hr_cleaning

November 29, 2025

Project: HR Data Cleaning and Standardization Objective: To prepare a raw, "messy" employee dataset for analysis by identifying errors, unifying data formats, and handling missing values. Scope of Work: The project involved a comprehensive data wrangling process, including: parsing and formatting inconsistent date strings, text cleaning (correcting typos in departments, skills, and job titles), standardizing phone numbers and SSNs using Regular Expressions (Regex), converting salary information into a numerical format, splitting composite attributes (e.g., Location, Full Name) into distinct columns.

Data loading and initial inspection.

[270]:
```
import pandas as pd
from datetime import datetime

df = pd.read_csv("messy_hr_data.csv")
df.head()
```

[270]:

	Employee_ID	Full_Name	SSN	Department	Job_Title	Salary	Joining_Date	Education	Skills	Performance_Rating	Email	Phone	Location
0	EMP-0926	Linda Jones	432-65-4227	Engineering	Officer - Manager	78k	2022-09-21	Master	Java,SQL	3	linda.jones@company.com	217.220.4686	
1	EMP-0909	James Davis	966429736	Finance	Consultant - Mid	$171,000	2019-12-26	B.Sc	SQL;Sales	4	james.davis@company.com	(370) 651-7833	
2	EMP-0933	James Davis	437 27 3267	IT Support	Officer - Mid	44000	2019-11-25	PhD	Java	3	james.davis@company.com	(557) 302-3186	
3	EMP-1263	Elizabeth Brown	739127077	IT Support	Lead Consultant	112k	2023-09-06	Bachelor	Sales | Java | Excell	4	elizabeth.brown@company.com	568-361-7994	
4	EMP-1402	James Johnson	691-79-3614	Marketing	Manager Developer	85000	2022-07-27	NaN	Excel | SQL | Management	4	james.johnson@company.com	(374) 314-5590	

```
0         Austin, TX
1             Remote
2  San Francisco, CA
3         Austin, TX
4  San Francisco, CA
```

[271]:
```
df.info
```

[271]:
```
<bound method DataFrame.info of      Employee_ID        Full_Name          SSN
Department  \
0       EMP-0926      Linda Jones  432-65-4227  Engineering
1       EMP-0909      James Davis    966429736      Finance
2       EMP-0933      James Davis  437 27 3267   IT Support
3       EMP-1263  Elizabeth Brown    739127077   IT Support
4       EMP-1402    James Johnson  691-79-3614    Marketing
…          …              …           …           …
2550    EMP-1269   Patricia Smith  764-57-7396   IT Support
2551    EMP-0523  Elizabeth Davis  643-15-2510      Finance
2552    EMP-2327   Patricia Jones  193-10-6811      Finance
2553    EMP-2153    Michael Jones    495633318    Marketing
2554    EMP-2028   Michael Garcia    713387016        Legal

                Job_Title      Salary Joining_Date    Education  \
0      Officer - Manager         78k   2022-09-21       Master
1       Consultant - Mid    $171,000   2019-12-26         B.Sc
2          Officer - Mid       44000   2019-11-25          PhD
3        Lead Consultant        112k   2023-09-06     Bachelor
4      Manager Developer       85000   2022-07-27          NaN
…                    …         …           …           …
2550      Mid Consultant      126000          NaN        Ph.D.
2551  Developer - Senior   79000 USD   2018-02-11       Master
2552  Developer - Junior         61k   10/08/2019          PhD
2553   Senior Consultant  138000 USD          NaN  High School
2554      Senior Analyst       94000   2023-05-28     Bachelor

                          Skills Performance_Rating  \
0                       Java,SQL                 3
1                      SQL;Sales                 4
2                           Java                 3
3             Sales | Java | Excell              4
4          Excel | SQL | Management              4
…                             …               …
2550  Power BI/Communication/Java                5
2551            Excel/Python/Sales               B
2552         SQL, Python, Power BI               3
2553    Python | Excell | Power BI               A
2554                    Management               2
```

```
                            Email            Phone           Location
0           linda.jones@company.com     217.220.4686         Austin, TX
1           james.davis@company.com   (370) 651-7833             Remote
2           james.davis@company.com   (557) 302-3186  San Francisco, CA
3       elizabeth.brown@company.com     568-361-7994         Austin, TX
4         james.johnson@company.com   (374) 314-5590  San Francisco, CA
…                               …              …                …
2550    patricia.smith@company.com  +1 564 720 3280         London, UK
2551  elizabeth.davis@company.com   (420) 236-6494         Berlin, DE
2552    patricia.jones@company.com  +1 216 123 3270         Austin, TX
2553     michael.jones@company.com   (249) 321-3430             Remote
2554    michael.garcia@company.com  +1 583 296 5341         London, UK

[2555 rows x 13 columns]>
```

[272]:
```
df.describe()
```

[272]:
```
       Employee_ID      Full_Name          SSN Department        Job_Title  \
count         2540           2540         2424       2540             2540
unique        2500            154         2385         18               60
top       EMP-2188  Michael Brown  251-19-3311      Sales  Manager Analyst
freq             2             44            2        375               65

       Salary Joining_Date    Education         Skills Performance_Rating  \
count    2416         2436         2458           2540                2319
unique    602         1778           15           1057                   7
top     96000      Pending  High School  Communication                   4
freq       16          131          477            121                 509

                          Email         Phone    Location
count                      2540          2540        2540
unique                       80          2500           6
top     james.davis@company.com  388-389-7102  London, UK
freq                         46             2         471
```

Dropping duplicates.

[273]:
```
sum(df.duplicated())
df = df.drop_duplicates()
```

Reseting the index after dropping rows.

[274]:
```
df = df.set_axis(range(1, len(df.index) + 1))
df.head()
```

[274]:
```
  Employee_ID        Full_Name          SSN   Department          Job_Title  \
1    EMP-0926      Linda Jones  432-65-4227  Engineering  Officer - Manager
```

```
2    EMP-0909      James Davis    966429736      Finance   Consultant - Mid
3    EMP-0933      James Davis  437 27 3267   IT Support      Officer - Mid
4    EMP-1263  Elizabeth Brown    739127077   IT Support    Lead Consultant
5    EMP-1402    James Johnson  691-79-3614    Marketing  Manager Developer

     Salary Joining_Date Education                   Skills  \
1       78k   2022-09-21    Master                 Java,SQL
2  $171,000   2019-12-26      B.Sc                SQL;Sales
3     44000   2019-11-25       PhD                     Java
4      112k   2023-09-06  Bachelor     Sales | Java | Excell
5     85000   2022-07-27       NaN  Excel | SQL | Management

  Performance_Rating                         Email           Phone  \
1                  3      linda.jones@company.com    217.220.4686
2                  4      james.davis@company.com  (370) 651-7833
3                  3      james.davis@company.com  (557) 302-3186
4                  4  elizabeth.brown@company.com    568-361-7994
5                  4    james.johnson@company.com  (374) 314-5590

            Location
1         Austin, TX
2             Remote
3  San Francisco, CA
4         Austin, TX
5  San Francisco, CA
```

Checking data for null values.

[275]:
```
df.isna().sum()
```

[275]:
```
Employee_ID            1
Full_Name              1
SSN                  116
Department             1
Job_Title              1
Salary               123
Joining_Date         105
Education             81
Skills                 1
Performance_Rating   219
Email                  1
Phone                  1
Location               1
dtype: int64
```

Dropping data where “Full_Name” is missing.

[276]:
```
df = df.dropna(subset=["Full_Name"])
df.isna().sum()
```

[276]:
```
Employee_ID            0
Full_Name              0
SSN                  115
Department             0
Job_Title              0
Salary               122
Joining_Date         104
Education             80
Skills                 0
Performance_Rating   218
Email                  0
Phone                  0
Location               0
dtype: int64
```

Standardizing name format.

[277]:
```
df.loc[:, "Full_Name"] = df["Full_Name"].str.title()
df["Full_Name"]
```

[277]:
```
1           Linda Jones
2           James Davis
3           James Davis
4       Elizabeth Brown
5         James Johnson
             ...
2497     Patricia Smith
2498    Elizabeth Davis
2499     Patricia Jones
2500      Michael Jones
2501     Michael Garcia
Name: Full_Name, Length: 2500, dtype: object
```

[278]:
```
df.columns
```

[278]:
```
Index(['Employee_ID', 'Full_Name', 'SSN', 'Department', 'Job_Title', 'Salary',
       'Joining_Date', 'Education', 'Skills', 'Performance_Rating', 'Email',
       'Phone', 'Location'],
      dtype='object')
```

Splitting "Full_Name" into 2 columns, moving new 2 columns to the front.

[279]:
```
df[["First_Name", "Last_Name"]] = df["Full_Name"].str.split(" ", n=1,␣
 ↪expand=True)
df = df.drop(columns=["Full_Name"])
```

```
df = df.loc[:, ["Employee_ID", "First_Name", "Last_Name", "SSN", "Department",␣
 ↪"Job_Title", "Salary", "Joining_Date", "Education", "Skills",␣
 ↪"Performance_Rating", "Email", "Phone", "Location"]]
df.head()
```

[279]:

	Employee_ID	First_Name	Last_Name	SSN	Department	Job_Title	Salary	Joining_Date	Education	Skills	Performance_Rating	Email	Phone	Location
1	EMP-0926	Linda	Jones	432-65-4227	Engineering	Officer - Manager	78k	2022-09-21	Master	Java,SQL	3	linda.jones@company.com	217.220.4686	Austin, TX
2	EMP-0909	James	Davis	966429736	Finance	Consultant - Mid	$171,000	2019-12-26	B.Sc	SQL;Sales	4	james.davis@company.com	(370) 651-7833	Remote
3	EMP-0933	James	Davis	437 27 3267	IT Support	Officer - Mid	44000	2019-11-25	PhD	Java	3	james.davis@company.com	(557) 302-3186	San Francisco, CA
4	EMP-1263	Elizabeth	Brown	739127077	IT Support	Lead Consultant	112k	2023-09-06	Bachelor	Sales | Java | Excell	4	elizabeth.brown@company.com	568-361-7994	Austin, TX
5	EMP-1402	James	Johnson	691-79-3614	Marketing	Manager Developer	85000	2022-07-27	NaN	Excel | SQL | Management	4	james.johnson@company.com	(374) 314-5590	San Francisco, CA

Removing formatting characters from SSN.

[280]:
```
df["SSN"] = df["SSN"].str.replace("-", "")
df["SSN"] = df["SSN"].str.replace(" ", "")
```

[281]:
```
df.head()
```

[281]:

	Employee_ID	First_Name	Last_Name	SSN	Department	Job_Title	Salary	Joining_Date	Education	Skills	Performance_Rating	Email	Phone	Location
1	EMP-0926	Linda	Jones	432654227	Engineering	Officer - Manager	78k	2022-09-21	Master	Java,SQL	3	linda.jones@company.com	217.220.4686	Austin, TX
2	EMP-0909	James	Davis	966429736	Finance	Consultant - Mid	$171,000	2019-12-26	B.Sc	SQL;Sales	4	james.davis@company.com	(370) 651-7833	Remote
3	EMP-0933	James	Davis	437273267	IT Support	Officer - Mid	44000	2019-11-25	PhD	Java	3	james.davis@company.com	(557) 302-3186	San Francisco, CA
4	EMP-1263	Elizabeth	Brown	739127077	IT Support	Lead Consultant	112k	2023-09-06	Bachelor	Sales | Java | Excell	4	elizabeth.brown@company.com	568-361-7994	Austin, TX
5	EMP-1402	James	Johnson	691793614	Marketing	Manager Developer	85000	2022-07-27	NaN	Excel | SQL | Management	4	james.johnson@company.com	(374) 314-5590	San Francisco, CA

```
1       78k   2022-09-21    Master                  Java,SQL
2  $171,000   2019-12-26      B.Sc                SQL;Sales
3     44000   2019-11-25       PhD                     Java
4      112k   2023-09-06  Bachelor     Sales | Java | Excell
5     85000   2022-07-27       NaN  Excel | SQL | Management

  Performance_Rating                         Email           Phone  \
1                  3      linda.jones@company.com    217.220.4686
2                  4      james.davis@company.com  (370) 651-7833
3                  3      james.davis@company.com  (557) 302-3186
4                  4  elizabeth.brown@company.com    568-361-7994
5                  4    james.johnson@company.com  (374) 314-5590

            Location
1         Austin, TX
2             Remote
3  San Francisco, CA
4         Austin, TX
5  San Francisco, CA
```

Checking for any anomalies in SSN.

[282]:
```
lengths = df["SSN"].str.len()
lengths.std()
```

[282]:
```
np.float64(0.0)
```

Removing SSN containing "X" values.

[283]:
```
df = df[~df["SSN"].str.contains("X", na=False)]
```

[284]:
```
df["Department"].unique()
```

[284]:
```
array(['Engineering', 'Finance', 'IT Support', 'Marketing', 'Sales',
       'Legal', 'HR', 'Enginering', 'Fin.', 'Human Resources', 'Eng.',
       'Engineering Dept', 'H.R.', 'Marketting', 'IT', 'Tech Support',
       'Mktg', 'Fiance'], dtype=object)
```

Mapping dictionary to fix typos and abbreviations in department names.

[285]:
```
department_mapping = {
    "Enginering" : "Engineering",
    "Eng." : "Engineering",
    "Engineering Dept" : "Engineering",

    "Fin." : "Finance",
    "Fiance": "Finance",

    "Tech Support" : "IT Support",
```

```
    "IT" : "IT Support",

    "Marketting" : "Marketing",
    "Mktg" : "Marketing",

    "Human Resources" : "HR",
    "H.R." : "HR"
}

df["Department"] = df["Department"].replace(department_mapping)
df["Department"].unique()
```

[285]:
```
array(['Engineering', 'Finance', 'IT Support', 'Marketing', 'Sales',
       'Legal', 'HR'], dtype=object)
```

[286]:
```
analysts = df[df["Job_Title"].str.contains("Developer")]
print(analysts["Job_Title"].unique())
```

```
['Manager Developer' 'Mid Developer' 'Director Developer'
 'Developer - Lead' 'Developer - Mid' 'Developer - Junior'
 'Developer - Senior' 'Developer - Director' 'Junior Developer'
 'Lead Developer' 'Developer - Manager' 'Senior Developer']
```

Mapping to standardize job titles.

[287]:
```
job_title_mapping = {
    "Mid Analyst" : "Analyst - Mid",
    "Senior Analyst" : "Analyst - Senior",
    "Junior Analyst" : "Analyst - Junior",
    "Director Analyst" : "Analyst - Director",
    "Manager Analyst" : "Analyst - Manager",
    "Lead Analyst" : "Analyst - Lead",

    "Mid Officer" : "Officer - Mid",
    "Senior Officer" : "Officer - Senior",
    "Junior Officer" : "Officer - Junior",
    "Director Officer" : "Officer - Director",
    "Manager Officer" : "Officer - Manager",
    "Lead Officer" : "Officer - Lead",

    "Mid Specialist" : "Specialist - Mid",
    "Senior Specialist" : "Specialist - Senior",
    "Junior Specialist" : "Specialist - Junior",
    "Director Specialist" : "Specialist - Director",
    "Manager Specialist" : "Specialist - Manager",
    "Lead Specialist" : "Specialist - Lead",

    "Mid Developer" : "Developer - Mid",
```

```
    "Senior Developer" : "Developer - Senior",
    "Junior Developer" : "Developer - Junior",
    "Director Developer" : "Developer - Director",
    "Manager Developer" : "Developer - Manager",
    "Lead Developer" : "Developer - Lead",

    "Mid Consultant" : "Consultant - Mid",
    "Senior Consultant" : "Consultant - Senior",
    "Junior Consultant" : "Consultant - Junior",
    "Director Consultant" : "Consultant - Director",
    "Manager Consultant" : "Consultant - Manager",
    "Lead Consultant" : "Consultant - Lead"

}

df["Job_Title"] = df["Job_Title"].replace(job_title_mapping)
```

[288]:
```
analysts = df[df["Job_Title"].str.contains("Consultant")]
print(analysts["Job_Title"].unique())
```

```
['Consultant - Mid' 'Consultant - Lead' 'Consultant - Director'
 'Consultant - Junior' 'Consultant - Manager' 'Consultant - Senior']
```

[289]:
```
df["Job_Title"].unique()
```

[289]:
```
array(['Officer - Manager', 'Consultant - Mid', 'Officer - Mid',
       'Consultant - Lead', 'Developer - Manager', 'Analyst - Mid',
       'Specialist - Lead', 'Consultant - Director', 'Officer - Senior',
       'Specialist - Mid', 'Consultant - Junior', 'Analyst - Senior',
       'Developer - Mid', 'Developer - Director', 'Analyst - Junior',
       'Consultant - Manager', 'Analyst - Director', 'Developer - Lead',
       'Specialist - Senior', 'Developer - Junior',
       'Specialist - Manager', 'Specialist - Junior',
       'Consultant - Senior', 'Developer - Senior', 'Officer - Director',
       'Analyst - Manager', 'Specialist - Director', 'Officer - Lead',
       'Analyst - Lead', 'Officer - Junior'], dtype=object)
```

[290]:
```
df["Salary"]
```

[290]:
```
1              78k
2         $171,000
3            44000
4             112k
5            85000
             ...
2497        126000
2498     79000 USD
2499           61k
```

```
2500    138000 USD
2501          94000
Name: Salary, Length: 2356, dtype: object
```

Removing currency suffixes and convrrting 'k' to notation '000'.

[291]:
```
df["Salary"] = df["Salary"].str.replace("k", "000", regex=False).str.
 ↪replace("USD", "", regex=False)
```

[292]:
```
df["Salary"].head()
```

[292]:
```
1       78000
2    $171,000
3       44000
4      112000
5       85000
Name: Salary, dtype: object
```

Removing currency symbols and thousands seperators.

[293]:
```
df["Salary"] = df["Salary"].str.replace("$", "", regex=False).str.replace(",",␣
 ↪"", regex=False)
```

[294]:
```
df["Salary"]
```

[294]:
```
1          78000
2         171000
3          44000
4         112000
5          85000
          …
2497      126000
2498      79000
2499       61000
2500     138000
2501       94000
Name: Salary, Length: 2356, dtype: object
```

Removing leading or trailing whitespaces.

[295]:
```
df["Salary"] = df["Salary"].str.strip()
df["Salary"]
```

[295]:
```
1          78000
2         171000
3          44000
4         112000
5          85000
          …
```

```
2497    126000
2498     79000
2499     61000
2500    138000
2501     94000
Name: Salary, Length: 2356, dtype: object
```

[296]:
```
df["Salary"] = "$" + df["Salary"]
df["Salary"]
```

[296]:
```
1        $78000
2       $171000
3        $44000
4       $112000
5        $85000
         …
2497    $126000
2498     $79000
2499     $61000
2500    $138000
2501     $94000
Name: Salary, Length: 2356, dtype: object
```

[297]:
```
df["Joining_Date"]
```

[297]:
```
1       2022-09-21
2       2019-12-26
3       2019-11-25
4       2023-09-06
5       2022-07-27
           …
2497           NaN
2498    2018-02-11
2499    10/08/2019
2500           NaN
2501    2023-05-28
Name: Joining_Date, Length: 2356, dtype: object
```

Custom function to handle multiple date formats.

[298]:
```
def clean_date_data(data_str):
    if not isinstance(data_str, str):
        return pd.NaT

    formats_to_try = [
        "%Y-%m-%d",
        "%m/%d/%Y",
        "%d.%m.%Y"
```

```
    ]

    for format in formats_to_try:
        try:
            return datetime.strptime(data_str.strip(), format)
        except ValueError:
            continue

    return pd.NaT
```

Apply cleaning function to the “Joining_Date” column.

[299]:
```
df["Joining_Date"] = df["Joining_Date"].apply(clean_date_data)
```

[300]:
```
df["Joining_Date"].head()
```

[300]:
```
1   2022-09-21
2   2019-12-26
3   2019-11-25
4   2023-09-06
5   2022-07-27
Name: Joining_Date, dtype: datetime64[ns]
```

[301]:
```
print(df["Joining_Date"].isna().sum())
```

```
222
```

[302]:
```
df["Education"]
```

[302]:
```
1            Master
2              B.Sc
3               PhD
4          Bachelor
5               NaN
           …
2497          Ph.D.
2498         Master
2499            PhD
2500    High School
2501       Bachelor
Name: Education, Length: 2356, dtype: object
```

[303]:
```
df["Education"].unique()
```

[303]:
```
array(['Master', 'B.Sc', 'PhD', 'Bachelor', nan, 'High School', 'Masters',
       'H.S.', "Master's", 'Doctorate', 'M.Sc', 'Bachelor Degree',
       'Bachelors', 'MBA', 'Ph.D.', 'BS'], dtype=object)
```

Mapping degree formats to standard categories.

[304]:
```
education_mapping = {
    "Master" : "M.Sc",
    "Masters" : "M.Sc",
    "Master's" : "M.Sc",
    "Bachelor" : "B.Sc",
    "Bachelor Degree" : "B.Sc",
    "Bachelors" : "B.Sc",
    "BS" : "B.Sc",
    "Doctorate" : "Ph.D.",
    "H.S." : "High School"
}
```

[305]:
```
df["Education"] = df["Education"].replace(education_mapping)
df["Education"].unique()
```

[305]:
```
array(['M.Sc', 'B.Sc', 'PhD', nan, 'High School', 'Ph.D.', 'MBA'],
      dtype=object)
```

[306]:
```
df["Skills"]
```

[306]:
```
1                          Java,SQL
2                         SQL;Sales
3                              Java
4              Sales | Java | Excell
5           Excel | SQL | Management
                     …
2497    Power BI/Communication/Java
2498              Excel/Python/Sales
2499           SQL, Python, Power BI
2500     Python | Excell | Power BI
2501                      Management
Name: Skills, Length: 2356, dtype: object
```

Unifying seperators, replacing semicolons and slashes with commas.

[307]:
```
df["Skills"] = df["Skills"].str.replace(";", ",", regex=False).str.replace(" |␣
↪", ",", regex=False).str.replace("/", ",", regex=False).str.replace(", ",␣
↪",", regex=False)
df["Skills"]
```

[307]:
```
1                          Java,SQL
2                         SQL,Sales
3                              Java
4                 Sales,Java,Excell
5              Excel,SQL,Management
                     …
2497    Power BI,Communication,Java
2498              Excel,Python,Sales
```

```
2499            SQL,Python,Power BI
2500         Python,Excell,Power BI
2501                     Management
Name: Skills, Length: 2356, dtype: object
```

Converting text strings into lists.

[308]:
```
df["Skills"] = df["Skills"].str.split(",")
df["Skills"]
```

[308]:
```
1                            [Java, SQL]
2                           [SQL, Sales]
3                                [Java]
4                  [Sales, Java, Excell]
5              [Excel, SQL, Management]
                       …
2497    [Power BI, Communication, Java]
2498              [Excel, Python, Sales]
2499            [SQL, Python, Power BI]
2500         [Python, Excell, Power BI]
2501                         [Management]
Name: Skills, Length: 2356, dtype: object
```

Mapping typos in column "Skills".

[309]:
```
skills_mapping = {
    "Excell" : "Excel",
    "python" : "Python",
    "Power BI" : "PowerBI"
}
```

Exploding the lists into seperate rows to clean individual elements.

[310]:
```
df_exploded = df.explode("Skills")
```

Stripping whitespaces and correcting typos. Grouping the cleaned skills back into lists for each employee.

[311]:
```
df_exploded["Skills"] = df_exploded["Skills"].str.strip()
df_exploded["Skills"] = df_exploded["Skills"].replace(skills_mapping)
df["Skills"] = df_exploded.groupby(level=0)["Skills"].agg(list)
```

[312]:
```
df_exploded["Skills"].unique()
```

[312]:
```
array(['Java', 'SQL', 'Sales', 'Excel', 'Management', 'Communication',
       'PowerBI', 'Python'], dtype=object)
```

[313]:
```
df["Performance_Rating"].unique()
```

[313]:
```
array(['3', '4', nan, '1', 'A', 'B', '2', '5'], dtype=object)
```

Exchanging letters from “Performance_Rating” for numbers (0-5).

[314]:
```
performance_rating_mapping = {
    "A" : "5",
    "B" : "4"
}
df["Performance_Rating"] = df["Performance_Rating"].
 ↪replace(performance_rating_mapping)
```

Checking if emails are created correctly.

[315]:
```
df["Correct_Email"] = df["First_Name"].str.lower() + "." + df["Last_Name"].str.
 ↪lower() + "@company.com"
```

[316]:
```
assert df["Email"].all() == df["Correct_Email"].all()
```

[317]:
```
df = df.drop(columns=["Correct_Email"])
```

[318]:
```
df["Phone"]
```

[318]:
```
1          217.220.4686
2        (370) 651-7833
3        (557) 302-3186
4          568-361-7994
5        (374) 314-5590
              …
2497    +1 564 720 3280
2498     (420) 236-6494
2499    +1 216 123 3270
2500     (249) 321-3430
2501    +1 583 296 5341
Name: Phone, Length: 2356, dtype: object
```

Removing country codes and parentheses. Replacing spaces and dots with standard hyphens.

[319]:
```
df["Phone"] = df["Phone"].str.replace(r'\+1\s|[\(\)]', '', regex=True)
df["Phone"] = df["Phone"].str.replace(r'[). ]+', '-', regex=True)
```

[320]:
```
df["Phone"].head()
```

[320]:
```
1    217-220-4686
2    370-651-7833
3    557-302-3186
4    568-361-7994
5    374-314-5590
Name: Phone, dtype: object
```

Splitting “Location” into “City” and “State/Country”.

[321]:
```
df[["City", "State/Country"]] = df["Location"].str.split(",", n=1, expand=True)
df = df.drop(columns="Location")
```

[328]:
```
df["State/Country"] = df["State/Country"].str.upper()
```

Saving the cleaned dataset.

[329]:
```
df.to_csv("cleaned_hr_data.csv", index=False)
```