# 01_eda

April 21, 2025

[1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import os
import sys
from pathlib import Path

# Add src directory to sys.path to import config
# This assumes the notebook is in RECSYS_FINAL/notebooks/
project_root = Path.cwd().parent # Should be RECSYS_FINAL
src_path = project_root / "src"
sys.path.append(str(src_path))

# Import config variables
import config

# Set some display options for pandas
pd.set_option('display.max_columns', 50)
pd.set_option('display.max_rows', 100)

# Plotting style
sns.set_style("whitegrid")
plt.rcParams['figure.figsize'] = (12, 6)

print(f"Project Root: {project_root}")
print(f"Configured Raw Data Path: {config.RAW_DATA_DIR}")
print(f"All Raw Files Found: {config.check_raw_data_exists()}")
```

```
Loading .env from: /Users/mohit/Desktop/everything/ATLAS/Semester
4/Pinnacle/recsys_final/.env
Database URI configured: Yes
Project Root: /Users/mohit/Desktop/everything/ATLAS/Semester
4/Pinnacle/recsys_final
Configured Raw Data Path: /Users/mohit/Desktop/everything/ATLAS/Semester
4/Pinnacle/recsys_final/data/raw
All raw data files found.
All Raw Files Found: True
```

[2]:

```
# Load all datasets using paths from config
try:
    assessments_df = pd.read_csv(config.ASSESSMENTS_CSV)
    courses_df = pd.read_csv(config.COURSES_CSV)
    student_assessment_df = pd.read_csv(config.STUDENT_ASSESSMENT_CSV)
    student_info_df = pd.read_csv(config.STUDENT_INFO_CSV)
    student_registration_df = pd.read_csv(config.STUDENT_REGISTRATION_CSV)
    student_vle_df = pd.read_csv(config.STUDENT_VLE_CSV)
    vle_df = pd.read_csv(config.VLE_CSV)
    print("All CSV files loaded successfully.")
except FileNotFoundError as e:
    print(f"Error loading files: {e}")
    print("Please ensure the CSV files are in the data/raw/ directory.")
    # Stop execution or handle appropriately

# Store dataframes in a dictionary for easier access
dataframes = {
    "assessments": assessments_df,
    "courses": courses_df,
    "student_assessment": student_assessment_df,
    "student_info": student_info_df,
    "student_registration": student_registration_df,
    "student_vle": student_vle_df,
    "vle": vle_df,
}
```

```
All CSV files loaded successfully.
```

[3]:

```
# Basic inspection of each dataframe
for name, df in dataframes.items():
    print(f"--- DataFrame: {name} ---")
    print(f"Shape: {df.shape}")
    print("Info:")
    df.info()
    print("\nHead:")
    print(df.head())
    print("\nMissing Values:")
    print(df.isnull().sum())
    # Only show describe() for dataframes with numerical columns
    numeric_cols = df.select_dtypes(include=np.number).columns
    if len(numeric_cols) > 0:
         print("\nDescribe (Numerical):")
         print(df.describe())
    # Describe categorical columns
    categorical_cols = df.select_dtypes(include=['object', 'category']).columns
    if len(categorical_cols) > 0:
        print("\nDescribe (Categorical):")
        print(df.describe(include=['object', 'category']))
```

```python
    print("-" * (len(name) + 22))
    print("\n")
```

```
--- DataFrame: assessments ---
Shape: (206, 6)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 206 entries, 0 to 205
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   code_module        206 non-null    object 
 1   code_presentation  206 non-null    object 
 2   id_assessment      206 non-null    int64  
 3   assessment_type    206 non-null    object 
 4   date               195 non-null    float64
 5   weight             206 non-null    float64
dtypes: float64(2), int64(1), object(3)
memory usage: 9.8+ KB

Head:
  code_module code_presentation  id_assessment assessment_type   date  weight
0         AAA             2013J           1752             TMA   19.0    10.0
1         AAA             2013J           1753             TMA   54.0    20.0
2         AAA             2013J           1754             TMA  117.0    20.0
3         AAA             2013J           1755             TMA  166.0    20.0
4         AAA             2013J           1756             TMA  215.0    30.0

Missing Values:
code_module           0
code_presentation     0
id_assessment         0
assessment_type       0
date                 11
weight                0
dtype: int64

Describe (Numerical):
       id_assessment        date      weight
count     206.000000  195.000000  206.000000
mean    26473.975728  145.005128   20.873786
std     10098.625521   76.001119   30.384224
min      1752.000000   12.000000    0.000000
25%     15023.250000   71.000000    0.000000
50%     25364.500000  152.000000   12.500000
75%     34891.750000  222.000000   24.250000
max     40088.000000  261.000000  100.000000
```

```
Describe (Categorical):
       code_module code_presentation assessment_type
count          206               206             206
unique           7                 4               3
top            FFF             2014J             TMA
freq            52                57             106
---------------------------------
```

```
--- DataFrame: courses ---
Shape: (22, 3)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22 entries, 0 to 21
Data columns (total 3 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   code_module                 22 non-null     object
 1   code_presentation           22 non-null     object
 2   module_presentation_length  22 non-null     int64
dtypes: int64(1), object(2)
memory usage: 660.0+ bytes

Head:
  code_module code_presentation  module_presentation_length
0         AAA             2013J                         268
1         AAA             2014J                         269
2         BBB             2013J                         268
3         BBB             2014J                         262
4         BBB             2013B                         240

Missing Values:
code_module                   0
code_presentation             0
module_presentation_length    0
dtype: int64

Describe (Numerical):
       module_presentation_length
count                   22.000000
mean                   255.545455
std                     13.654677
min                    234.000000
25%                    241.000000
50%                    261.500000
75%                    268.000000
max                    269.000000
```

```
Describe (Categorical):
       code_module code_presentation
count           22                22
unique           7                 4
top            BBB             2014J
freq             4                 7
-----------------------------



--- DataFrame: student_assessment ---
Shape: (173912, 5)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 173912 entries, 0 to 173911
Data columns (total 5 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   id_assessment   173912 non-null  int64
 1   id_student      173912 non-null  int64
 2   date_submitted  173912 non-null  int64
 3   is_banked       173912 non-null  int64
 4   score           173739 non-null  float64
dtypes: float64(1), int64(4)
memory usage: 6.6 MB

Head:
   id_assessment  id_student  date_submitted  is_banked  score
0           1752       11391              18          0   78.0
1           1752       28400              22          0   70.0
2           1752       31604              17          0   72.0
3           1752       32885              26          0   69.0
4           1752       38053              19          0   79.0

Missing Values:
id_assessment       0
id_student          0
date_submitted      0
is_banked           0
score             173
dtype: int64

Describe (Numerical):
       id_assessment    id_student  date_submitted      is_banked  \
count  173912.000000  1.739120e+05   173912.000000  173912.000000
mean    26553.803556  7.051507e+05      116.032942       0.010977
std      8829.784254  5.523952e+05       71.484148       0.104194
min      1752.000000  6.516000e+03      -11.000000       0.000000
25%     15022.000000  5.044290e+05       51.000000       0.000000
```

```
50%     25359.000000  5.852080e+05      116.000000       0.000000
75%     34883.000000  6.344980e+05      173.000000       0.000000
max     37443.000000  2.698588e+06      608.000000       1.000000

               score
count  173739.000000
mean       75.799573
std        18.798107
min         0.000000
25%        65.000000
50%        80.000000
75%        90.000000
max       100.000000
----------------------------------------


--- DataFrame: student_info ---
Shape: (32593, 12)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32593 entries, 0 to 32592
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype 
---  ------                --------------  ----- 
 0   code_module           32593 non-null  object
 1   code_presentation     32593 non-null  object
 2   id_student            32593 non-null  int64 
 3   gender                32593 non-null  object
 4   region                32593 non-null  object
 5   highest_education     32593 non-null  object
 6   imd_band              31482 non-null  object
 7   age_band              32593 non-null  object
 8   num_of_prev_attempts  32593 non-null  int64 
 9   studied_credits       32593 non-null  int64 
 10  disability            32593 non-null  object
 11  final_result          32593 non-null  object
dtypes: int64(3), object(9)
memory usage: 3.0+ MB

Head:
  code_module code_presentation  id_student gender                region  \
0         AAA             2013J       11391      M   East Anglian Region   
1         AAA             2013J       28400      F              Scotland   
2         AAA             2013J       30268      F  North Western Region   
3         AAA             2013J       31604      F     South East Region   
4         AAA             2013J       32885      F  West Midlands Region   

       highest_education imd_band age_band  num_of_prev_attempts  \
0       HE Qualification  90-100%     55<=                    0
1       HE Qualification   20-30%    35-55                    0
2  A Level or Equivalent   30-40%    35-55                    0
3  A Level or Equivalent   50-60%    35-55                    0
4     Lower Than A Level   50-60%     0-35                    0

   studied_credits disability final_result
0              240          N         Pass
1               60          N         Pass
2               60          Y    Withdrawn
3               60          N         Pass
4               60          N         Pass

Missing Values:
code_module                0
code_presentation          0
id_student                 0
gender                     0
region                     0
highest_education          0
imd_band                1111
age_band                   0
num_of_prev_attempts       0
studied_credits            0
disability                 0
final_result               0
dtype: int64

Describe (Numerical):
         id_student  num_of_prev_attempts  studied_credits
count  3.259300e+04          32593.000000     32593.000000
mean   7.066877e+05              0.163225        79.758691
std    5.491673e+05              0.479758        41.071900
min    3.733000e+03              0.000000        30.000000
25%    5.085730e+05              0.000000        60.000000
50%    5.903100e+05              0.000000        60.000000
75%    6.444530e+05              0.000000       120.000000
max    2.716795e+06              6.000000       655.000000

Describe (Categorical):
       code_module code_presentation gender    region      highest_education  \
count        32593             32593  32593     32593                  32593
unique           7                 4      2        13                      5
top            BBB             2014J      M  Scotland  A Level or Equivalent
freq          7909             11260  17875      3446                  14045

       imd_band age_band disability final_result
count     31482    32593      32593        32593
```

```
unique       10       3          2           4
top      20-30%    0-35          N        Pass
freq       3654   22944      29429       12361
----------------------------------


--- DataFrame: student_registration ---
Shape: (32593, 5)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32593 entries, 0 to 32592
Data columns (total 5 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   code_module          32593 non-null  object
 1   code_presentation    32593 non-null  object
 2   id_student           32593 non-null  int64
 3   date_registration    32548 non-null  float64
 4   date_unregistration  10072 non-null  float64
dtypes: float64(2), int64(1), object(2)
memory usage: 1.2+ MB

Head:
  code_module code_presentation  id_student  date_registration  \
0         AAA             2013J       11391             -159.0
1         AAA             2013J       28400              -53.0
2         AAA             2013J       30268              -92.0
3         AAA             2013J       31604              -52.0
4         AAA             2013J       32885             -176.0

   date_unregistration
0                  NaN
1                  NaN
2                 12.0
3                  NaN
4                  NaN

Missing Values:
code_module                0
code_presentation          0
id_student                 0
date_registration         45
date_unregistration    22521
dtype: int64

Describe (Numerical):
         id_student  date_registration  date_unregistration
count  3.259300e+04       32548.000000         10072.000000
```

```
mean   7.066877e+05         -69.411300           49.757645
std    5.491673e+05          49.260522           82.460890
min    3.733000e+03        -322.000000         -365.000000
25%    5.085730e+05        -100.000000           -2.000000
50%    5.903100e+05         -57.000000           27.000000
75%    6.444530e+05         -29.000000          109.000000
max    2.716795e+06         167.000000          444.000000

Describe (Categorical):
       code_module code_presentation
count        32593             32593
unique           7                 4
top            BBB             2014J
freq          7909             11260
------------------------------------------


--- DataFrame: student_vle ---
Shape: (10655280, 6)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10655280 entries, 0 to 10655279
Data columns (total 6 columns):
 #   Column             Dtype
---  ------             -----
 0   code_module        object
 1   code_presentation  object
 2   id_student         int64
 3   id_site            int64
 4   date               int64
 5   sum_click          int64
dtypes: int64(4), object(2)
memory usage: 487.8+ MB

Head:
  code_module code_presentation  id_student  id_site  date  sum_click
0         AAA             2013J       28400   546652   -10          4
1         AAA             2013J       28400   546652   -10          1
2         AAA             2013J       28400   546652   -10          1
3         AAA             2013J       28400   546614   -10         11
4         AAA             2013J       28400   546714   -10          1

Missing Values:
code_module          0
code_presentation    0
id_student           0
id_site              0
date                 0
```

```
sum_click            0
dtype: int64

Describe (Numerical):
         id_student       id_site          date     sum_click
count  1.065528e+07  1.065528e+07  1.065528e+07  1.065528e+07
mean   7.333336e+05  7.383234e+05  9.517400e+01  3.716946e+00
std    5.827060e+05  1.312196e+05  7.607130e+01  8.849047e+00
min    6.516000e+03  5.267210e+05 -2.500000e+01  1.000000e+00
25%    5.077430e+05  6.735190e+05  2.500000e+01  1.000000e+00
50%    5.882360e+05  7.300690e+05  8.600000e+01  2.000000e+00
75%    6.464840e+05  8.770300e+05  1.560000e+02  3.000000e+00
max    2.698588e+06  1.049562e+06  2.690000e+02  6.977000e+03

Describe (Categorical):
       code_module code_presentation
count     10655280          10655280
unique           7                 4
top            FFF             2014J
freq       4014499           3619452
---------------------------------


--- DataFrame: vle ---
Shape: (6364, 6)
Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6364 entries, 0 to 6363
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   id_site            6364 non-null   int64
 1   code_module        6364 non-null   object
 2   code_presentation  6364 non-null   object
 3   activity_type      6364 non-null   object
 4   week_from          1121 non-null   float64
 5   week_to            1121 non-null   float64
dtypes: float64(2), int64(1), object(3)
memory usage: 298.4+ KB

Head:
   id_site code_module code_presentation activity_type  week_from  week_to
0   546943         AAA             2013J      resource        NaN      NaN
1   546712         AAA             2013J     oucontent        NaN      NaN
2   546998         AAA             2013J      resource        NaN      NaN
3   546888         AAA             2013J           url        NaN      NaN
4   547035         AAA             2013J      resource        NaN      NaN
```

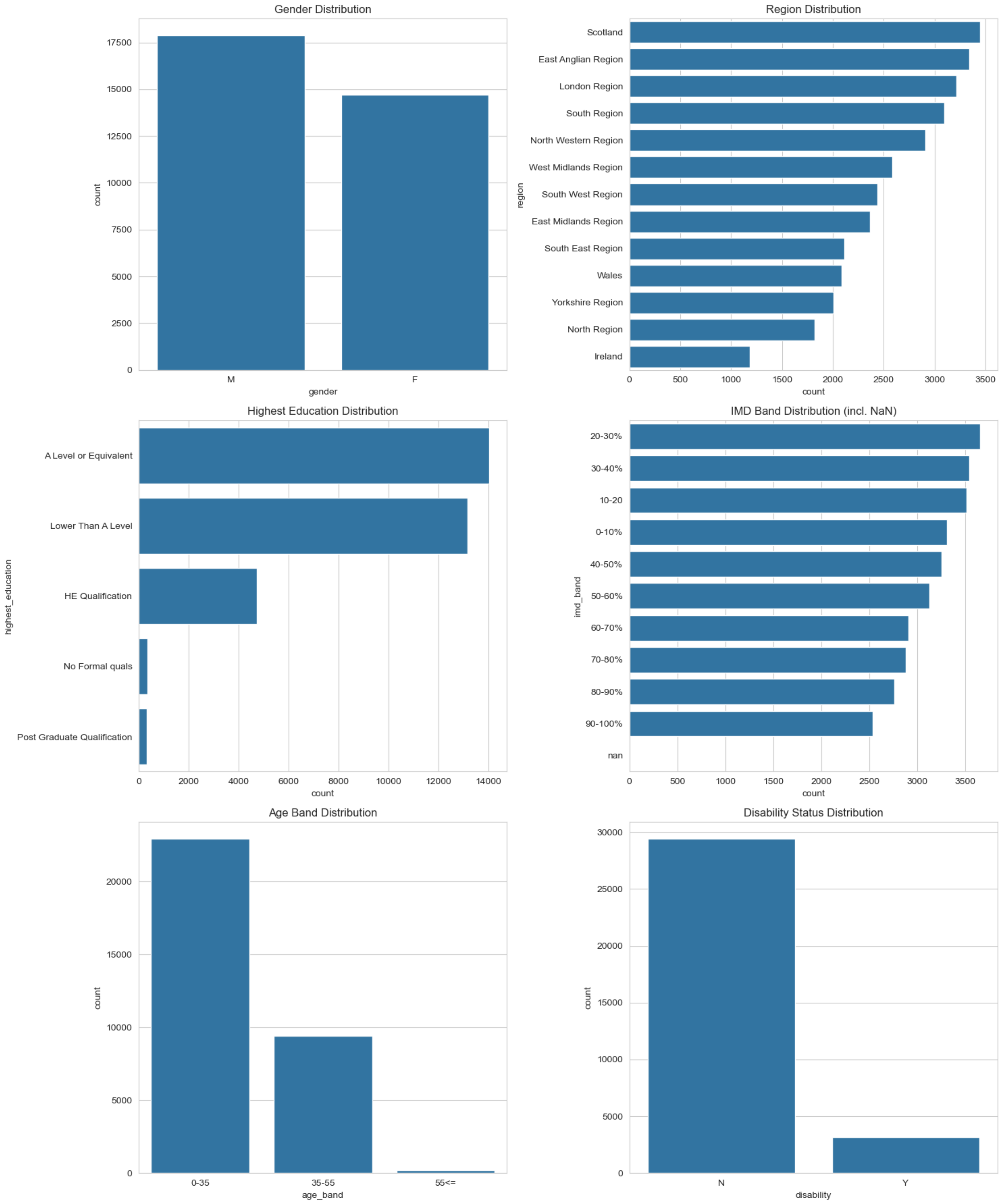
```
Missing Values:
id_site                 0
code_module             0
code_presentation       0
activity_type           0
week_from            5243
week_to              5243
dtype: int64

Describe (Numerical):
            id_site    week_from      week_to
count  6.364000e+03  1121.000000  1121.000000
mean   7.260991e+05    15.204282    15.214987
std    1.283151e+05     8.792865     8.779806
min    5.267210e+05     0.000000     0.000000
25%    6.615928e+05     8.000000     8.000000
50%    7.300965e+05    15.000000    15.000000
75%    8.140162e+05    22.000000    22.000000
max    1.077905e+06    29.000000    29.000000

Describe (Categorical):
       code_module code_presentation activity_type
count         6364              6364          6364
unique           7                 4            20
top            FFF             2013J      resource
freq          1967              1772          2660
-------------------------
```

[4]:

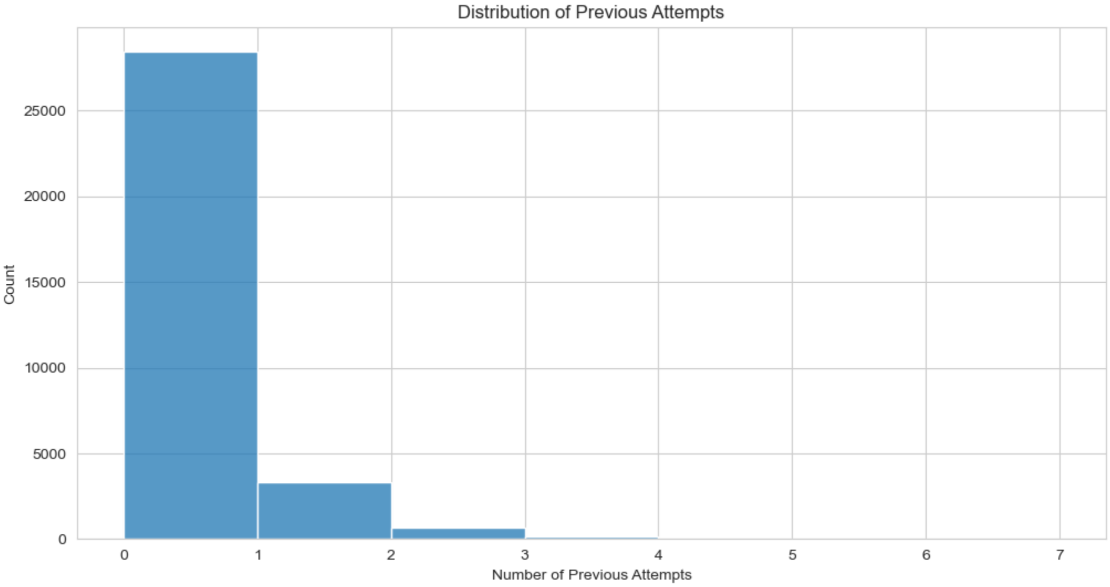
```
print("--- Analyzing student_info ---")
student_info = dataframes['student_info']

# Distributions of categorical features
fig, axes = plt.subplots(3, 2, figsize=(15, 18))
sns.countplot(ax=axes[0, 0], x='gender', data=student_info)
axes[0, 0].set_title('Gender Distribution')

sns.countplot(ax=axes[0, 1], y='region', data=student_info,␣
 ↪order=student_info['region'].value_counts().index)
axes[0, 1].set_title('Region Distribution')

sns.countplot(ax=axes[1, 0], y='highest_education', data=student_info,␣
 ↪order=student_info['highest_education'].value_counts().index)
axes[1, 0].set_title('Highest Education Distribution')
```

```
sns.countplot(ax=axes[1, 1], y='imd_band', data=student_info,␣
 ↪order=student_info['imd_band'].value_counts(dropna=False).index) # Show NaNs
axes[1, 1].set_title('IMD Band Distribution (incl. NaN)')

sns.countplot(ax=axes[2, 0], x='age_band', data=student_info,␣
 ↪order=student_info['age_band'].value_counts().index)
axes[2, 0].set_title('Age Band Distribution')

sns.countplot(ax=axes[2, 1], x='disability', data=student_info)
axes[2, 1].set_title('Disability Status Distribution')

plt.tight_layout()
plt.show()

# Explore numerical features
print("\nPrevious Attempts Distribution:")
sns.histplot(student_info['num_of_prev_attempts'],␣
 ↪bins=range(student_info['num_of_prev_attempts'].max() + 2), kde=False)
plt.title('Distribution of Previous Attempts')
plt.xlabel('Number of Previous Attempts')
plt.ylabel('Count')
plt.show()

print("\nStudied Credits Distribution:")
sns.histplot(student_info['studied_credits'], bins=20, kde=True)
plt.title('Distribution of Studied Credits')
plt.xlabel('Studied Credits')
plt.ylabel('Count')
plt.show()

# Check final result distribution
print("\nFinal Result Distribution:")
sns.countplot(y='final_result', data=student_info,␣
 ↪order=student_info['final_result'].value_counts().index)
plt.title('Final Result Distribution')
plt.show()
```

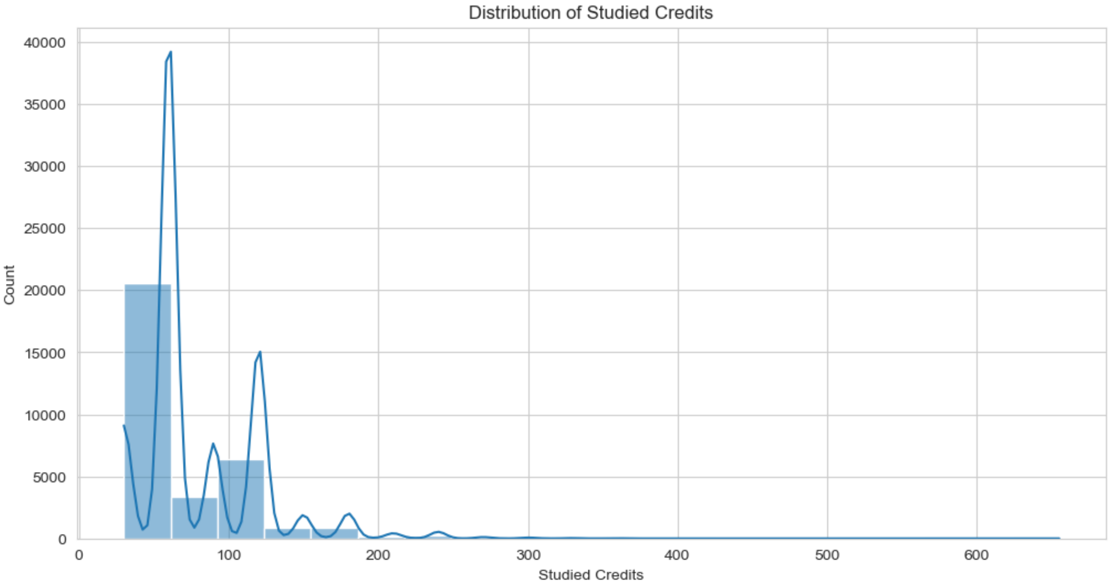
```
--- Analyzing student_info ---
```



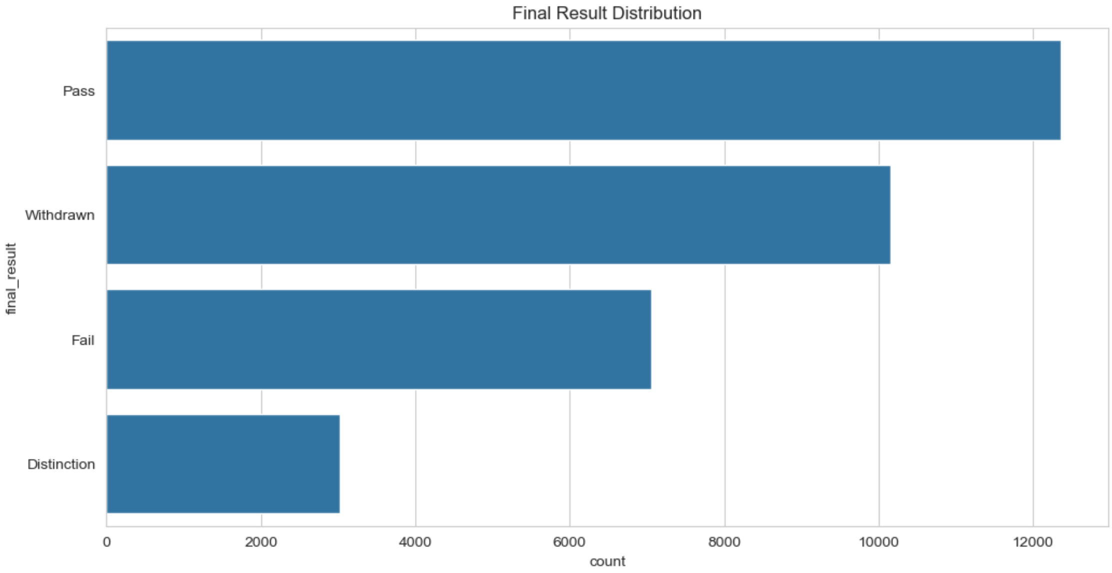
```
Previous Attempts Distribution:
```



```
Studied Credits Distribution:
```



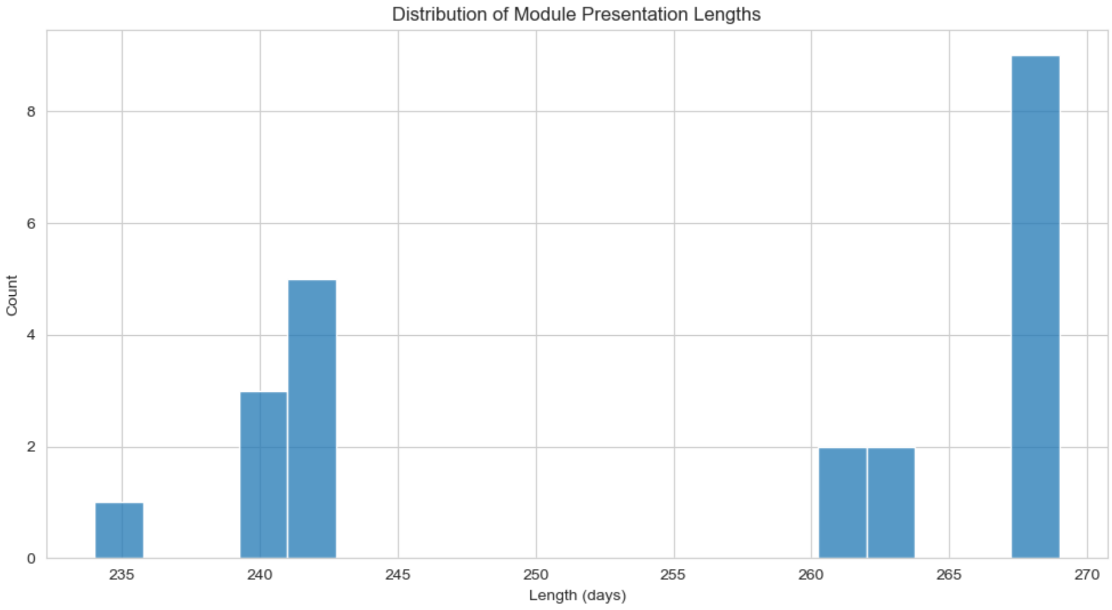
```
Final Result Distribution:
```



[5]:
```
print("--- Analyzing courses ---")
courses = dataframes['courses']

print(f"Number of unique modules: {courses['code_module'].nunique()}")
print(f"Number of unique presentations: {courses.shape[0]}") # Each row is a␣
  ↪presentation
print("\nModules and their number of presentations:")
print(courses['code_module'].value_counts())

print("\nDistribution of Module Presentation Lengths:")
sns.histplot(courses['module_presentation_length'], bins=20, kde=False)
plt.title('Distribution of Module Presentation Lengths')
plt.xlabel('Length (days)')
plt.ylabel('Count')
plt.show()

# Create presentation_id for later use (might do this again in preprocessing)
courses['presentation_id'] = courses['code_module'] + '_' +␣
  ↪courses['code_presentation']
print(f"\nCheck unique presentation IDs: {courses['presentation_id'].nunique()}␣
  ↪(should match shape[0])")
```

```
--- Analyzing courses ---
Number of unique modules: 7
Number of unique presentations: 22

Modules and their number of presentations:
code_module
```

```
BBB    4
DDD    4
FFF    4
EEE    3
GGG    3
AAA    2
CCC    2
Name: count, dtype: int64

Distribution of Module Presentation Lengths:
```

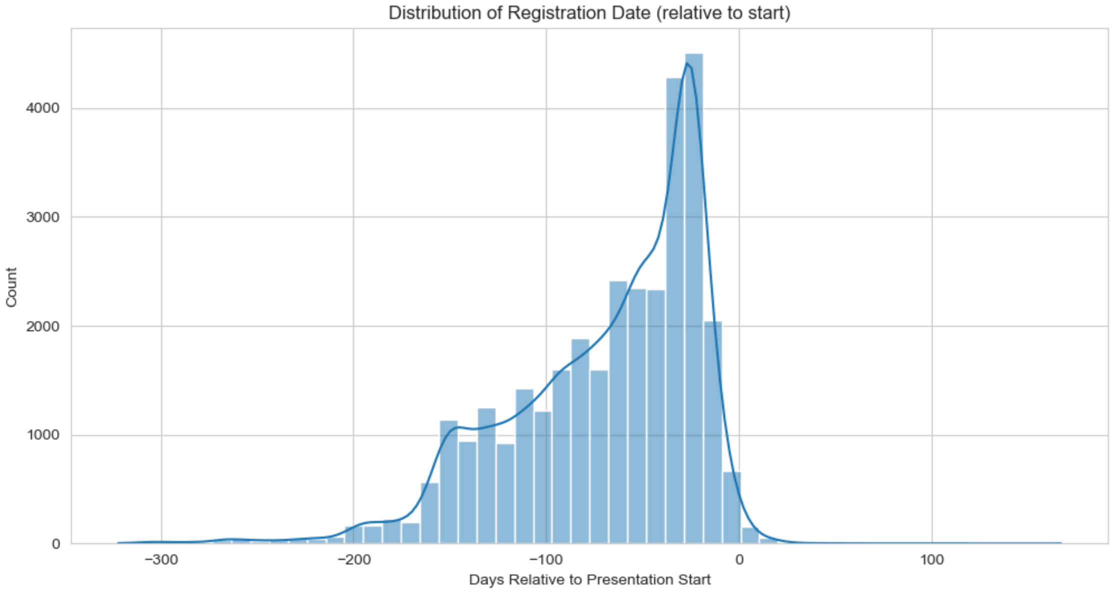


```
Check unique presentation IDs: 22 (should match shape[0])
```

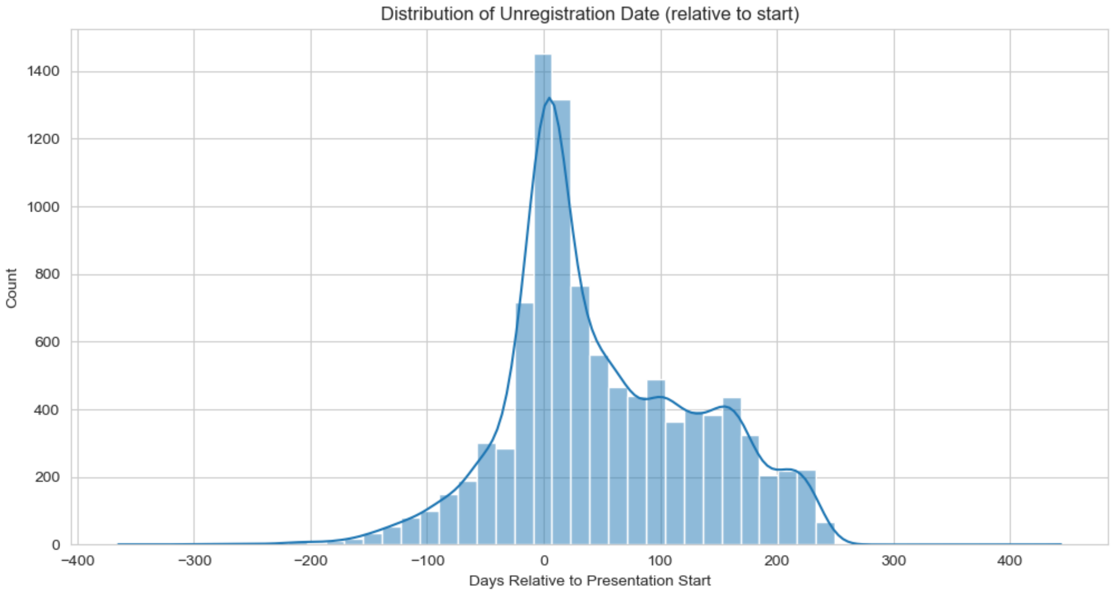
[6]:
```python
print("--- Analyzing student_registration ---")
student_reg = dataframes['student_registration']

# Check for multiple registrations per student per presentation (should be␣
  ↪unique ideally)
reg_duplicates = student_reg.duplicated(subset=['id_student', 'code_module',␣
  ↪'code_presentation']).sum()
print(f"\nDuplicate registrations (same student, same presentation):␣
  ↪{reg_duplicates}")

# Distribution of registration dates (relative to presentation start, which is␣
  ↪0)
print("\nDistribution of Registration Dates:")
```

```python
sns.histplot(student_reg['date_registration'].dropna(), bins=50, kde=True)
plt.title('Distribution of Registration Date (relative to start)')
plt.xlabel('Days Relative to Presentation Start')
plt.ylabel('Count')
plt.show()
# Note: Negative values mean registered before start, positive after.

# Distribution of unregistration dates
print("\nDistribution of Unregistration Dates:")
sns.histplot(student_reg['date_unregistration'].dropna(), bins=50, kde=True)
plt.title('Distribution of Unregistration Date (relative to start)')
plt.xlabel('Days Relative to Presentation Start')
plt.ylabel('Count')
plt.show()

print(f"\nPercentage of registrations with an unregistration date: \
{student_reg['date_unregistration'].notnull().mean() * 100:.2f}%")
```

```
--- Analyzing student_registration ---

Duplicate registrations (same student, same presentation): 0

Distribution of Registration Dates:
```



```
Distribution of Unregistration Dates:
```

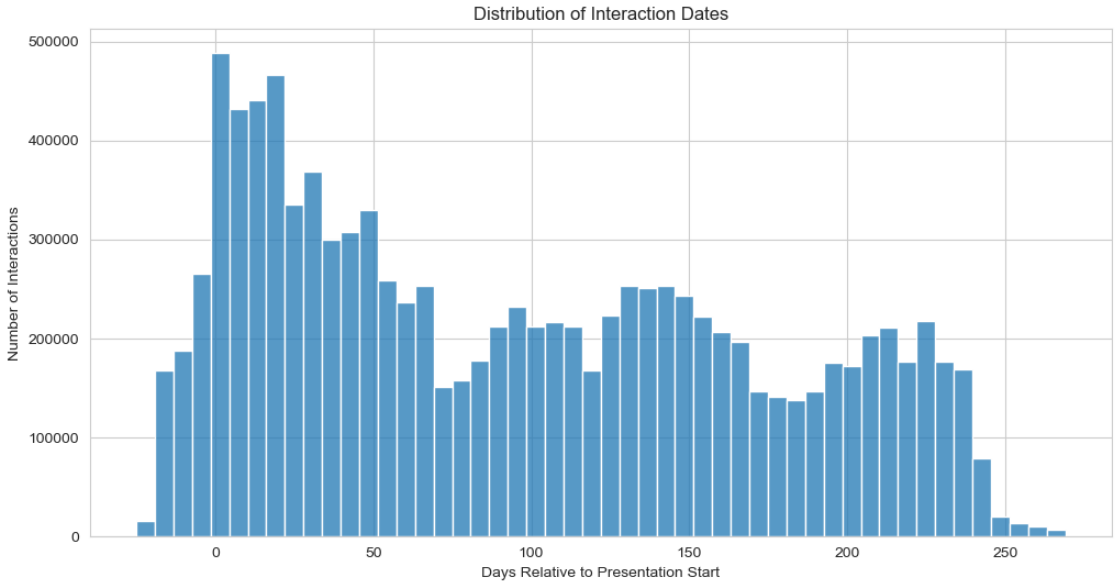


```
Percentage of registrations with an unregistration date: 30.90%
```

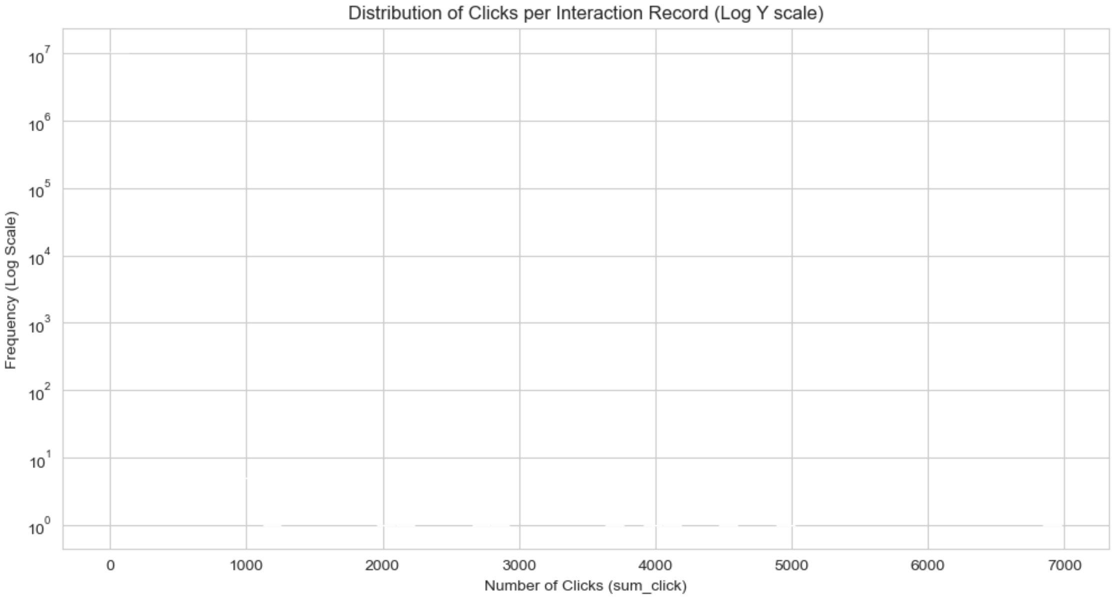
[7]:

```python
print("--- Analyzing student_vle ---")
student_vle = dataframes['student_vle']

print(f"Total interaction records: {student_vle.shape[0]}")
print(f"Unique students in VLE logs: {student_vle['id_student'].nunique()}")
print(f"Unique VLE items interacted with: {student_vle['id_site'].nunique()}")

# Distribution of interaction dates
print("\nDistribution of Interaction Dates (relative to start):")
sns.histplot(student_vle['date'], bins=50, kde=False)
plt.title('Distribution of Interaction Dates')
plt.xlabel('Days Relative to Presentation Start')
plt.ylabel('Number of Interactions')
plt.show()

# Distribution of sum_click
# Handle potential outliers by looking at quantiles
print("\nDistribution of Clicks per Interaction Record:")
print(student_vle['sum_click'].describe(percentiles=[.25, .5, .75, .9, .95, .99]))
# Plotting might be skewed, consider log scale or capping
sns.histplot(student_vle['sum_click'], bins=50, log_scale=(False, True)) # Log scale for y-axis
plt.title('Distribution of Clicks per Interaction Record (Log Y scale)')
plt.xlabel('Number of Clicks (sum_click)')
plt.ylabel('Frequency (Log Scale)')
plt.show()

# Interactions per student (across all their registered courses)
interactions_per_student = student_vle.groupby('id_student').size()
print("\nInteractions per Student (Summary Stats):")
print(interactions_per_student.describe())
sns.histplot(interactions_per_student, bins=50, log_scale=True)
plt.title('Distribution of Total Interaction Records per Student (Log Scale)')
plt.xlabel('Number of Interaction Records (Log Scale)')
plt.ylabel('Number of Students (Log Scale)')
plt.show()

# Clicks per student
clicks_per_student = student_vle.groupby('id_student')['sum_click'].sum()
print("\nTotal Clicks per Student (Summary Stats):")
print(clicks_per_student.describe())
sns.histplot(clicks_per_student, bins=50, log_scale=True)
plt.title('Distribution of Total Clicks per Student (Log Scale)')
plt.xlabel('Total Clicks (Log Scale)')
plt.ylabel('Number of Students (Log Scale)')
plt.show()
```

```
--- Analyzing student_vle ---
Total interaction records: 10655280
Unique students in VLE logs: 26074
Unique VLE items interacted with: 6268

Distribution of Interaction Dates (relative to start):
```

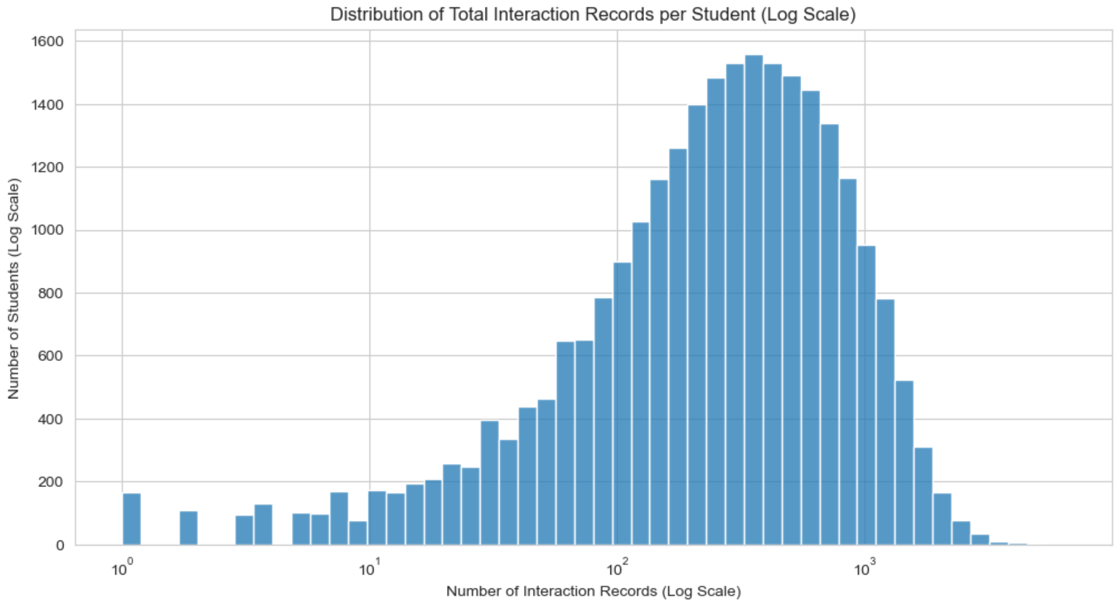


```
Distribution of Clicks per Interaction Record:
count    1.065528e+07
mean     3.716946e+00
std      8.849047e+00
min      1.000000e+00
25%      1.000000e+00
50%      2.000000e+00
75%      3.000000e+00
90%      8.000000e+00
95%      1.200000e+01
99%      3.400000e+01
max      6.977000e+03
Name: sum_click, dtype: float64
```

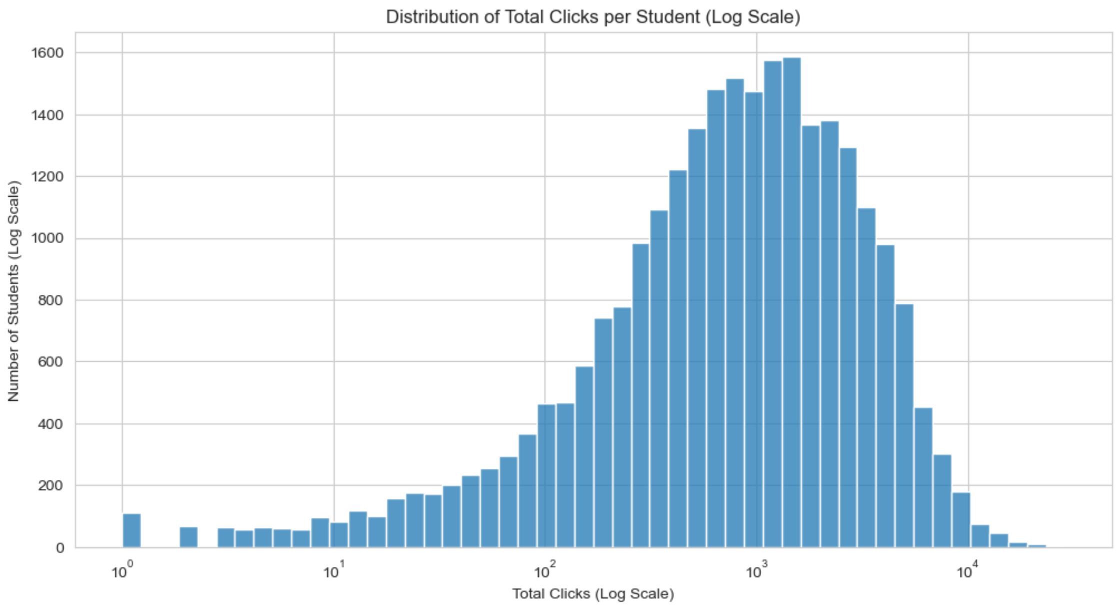


```
Interactions per Student (Summary Stats):
count    26074.000000
mean       408.655365
std        430.608121
min          1.000000
25%        108.000000
50%        270.000000
75%        570.000000
max       6389.000000
```

```
dtype: float64
```



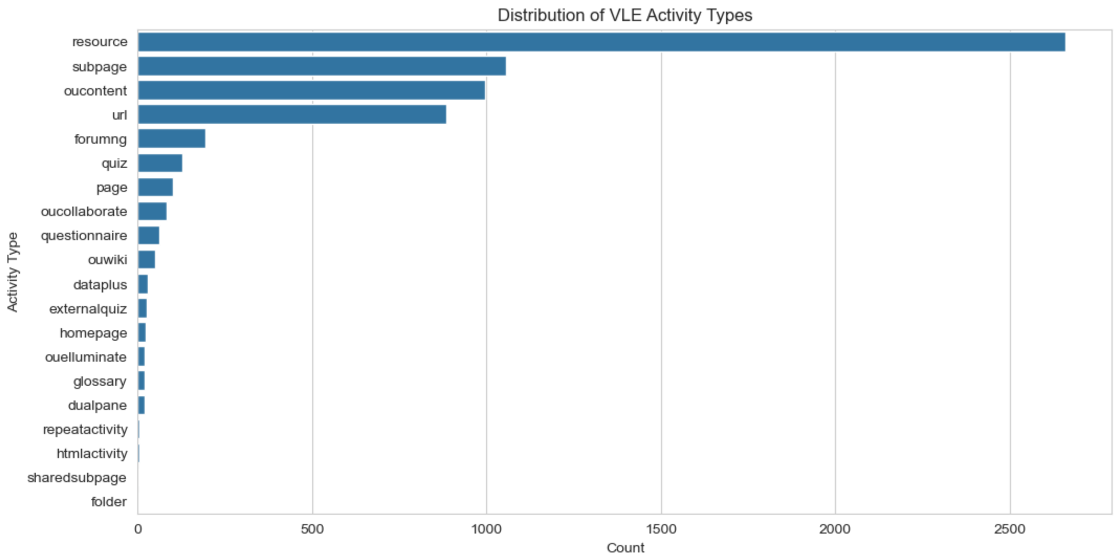
```
Total Clicks per Student (Summary Stats):
count    26074.000000
mean      1518.949873
std       1935.994635
min          1.000000
25%        298.000000
50%        824.000000
75%       2018.000000
max      28615.000000
Name: sum_click, dtype: float64
```



[8]:

```
print("--- Analyzing vle ---")
vle = dataframes['vle']

print(f"Total VLE items defined: {vle.shape[0]}")
print(f"Unique VLE item IDs (id_site): {vle['id_site'].nunique()}") # Should␣
  ↪match shape[0]

# Distribution of activity types
print("\nDistribution of VLE Activity Types:")
activity_counts = vle['activity_type'].value_counts()
sns.barplot(y=activity_counts.index, x=activity_counts.values)
plt.title('Distribution of VLE Activity Types')
plt.xlabel('Count')
plt.ylabel('Activity Type')
plt.show()

# Weeks - presence of week_from/week_to
print(f"\nPercentage of VLE items with 'week_from': {vle['week_from'].notnull().
  ↪mean()*100:.2f}%")
print(f"Percentage of VLE items with 'week_to': {vle['week_to'].notnull().
  ↪mean()*100:.2f}%")
```

```
--- Analyzing vle ---
Total VLE items defined: 6364
Unique VLE item IDs (id_site): 6364

Distribution of VLE Activity Types:
```

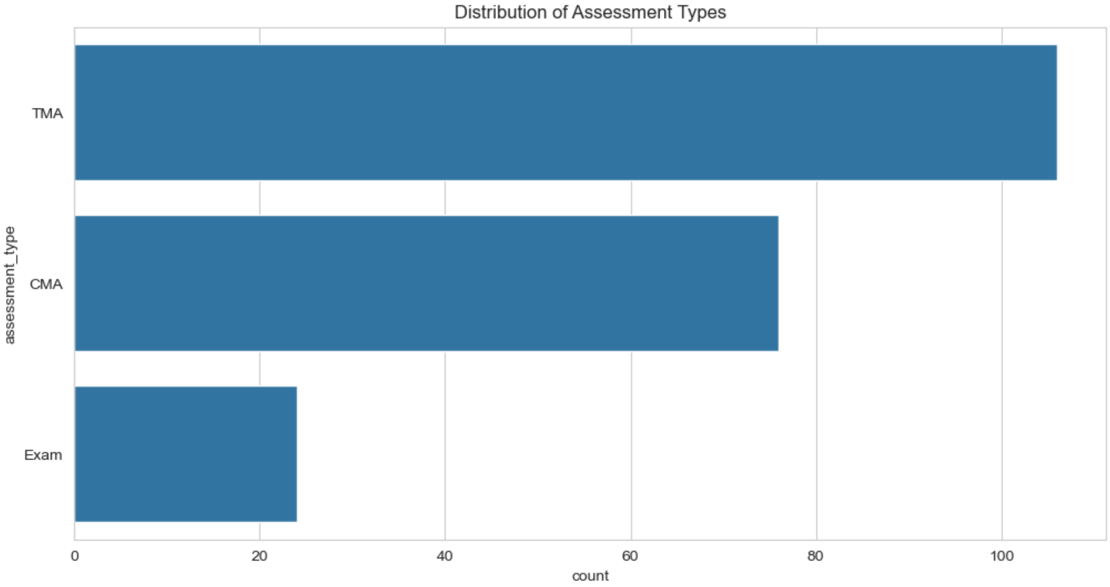


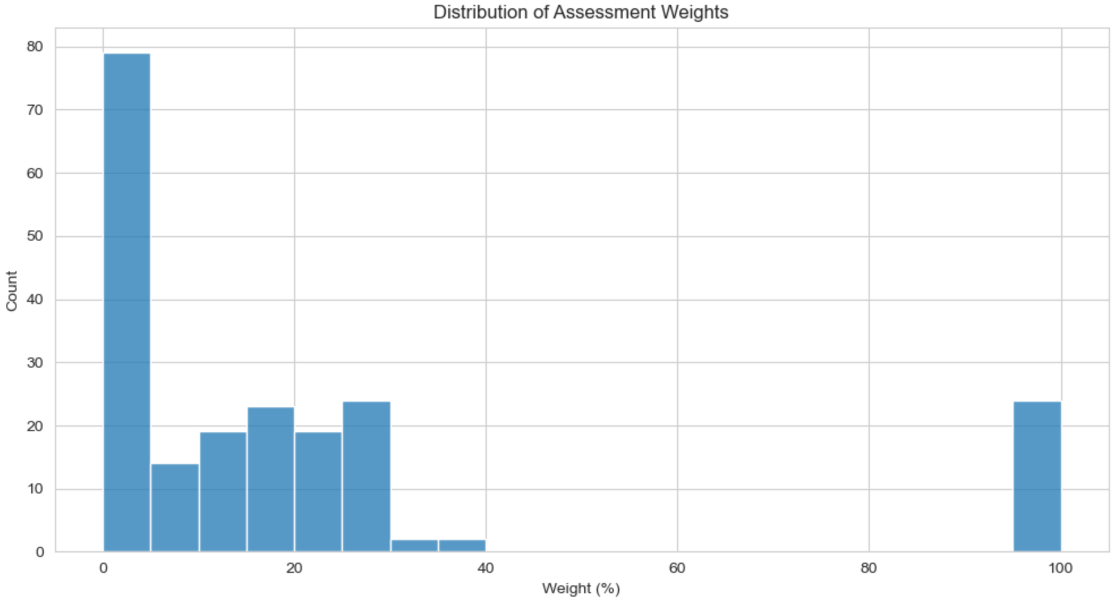
```
Percentage of VLE items with 'week_from': 17.61%
Percentage of VLE items with 'week_to': 17.61%
```

[9]:

```python
print("--- Analyzing assessments & student_assessment ---")
assessments = dataframes['assessments']
student_assessment = dataframes['student_assessment']

print(f"Total assessments defined: {assessments.shape[0]}")
print(f"Unique assessment IDs: {assessments['id_assessment'].nunique()}") # Should match shape[0]

# Assessment types
print("\nDistribution of Assessment Types:")
sns.countplot(y='assessment_type', data=assessments, order=assessments['assessment_type'].value_counts().index)
plt.title('Distribution of Assessment Types')
plt.show()

# Distribution of assessment weights
sns.histplot(assessments['weight'], bins=20, kde=False)
plt.title('Distribution of Assessment Weights')
plt.xlabel('Weight (%)')
plt.ylabel('Count')
plt.show()

# Explore student scores
print("\nDistribution of Student Scores:")
print(student_assessment['score'].describe())
sns.histplot(student_assessment['score'].dropna(), bins=20, kde=True)
plt.title('Distribution of Student Assessment Scores')
plt.xlabel('Score')
plt.ylabel('Count')
plt.show()

# Pass/Fail based on score >= 40 (common threshold)
student_assessment['passed'] = student_assessment['score'] >= 40
print("\nPass Rate (based on score >= 40):")
print(student_assessment['passed'].value_counts(normalize=True))

# Submission relative to deadline
assessments_renamed = assessments.rename(columns={'date': 'deadline'})
student_assessment_merged = pd.merge(
    student_assessment,
    assessments_renamed[['id_assessment', 'deadline']],
    on='id_assessment',
    how='left'
)
student_assessment_merged['days_early'] = student_assessment_merged['deadline'] - student_assessment_merged['date_submitted']

print("\nDistribution of Submission Time Relative to Deadline:")
print(student_assessment_merged['days_early'].describe())
sns.histplot(student_assessment_merged['days_early'].dropna(), bins=50)
plt.title('Submission Time Relative to Deadline (Positive = Early)')
plt.xlabel('Days Early')
plt.ylabel('Count')
plt.show()
```

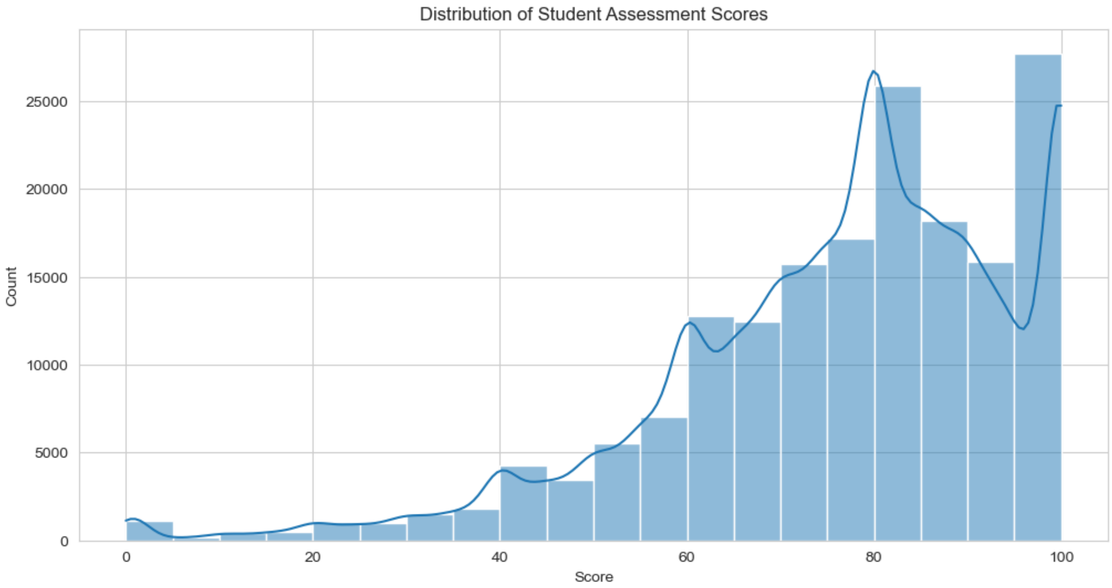
```
--- Analyzing assessments & student_assessment ---
Total assessments defined: 206
Unique assessment IDs: 206

Distribution of Assessment Types:
```



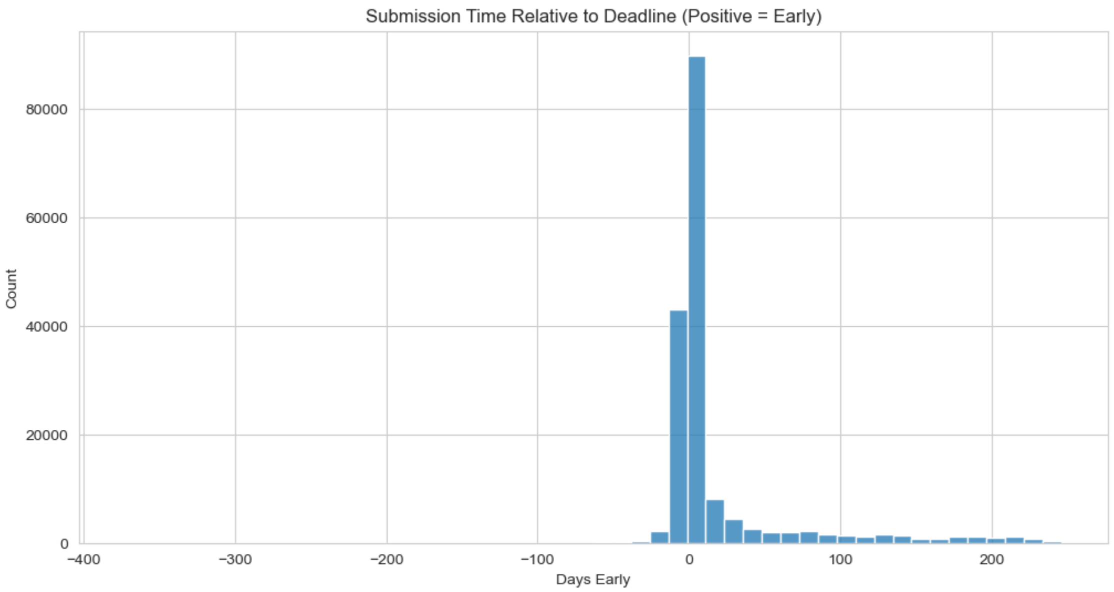


```
Distribution of Student Scores:
count    173739.000000
mean         75.799573
std          18.798107
min           0.000000
25%          65.000000
50%          80.000000
```

```
75%            90.000000
max           100.000000
Name: score, dtype: float64
```



```
Pass Rate (based on score >= 40):
passed
True     0.955431
False    0.044569
Name: proportion, dtype: float64

Distribution of Submission Time Relative to Deadline:
count    171047.000000
mean         16.657989
std          45.945880
min        -372.000000
25%          -2.000000
50%           1.000000
75%           6.000000
max         246.000000
Name: days_early, dtype: float64
```

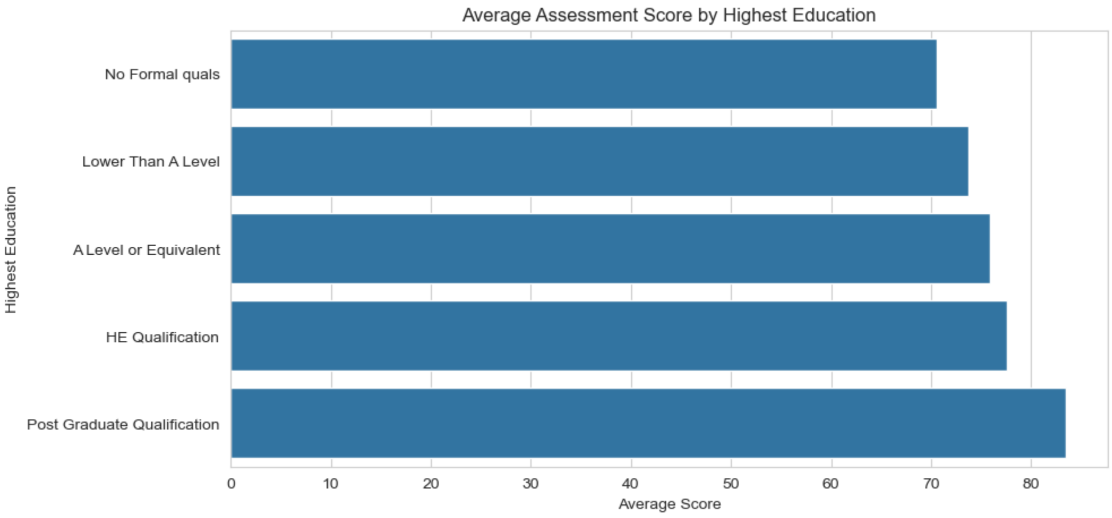


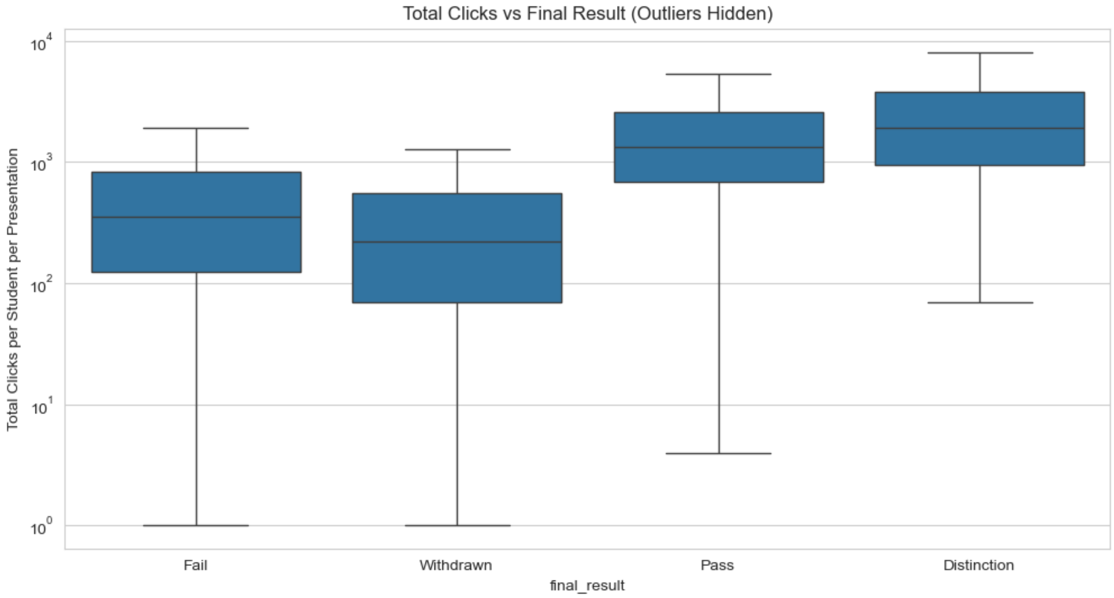
[10]:

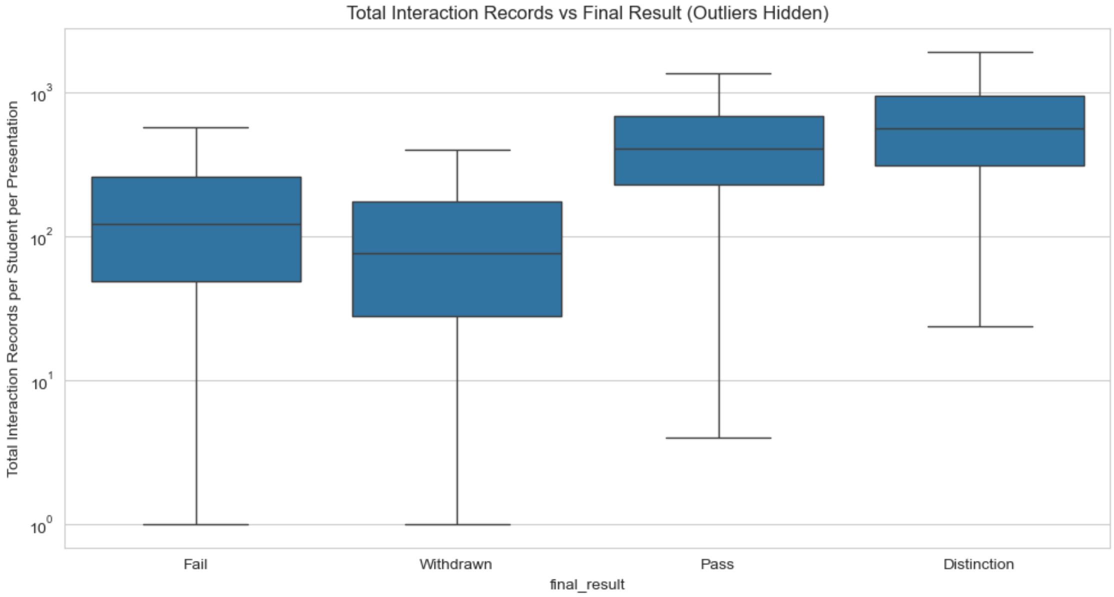
```
# Example: Average score per highest education level
student_info_subset = student_info[['id_student', 'highest_education']]
student_scores_merged = pd.merge(student_assessment, student_info_subset,␣
 ↪on='id_student')

avg_score_by_edu = student_scores_merged.groupby('highest_education')['score'].
 ↪mean().sort_values()

print("\nAverage Assessment Score by Highest Education:")
print(avg_score_by_edu)

plt.figure(figsize=(10, 5))
sns.barplot(x=avg_score_by_edu.values, y=avg_score_by_edu.index)
plt.title('Average Assessment Score by Highest Education')
plt.xlabel('Average Score')
plt.ylabel('Highest Education')
plt.show()

# Example: Interactions vs Final Result
student_interactions = student_vle.groupby(['id_student', 'code_module',␣
 ↪'code_presentation']).size().reset_index(name='total_interactions')
student_clicks = student_vle.groupby(['id_student', 'code_module',␣
 ↪'code_presentation'])['sum_click'].sum().reset_index(name='total_clicks')

student_activity = pd.merge(student_interactions, student_clicks,␣
 ↪on=['id_student', 'code_module', 'code_presentation'])
```

```
student_activity_results = pd.merge(student_info[['id_student', 'code_module',␣
 ↪'code_presentation', 'final_result']], student_activity, on=['id_student',␣
 ↪'code_module', 'code_presentation'])

plt.figure(figsize=(12, 6))
sns.boxplot(data=student_activity_results, x='final_result', y='total_clicks',␣
 ↪showfliers=False, # Hide outliers for clarity
            order=['Fail', 'Withdrawn', 'Pass', 'Distinction'])
plt.title('Total Clicks vs Final Result (Outliers Hidden)')
plt.ylabel('Total Clicks per Student per Presentation')
plt.yscale('log') # Log scale often useful for clicks
plt.show()

plt.figure(figsize=(12, 6))
sns.boxplot(data=student_activity_results, x='final_result',␣
 ↪y='total_interactions', showfliers=False,
            order=['Fail', 'Withdrawn', 'Pass', 'Distinction'])
plt.title('Total Interaction Records vs Final Result (Outliers Hidden)')
plt.ylabel('Total Interaction Records per Student per Presentation')
plt.yscale('log')
plt.show()
```

```
Average Assessment Score by Highest Education:
highest_education
No Formal quals                70.601852
Lower Than A Level             73.677280
A Level or Equivalent          75.825197
HE Qualification               77.550154
Post Graduate Qualification    83.489118
Name: score, dtype: float64
```




Total Clicks vs Final Result (Outliers Hidden)
Total Clicks per Student per Presentation
$10^4$
$10^3$
$10^2$
$10^1$
$10^0$
Fail
Withdrawn
Pass
Distinction
final_result


Total Interaction Records vs Final Result (Outliers Hidden)
Total Interaction Records per Student per Presentation
$10^3$
$10^2$
$10^1$
$10^0$
Fail
Withdrawn
Pass
Distinction
final_result