Data Visualization Lab

Estimated time needed: **45 to 60** minutes

In this assignment you will be focusing on the visualization of data.

The data set will be presented to you in the form of a RDBMS.

You will have to use SQL queries to extract the data.

Objectives

In this lab you will perform the following:

- Visualize the distribution of data.
- Visualize the relationship between two features.
- Visualize composition of data.
- Visualize comparison of data.

Demo: How to work with database

Download database file.

In [78]:
```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM
```

```
--2023-09-16 02:13:32--  https://cf-courses-data.s3.us.cloud-object-storage.a
ppdomain.cloud/IBM-DA0321EN-SkillsNetwork/LargeData/m4_survey_data.sqlite
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-cour
ses-data.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-
courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:44
3... connected.
HTTP request sent, awaiting response... 200 OK
Length: 36679680 (35M) [application/octet-stream]
Saving to: ‘m4_survey_data.sqlite.13’

m4_survey_data.sqli 100%[===================>]  34.98M  24.3MB/s    in 1.4s

2023-09-16 02:13:35 (24.3 MB/s) - ‘m4_survey_data.sqlite.13’ saved [36679680/
36679680]
```

Connect to the database.

In [79]:
```
import sqlite3
conn = sqlite3.connect("m4_survey_data.sqlite") # open a database connectior
```

Import pandas module.

In [80]:
```
import pandas as pd
```

Demo: How to run an sql query

In [81]:
```
# print how many rows are there in the table named 'master'
QUERY = """
SELECT COUNT(*)
FROM master
"""

# the read_sql_query runs the sql query and returns the data as a dataframe
df = pd.read_sql_query(QUERY,conn)
df.head()
```

Out[81]:

	COUNT(*)
0	11398

Demo: How to list all tables

In [82]:
```
# print all the tables names in the database
QUERY = """
SELECT name as Table_Name FROM
sqlite_master WHERE
type = 'table'
"""
```

```
# the read_sql_query runs the sql query and returns the data as a dataframe
pd.read_sql_query(QUERY,conn)
```

Out[82]:

	Table_Name
0	EduOther
1	DevType
2	LastInt
3	JobFactors
4	WorkPlan
5	WorkChallenge
6	LanguageWorkedWith
7	LanguageDesireNextYear
8	DatabaseWorkedWith
9	DatabaseDesireNextYear
10	PlatformWorkedWith
11	PlatformDesireNextYear
12	WebFrameWorkedWith
13	WebFrameDesireNextYear
14	MiscTechWorkedWith
15	MiscTechDesireNextYear
16	DevEnviron
17	Containers
18	SOVisitTo
19	SONewContent
20	Gender
21	Sexuality
22	Ethnicity
23	master

In [86]:

```
QUERY = """
SELECT *
from master
"""
dkn=pd.read_sql_query(QUERY,conn)
dkn.head()
```

Out[86]:

	index	Respondent	MainBranch	Hobbyist	OpenSourcer	OpenSource	Employment
0	0	4	I am a developer by profession	No	Never	The quality of OSS and closed source software ...	Employed full-time
1	1	9	I am a developer by profession	Yes	Once a month or more often	The quality of OSS and closed source software ...	Employed full-time
2	2	13	I am a developer by profession	Yes	Less than once a month but more than once per ...	OSS is, on average, of HIGHER quality than pro...	Employed full-time
3	3	16	I am a developer by profession	Yes	Never	The quality of OSS and closed source software ...	Employed full-time
4	4	17	I am a developer by profession	Yes	Less than once a month but more than once per ...	The quality of OSS and closed source software ...	Employed full-time

5 rows × 63 columns

In [89]:
```
dkn.columns
```

Out[89]:
```
Index(['index', 'Respondent', 'MainBranch', 'Hobbyist', 'OpenSourcer',
       'OpenSource', 'Employment', 'Country', 'Student', 'EdLevel',
       'UndergradMajor', 'OrgSize', 'YearsCode', 'Age1stCode', 'YearsCodePr
o',
       'CareerSat', 'JobSat', 'MgrIdiot', 'MgrMoney', 'MgrWant', 'JobSeek',
       'LastHireDate', 'FizzBuzz', 'ResumeUpdate', 'CurrencySymbol',
       'CurrencyDesc', 'CompTotal', 'CompFreq', 'ConvertedComp', 'WorkWeekH
rs',
       'WorkRemote', 'WorkLoc', 'ImpSyn', 'CodeRev', 'CodeRevHrs', 'UnitTes
ts',
       'PurchaseHow', 'PurchaseWhat', 'OpSys', 'BlockchainOrg', 'Blockchain
Is',
       'BetterLife', 'ITperson', 'OffOn', 'SocialMedia', 'Extraversion',
       'ScreenName', 'SOVisit1st', 'SOVisitFreq', 'SOFindAnswer',
       'SOTimeSaved', 'SOHowMuchTime', 'SOAccount', 'SOPartFreq', 'SOJobs',
       'EntTeams', 'SOComm', 'WelcomeChange', 'Age', 'Trans', 'Dependents',
       'SurveyLength', 'SurveyEase'],
      dtype='object')
```

In [49]:
```
QUERY = """
SELECT DISTINCT DatabaseWorkedWith, Respondent
from DatabaseWorkedWith
"""
# the read_sql_query runs the sql query and returns the data as a dataframe
ran_pd=pd.read_sql_query(QUERY,conn)
ran_pd.head(50)
```

Out[49]:

	DatabaseWorkedWith	Respondent
0	MySQL	4
1	SQLite	4
2	DynamoDB	9
3	PostgreSQL	9
4	SQLite	9
5	Couchbase	13
6	DynamoDB	13
7	Firebase	13
8	MySQL	13
9	MongoDB	16
10	Microsoft SQL Server	16
11	MySQL	16
12	MongoDB	17
13	PostgreSQL	17
14	DynamoDB	19
15	Firebase	19
16	Microsoft SQL Server	19
17	MySQL	19
18	SQLite	19
19	Elasticsearch	20
20	MariaDB	20
21	MongoDB	20
22	Microsoft SQL Server	20
23	Elasticsearch	22
24	MySQL	22
25	Oracle	22
26	Redis	22
27	Oracle	23
28	SQLite	23
29	Firebase	24
30	MongoDB	24
31	MySQL	24

	DatabaseWorkedWith	Respondent
32	MySQL	25
33	Microsoft SQL Server	26
34	MySQL	26
35	Redis	26
36	SQLite	26
37	Firebase	29
38	MongoDB	29
39	MySQL	29
40	MongoDB	32
41	PostgreSQL	32
42	Redis	32
43	Microsoft SQL Server	38
44	Microsoft SQL Server	39
45	PostgreSQL	39
46	Redis	39
47	SQLite	39
48	Firebase	43
49	MySQL	43

Demo: How to run a group by query

In [50]:
```
QUERY = """
SELECT Age,COUNT(*) as count
FROM master
group by age
order by age
"""
pd.read_sql_query(QUERY,conn)
```

Out[50]:

	Age	count
0	NaN	287
1	16.0	3
2	17.0	6
3	18.0	29
4	19.0	78
5	20.0	109
6	21.0	203
7	22.0	406
8	23.0	581
9	24.0	679
10	25.0	738
11	26.0	720
12	27.0	724
13	28.0	787
14	29.0	697
15	30.0	651
16	31.0	531
17	32.0	489
18	33.0	483
19	34.0	395
20	35.0	393
21	36.0	308
22	37.0	280
23	38.0	279
24	39.0	232
25	40.0	187
26	41.0	136
27	42.0	162
28	43.0	100
29	44.0	95
30	45.0	85
31	46.0	66

	Age	count
32	47.0	68
33	48.0	64
34	49.0	66
35	50.0	57
36	51.0	29
37	52.0	41
38	53.0	32
39	54.0	26
40	55.0	13
41	56.0	16
42	57.0	11
43	58.0	12
44	59.0	11
45	60.0	2
46	61.0	10
47	62.0	5
48	63.0	7
49	65.0	2
50	66.0	1
51	67.0	1
52	69.0	1
53	71.0	2
54	72.0	1
55	99.0	1

Demo: How to describe a table

In [51]:
```
table_name = 'master'  # the table you wish to describe

QUERY = """
SELECT sql FROM sqlite_master
WHERE name= '{}'
""".format(table_name)
```

```
df = pd.read_sql_query(QUERY,conn)
print(df.iat[0,0])
```

```
CREATE TABLE "master" (
"index" INTEGER,
  "Respondent" INTEGER,
  "MainBranch" TEXT,
  "Hobbyist" TEXT,
  "OpenSourcer" TEXT,
  "OpenSource" TEXT,
  "Employment" TEXT,
  "Country" TEXT,
  "Student" TEXT,
  "EdLevel" TEXT,
  "UndergradMajor" TEXT,
  "OrgSize" TEXT,
  "YearsCode" TEXT,
  "Age1stCode" TEXT,
  "YearsCodePro" TEXT,
  "CareerSat" TEXT,
  "JobSat" TEXT,
  "MgrIdiot" TEXT,
  "MgrMoney" TEXT,
  "MgrWant" TEXT,
  "JobSeek" TEXT,
  "LastHireDate" TEXT,
  "FizzBuzz" TEXT,
  "ResumeUpdate" TEXT,
  "CurrencySymbol" TEXT,
  "CurrencyDesc" TEXT,
  "CompTotal" REAL,
  "CompFreq" TEXT,
  "ConvertedComp" REAL,
  "WorkWeekHrs" REAL,
  "WorkRemote" TEXT,
  "WorkLoc" TEXT,
  "ImpSyn" TEXT,
  "CodeRev" TEXT,
  "CodeRevHrs" REAL,
  "UnitTests" TEXT,
  "PurchaseHow" TEXT,
  "PurchaseWhat" TEXT,
  "OpSys" TEXT,
  "BlockchainOrg" TEXT,
  "BlockchainIs" TEXT,
  "BetterLife" TEXT,
  "ITperson" TEXT,
  "OffOn" TEXT,
  "SocialMedia" TEXT,
  "Extraversion" TEXT,
  "ScreenName" TEXT,
  "SOVisit1st" TEXT,
  "SOVisitFreq" TEXT,
  "SOFindAnswer" TEXT,
  "SOTimeSaved" TEXT,
  "SOHowMuchTime" TEXT,
  "SOAccount" TEXT,
  "SOPartFreq" TEXT,
  "SOJobs" TEXT,
```

```
    "EntTeams" TEXT,
    "SOComm" TEXT,
    "WelcomeChange" TEXT,
    "Age" REAL,
    "Trans" TEXT,
    "Dependents" TEXT,
    "SurveyLength" TEXT,
    "SurveyEase" TEXT
)
```

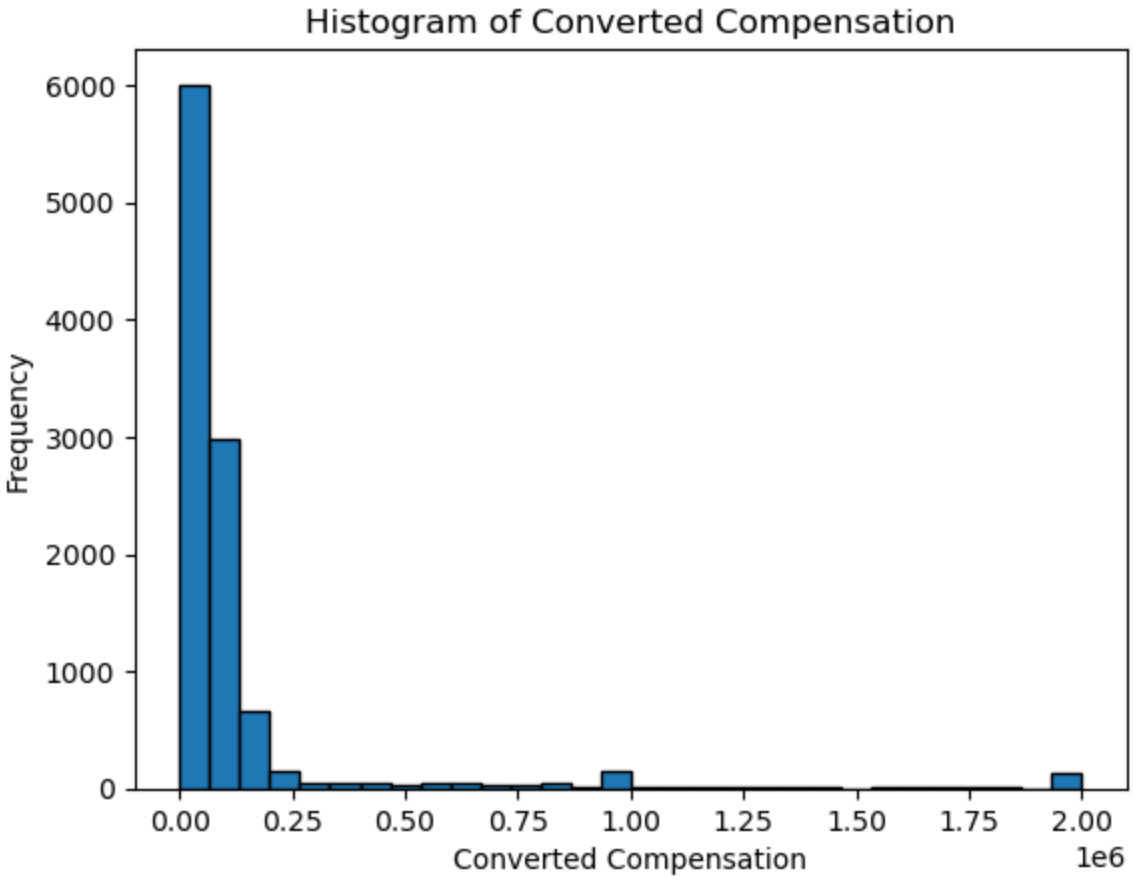
Hands-on Lab

Visualizing distribution of data

Histograms

Plot a histogram of `ConvertedComp.`

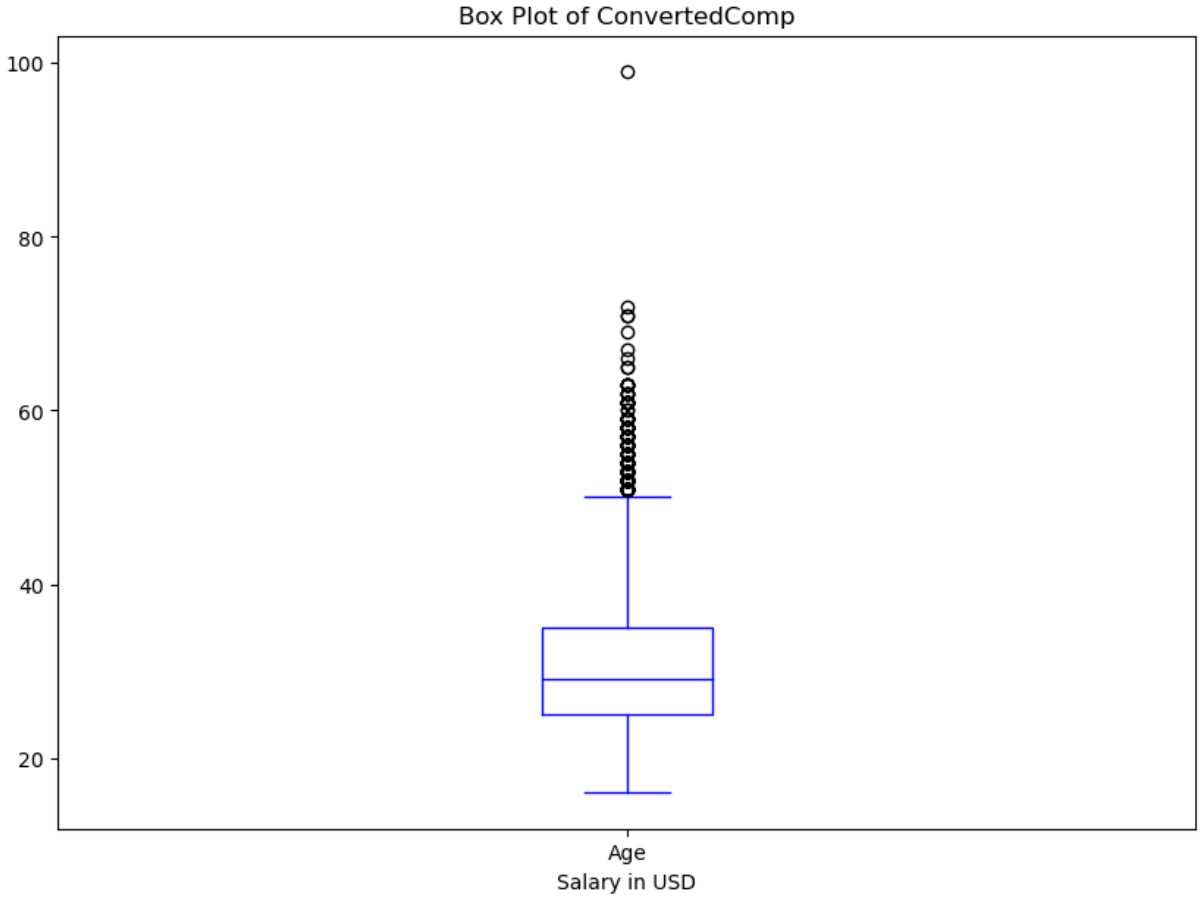
In [52]:
```
# your code goes here
import matplotlib.pyplot as plt

# SQL query to select the "ConvertedComp" column
query = """
SELECT ConvertedComp
FROM master
"""

# Read the data into a DataFrame
df = pd.read_sql_query(query, conn)

# Plot a histogram
plt.hist(df['ConvertedComp'], bins=30, edgecolor='k')
plt.xlabel('Converted Compensation')
plt.ylabel('Frequency')
plt.title('Histogram of Converted Compensation')
plt.show()
```



Box Plots

Plot a box plot of `Age.`

In [53]:
```
# your code goes here
query = """
SELECT Age
FROM master
"""

df = pd.read_sql_query(query, conn)
df_con=df['Age']

plt.figure(figsize=(8, 6))
df_con.plot(kind='box', figsize=(10, 7), color='blue', vert=True)
plt.title('Box Plot of ConvertedComp')
plt.xlabel('Salary in USD')
plt.show()
```



Visualizing relationships in data

Scatter Plots

plot of Age and WorkWeekHrsCreate a scatter plot of `Age` and `WorkWeekHrs.`

In [54]:
```
# your code goes here

query = """
SELECT Age, WorkWeekHrs
FROM master
"""

df = pd.read_sql_query(query, conn)
df_con=df['Age']
df_Work=df['WorkWeekHrs']

#Create figure and axes
fig, ax = plt.subplots(figsize=(8, 4))

# Customizing Scatter Plot
ax.scatter(df_con, df_Work,
           marker='o', #setting up the markers
           s = 20, #setting up the size of the markers
           color='darkblue')#the color for the marker
```

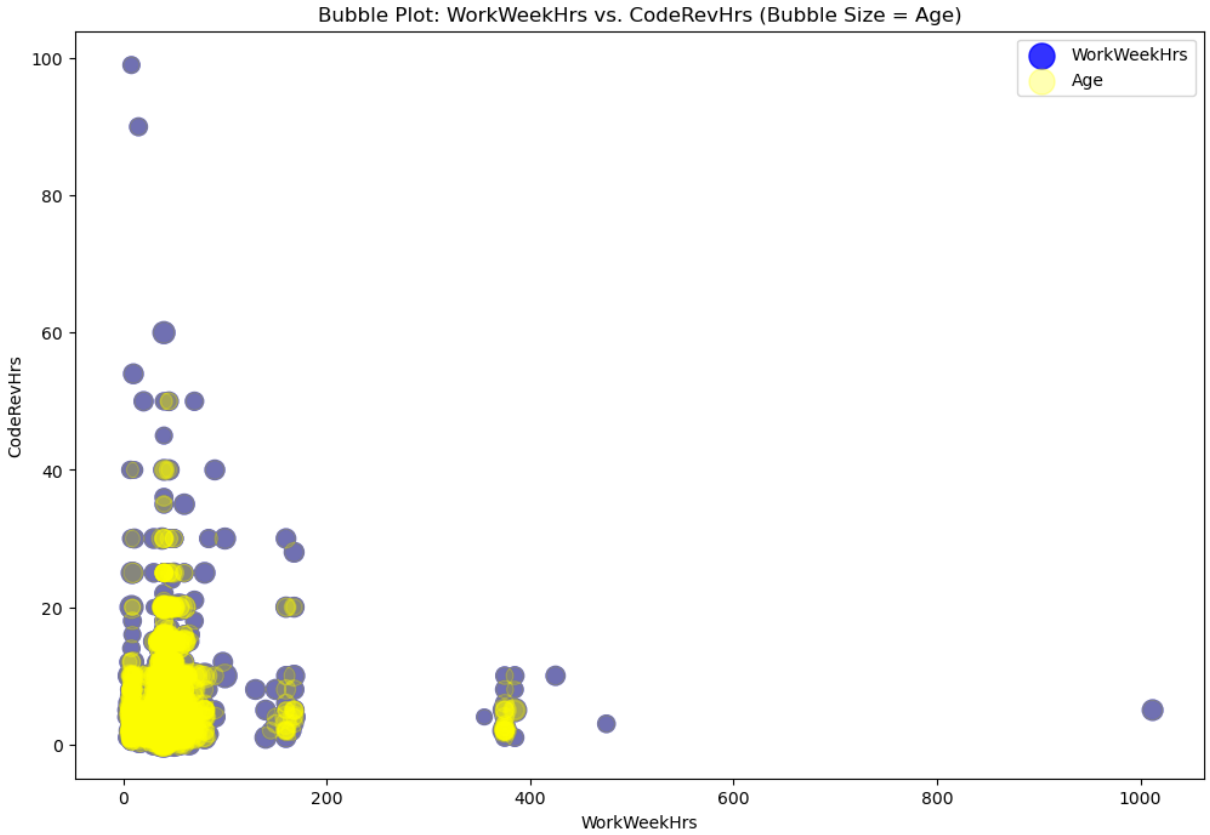
```
#add title
plt.title('plot of Age and WorkWeekHrs')
#add labels
plt.xlabel('Age')
plt.ylabel('Work hours per week')
#including grid
plt.grid(True)


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

Bubble Plots

Create a bubble plot of `WorkWeekHrs` and `CodeRevHrs`, use `Age` column as bubble size.

In [55]:
```
import matplotlib.pyplot as plt

# SQL query to select Age, WorkWeekHrs, and CodeRevHrs
query = """
SELECT Age, WorkWeekHrs, CodeRevHrs
FROM master
"""

# Read data into a DataFrame
df = pd.read_sql_query(query, conn)

# Extract data columns
age = df['Age']
work_week_hours = df['WorkWeekHrs']
code_review_hours = df['CodeRevHrs']
```

```
# Increase the size of the bubbles by multiplying the 'age' values
bubble_size = age * 4  # Adjust the multiplier as needed

# Create a bubble plot with increased bubble size and custom colors
plt.figure(figsize=(12, 8))
plt.scatter(work_week_hours, code_review_hours, s=bubble_size, c='blue', alp
plt.scatter(work_week_hours, code_review_hours, s=bubble_size, c='yellow', a

# Customize the plot
plt.xlabel('WorkWeekHrs')
plt.ylabel('CodeRevHrs')
plt.title('Bubble Plot: WorkWeekHrs vs. CodeRevHrs (Bubble Size = Age)')
plt.legend()

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



Visualizing composition of data

Pie Charts

Create a pie chart of the top 5 databases that respondents wish to learn next year. Label the pie chart with database names. Display percentages of each database on the pie chart.

In [56]:
```
import pandas as pd
```

```
# SQL query to retrieve the rank and count of Python in the desired language
QUERY = """
SELECT *
from LanguageDesireNextYear

"""

# Execute the query and store the result in a DataFrame
result_df = pd.read_sql_query(QUERY, conn)

# Print the result
result_df.head(60)
```

Out[56]:

	Respondent	LanguageDesireNextYear
0	4	C
1	4	C#
2	4	JavaScript
3	4	SQL
4	9	Bash/Shell/PowerShell
5	9	C
6	9	HTML/CSS
7	9	JavaScript
8	9	Ruby
9	9	Rust
10	9	SQL
11	9	TypeScript
12	9	WebAssembly
13	9	Other(s):
14	13	Bash/Shell/PowerShell
15	13	HTML/CSS
16	13	JavaScript
17	13	Rust
18	13	SQL
19	13	TypeScript
20	13	WebAssembly
21	16	C#
22	16	HTML/CSS
23	16	JavaScript
24	16	TypeScript
25	16	WebAssembly
26	16	Other(s):
27	17	Bash/Shell/PowerShell
28	17	HTML/CSS
29	17	Java
30	17	JavaScript
31	17	TypeScript

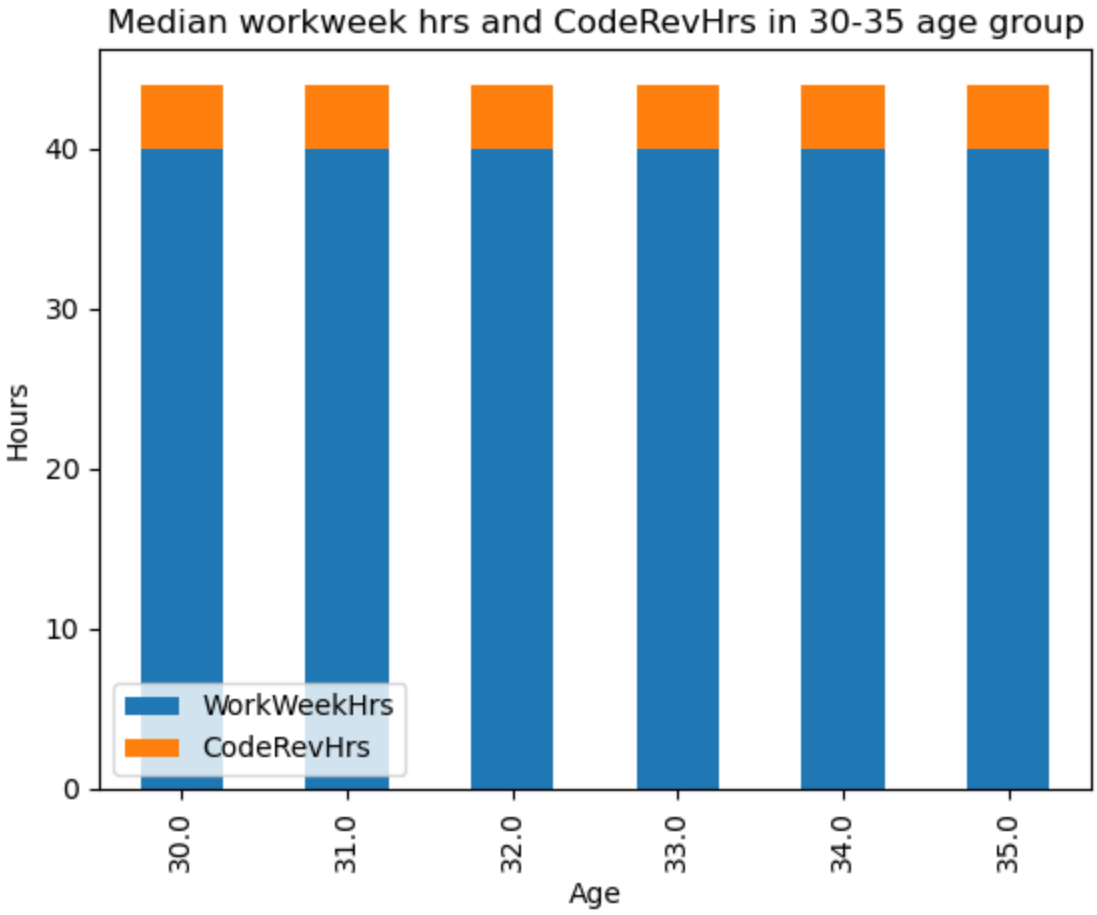
	Respondent	LanguageDesireNextYear
32	17	WebAssembly
33	19	HTML/CSS
34	19	JavaScript
35	20	Bash/Shell/PowerShell
36	20	C#
37	20	HTML/CSS
38	20	Java
39	20	JavaScript
40	20	PHP
41	20	Python
42	20	R
43	20	SQL
44	22	Bash/Shell/PowerShell
45	22	C++
46	22	HTML/CSS
47	22	JavaScript
48	22	Python
49	22	Ruby
50	22	SQL
51	22	TypeScript
52	23	Bash/Shell/PowerShell
53	23	Go
54	23	HTML/CSS
55	23	Java
56	23	JavaScript
57	23	Kotlin
58	23	Objective-C
59	23	Python

In [ ]:

Stacked Charts

Create a stacked chart of median `WorkWeekHrs` and `CodeRevHrs` for the age group 30 to 35.

In [57]:
```
import matplotlib.pyplot as plt
QUERY = """
SELECT Age,WorkWeekHrs,CodeRevHrs
FROM master
WHERE Age BETWEEN 30 AND 35
"""
df_2 = pd.read_sql_query(QUERY,conn)

group_df=df_2.groupby('Age').median()

group_df.plot(kind='bar',stacked=True)
plt.title('Median workweek hrs and CodeRevHrs in 30-35 age group')
plt.xlabel('Age')
plt.ylabel('Hours')
plt.show()
```

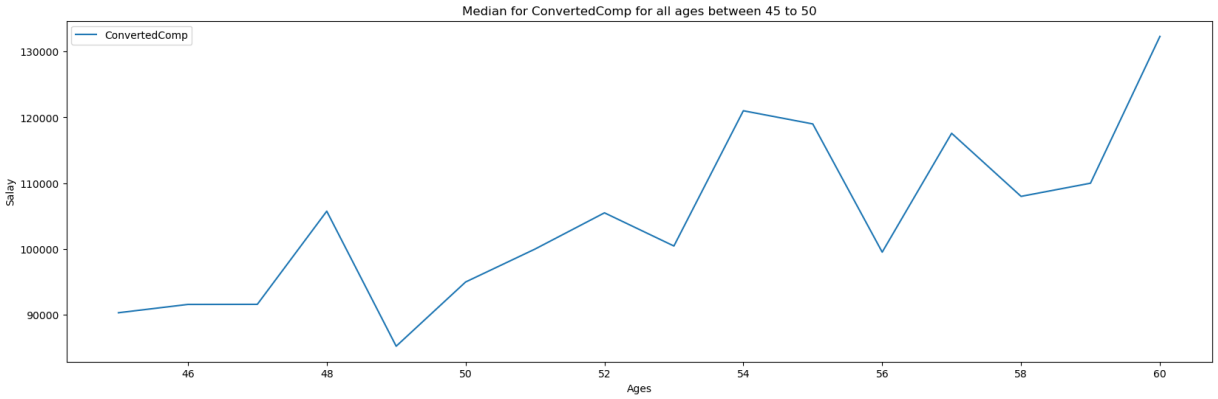


Visualizing comparison of data

Line Chart

Plot the median `ConvertedComp` for all ages from 45 to 60.

In [67]:
```
# your code goes here
import pandas as pd

# Establish a database connection

# SQL query to select ConvertedComp for individuals between ages 45 and 60
QUERY = """
SELECT ConvertedComp,Age
FROM master
WHERE Age BETWEEN 45 AND 60
"""

# Read data from the database into a DataFrame
df_8= pd.read_sql_query(QUERY, conn)

# Group the DataFrame by 'Age' and calculate the median
group_df_5 = df_8.groupby('Age').median()

group_df_5.plot(kind='line', figsize=(20, 6)) # add to subplot
plt.title ('Median for ConvertedComp for all ages between 45 to 50')
plt.ylabel('Salay')
plt.xlabel('Ages')

plt.show()
```



Bar Chart

SQL query to select the 'MainBranch' column

QUERY = """ SELECT MainBranch FROM master """

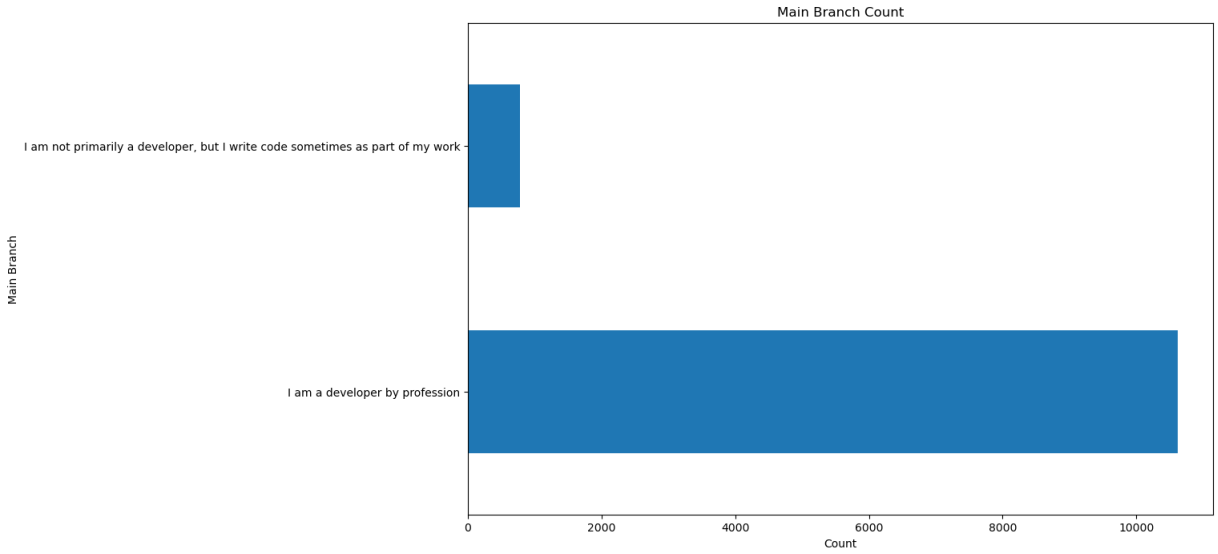
Read data from the database into a DataFrame

df_9 = pd.read_sql_query(QUERY, conn)

Create a count plot for the 'MainBranch' column

plt.figure(figsize=(12, 8)) df_9['MainBranch'].value_counts().plot(kind='barh') plt.title('Main Branch Count') plt.xlabel('Count') plt.ylabel('Main Branch') plt.show()
Create a horizontal bar chart using column `MainBranch.`

In [93]:
```
# SQL query to select the 'MainBranch' column
QUERY = """
SELECT MainBranch
FROM master
"""

# Read data from the database into a DataFrame
df_9 = pd.read_sql_query(QUERY, conn)

# Create a count plot for the 'MainBranch' column
plt.figure(figsize=(12, 8))
df_9['MainBranch'].value_counts().plot(kind='barh')
plt.title('Main Branch Count')
plt.xlabel('Count')
plt.ylabel('Main Branch')
plt.show()
```



In [95]:
```
QUERY = """
SELECT MainBranch,OpenSourcer,JobSat
FROM master
"""
df_9 = pd.read_sql_query(QUERY, conn)
df_9.head(50)
```

Out[95]:

	MainBranch	OpenSourcer	JobSat
0	I am a developer by profession	Never	Slightly satisfied
1	I am a developer by profession	Once a month or more often	Slightly satisfied
2	I am a developer by profession	Less than once a month but more than once per ...	Very satisfied
3	I am a developer by profession	Never	Slightly satisfied
4	I am a developer by profession	Less than once a month but more than once per ...	Neither satisfied nor dissatisfied
5	I am a developer by profession	Never	Very satisfied
6	I am not primarily a developer, but I write co...	Never	Slightly dissatisfied
7	I am a developer by profession	Less than once per year	Very dissatisfied
8	I am a developer by profession	Less than once per year	Slightly satisfied
9	I am a developer by profession	Never	Very satisfied
10	I am a developer by profession	Never	Very satisfied
11	I am a developer by profession	Less than once per year	Very satisfied
12	I am a developer by profession	Less than once a month but more than once per ...	Slightly satisfied
13	I am a developer by profession	Never	Slightly satisfied
14	I am a developer by profession	Never	Very satisfied
15	I am a developer by profession	Less than once per year	Very satisfied
16	I am a developer by profession	Less than once per year	Slightly dissatisfied
17	I am a developer by profession	Once a month or more often	Neither satisfied nor dissatisfied
18	I am a developer by profession	Less than once per year	Very satisfied
19	I am a developer by profession	Once a month or more often	Very satisfied

	MainBranch	OpenSourcer	JobSat
20	I am not primarily a developer, but I write co...	Less than once a month but more than once per ...	Slightly satisfied
21	I am a developer by profession	Once a month or more often	Slightly satisfied
22	I am a developer by profession	Never	Slightly dissatisfied
23	I am not primarily a developer, but I write co...	Never	Very satisfied
24	I am a developer by profession	Less than once a month but more than once per ...	Neither satisfied nor dissatisfied
25	I am a developer by profession	Less than once per year	Very satisfied
26	I am a developer by profession	Less than once a month but more than once per ...	Slightly dissatisfied
27	I am not primarily a developer, but I write co...	Never	Very satisfied
28	I am a developer by profession	Less than once a month but more than once per ...	Slightly dissatisfied
29	I am a developer by profession	Less than once per year	Slightly dissatisfied
30	I am a developer by profession	Never	Slightly satisfied
31	I am a developer by profession	Never	Very satisfied
32	I am a developer by profession	Never	Slightly dissatisfied
33	I am a developer by profession	Never	Very satisfied
34	I am a developer by profession	Less than once a month but more than once per ...	Neither satisfied nor dissatisfied
35	I am a developer by profession	Less than once a month but more than once per ...	Very satisfied
36	I am a developer by profession	Less than once a month but more than once per ...	Very satisfied
37	I am a developer by profession	Never	Slightly satisfied
38	I am not primarily a developer, but I write co...	Once a month or more often	Slightly dissatisfied
39	I am a developer by profession	Never	Slightly satisfied

	MainBranch	OpenSourcer	JobSat
40	I am a developer by profession	Less than once a month but more than once per ...	Very dissatisfied
41	I am a developer by profession	Never	Very satisfied
42	I am a developer by profession	Never	Slightly dissatisfied
43	I am a developer by profession	Less than once per year	Very satisfied
44	I am not primarily a developer, but I write co...	Never	Slightly satisfied
45	I am a developer by profession	Once a month or more often	Very satisfied
46	I am a developer by profession	Never	Slightly satisfied
47	I am a developer by profession	Never	Very dissatisfied
48	I am a developer by profession	Never	Very satisfied
49	I am a developer by profession	Once a month or more often	Slightly satisfied

In [74]:
```
QUERY = """
SELECT MainBranch
FROM master
"""

# Read data from the database into a DataFrame
df_9 = pd.read_sql_query(QUERY, conn)

# Find the most common response (mode) in the 'MainBranch' column
majority_response = df_9['MainBranch'].mode().values[0]

# Close the database connection when done
conn.close()

print(f"The majority of survey responders are in the '{majority_response}' c
```

```
The majority of survey responders are in the 'I am a developer by profession'
category.
```

Close the database connection.

In [ ]:
```
conn.close()
```

Authors

Ramesh Sannareddy

Other Contributors

Rav Ahuja

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-10-17	0.1	Ramesh Sannareddy	Created initial version of the lab