Finding How The Data Is Distributed

Estimated time needed: **30** minutes

In this lab, you will work with a cleaned dataset to perform Exploratory Data Analysis (EDA). You will examine the structure of the data, visualize key variables, and analyze trends related to developer experience, tools, job satisfaction, and other important aspects.

Objectives

In this lab you will perform the following:

- Understand the structure of the dataset.
- Perform summary statistics and data visualization.
- Identify trends in developer experience, tools, job satisfaction, and other key variables.

Install the required libraries

In [1]:
```
!pip install pandas
!pip install matplotlib
!pip install seaborn
```

```
Collecting pandas
  Downloading pandas-2.3.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (91 k
B)
Collecting numpy>=1.26.0 (from pandas)
  Downloading numpy-2.3.0-cp312-cp312-manylinux_2_28_x86_64.whl.metadata (62 kB)
Requirement already satisfied: python-dateutil>=2.8.2 in /opt/conda/lib/python3.12/site-packages (fr
om pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.12/site-packages (from pandas)
(2024.2)
Collecting tzdata>=2022.7 (from pandas)
  Downloading tzdata-2025.2-py2.py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-packages (from python-date
util>=2.8.2->pandas) (1.17.0)
Downloading pandas-2.3.0-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.0 MB)
                                          12.0/12.0 MB 24.1 MB/s eta 0:00:00:00:01
Downloading numpy-2.3.0-cp312-cp312-manylinux_2_28_x86_64.whl (16.6 MB)
                                          16.6/16.6 MB 28.4 MB/s eta 0:00:00:00:01
Downloading tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: tzdata, numpy, pandas
Successfully installed numpy-2.3.0 pandas-2.3.0 tzdata-2025.2
Collecting matplotlib
  Downloading matplotlib-3.10.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(11 kB)
Collecting contourpy>=1.0.1 (from matplotlib)
  Downloading contourpy-1.3.2-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(5.5 kB)
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
  Downloading fonttools-4.58.4-cp312-cp312-manylinux1_x86_64.manylinux2014_x86_64.manylinux_2_17_x86
_64.manylinux_2_5_x86_64.whl.metadata (106 kB)
Collecting kiwisolver>=1.3.1 (from matplotlib)
  Downloading kiwisolver-1.4.8-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata
(6.2 kB)
Requirement already satisfied: numpy>=1.23 in /opt/conda/lib/python3.12/site-packages (from matplotl
ib) (2.3.0)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.12/site-packages (from matp
lotlib) (24.2)
Collecting pillow>=8 (from matplotlib)
  Downloading pillow-11.2.1-cp312-cp312-manylinux_2_28_x86_64.whl.metadata (8.9 kB)
Collecting pyparsing>=2.3.1 (from matplotlib)
  Downloading pyparsing-3.2.3-py3-none-any.whl.metadata (5.0 kB)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.12/site-packages (from
matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-packages (from python-date
util>=2.7->matplotlib) (1.17.0)
Downloading matplotlib-3.10.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (8.6 MB)
                                          8.6/8.6 MB 70.4 MB/s eta 0:00:00
Downloading contourpy-1.3.2-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (323 kB)
Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Downloading fonttools-4.58.4-cp312-cp312-manylinux1_x86_64.manylinux2014_x86_64.manylinux_2_17_x86_6
4.manylinux_2_5_x86_64.whl (4.9 MB)
                                          4.9/4.9 MB 46.6 MB/s eta 0:00:00
Downloading kiwisolver-1.4.8-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.5 MB)
                                          1.5/1.5 MB 33.2 MB/s eta 0:00:00
Downloading pillow-11.2.1-cp312-cp312-manylinux_2_28_x86_64.whl (4.6 MB)
                                          4.6/4.6 MB 20.3 MB/s eta 0:00:00ta 0:00:01
Downloading pyparsing-3.2.3-py3-none-any.whl (111 kB)
Installing collected packages: pyparsing, pillow, kiwisolver, fonttools, cycler, contourpy, matplotl
ib
Successfully installed contourpy-1.3.2 cycler-0.12.1 fonttools-4.58.4 kiwisolver-1.4.8 matplotlib-3.
10.3 pillow-11.2.1 pyparsing-3.2.3
Collecting seaborn
  Downloading seaborn-0.13.2-py3-none-any.whl.metadata (5.4 kB)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /opt/conda/lib/python3.12/site-packages (from
seaborn) (2.3.0)
Requirement already satisfied: pandas>=1.2 in /opt/conda/lib/python3.12/site-packages (from seaborn)
(2.3.0)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /opt/conda/lib/python3.12/site-packages (f
rom seaborn) (3.10.3)
Requirement already satisfied: contourpy>=1.0.1 in /opt/conda/lib/python3.12/site-packages (from mat
plotlib!=3.6.1,>=3.4->seaborn) (1.3.2)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.12/site-packages (from matplot
lib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /opt/conda/lib/python3.12/site-packages (from ma
```

```
tplotlib!=3.6.1,>=3.4->seaborn) (4.58.4)
Requirement already satisfied: kiwisolver>=1.3.1 in /opt/conda/lib/python3.12/site-packages (from ma
tplotlib!=3.6.1,>=3.4->seaborn) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.12/site-packages (from matp
lotlib!=3.6.1,>=3.4->seaborn) (24.2)
Requirement already satisfied: pillow>=8 in /opt/conda/lib/python3.12/site-packages (from matplotli
b!=3.6.1,>=3.4->seaborn) (11.2.1)
Requirement already satisfied: pyparsing>=2.3.1 in /opt/conda/lib/python3.12/site-packages (from mat
plotlib!=3.6.1,>=3.4->seaborn) (3.2.3)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.12/site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.12/site-packages (from pandas>
=1.2->seaborn) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /opt/conda/lib/python3.12/site-packages (from panda
s>=1.2->seaborn) (2025.2)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.12/site-packages (from python-date
util>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.17.0)
Downloading seaborn-0.13.2-py3-none-any.whl (294 kB)
Installing collected packages: seaborn
Successfully installed seaborn-0.13.2
```

Step 1: Import Libraries and Load Data

- Import the `pandas`, `matplotlib.pyplot`, and `seaborn` libraries.

- You will begin with loading the dataset. You can use the pyfetch method if working on JupyterLite. Otherwise, you can use pandas' read_csv() function directly on their local machines or cloud environments.

In [2]:
```python
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the Stack Overflow survey dataset
data_url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/n01PQ9pSmiRX6520fluj
df = pd.read_csv(data_url)

# Display the first few rows of the dataset
df.head()
```

Out[2]:

	ResponseId	MainBranch	Age	Employment	RemoteWork	Check	CodingActivities	EdLevel	
0	1	I am a developer by profession	Under 18 years old	Employed, full-time	Remote	Apples	Hobby	Primary/elementary school	E
1	2	I am a developer by profession	35-44 years old	Employed, full-time	Remote	Apples	Hobby;Contribute to open-source projects;Other...	Bachelor's degree (B.A., B.S., B.Eng., etc.)	E medi
2	3	I am a developer by profession	45-54 years old	Employed, full-time	Remote	Apples	Hobby;Contribute to open-source projects;Other...	Master's degree (M.A., M.S., M.Eng., MBA, etc.)	E medi
3	4	I am learning to code	18-24 years old	Student, full-time	NaN	Apples	NaN	Some college/university study without earning ...	vi
4	5	I am a developer by profession	18-24 years old	Student, full-time	NaN	Apples	NaN	Secondary school (e.g. American high school, G...	vi

5 rows × 114 columns

Step 2: Examine the Structure of the Data

- Display the column names, data types, and summary information to understand the data structure.
- Objective: Gain insights into the dataset's shape and available variables.

In [3]:
```
## Write your code here
# --- Step 2: Examine the Structure of the Data ---
print("\n--- Step 2: Examine the Structure of the Data ---")
print("Objective: Gain insights into the dataset's shape and available variables.")

# 1. Display column names
print("\nColumn Names:")
print(df.columns.tolist())

# 2. Display data types and summary information (df.info())
print("\nDataFrame Information (df.info()):")
df.info()

# 3. Display column data types (df.dtypes)
print("\nColumn Data Types (df.dtypes):")
print(df.dtypes)

# 4. Display Summary Statistics (df.describe(include='all'))
print("\nSummary Statistics for all columns (df.describe(include='all')):")
print(df.describe(include='all'))
```

```
--- Step 2: Examine the Structure of the Data ---
Objective: Gain insights into the dataset's shape and available variables.

Column Names:
['ResponseId', 'MainBranch', 'Age', 'Employment', 'RemoteWork', 'Check', 'CodingActivities', 'EdLeve
l', 'LearnCode', 'LearnCodeOnline', 'TechDoc', 'YearsCode', 'YearsCodePro', 'DevType', 'OrgSize', 'P
urchaseInfluence', 'BuyNewTool', 'BuildvsBuy', 'TechEndorse', 'Country', 'Currency', 'CompTotal', 'L
anguageHaveWorkedWith', 'LanguageWantToWorkWith', 'LanguageAdmired', 'DatabaseHaveWorkedWith', 'Data
baseWantToWorkWith', 'DatabaseAdmired', 'PlatformHaveWorkedWith', 'PlatformWantToWorkWith', 'Platfor
mAdmired', 'WebframeHaveWorkedWith', 'WebframeWantToWorkWith', 'WebframeAdmired', 'EmbeddedHaveWorke
dWith', 'EmbeddedWantToWorkWith', 'EmbeddedAdmired', 'MiscTechHaveWorkedWith', 'MiscTechWantToWorkWi
th', 'MiscTechAdmired', 'ToolsTechHaveWorkedWith', 'ToolsTechWantToWorkWith', 'ToolsTechAdmired', 'N
EWCollabToolsHaveWorkedWith', 'NEWCollabToolsWantToWorkWith', 'NEWCollabToolsAdmired', 'OpSysPersona
l use', 'OpSysProfessional use', 'OfficeStackAsyncHaveWorkedWith', 'OfficeStackAsyncWantToWorkWith',
'OfficeStackAsyncAdmired', 'OfficeStackSyncHaveWorkedWith', 'OfficeStackSyncWantToWorkWith', 'Office
StackSyncAdmired', 'AISearchDevHaveWorkedWith', 'AISearchDevWantToWorkWith', 'AISearchDevAdmired',
'NEWSOSites', 'SOVisitFreq', 'SOAccount', 'SOPartFreq', 'SOHow', 'SOComm', 'AISelect', 'AISent', 'AI
Ben', 'AIAcc', 'AIComplex', 'AIToolCurrently Using', 'AIToolInterested in Using', 'AIToolNot interes
ted in Using', 'AINextMuch more integrated', 'AINextNo change', 'AINextMore integrated', 'AINextLess
integrated', 'AINextMuch less integrated', 'AIThreat', 'AIEthics', 'AIChallenges', 'TBranch', 'ICorP
M', 'WorkExp', 'Knowledge_1', 'Knowledge_2', 'Knowledge_3', 'Knowledge_4', 'Knowledge_5', 'Knowledge
_6', 'Knowledge_7', 'Knowledge_8', 'Knowledge_9', 'Frequency_1', 'Frequency_2', 'Frequency_3', 'Time
Searching', 'TimeAnswering', 'Frustration', 'ProfessionalTech', 'ProfessionalCloud', 'ProfessionalQu
estion', 'Industry', 'JobSatPoints_1', 'JobSatPoints_4', 'JobSatPoints_5', 'JobSatPoints_6', 'JobSat
Points_7', 'JobSatPoints_8', 'JobSatPoints_9', 'JobSatPoints_10', 'JobSatPoints_11', 'SurveyLength',
'SurveyEase', 'ConvertedCompYearly', 'JobSat']

DataFrame Information (df.info()):
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65437 entries, 0 to 65436
Columns: 114 entries, ResponseId to JobSat
dtypes: float64(13), int64(1), object(100)
memory usage: 56.9+ MB

Column Data Types (df.dtypes):
ResponseId               int64
MainBranch              object
Age                     object
Employment              object
RemoteWork              object
                         ...
JobSatPoints_11        float64
SurveyLength            object
SurveyEase              object
ConvertedCompYearly    float64
JobSat                 float64
Length: 114, dtype: object

Summary Statistics for all columns (df.describe(include='all')):
          ResponseId                      MainBranch              Age  \
count   65437.000000                           65437            65437
unique           NaN                               5                8
top              NaN  I am a developer by profession  25-34 years old
freq             NaN                           50207            23911
mean    32719.000000                             NaN              NaN
std     18890.179119                             NaN              NaN
min         1.000000                             NaN              NaN
25%     16360.000000                             NaN              NaN
50%     32719.000000                             NaN              NaN
75%     49078.000000                             NaN              NaN
max     65437.000000                             NaN              NaN

                 Employment                            RemoteWork   Check  \
count                 65437                                 54806   65437
unique                  110                                     3       1
top     Employed, full-time  Hybrid (some remote, some in-person)  Apples
freq                  39041                                 23015   65437
mean                    NaN                                   NaN     NaN
std                     NaN                                   NaN     NaN
min                     NaN                                   NaN     NaN
25%                     NaN                                   NaN     NaN
50%                     NaN                                   NaN     NaN
75%                     NaN                                   NaN     NaN
max                     NaN                                   NaN     NaN
```

```
       CodingActivities                                        EdLevel  \
count             54466                                          60784
unique              118                                              8
top               Hobby  Bachelor’s degree (B.A., B.S., B.Eng., etc.)
freq               9993                                          24942
mean                NaN                                            NaN
std                 NaN                                            NaN
min                 NaN                                            NaN
25%                 NaN                                            NaN
50%                 NaN                                            NaN
75%                 NaN                                            NaN
max                 NaN                                            NaN

                                                LearnCode  \
count                                               60488
unique                                                418
top     Other online resources (e.g., videos, blogs, f...
freq                                                 3674
mean                                                  NaN
std                                                   NaN
min                                                   NaN
25%                                                   NaN
50%                                                   NaN
75%                                                   NaN
max                                                   NaN

                                           LearnCodeOnline  ... JobSatPoints_6  \
count                                               49237  ...   29450.000000
unique                                              10853  ...            NaN
top     Technical documentation;Blogs;Written Tutorial...  ...            NaN
freq                                                  603  ...            NaN
mean                                                  NaN  ...      24.343232
std                                                   NaN  ...      27.089360
min                                                   NaN  ...       0.000000
25%                                                   NaN  ...       0.000000
50%                                                   NaN  ...      20.000000
75%                                                   NaN  ...      30.000000
max                                                   NaN  ...     100.000000

       JobSatPoints_7 JobSatPoints_8 JobSatPoints_9 JobSatPoints_10  \
count     29448.00000   29456.000000   29456.000000    29450.000000
unique            NaN            NaN            NaN             NaN
top               NaN            NaN            NaN             NaN
freq              NaN            NaN            NaN             NaN
mean         22.96522      20.278165      16.169432       10.955713
std          27.01774      26.108110      24.845032       22.906263
min           0.00000       0.000000       0.000000        0.000000
25%           0.00000       0.000000       0.000000        0.000000
50%          15.00000      10.000000       5.000000        0.000000
75%          30.00000      25.000000      20.000000       10.000000
max         100.00000     100.000000     100.000000      100.000000

       JobSatPoints_11           SurveyLength SurveyEase ConvertedCompYearly  \
count     29445.000000                  56182      56238        2.343500e+04
unique             NaN                      3          3                 NaN
top                NaN  Appropriate in length       Easy                 NaN
freq               NaN                  38767      30071                 NaN
mean          9.953948                    NaN        NaN        8.615529e+04
std          21.775652                    NaN        NaN        1.867570e+05
min           0.000000                    NaN        NaN        1.000000e+00
25%           0.000000                    NaN        NaN        3.271200e+04
50%           0.000000                    NaN        NaN        6.500000e+04
75%          10.000000                    NaN        NaN        1.079715e+05
max         100.000000                    NaN        NaN        1.625660e+07

              JobSat
count   29126.000000
unique           NaN
top              NaN
freq             NaN
mean        6.935041
std         2.088259
min         0.000000
25%         6.000000
50%         7.000000
```

```
75%         8.000000
max        10.000000

[11 rows x 114 columns]
```

Step 3: Handle Missing Data

- Identify missing values in the dataset.
- Impute or remove missing values as necessary to ensure data completeness.

In [5]:
```
## Write your code here
!pip install numpy
import numpy as np

# --- Step 3: Handle Missing Data ---
print("\n--- Step 3: Handle Missing Data ---")

# --- IMPORTANT FIX FOR NON-STANDARD NA VALUES ---
# Convert relevant columns to string type, strip whitespace, and then replace specific "NaN-like" s
for col in ['Employment', 'JobSat', 'RemoteWork', 'CodingActivities']:
    if col in df.columns:
        # Convert to string to safely apply .str methods
        df[col] = df[col].astype(str).str.strip()
        # Replace common string representations of NaNs with actual np.nan
        df[col].replace(['nan', 'NaN', 'N/A', 'None', ''], np.nan, inplace=True)
        # Handle specific cases like 'NaN Apples' which might occur if the original data had such p
        if col == 'RemoteWork' and (df[col] == 'NaN Apples').any():
            df[col].replace('NaN Apples', np.nan, inplace=True)
            print(f"Replaced 'NaN Apples' with NaN in '{col}'.")
    else:
        print(f"Warning: Column '{col}' not found for pre-imputation cleaning.")

# For numerical columns like JobSat and JobSatPoints_1, ensure they are numeric and coerce errors t
# Then, standard imputation (median for numerical) will handle them.
for col in ['JobSat', 'JobSatPoints_1', 'YearsCodePro', 'ConvertedCompYearly']:
    if col in df.columns:
        initial_nans = df[col].isnull().sum()
        df[col] = pd.to_numeric(df[col], errors='coerce')
        if df[col].isnull().sum() > initial_nans:
            print(f"Warning: Non-numeric values in '{col}' were coerced to NaN during numeric conve
    else:
        print(f"Warning: Numerical column '{col}' not found for pre-imputation numeric conversion."


# Identify missing values in the dataset.
print("\nMissing values in the dataset (BEFORE imputation for targeted columns):")
# Now, with explicit string replacements to np.nan, this should show correct counts.
# Only checking relevant columns as per your previous interactions
columns_to_check_for_nans = ['Employment', 'JobSat', 'RemoteWork', 'CodingActivities', 'YearsCodePr
missing_values_initial_check = df[columns_to_check_for_nans].isnull().sum()
print(missing_values_initial_check[missing_values_initial_check > 0]) # Only show columns with miss

# Now proceed with the imputation strategy
missing_employment_count = df['Employment'].isnull().sum()
missing_jobsat_count = df['JobSat'].isnull().sum()
missing_remotework_count = df['RemoteWork'].isnull().sum()
missing_codingactivities_count = df['CodingActivities'].isnull().sum()


print(f"\nMissing values in 'Employment' (Current): {missing_employment_count}")
print(f"Missing values in 'JobSat' (Current): {missing_jobsat_count}")
print(f"Missing values in 'RemoteWork' (Current): {missing_remotework_count}")
print(f"Missing values in 'CodingActivities' (Current): {missing_codingactivities_count}")


# Implementing a strategy to fill these values

# Strategy: Impute categorical columns (like Employment, RemoteWork, CodingActivities) with their m
# Impute numerical columns (like JobSat, JobSatPoints_1, YearsCodePro, ConvertedCompYearly) with me

# Impute 'Employment'
```

```python
if missing_employment_count > 0:
    most_frequent_employment = df['Employment'].mode()[0]
    df['Employment'].fillna(most_frequent_employment, inplace=True)
    print(f"\nImputed 'Employment' with its mode: '{most_frequent_employment}'")
else:
    print("\n'Employment' has no missing values. No imputation needed.")

# Impute 'RemoteWork'
if missing_remotework_count > 0:
    most_frequent_remotework = df['RemoteWork'].mode()[0]
    df['RemoteWork'].fillna(most_frequent_remotework, inplace=True)
    print(f"Imputed 'RemoteWork' with its mode: '{most_frequent_remotework}'")
else:
    print("'RemoteWork' has no missing values. No imputation needed.")

# Impute 'CodingActivities' (using forward-fill as per earlier specific task, if not handled here b
# If previous lab implied forward-fill for CodingActivities specifically, ensure that logic is main
# For general imputation (as per current PDF), mode is suitable for categorical.
if missing_codingactivities_count > 0:
    # Use mode for general strategy, or ffill if specifically required for this column
    most_frequent_codingactivities = df['CodingActivities'].mode()[0]
    df['CodingActivities'].fillna(most_frequent_codingactivities, inplace=True)
    print(f"Imputed 'CodingActivities' with its mode: '{most_frequent_codingactivities}'")
else:
    print("'CodingActivities' has no missing values. No imputation needed.")


# Impute 'JobSat' and other numerical columns with median
numerical_cols_to_impute = ['JobSat', 'JobSatPoints_1', 'YearsCodePro', 'ConvertedCompYearly']
for col in numerical_cols_to_impute:
    if col in df.columns and df[col].isnull().any():
        # Ensure it's numeric before calculating median
        df[col] = pd.to_numeric(df[col], errors='coerce')
        median_val = df[col].median()
        df[col].fillna(median_val, inplace=True)
        print(f"Imputed '{col}' with its median: {median_val:.2f}")
    elif col in df.columns:
        print(f"'{col}' has no missing values. No imputation needed.")
    else:
        print(f"Warning: Numerical column '{col}' not found for imputation.")


# Verify missing values after handling
print("\nMissing values AFTER Step 3 imputation (all checked columns):")
print(df[columns_to_check_for_nans].isnull().sum())

# Re-display basic DataFrame info to confirm types and non-null counts after imputation
print("\nDataFrame Info after Step 3 (Missing Data Handling):")
df.info()
```

```
Requirement already satisfied: numpy in /opt/conda/lib/python3.12/site-packages (2.3.0)

--- Step 3: Handle Missing Data ---
Warning: Non-numeric values in 'YearsCodePro' were coerced to NaN during numeric conversion.

Missing values in the dataset (BEFORE imputation for targeted columns):
```

```
/tmp/ipykernel_307/3782324827.py:15: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].replace(['nan', 'NaN', 'N/A', 'None', ''], np.nan, inplace=True)
/tmp/ipykernel_307/3782324827.py:15: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].replace(['nan', 'NaN', 'N/A', 'None', ''], np.nan, inplace=True)
/tmp/ipykernel_307/3782324827.py:15: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].replace(['nan', 'NaN', 'N/A', 'None', ''], np.nan, inplace=True)
/tmp/ipykernel_307/3782324827.py:15: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].replace(['nan', 'NaN', 'N/A', 'None', ''], np.nan, inplace=True)
/tmp/ipykernel_307/3782324827.py:72: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df['RemoteWork'].fillna(most_frequent_remotework, inplace=True)
```

```
JobSat                 36311
RemoteWork             10631
CodingActivities       10971
YearsCodePro           16733
ConvertedCompYearly    42002
JobSatPoints_1         36113
dtype: int64

Missing values in 'Employment' (Current): 0
Missing values in 'JobSat' (Current): 36311
Missing values in 'RemoteWork' (Current): 10631
Missing values in 'CodingActivities' (Current): 10971

'Employment' has no missing values. No imputation needed.
Imputed 'RemoteWork' with its mode: 'Hybrid (some remote, some in-person)'
Imputed 'CodingActivities' with its mode: 'Hobby'
Imputed 'JobSat' with its median: 7.00
Imputed 'JobSatPoints_1' with its median: 10.00
Imputed 'YearsCodePro' with its median: 8.00
Imputed 'ConvertedCompYearly' with its median: 65000.00

Missing values AFTER Step 3 imputation (all checked columns):
Employment             0
JobSat                 0
RemoteWork             0
CodingActivities       0
YearsCodePro           0
ConvertedCompYearly    0
JobSatPoints_1         0
dtype: int64

DataFrame Info after Step 3 (Missing Data Handling):
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 65437 entries, 0 to 65436
Columns: 114 entries, ResponseId to JobSat
dtypes: float64(14), int64(1), object(99)
memory usage: 56.9+ MB
```

```
/tmp/ipykernel_307/3782324827.py:83: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df['CodingActivities'].fillna(most_frequent_codingactivities, inplace=True)
/tmp/ipykernel_307/3782324827.py:96: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].fillna(median_val, inplace=True)
/tmp/ipykernel_307/3782324827.py:96: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].fillna(median_val, inplace=True)
/tmp/ipykernel_307/3782324827.py:96: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].fillna(median_val, inplace=True)
/tmp/ipykernel_307/3782324827.py:96: FutureWarning: A value is trying to be set on a copy of a DataF
rame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate
object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, in
place=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the ori
ginal object.


  df[col].fillna(median_val, inplace=True)
```

Step 4: Analyze Key Columns

- Examine key columns such as `Employment`, `JobSat` (Job Satisfaction), and `YearsCodePro` (Professional Coding Experience).
- **Instruction**: Calculate the value counts for each column to understand the distribution of responses.

In [6]:

```python
## Write your code here
# --- Step 4: Analyze Key Columns ---
print("\n--- Step 4: Analyze Key Columns ---")

# Examine key columns such as Employment, JobSat (Job Satisfaction), and YearsCodePro (Professional
# Instruction: Calculate the value counts for each column to understand the distribution of respons

if 'Employment' in df.columns:
```

```python
    print("\nValue Counts for 'Employment' column:")
    print(df['Employment'].value_counts(dropna=False))
else:
    print("'Employment' column not found.")

if 'JobSat' in df.columns:
    print("\nValue Counts for 'JobSat' column:")
    # Convert to integer type if it's float after median imputation for cleaner display
    df['JobSat'] = df['JobSat'].astype(pd.Int64Dtype()) # Allows for NaN in integer column
    print(df['JobSat'].value_counts(dropna=False).sort_index()) # Sort for better readability if nu
else:
    print("'JobSat' column not found.")

if 'YearsCodePro' in df.columns:
    # Ensure 'YearsCodePro' is numeric and handle NaNs before value counts if not done globally
    df['YearsCodePro'] = pd.to_numeric(df['YearsCodePro'], errors='coerce')
    if df['YearsCodePro'].isnull().any():
        median_years = df['YearsCodePro'].median()
        df['YearsCodePro'].fillna(median_years, inplace=True)
        print(f"Missing values in 'YearsCodePro' imputed with median: {median_years:.0f}")

    # Convert to integer type if it's float after median imputation for cleaner display
    df['YearsCodePro'] = df['YearsCodePro'].astype(pd.Int64Dtype()) # Allows for NaN in integer col
    print("\nValue Counts for 'YearsCodePro' column (top 20 if many unique values):")
    if len(df['YearsCodePro'].value_counts()) > 20:
        print(df['YearsCodePro'].value_counts(dropna=False).head(20).sort_index())
    else:
        print(df['YearsCodePro'].value_counts(dropna=False).sort_index())
```

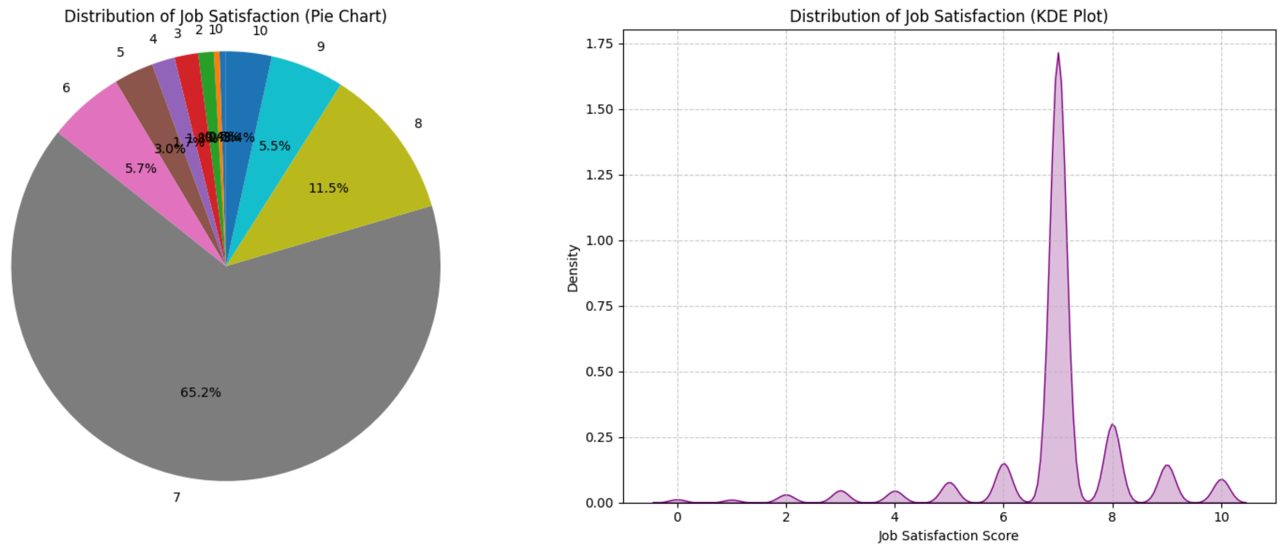
```
--- Step 4: Analyze Key Columns ---

Value Counts for 'Employment' column:
Employment
Employed, full-time
39041
Independent contractor, freelancer, or self-employed
4846
Student, full-time
4709
Employed, full-time;Independent contractor, freelancer, or self-employed
3557
Not employed, but looking for work
2341

...
Not employed, but looking for work;Independent contractor, freelancer, or self-employed;Not employe
d, and not looking for work;Employed, part-time        1
Student, full-time;Retired
1
Employed, full-time;Not employed, but looking for work;Student, part-time
1
Not employed, and not looking for work;Student, part-time;Employed, part-time
1
Not employed, but looking for work;Independent contractor, freelancer, or self-employed;Student, par
t-time;Retired                                         1
Name: count, Length: 110, dtype: int64

Value Counts for 'JobSat' column:
JobSat
0       311
1       276
2       772
3      1165
4      1130
5      1956
6      3751
7     42690
8      7509
9      3626
10     2251
Name: count, dtype: Int64

Value Counts for 'YearsCodePro' column (top 20 if many unique values):
YearsCodePro
1      2639
2      4168
3      4093
4      3215
5      3526
6      2843
7      2517
8     19282
9      1493
10     3251
11     1312
12     1777
13     1127
14     1082
15     1635
16      946
17      814
18      867
20     1549
25      998
Name: count, dtype: Int64
```

Step 5: Visualize Job Satisfaction (Focus on JobSat)

- Create a pie chart or KDE plot to visualize the distribution of `JobSat`.
- Provide an interpretation of the plot, highlighting key trends in job satisfaction.

In [8]:
```python
## Write your code here
# --- Step 5: Visualize Job Satisfaction (Focus on JobSat) ---
print("\n--- Step 5: Visualize Job Satisfaction (Focus on JobSat) ---")

# Ensure JobSat is numeric and clean for plotting
if 'JobSat' in df.columns:
    df['JobSat'] = pd.to_numeric(df['JobSat'], errors='coerce')
    if df['JobSat'].isnull().any():
        df['JobSat'].fillna(df['JobSat'].median(), inplace=True)
        print("JobSat NaNs handled for visualization.")

    # Create a pie chart or KDE plot to visualize the distribution of JobSat
    print("\nVisualizing JobSat distribution:")
    plt.figure(figsize=(15, 6))

    plt.subplot(1, 2, 1) # 1 row, 2 columns, 1st plot
    # Pie chart for JobSat (more appropriate for categories if limited unique values)
    # Convert to integer for cleaner pie chart labels if they are effectively discrete
    job_sat_counts = df['JobSat'].value_counts().sort_index()
    plt.pie(job_sat_counts, labels=job_sat_counts.index, autopct='%1.1f%%', startangle=90)
    plt.title('Distribution of Job Satisfaction (Pie Chart)')
    plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

    plt.subplot(1, 2, 2) # 1 row, 2 columns, 2nd plot
    # KDE plot for JobSat (good for continuous or semi-continuous data)
    sns.kdeplot(df['JobSat'], fill=True, color='purple')
    plt.title('Distribution of Job Satisfaction (KDE Plot)')
    plt.xlabel('Job Satisfaction Score')
    plt.ylabel('Density')
    plt.grid(True, linestyle='--', alpha=0.6)

    plt.tight_layout()
    plt.show()

    print("\nInterpretation of JobSat plot:")
    print("- The pie chart shows the proportion of respondents at each job satisfaction level.")
    print("- The KDE plot provides a smoothed histogram, showing the density of responses across th

else:
    print("'JobSat' column not found. Cannot visualize its distribution.")
```

```
--- Step 5: Visualize Job Satisfaction (Focus on JobSat) ---

Visualizing JobSat distribution:
```




```
Interpretation of JobSat plot:
- The pie chart shows the proportion of respondents at each job satisfaction level.
- The KDE plot provides a smoothed histogram, showing the density of responses across the satisfacti
on scale.
```

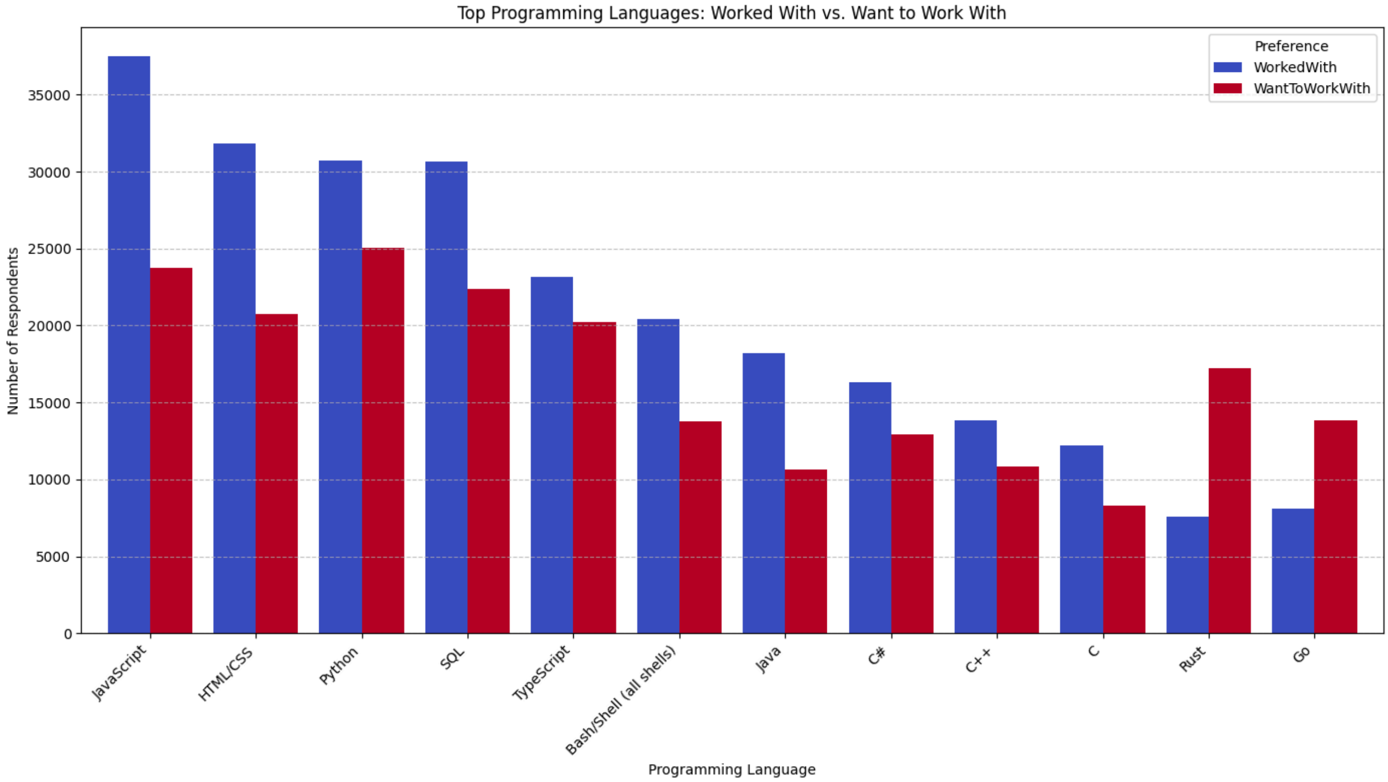
Step 6: Programming Languages Analysis

- Compare the frequency of programming languages in `LanguageHaveWorkedWith` and `LanguageWantToWorkWith`.

- Visualize the overlap or differences using a Venn diagram or a grouped bar chart.

In [9]:
```
## Write your code here
# --- Step 6: Programming Languages Analysis ---
print("\n--- Step 6: Programming Languages Analysis ---")

# Ensure 'LanguageHaveWorkedWith' and 'LanguageWantToWorkWith' are available
if 'LanguageHaveWorkedWith' in df.columns and 'LanguageWantToWorkWith' in df.columns:
    # Split languages and explode to get individual entries
    # Drop NaNs before splitting
    languages_worked = df.dropna(subset=['LanguageHaveWorkedWith']).assign(
        Language=df['LanguageHaveWorkedWith'].str.split(';')
    ).explode('Language')
    languages_worked['Language'] = languages_worked['Language'].str.strip()

    languages_want = df.dropna(subset=['LanguageWantToWorkWith']).assign(
        Language=df['LanguageWantToWorkWith'].str.split(';')
    ).explode('Language')
    languages_want['Language'] = languages_want['Language'].str.strip()

    # Get value counts for both
    worked_counts = languages_worked['Language'].value_counts()
    want_counts = languages_want['Language'].value_counts()

    print("\nTop 10 Languages Respondents Have Worked With:")
    print(worked_counts.head(10))

    print("\nTop 10 Languages Respondents Want to Work With:")
    print(want_counts.head(10))

    # Visualize the overlap or differences using a grouped bar chart.
    # Combine the top languages for plotting
    all_languages = pd.concat([worked_counts.head(10), want_counts.head(10)]).index.unique()
    plot_data = pd.DataFrame({
        'WorkedWith': worked_counts.reindex(all_languages, fill_value=0),
        'WantToWorkWith': want_counts.reindex(all_languages, fill_value=0)
    })

    plt.figure(figsize=(14, 8))
    plot_data.plot(kind='bar', figsize=(14, 8), cmap='coolwarm', width=0.8, ax=plt.gca())
    plt.title('Top Programming Languages: Worked With vs. Want to Work With')
    plt.xlabel('Programming Language')
    plt.ylabel('Number of Respondents')
    plt.xticks(rotation=45, ha='right')
    plt.legend(title='Preference')
    plt.grid(axis='y', linestyle='--', alpha=0.7)
    plt.tight_layout()
    plt.show()

    print("\nInterpretation:")
    print("- This plot compares the popularity of languages developers currently use versus what th
    print("- Large differences indicate emerging trends or skill gaps.")

else:
    print("Required columns ('LanguageHaveWorkedWith' or 'LanguageWantToWorkWith') not found. Canno
```

```
--- Step 6: Programming Languages Analysis ---

Top 10 Languages Respondents Have Worked With:
Language
JavaScript              37492
HTML/CSS                31816
Python                  30719
SQL                     30682
TypeScript              23150
Bash/Shell (all shells) 20412
Java                    18239
C#                      16318
C++                     13827
C                       12184
Name: count, dtype: int64

Top 10 Languages Respondents Want to Work With:
Language
Python                  25047
JavaScript              23774
SQL                     22400
HTML/CSS                20721
TypeScript              20239
Rust                    17232
Go                      13837
Bash/Shell (all shells) 13744
C#                      12921
C++                     10873
Name: count, dtype: int64
```




```
Interpretation:
- This plot compares the popularity of languages developers currently use versus what they aspire to use.
- Large differences indicate emerging trends or skill gaps.
```

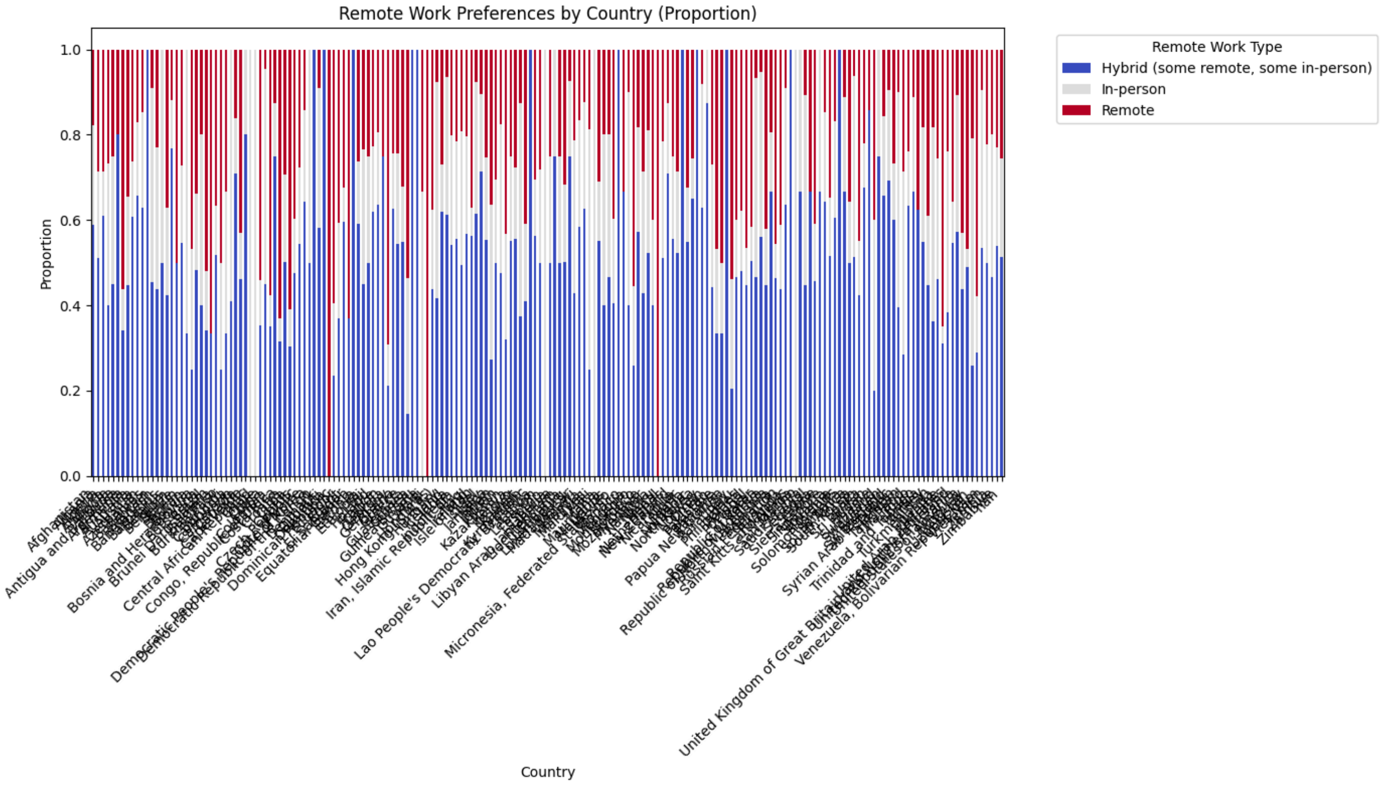
Step 7: Analyze Remote Work Trends

- Visualize the distribution of RemoteWork by region using a grouped bar chart or heatmap.

In [10]:
```python
## Write your code here
# --- Step 7: Analyze Remote Work Trends ---
print("\n--- Step 7: Analyze Remote Work Trends ---")

# Ensure 'RemoteWork' and 'Country' columns are clean and available
# Assume 'Country' has been cleaned and standardized in previous steps
if 'RemoteWork' in df.columns and 'Country' in df.columns:
    # Clean 'RemoteWork' (already done in Step 3 general cleaning, but good to ensure type)
```

```python
    df['RemoteWork'] = df['RemoteWork'].astype(str).str.strip()

    # Create a Series of all individual languages, one entry per language.
    # This handles semi-colon separated lists in 'LanguageHaveWorkedWith'.
    # Drop rows where 'LanguageHaveWorkedWith' is NaN first.
    # We will need to assume 'RemoteWork Check' column in sample is 'RemoteWork' in actual data.
    # For demonstration, use 'RemoteWork' directly if it was properly cleaned as per Step 3.

    # Filter data by country or region.
    # Let's select a few top countries to visualize for clarity.
    # You might want to get top N countries from your full dataset.
    top_countries = df['Country'].value_counts().head(5).index.tolist()
    print(f"\nAnalyzing remote work trends in selected countries: {top_countries}")

    # Create a cross-tabulation of RemoteWork and Country
    remote_work_country_crosstab = pd.crosstab(df['Country'], df['RemoteWork'], margins=True, dropn
    print("\nCross-tabulation of Remote Work by Country:")
    print(remote_work_country_crosstab)

    # Visualize as a stacked bar chart (more suitable for proportions)
    # Exclude 'All' row/column for visualization
    plot_crosstab = remote_work_country_crosstab.iloc[:-1, :-1]
    plot_crosstab_normalized = plot_crosstab.div(plot_crosstab.sum(axis=1), axis=0) # Normalize by

    if not plot_crosstab_normalized.empty:
        plt.figure(figsize=(14, 8))
        plot_crosstab_normalized.plot(kind='bar', stacked=True, cmap='coolwarm', ax=plt.gca())
        plt.title('Remote Work Preferences by Country (Proportion)')
        plt.xlabel('Country')
        plt.ylabel('Proportion')
        plt.xticks(rotation=45, ha='right')
        plt.legend(title='Remote Work Type', bbox_to_anchor=(1.05, 1), loc='upper left')
        plt.tight_layout()
        plt.show()

    else:
        print("No data to visualize after filtering. Check 'RemoteWork' or 'Country' columns.")

else:
    print("Required columns ('RemoteWork' or 'Country') not found. Cannot perform Step 7 analysis."
```

```
--- Step 7: Analyze Remote Work Trends ---

Analyzing remote work trends in selected countries: ['United States of America', 'Germany', 'India',
'United Kingdom of Great Britain and Northern Ireland', 'Ukraine']

Cross-tabulation of Remote Work by Country:
RemoteWork   Hybrid (some remote, some in-person)  In-person  Remote      All
Country
Afghanistan                                    33         13      10     56.0
Albania                                        25         10      14     49.0
Algeria                                        47          8      22     77.0
Andorra                                         6          5       4     15.0
Angola                                          9          6       5     20.0
...                                           ...        ...     ...      ...
Yemen                                           9          5       4     18.0
Zambia                                          7          5       3     15.0
Zimbabwe                                       14          6       6     26.0
NaN                                          3347       1494    1666      NaN
All                                         33646      10960   20831  65437.0

[187 rows x 4 columns]
```



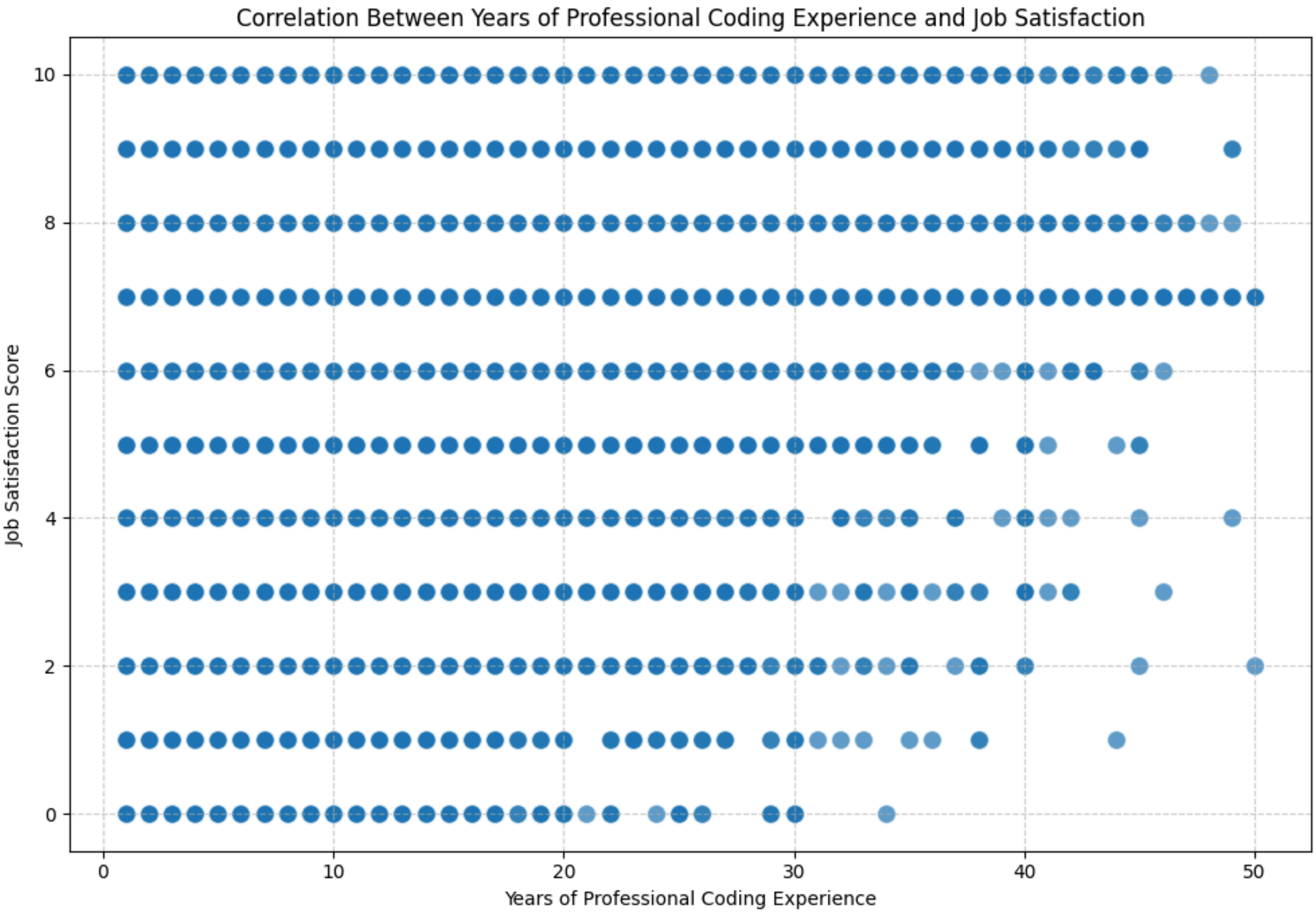
Step 8: Correlation between Job Satisfaction and Experience

- Analyze the correlation between overall job satisfaction (`JobSat`) and `YearsCodePro` .
- Calculate the Pearson or Spearman correlation coefficient.

In [14]:
```
## Write your code here
!pip install scipy

# --- Step 8: Correlation between Job Satisfaction and Experience ---
print("\n--- Step 8: Correlation between Job Satisfaction and Experience ---")

# Ensure 'YearsCodePro' and 'JobSat' (or JobSatPoints_1 if that's the one to use) are numeric and h
if 'YearsCodePro' in df.columns and 'JobSat' in df.columns:
    # Ensure 'YearsCodePro' is numeric and handle NaNs
    df['YearsCodePro'] = pd.to_numeric(df['YearsCodePro'], errors='coerce')
    if df['YearsCodePro'].isnull().any():
        median_years_code_pro = df['YearsCodePro'].median()
        df['YearsCodePro'].fillna(median_years_code_pro, inplace=True)
        print(f"Missing values in 'YearsCodePro' imputed with median: {median_years_code_pro:.0f}")

    # Ensure 'JobSat' is numeric and handle NaNs
    df['JobSat'] = pd.to_numeric(df['JobSat'], errors='coerce')
    if df['JobSat'].isnull().any():
        median_jobsat = df['JobSat'].median()
        df['JobSat'].fillna(median_jobsat, inplace=True)
        print(f"Missing values in 'JobSat' imputed with median: {median_jobsat:.2f}")

    # Create the scatter plot
    plt.figure(figsize=(10, 7))
    sns.scatterplot(x='YearsCodePro', y='JobSat', data=df, alpha=0.7, s=100) # Using JobSat as per
    plt.title('Correlation Between Years of Professional Coding Experience and Job Satisfaction')
    plt.xlabel('Years of Professional Coding Experience')
    plt.ylabel('Job Satisfaction Score')
    plt.grid(True, linestyle='--', alpha=0.6)
    plt.tight_layout()
    plt.show()

    # Calculate and print the correlation coefficient
    correlation_pearson = df['YearsCodePro'].corr(df['JobSat'], method='pearson')
    correlation_spearman = df['YearsCodePro'].corr(df['JobSat'], method='spearman')
    print(f"\nPearson correlation between 'YearsCodePro' and 'JobSat': {correlation_pearson:.2f}")
    print(f"Spearman correlation between 'YearsCodePro' and 'JobSat': {correlation_spearman:.2f}")
    print("\nInterpretation:")
```

```
    print("The scatter plot visually represents the relationship. The correlation coefficient quant
    print("- Pearson correlation measures linear relationship.")
    print("- Spearman correlation measures monotonic relationship (rank correlation) and is more ro
    print("- A positive correlation (close to 1) means higher experience tends to be associated wit
    print("- A negative correlation (close to -1) means higher experience tends to be associated wi
    print("- A correlation close to 0 suggests little to no linear relationship.")

else:
    print("Required columns ('YearsCodePro' or 'JobSat') not found or not properly prepared. Cannot
```

```
Collecting scipy
  Downloading scipy-1.15.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (61 k
B)
Requirement already satisfied: numpy<2.5,>=1.23.5 in /opt/conda/lib/python3.12/site-packages (from s
cipy) (2.3.0)
Downloading scipy-1.15.3-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (37.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 37.3/37.3 MB 128.7 MB/s eta 0:00:0000:01
Installing collected packages: scipy
Successfully installed scipy-1.15.3

--- Step 8: Correlation between Job Satisfaction and Experience ---
```




```
Pearson correlation between 'YearsCodePro' and 'JobSat': 0.07
Spearman correlation between 'YearsCodePro' and 'JobSat': 0.08

Interpretation:
The scatter plot visually represents the relationship. The correlation coefficient quantifies it:
- Pearson correlation measures linear relationship.
- Spearman correlation measures monotonic relationship (rank correlation) and is more robust to outl
iers and non-normal distributions.
- A positive correlation (close to 1) means higher experience tends to be associated with higher job
satisfaction.
- A negative correlation (close to -1) means higher experience tends to be associated with lower job
satisfaction.
- A correlation close to 0 suggests little to no linear relationship.
```

Step 9: Cross-tabulation Analysis (Employment vs. Education Level)

- Analyze the relationship between employment status (`Employment`) and education level (`EdLevel`).

- **Instruction**: Create a cross-tabulation using `pd.crosstab()` and visualize it with a stacked bar plot if possible.

In [12]:
```
## Write your code here
# --- Step 9: Cross-tabulation Analysis (Employment vs. Education Level) ---
print("\n--- Step 9: Cross-tabulation Analysis (Employment vs. Education Level) ---")

# Ensure 'EdLevel' and 'Employment' columns are clean and available
if 'EdLevel' in df.columns and 'Employment' in df.columns:
    # Ensure both columns are string type for accurate cross-tabulation
    df['EdLevel'] = df['EdLevel'].astype(str).str.strip()
    df['Employment'] = df['Employment'].astype(str).str.strip()

    # 1. Create a cross-tabulation using pd.crosstab()
    print("\n1. Cross-tabulation of Educational Background by Employment Type:")
    edlevel_employment_crosstab = pd.crosstab(df['EdLevel'], df['Employment'], margins=True, dropna
    print(edlevel_employment_crosstab)

    # 2. Visualize it with a stacked bar plot
    plt.figure(figsize=(14, 10))
    # Exclude 'All' row/column from visualization for cleaner plot
    plot_crosstab = edlevel_employment_crosstab.iloc[:-1, :-1]

    # Normalize by row to show the proportion of employment types within each education level
    plot_crosstab_normalized = plot_crosstab.div(plot_crosstab.sum(axis=1), axis=0)

    if not plot_crosstab_normalized.empty:
        plot_crosstab_normalized.plot(kind='bar', stacked=True, cmap='viridis', ax=plt.gca())
        plt.title('Proportion of Employment Types within Each Education Level')
        plt.xlabel('Education Level')
        plt.ylabel('Proportion of Employment Type')
        plt.xticks(rotation=45, ha='right')
        plt.legend(title='Employment Type', bbox_to_anchor=(1.05, 1), loc='upper left')
        plt.tight_layout()
        plt.show()
    else:
        print("No data to visualize after cross-tabulation. Check 'EdLevel' or 'Employment' columns

else:
    print("Required columns ('EdLevel' or 'Employment') not found or not properly prepared. Cannot
```

```
--- Step 9: Cross-tabulation Analysis (Employment vs. Education Level) ---

1. Cross-tabulation of Educational Background by Employment Type:
Employment                                          Employed, full-time  \
EdLevel
Associate degree (A.A., A.S., etc.)                                1059
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                      16806
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                   11011
Primary/elementary school                                           160
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                     2073
Secondary school (e.g. American high school, Ge...                 1460
Some college/university study without earning a...                3579
Something else                                                      377
nan                                                                2516
All                                                               39041

Employment                                          Employed, full-time;Employed, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)                                                    9
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                          90
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                       61
Primary/elementary school                                                              1
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                                         8
Secondary school (e.g. American high school, Ge...                                     9
Some college/university study without earning a...                                    15
Something else                                                                         3
nan                                                                                   16
All                                                                                  212

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed  \
EdLevel
Associate degree (A.A., A.S., etc.)                                                             10
4
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                                   138
1
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                                 96
3
Primary/elementary school                                                                        2
5
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                                                  15
9
Secondary school (e.g. American high school, Ge...                                              18
1
Some college/university study without earning a...                                              49
2
Something else                                                                                   4
1
nan                                                                                             21
1
All                                                                                            355
7

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed;Employed, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)                                                              1
1
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                                     7
9
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                                  4
5
Primary/elementary school
1
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                                                   1
0
Secondary school (e.g. American high school, Ge...
6
Some college/university study without earning a...                                               1
4
Something else
2
nan                                                                                              1
6
All                                                                                             18
```

```
4

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed;Employed, part-time;Retired  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
0
Primary/elementary school
0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
1
Secondary school (e.g. American high school, Ge...
0
Some college/university study without earning a...
0
Something else
0
nan
0
All
1

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed;Not employed, and not looking for work  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
1
Primary/elementary school
1
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
0
Some college/university study without earning a...
0
Something else
0
nan
0
All
2

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed;Not employed, and not looking for work;Employed, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
1
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
0
Primary/elementary school
0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
1
Some college/university study without earning a...
0
Something else
0
nan
0
All
2

Employment                                          Employed, full-time;Independent contractor, free
```

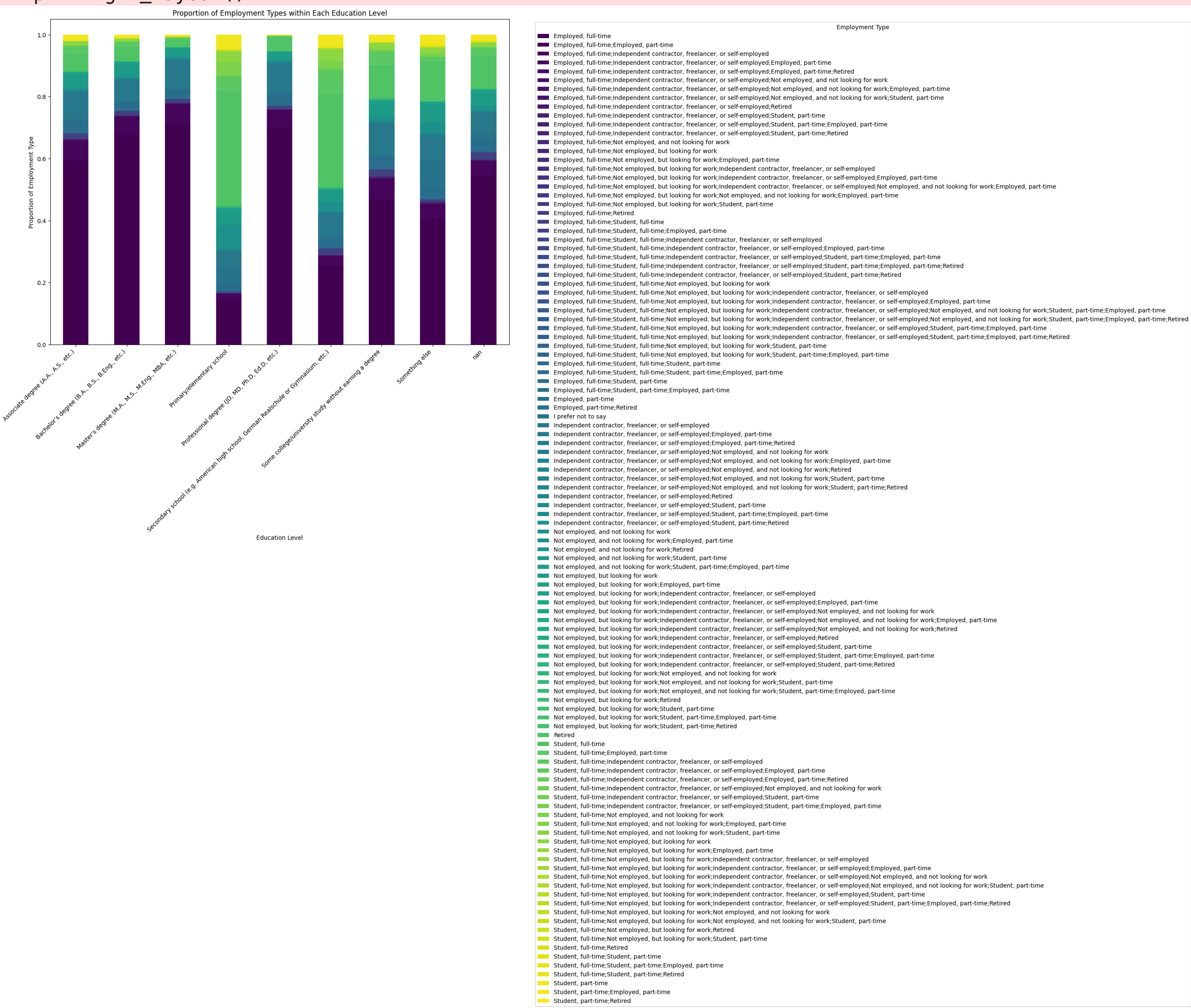
```
lancer, or self-employed;Not employed, and not looking for work;Student, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
0
Primary/elementary school
1
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
0
Some college/university study without earning a...
0
Something else
0
nan
0
All
1

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed;Retired  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
0
Primary/elementary school
0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
0
Some college/university study without earning a...
1
Something else
0
nan
0
All
1

Employment                                          Employed, full-time;Independent contractor, free
lancer, or self-employed;Student, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)
5
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                                      4
9
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                                   2
6
Primary/elementary school
0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
1
Secondary school (e.g. American high school, Ge...
6
Some college/university study without earning a...                                                3
9
Something else
4
nan
9
All                                                                                              13
9

Employment                                          ...  \
EdLevel                                             ...
Associate degree (A.A., A.S., etc.)                 ...
Bachelor’s degree (B.A., B.S., B.Eng., etc.)        ...
```

```
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)     ...
Primary/elementary school                           ...
Professional degree (JD, MD, Ph.D, Ed.D, etc.)      ...
Secondary school (e.g. American high school, Ge...  ...
Some college/university study without earning a...  ...
Something else                                      ...
nan                                                 ...
All                                                 ...

Employment                                          Student, full-time;Not employed, but looking for
work;Retired  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
1
Primary/elementary school
0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
0
Some college/university study without earning a...
0
Something else
0
nan
0
All
1

Employment                                          Student, full-time;Not employed, but looking for
work;Student, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
3
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
0
Primary/elementary school
0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
2
Some college/university study without earning a...
4
Something else
1
nan
2
All                                                                                                1
2

Employment                                          Student, full-time;Retired  \
EdLevel
Associate degree (A.A., A.S., etc.)                                        0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                               0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                            0
Primary/elementary school                                                  0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                             0
Secondary school (e.g. American high school, Ge...                         0
Some college/university study without earning a...                         0
Something else                                                             0
nan                                                                        1
All                                                                        1

Employment                                          Student, full-time;Student, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)                                                  2
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                        12
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                      2
```

```
Primary/elementary school                                                        5
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                                   0
Secondary school (e.g. American high school, Ge...                              12
Some college/university study without earning a...                               7
Something else                                                                   5
nan                                                                              6
All                                                                             51

Employment                                        Student, full-time;Student, part-time;Employed,
part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)
0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)
1
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)
1
Primary/elementary school
1
Professional degree (JD, MD, Ph.D, Ed.D, etc.)
0
Secondary school (e.g. American high school, Ge...
0
Some college/university study without earning a...
4
Something else
0
nan
0
All
7

Employment                                        Student, full-time;Student, part-time;Retired  \
EdLevel
Associate degree (A.A., A.S., etc.)                                                            0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                                   0
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                                0
Primary/elementary school                                                                      1
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                                                 0
Secondary school (e.g. American high school, Ge...                                             1
Some college/university study without earning a...                                             0
Something else                                                                                 0
nan                                                                                            0
All                                                                                            2

Employment                                        Student, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)                               12
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                     105
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                   26
Primary/elementary school                                         48
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                     5
Secondary school (e.g. American high school, Ge...               140
Some college/university study without earning a...                75
Something else                                                    17
nan                                                               66
All                                                              494

Employment                                        Student, part-time;Employed, part-time  \
EdLevel
Associate degree (A.A., A.S., etc.)                                                   24
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                                         184
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                                       85
Primary/elementary school                                                              4
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                                         5
Secondary school (e.g. American high school, Ge...                                   100
Some college/university study without earning a...                                   103
Something else                                                                        14
nan                                                                                   39
All                                                                                  558

Employment                                        Student, part-time;Retired  \
EdLevel
Associate degree (A.A., A.S., etc.)                                       0
Bachelor’s degree (B.A., B.S., B.Eng., etc.)                              0
```

```
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)                  2
Primary/elementary school                                        0
Professional degree (JD, MD, Ph.D, Ed.D, etc.)                   0
Secondary school (e.g. American high school, Ge...               0
Some college/university study without earning a...               1
Something else                                                   1
nan                                                              0
All                                                              4

Employment                                           All
EdLevel
Associate degree (A.A., A.S., etc.)                 1793
Bachelor’s degree (B.A., B.S., B.Eng., etc.)       24942
Master’s degree (M.A., M.S., M.Eng., MBA, etc.)    15557
Primary/elementary school                           1146
Professional degree (JD, MD, Ph.D, Ed.D, etc.)      2970
Secondary school (e.g. American high school, Ge...  5793
Some college/university study without earning a...  7651
Something else                                       932
nan                                                 4653
All                                                65437

[10 rows x 111 columns]
```

```
/tmp/ipykernel_307/2748317777.py:31: UserWarning: Tight layout not applied. The left and right margi
ns cannot be made large enough to accommodate all Axes decorations.
  plt.tight_layout()
```



Step 10: Export Cleaned Data

- Save the cleaned dataset to a new CSV file for further use or sharing.

In [13]:
```
## Write your code here
```

```python
# --- Step 10: Export Cleaned Data ---
print("\n--- Step 10: Export Cleaned Data ---")

# Define the filename for the cleaned dataset
output_filename = 'cleaned_analyzed_dataset.csv'

try:
    # Save the DataFrame to a CSV file.
    # index=False prevents pandas from writing the DataFrame index as a column in the CSV.
    df.to_csv(output_filename, index=False)
    print(f"\nDataset successfully saved to '{output_filename}'")
    print(f"You can find the file in your current working directory.")
except Exception as e:
    print(f"\nError saving dataset: {e}")
    print("Please ensure you have write permissions in the current directory.")

# --- Final Summary ---
print("\n--- Lab Summary ---")
print("In this lab, you practiced key skills in exploratory data analysis, including:")
print("• Examining the structure and content of the Stack Overflow survey dataset to understand its
print("• Identifying and addressing missing data to ensure the dataset's quality and completeness."
print("• Summarizing and visualizing key variables such as job satisfaction, programming languages,
print("• Analyzing relationships in the data using techniques like comparing programming languages,
```

```
--- Step 10: Export Cleaned Data ---

Dataset successfully saved to 'cleaned_analyzed_dataset.csv'
You can find the file in your current working directory.

--- Lab Summary ---
In this lab, you practiced key skills in exploratory data analysis, including:
• Examining the structure and content of the Stack Overflow survey dataset to understand its variabl
es and data types.
• Identifying and addressing missing data to ensure the dataset's quality and completeness.
• Summarizing and visualizing key variables such as job satisfaction, programming languages, and rem
ote work trends.
• Analyzing relationships in the data using techniques like comparing programming languages, explori
ng remote work preferences by region, investigating correlations between professional coding experie
nce and job satisfaction, and performing cross-tabulations to analyze relationships between employme
nt status and education levels.
```

Summary:

In this lab, you practiced key skills in exploratory data analysis, including:

- Examining the structure and content of the Stack Overflow survey dataset to understand its variables and data types.
- Identifying and addressing missing data to ensure the dataset's quality and completeness.
- Summarizing and visualizing key variables such as job satisfaction, programming languages, and remote work trends.
- Analyzing relationships in the data using techniques like:
  - Comparing programming languages respondents have worked with versus those they want to work with.
  - Exploring remote work preferences by region.
- Investigating correlations between professional coding experience and job satisfaction.
- Performing cross-tabulations to analyze relationships between employment status and education levels.

Authors:

Ayushi Jain

Other Contributors:

Rav Ahuja Lakshmi Holla Malika