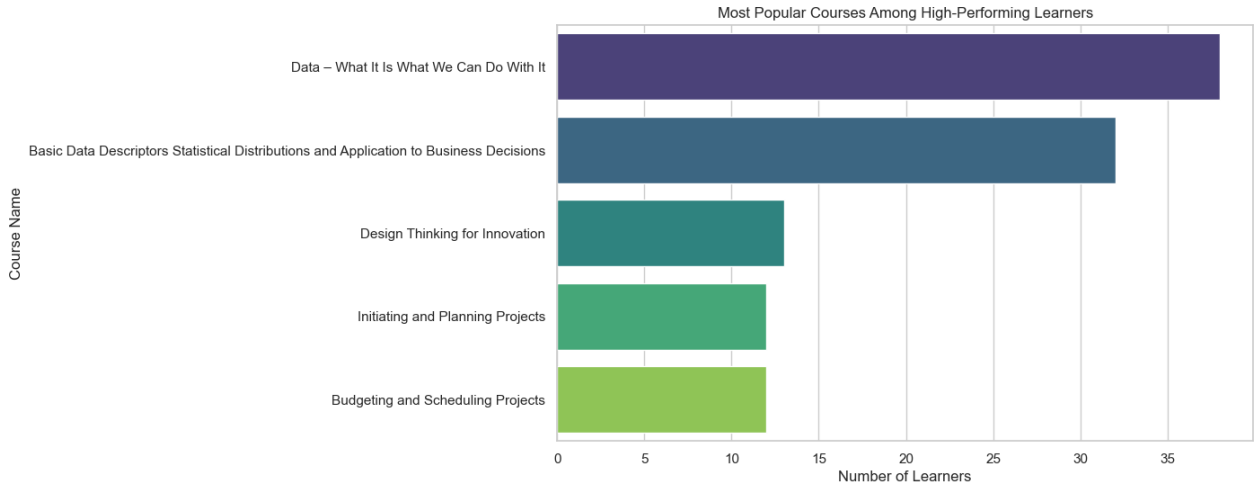
In [158]:
```
sns.set(style="whitegrid")

# Plot the most popular programs
plt.figure(figsize=(10, 6))
top_programs = high_performers_df['Program Name'].value_counts().nlargest(5)
sns.barplot(x=top_programs, y=top_programs.index, palette="viridis")
plt.title('Most Popular Programs Among High-Performing Learners')
plt.xlabel('Number of Learners')
plt.ylabel('Program Name')
plt.show()
```



In [159]:
```
plt.figure(figsize=(10, 8))
top_skills = high_performers_df['Skills Learned'].str.split('; ').explode().value_counts().nlargest(10)
plt.pie(top_skills, labels=top_skills.index, autopct='%1.1f%%', startangle=140, colors=sns.color_palette("Blues"))
plt.title('Most Popular Skills Among High-Performing Learners')
plt.show()
```



In [160]:
```
plt.figure(figsize=(10, 6))
top_courses = high_performers_df['Course Name'].value_counts().nlargest(5)
sns.barplot(x=top_courses, y=top_courses.index, palette="viridis")
plt.title('Most Popular Courses Among High-Performing Learners')
plt.xlabel('Number of Learners')
plt.ylabel('Course Name')
plt.show()
```



In [161]:
```
average_learning_hours_high_performers = high_performers_df['Learning Hours Spent'].mean()
print(f'The average learning hours for high-performing learners is: {average_learning_hours_high_performers:.2f} hours
```

```
The average learning hours for high-performing learners is: 5.31 hours
```

In [162]:
```
completed_learners_df = df[df['Completed'] == 'Yes']
average_learning_hours_completed = completed_learners_df['Learning Hours Spent'].mean()
print(f'The average learning hours for all learners who have completed the course is: {average_learning_hours_complete
```

```
The average learning hours for all learners who have completed the course is: 5.48 hours
```

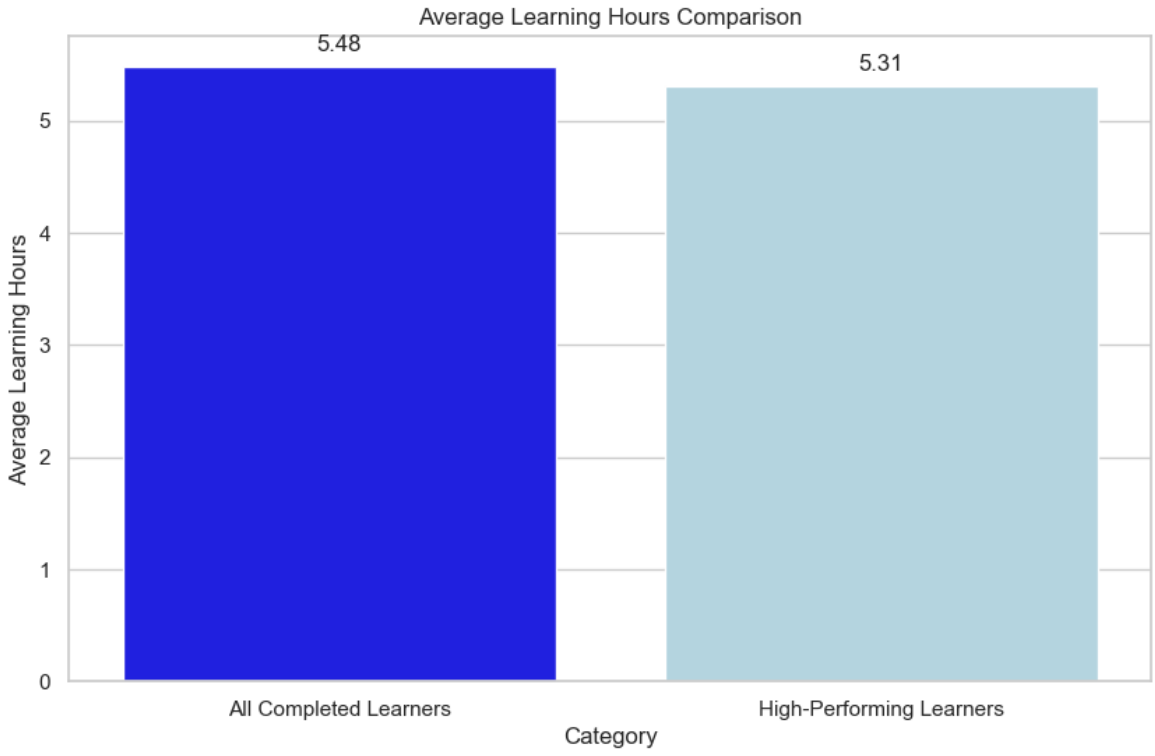
In [173]:
```
df=df_test
df
```

Out[173]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Cou Gra
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
...	...	...	...	...	...	...	...	...	...	...	...	
6630	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-02-15 11:48:06	2023-03-17 06:11:14	7.31	90
6631	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	2023-02-04 14:10:21	1900-01-01 00:00:00	2.09	33
6632	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
6633	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-12-02 12:04:42	2023-02-04 14:01:56	6.34	90
6634	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0

6635 rows × 15 columns

In [175]:
```
data = {'Category': ['All Completed Learners', 'High-Performing Learners'],
        'Average Learning Hours': [average_learning_hours_completed, average_learning_hours_high_performers]}

df_2 = pd.DataFrame(data)

# Create a grouped bar plot using Seaborn
plt.figure(figsize=(10, 6))
sns.barplot(x='Category', y='Average Learning Hours', data=df_2, palette=['blue', 'lightblue'])
plt.ylabel('Average Learning Hours')
plt.title('Average Learning Hours Comparison')

# Adding annotations to the top of the bars
for i, value in enumerate(df_2['Average Learning Hours']):
    plt.text(i, value + 0.1, f'{value:.2f}', ha='center', va='bottom')

plt.show()
```



In [176]:
```
program_skills_dict = {}

for index, row in df.iterrows():
    program_name = row['Program Name']
    skills = row['Skills Learned'].split('; ')

    if program_name not in program_skills_dict:
        program_skills_dict[program_name] = {'total_courses': 0, 'skills': {}}

    program_skills_dict[program_name]['total_courses'] += 1

    for skill in skills:
        if skill in program_skills_dict[program_name]['skills']:
            program_skills_dict[program_name]['skills'][skill] += 1
        else:
            program_skills_dict[program_name]['skills'][skill] = 1

# Creating a list of dictionaries to store program name, the most popular skill, and the percentage
data = []

for program, program_data in program_skills_dict.items():
    most_popular_skill = max(program_data['skills'], key=program_data['skills'].get)
    percentage_courses_with_skill = (program_data['skills'][most_popular_skill] / program_data['total_courses']) * 100
    data.append({
        'Program Name': program,
        'Most Popular Skill': most_popular_skill,
        'Popularity Percentage ': percentage_courses_with_skill
    })

# Creating a dataframe from the list of dictionaries
program_most_popular_skill = pd.DataFrame(data)
program_most_popular_skill
```

Out[176]:

	Program Name	Most Popular Skill	Popularity Percentage
0	Six Sigma Black Belt Learning Program	Six Sigma	62.500000
1	School of Analytics-Basic	Analysis	100.000000
2	Supply Chain Planning Learning Program	Supply Chain	100.000000
3	School of Visualization ONE IT	Tableau Software	100.000000
4	Teamwork Skills Learning Program	Team Building	100.000000
...	...	...	...
129	Google Data Analytics Learning Program	Tableau Software	100.000000
130	Communication Roles Learning Program	Marketing	66.666667
131	Solve Business Problems- AI & ML ONE IT	Machine Learning	100.000000
132	Leading Positive Change Learning Program	Change Management	100.000000
133	Business Strategy Learning Program	Business Strategy	100.000000

134 rows × 3 columns

In [188]:
```
df
```

Out[188]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Cou Gra
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
...	...	...	...	...	...	...	...	...	...	...	...	
6630	Person_1408	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-02-15 11:48:06	2023-03-17 06:11:14	7.31	90
6631	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Excellence	6.8	No	2023-02-04 14:10:21	1900-01-01 00:00:00	2.09	33
6632	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	Operations Management Learning Program	Supply Chain Operations	7.6	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0
6633	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-12-02 12:04:42	2023-02-04 14:01:56	6.34	90
6634	Person_1409	TQM and E&P	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01 00:00:00	0.00	0

6635 rows × 15 columns

In [191]:
```
df.head(12)
```

Out[191]:

	Name	Division	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Course Grade	L
0	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Define Phase for the 6 σ Black Belt	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	Trigor Integr Six
1	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Analyze Phase for the 6 σ Black Belt	7.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	Six T A L
2	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Measure Phase for the 6 σ Black Belt	10.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	A F Measu G
3	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	DFSS for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	Trigor Inte Leade
4	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Team Management for the 6 σ Black Belt	2.5	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	Mana R
5	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	Organization Planning and Development for the ...	6.2	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	P Trigor Leade
6	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Control Phase for the 6 σ Black Belt	6.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	Trigor Inte
7	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	No	2023-03-24 05:29:17	1900-01-01	0.01	0.0	Visua Data A Anal
8	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	S S A
9	Person_1	Operations - TSK	Steel & Mills - TSK	Steel Melting Shop & LCP TSK	Six Sigma Black Belt Learning Program	The Improve Phase for the 6 σ Black Belt	5.0	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	F
10	Person_2	VP One SC	Integrated Planning & Service	Master Production Planning & Margin Mgmt	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Not Started Yet	1900-01-01 00:00:00	1900-01-01	0.00	0.0	Visua Data A Anal
11	Person_2	VP One SC	Integrated Planning & Service	Master Production Planning & Margin Mgmt	Supply Chain Planning Learning Program	Supply Chain Planning	8.4	No	2023-04-24 07:02:35	1900-01-01	0.09	0.0	Supply D Fore Chair

In [192]:
```
df['Completion Status'] = df['Completed'].apply(lambda x: 1 if x == 'Yes' else 0)

# Group by person and aggregate information
person_data = df.groupby('Name').agg({
    'Course Name': 'count',            # Number of courses enrolled
    'Completion Status': 'sum'         # Number of courses completed
}).reset_index()

# Calculate completion percentage
person_data['Completion Percentage'] = (person_data['Completion Status'] / person_data['Course Name']) * 100

# Rename columns for clarity
person_data.columns = ['Person', 'Enrolled Courses', 'Completed Courses', 'Completion Percentage']

person_data
```

Out[192]:

	Person	Enrolled Courses	Completed Courses	Completion Percentage
0	Person_1	10	0	0.000000
1	Person_10	6	0	0.000000
2	Person_100	6	0	0.000000
3	Person_1000	6	0	0.000000
4	Person_1001	5	0	0.000000
...	...	...	...	...
1404	Person_995	6	1	16.666667
1405	Person_996	3	3	100.000000
1406	Person_997	4	0	0.000000
1407	Person_998	4	0	0.000000
1408	Person_999	3	0	0.000000

1409 rows × 4 columns

In [193]:
```
completed_persons_new_dataframe = person_data[person_data['Completed Courses'] > 0]
completed_persons_new_dataframe
```

Out[193]:

	Person	Enrolled Courses	Completed Courses	Completion Percentage
5	Person_1002	5	2	40.000000
10	Person_1007	3	1	33.333333
22	Person_1018	2	2	100.000000
29	Person_1024	4	4	100.000000
32	Person_1027	10	4	40.000000
...	...	...	...	...
1388	Person_980	7	1	14.285714
1392	Person_984	6	1	16.666667
1395	Person_987	4	1	25.000000
1404	Person_995	6	1	16.666667
1405	Person_996	3	3	100.000000

294 rows × 4 columns

In [195]:
```
high_performers_df
```

Out[195]:

	Group	Department	Program Name	Course Name	Duration(in hrs)	Completed	Enrollment Time	Completion Time	Learning Hours Spent	Course Grade	Skills Learned	Enrollment YearMonth
	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Yes	2023-07-02 10:31:36	2023-02-28 12:03:05	7.37	94.95	General Statistics; Statistical Analysis; Prob...	2023-07
	Projects & Construction TSK	Cold Rolling Mills - POG	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-07-02 10:27:38	2023-02-28 06:39:02	7.70	95.00	Data Visualization; Data Analysis; Analysis; P...	2023-07
	Projects & Construction TSK	Cold Rolling Mills - POG	Project Mgmt Learning Program-Generic	Initiating and Planning Projects	4.8	Yes	2023-07-02 10:34:42	2023-07-05 09:10:08	3.68	100.00	Project; Planning; Project Management; Project...	2023-07
	Electrical Maintenance	Iron Making Electrical Maintenance	School of Analytics-Basic	Basic Data Descriptors Statistical Distributio...	8.9	Yes	2023-09-04 07:41:14	2023-12-04 03:30:03	2.78	89.32	General Statistics; Statistical Analysis; Prob...	2023-09
	Electrical Maintenance	Iron Making Electrical Maintenance	IT Security Learning Program	IT Security: Defense against the digital dark ...	13.4	Yes	2023-03-17 06:21:05	2023-09-04 07:35:21	13.29	96.70	Wireless; Network Security; System Security; C...	2023-03
	...	...	...	...	...	...	...	...	...	...	...	...
	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Supply Market Analysis	2.7	Yes	2023-05-13 07:15:40	2023-05-14 14:22:09	0.52	100.00	Supply Chain; Risk; Market Analysis; Market (E...	2023-05
	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Procurement & Sourcing Introduction	1.7	Yes	2023-09-05 10:16:48	2023-09-05 13:11:15	1.01	100.00	Supply Chain; Sources; Supplier Relationship M...	2023-09
	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Strategic Sourcing	3.0	Yes	2023-10-05 04:18:59	2023-10-05 13:45:01	0.41	87.50	Sources; Leadership and Management	2023-10
	Manufacturing Flat Products	CRC - West	Strategic Procurement Learning Program	Procurement Basics	3.9	Yes	2023-09-05 13:28:29	2023-10-05 09:47:24	1.49	100.00	Supply Chain; Procurement; Leadership and Mana...	2023-09
	Manufacturing Flat Products	CRC - West	School of Analytics-Basic	Data – What It Is What We Can Do With It	5.8	Yes	2023-03-13 14:17:21	2023-03-13 16:03:40	1.72	90.00	Data Visualization; Data Analysis; Analysis; P...	2023-03

In [196]:
```
unique_persons_count = high_performers_df['Name'].nunique()
print("Number of unique persons:", unique_persons_count)
```

```
Number of unique persons: 67
```

In [197]:
```
person_courses_dict = {}

# Populate the dictionary
for _, row in high_performers_df.iterrows():
    person = row['Name']
    course = row['Course Name']

    if person in person_courses_dict:
        person_courses_dict[person].append(course)
    else:
        person_courses_dict[person] = [course]
person_courses_dict
```

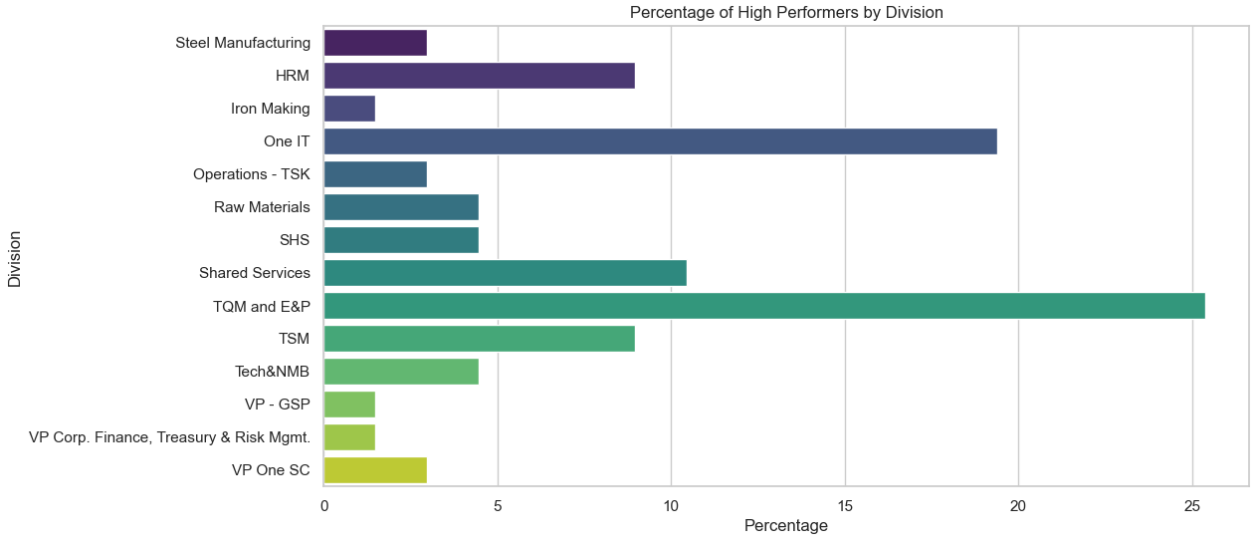
```
 'Design Thinking for Innovation'],
 'Person_1310': ['Excel Skills for Business: Essentials',
  'Excel Skills for Business: Intermediate II',
  'Excel Skills for Business: Advanced',
  'Excel Skills for Business: Intermediate I',
  'Basic Data Descriptors Statistical Distributions and Application to Business Decisions',
  'Data – What It Is What We Can Do With It'],
 'Person_1346': ['Machine Learning with Python',
  'Deep Neural Networks with PyTorch',
  'Introduction to Deep Learning & Neural Networks with Keras',
  'Design Thinking for Innovation'],
 'Person_1347': ['Strategic Procurement and Sourcing Conclusions',
  'Procurement Negotiation',
  'Supplier Management',
  'Supply Market Analysis',
  'Procurement & Sourcing Introduction',
  'Strategic Sourcing',
  'Procurement Basics',
  'Data – What It Is What We Can Do With It']}
```

In [218]:
```
division_unique_persons = high_performers_df.groupby('Division')['Name'].nunique().reset_index()

# Rename the column for clarity
division_unique_persons.columns = ['Division', 'Number of Unique Persons']

# Calculate the percentage of high performers in each division
division_unique_persons['Percentage of High Performers'] = (division_unique_persons['Number of Unique Persons'] / 67)
division_unique_persons
```

Out[218]:

	Division	Number of Unique Persons	Percentage of High Performers
0	Steel Manufacturing	2	2.985075
1	HRM	6	8.955224
2	Iron Making	1	1.492537
3	One IT	13	19.402985
4	Operations - TSK	2	2.985075
5	Raw Materials	3	4.477612
6	SHS	3	4.477612
7	Shared Services	7	10.447761
8	TQM and E&P	17	25.373134
9	TSM	6	8.955224
10	Tech&NMB	3	4.477612
11	VP - GSP	1	1.492537
12	VP Corp. Finance, Treasury & Risk Mgmt.	1	1.492537
13	VP One SC	2	2.985075

In [219]:
```
plt.figure(figsize=(12, 6))
sns.barplot(x='Percentage of High Performers', y='Division', data=division_unique_persons, palette='viridis')
plt.title('Percentage of High Performers by Division')
plt.xlabel('Percentage')
plt.ylabel('Division')
plt.show()
```

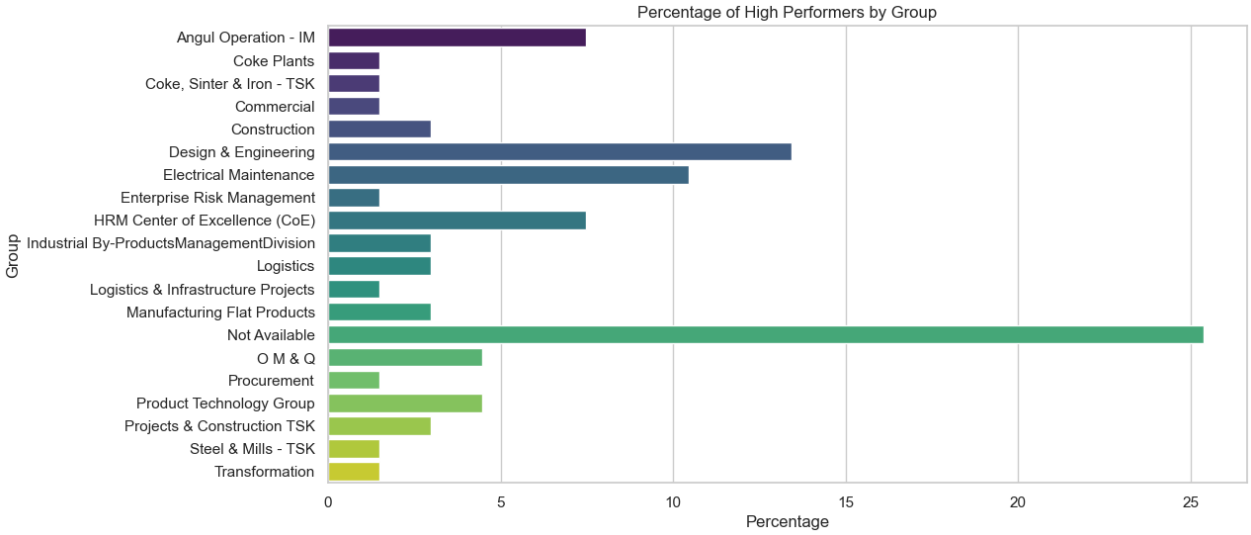


In [220]:
```
group_unique_persons = high_performers_df.groupby('Group')['Name'].nunique().reset_index()
group_unique_persons.columns = ['Group', 'Number of Unique Persons']
group_unique_persons['Percentage of High Performers'] = (group_unique_persons['Number of Unique Persons'] / 67) * 100
group_unique_persons
```

Out[220]:

	Group	Number of Unique Persons	Percentage of High Performers
0	Angul Operation - IM	5	7.462687
1	Coke Plants	1	1.492537
2	Coke, Sinter & Iron - TSK	1	1.492537
3	Commercial	1	1.492537
4	Construction	2	2.985075
5	Design & Engineering	9	13.432836
6	Electrical Maintenance	7	10.447761
7	Enterprise Risk Management	1	1.492537
8	HRM Center of Excellence (CoE)	5	7.462687
9	Industrial By-ProductsManagementDivision	2	2.985075
10	Logistics	2	2.985075
11	Logistics & Infrastructure Projects	1	1.492537
12	Manufacturing Flat Products	2	2.985075
13	Not Available	17	25.373134
14	O M & Q	3	4.477612
15	Procurement	1	1.492537
16	Product Technology Group	3	4.477612
17	Projects & Construction TSK	2	2.985075
18	Steel & Mills - TSK	1	1.492537
19	Transformation	1	1.492537

In [221]:
```
plt.figure(figsize=(12, 6))
sns.barplot(x='Percentage of High Performers', y='Group', data=group_unique_persons, palette='viridis')
plt.title('Percentage of High Performers by Group')
plt.xlabel('Percentage')
plt.ylabel('Group')
plt.show()
```

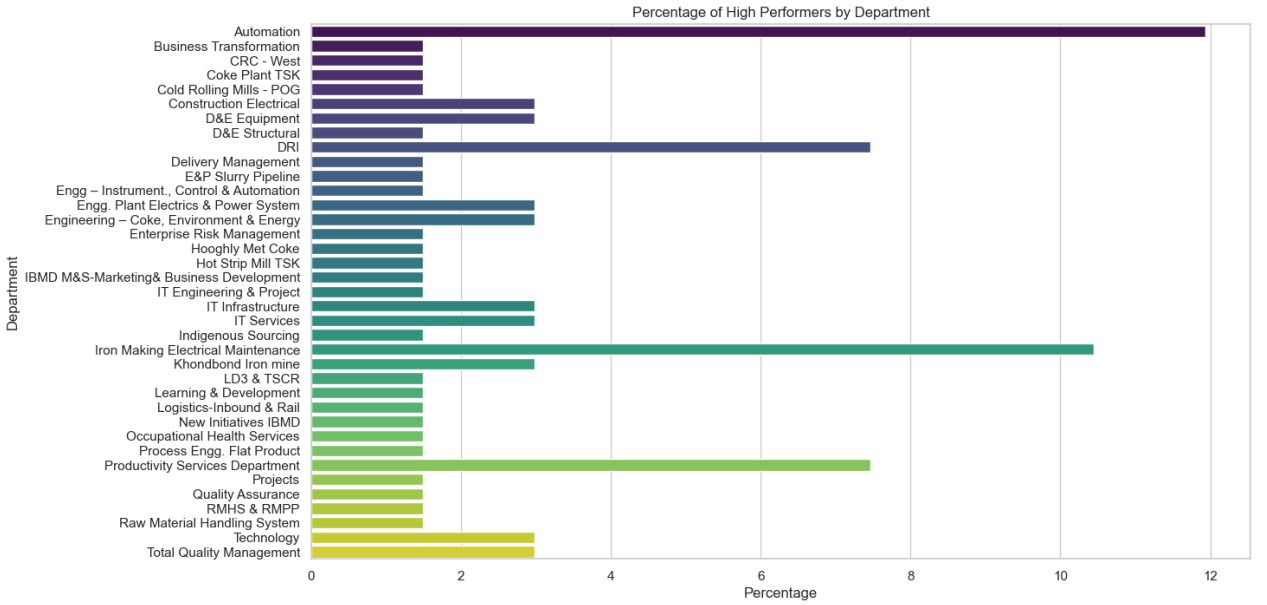


In [222]:
```
length = 67  # Total number of unique high performers
department_unique_persons = high_performers_df.groupby('Department')['Name'].nunique().reset_index()
department_unique_persons.columns = ['Department', 'Number of Unique Persons']
department_unique_persons['Percentage of High Performers'] = (department_unique_persons['Number of Unique Persons'] /
department_unique_persons
```

Out[222]:

	Department	Number of Unique Persons	Percentage of High Performers
0	Automation	8	11.940299
1	Business Transformation	1	1.492537
2	CRC - West	1	1.492537
3	Coke Plant TSK	1	1.492537
4	Cold Rolling Mills - POG	1	1.492537
5	Construction Electrical	2	2.985075
6	D&E Equipment	2	2.985075
7	D&E Structural	1	1.492537
8	DRI	5	7.462687
9	Delivery Management	1	1.492537
10	E&P Slurry Pipeline	1	1.492537
11	Engg – Instrument., Control & Automation	1	1.492537
12	Engg. Plant Electrics & Power System	2	2.985075
13	Engineering – Coke, Environment & Energy	2	2.985075
14	Enterprise Risk Management	1	1.492537
15	Hooghly Met Coke	1	1.492537
16	Hot Strip Mill TSK	1	1.492537
17	IBMD M&S-Marketing& Business Development	1	1.492537
18	IT Engineering & Project	1	1.492537
19	IT Infrastructure	2	2.985075
20	IT Services	2	2.985075
21	Indigenous Sourcing	1	1.492537
22	Iron Making Electrical Maintenance	7	10.447761
23	Khondbond Iron mine	2	2.985075
24	LD3 & TSCR	1	1.492537
25	Learning & Development	1	1.492537
26	Logistics-Inbound & Rail	1	1.492537
27	New Initiatives IBMD	1	1.492537
28	Occupational Health Services	1	1.492537
29	Process Engg. Flat Product	1	1.492537
30	Productivity Services Department	5	7.462687
31	Projects	1	1.492537
32	Quality Assurance	1	1.492537
33	RMHS & RMPP	1	1.492537
34	Raw Material Handling System	1	1.492537
35	Technology	2	2.985075
36	Total Quality Management	2	2.985075

In [223]:
```
plt.figure(figsize=(14, 8))
sns.barplot(x='Percentage of High Performers', y='Department', data=department_unique_persons, palette='viridis')
plt.title('Percentage of High Performers by Department')
plt.xlabel('Percentage')
plt.ylabel('Department')
plt.show()
```

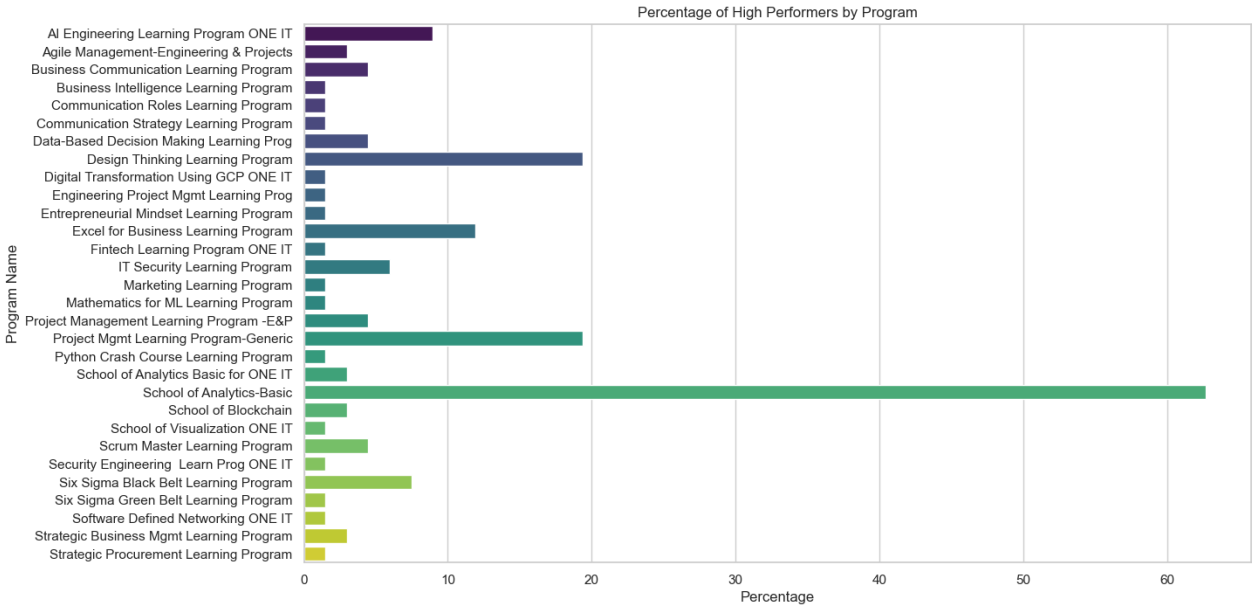


In [224]:
```
total_high_performers = 67

program_unique_persons = high_performers_df.groupby('Program Name')['Name'].nunique().reset_index()

program_unique_persons.columns = ['Program Name', 'Number of Unique Persons']

program_unique_persons['Percentage of High Performers'] = (program_unique_persons['Number of Unique Persons'] / total_

program_unique_persons
```

Out[224]:

	Program Name	Number of Unique Persons	Percentage of High Performers
0	AI Engineering Learning Program ONE IT	6	8.955224
1	Agile Management-Engineering & Projects	2	2.985075
2	Business Communication Learning Program	3	4.477612
3	Business Intelligence Learning Program	1	1.492537
4	Communication Roles Learning Program	1	1.492537
5	Communication Strategy Learning Program	1	1.492537
6	Data-Based Decision Making Learning Prog	3	4.477612
7	Design Thinking Learning Program	13	19.402985
8	Digital Transformation Using GCP ONE IT	1	1.492537
9	Engineering Project Mgmt Learning Prog	1	1.492537
10	Entrepreneurial Mindset Learning Program	1	1.492537
11	Excel for Business Learning Program	8	11.940299
12	Fintech Learning Program ONE IT	1	1.492537
13	IT Security Learning Program	4	5.970149
14	Marketing Learning Program	1	1.492537
15	Mathematics for ML Learning Program	1	1.492537
16	Project Management Learning Program -E&P	3	4.477612
17	Project Mgmt Learning Program-Generic	13	19.402985
18	Python Crash Course Learning Program	1	1.492537
19	School of Analytics Basic for ONE IT	2	2.985075
20	School of Analytics-Basic	42	62.686567
21	School of Blockchain	2	2.985075
22	School of Visualization ONE IT	1	1.492537
23	Scrum Master Learning Program	3	4.477612
24	Security Engineering Learn Prog ONE IT	1	1.492537
25	Six Sigma Black Belt Learning Program	5	7.462687
26	Six Sigma Green Belt Learning Program	1	1.492537
27	Software Defined Networking ONE IT	1	1.492537
28	Strategic Business Mgmt Learning Program	2	2.985075
29	Strategic Procurement Learning Program	1	1.492537

In [225]:
```
plt.figure(figsize=(14, 8))
sns.barplot(x='Percentage of High Performers', y='Program Name', data=program_unique_persons, palette='viridis')
plt.title('Percentage of High Performers by Program')
plt.xlabel('Percentage')
plt.ylabel('Program Name')
plt.show()
```



In [226]:
```
course_stats = high_performers_df.groupby('Course Name')['Name'].agg(['nunique', lambda x: len(x) / len(high_performer
course_stats.columns = ['Course Name', 'Number of Unique Persons', 'Percentage of High Performers']
course_stats
```

Out[226]:

	Course Name	Number of Unique Persons	Percentage of High Performers
0	AI Capstone Project with Deep Learning	1	0.377358
1	Applied Data Science Capstone	2	0.754717
2	Artificial Intelligence in Marketing	1	0.377358
3	Basic Data Descriptors Statistical Distributio...	32	12.075472
4	Budgeting and Scheduling Projects	12	4.528302
...	...	...	...
73	The Improve Phase for the 6 σ Black Belt	1	0.377358
74	The Measure Phase for the 6 σ Black Belt	1	0.377358
75	The Nuts and Bolts of Public Relations	1	0.377358
76	Transacting on the Blockchain	1	0.377358
77	Visual Analytics with Tableau	1	0.377358

78 rows × 3 columns

In [ ]: