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 [1]:
```
!wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DA0321
```

```
--2024-12-10 01:36:54--  https://cf-courses-data.s3.us.cloud-object-storage.appdomai
n.cloud/IBM-DA0321EN-SkillsNetwork/LargeData/m4_survey_data.sqlite
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-courses-dat
a.s3.us.cloud-object-storage.appdomain.cloud)... 169.63.118.104, 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|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 36679680 (35M) [application/octet-stream]
Saving to: ‘m4_survey_data.sqlite’

m4_survey_data.sqli 100%[===================>]  34.98M  40.5MB/s    in 0.9s

2024-12-10 01:36:56 (40.5 MB/s) - ‘m4_survey_data.sqlite’ saved [36679680/36679680]
```

Connect to the database.

In [3]:
```
import sqlite3
conn = sqlite3.connect("m4_survey_data.sqlite") # open a database connection
```

Import pandas module.

In [10]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

Demo: How to run an sql query

In [11]:
```
# 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[11]:

	COUNT(*)
0	11398

Demo: How to list all tables

In [12]:
```
# 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[12]:

	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

Demo: How to run a group by query

In [13]:
```
QUERY = """
SELECT Age,COUNT(*) as count
FROM master
group by age
order by age
```

```
"""
pd.read_sql_query(QUERY,conn)
```

Out[13]:

	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

	Age	count
30	45.0	85
31	46.0	66
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 [14]:
```
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

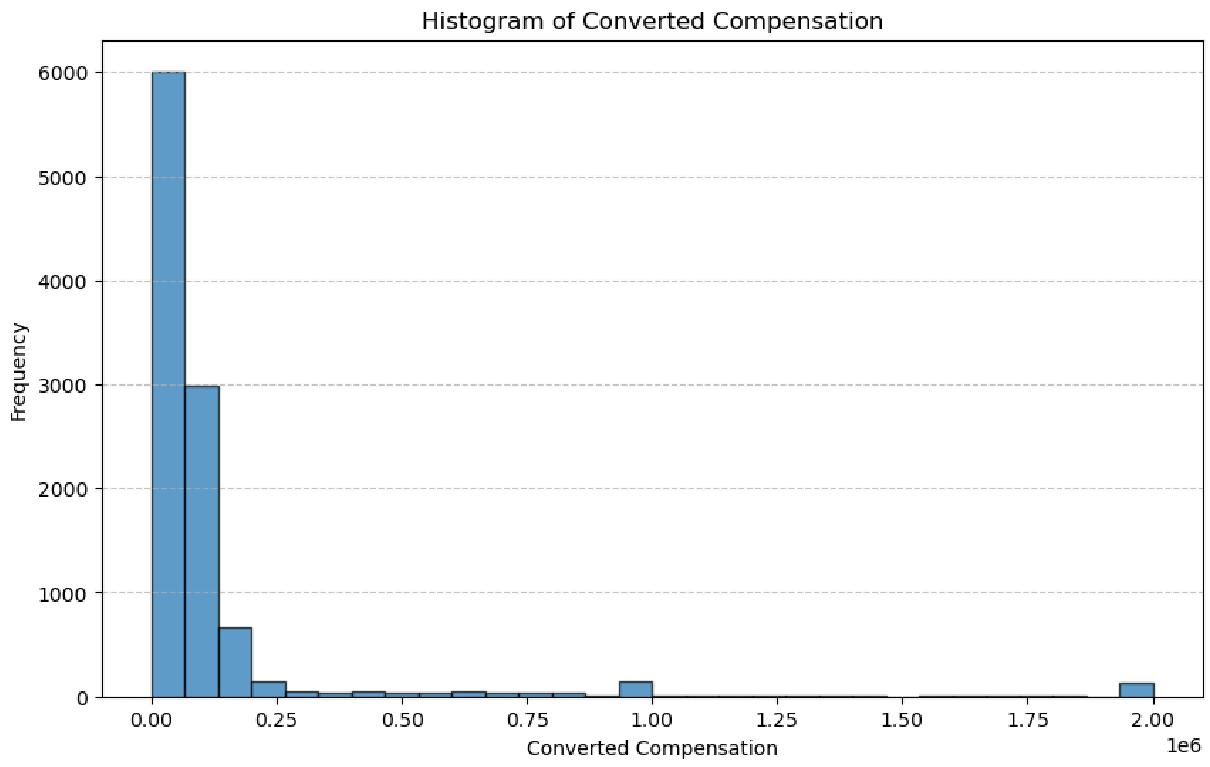
Histograms

Plot a histogram of `ConvertedComp.`

In [16]:

```
# Query to get the 'ConvertedComp' column
query = "SELECT ConvertedComp FROM master"
df = pd.read_sql_query(query, conn)

# Plot the histogram
plt.figure(figsize=(10, 6))
plt.hist(df['ConvertedComp'].dropna(), bins=30, edgecolor='black', alpha=0.7)
plt.title('Histogram of Converted Compensation')
plt.xlabel('Converted Compensation')
plt.ylabel('Frequency')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

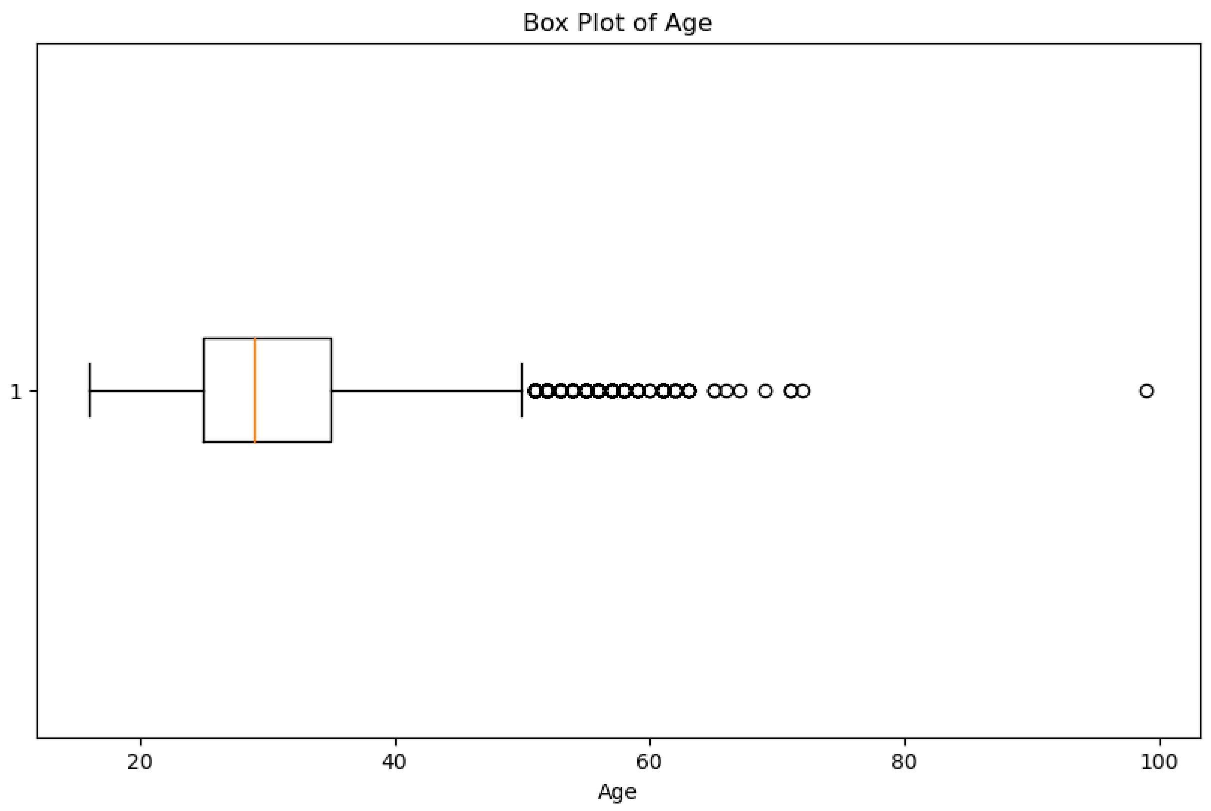


Box Plots

Plot a box plot of `Age.`

In [17]:
```
# Query to get the 'Age' column
query = "SELECT Age FROM master"
df = pd.read_sql_query(query, conn)

# Plot the box plot
plt.figure(figsize=(10, 6))
plt.boxplot(df['Age'].dropna(), vert=False)
plt.title('Box Plot of Age')
plt.xlabel('Age')
plt.show()# your code goes here
```



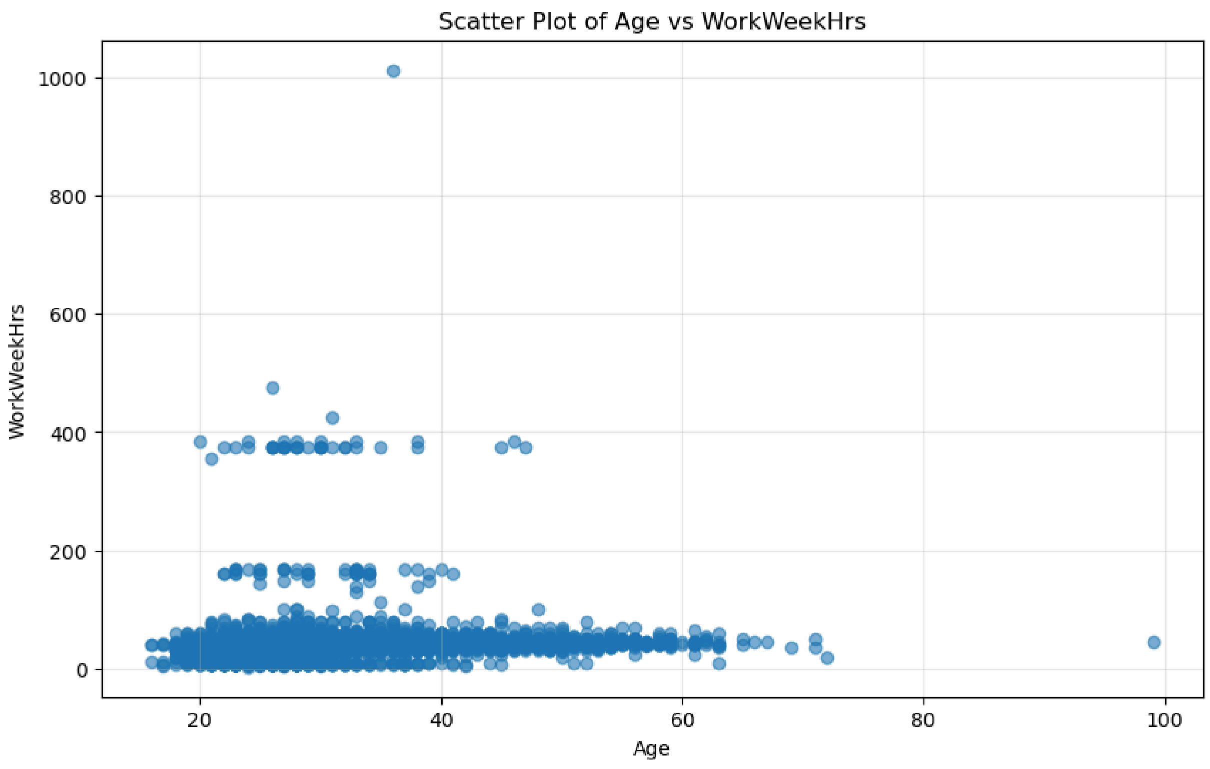
Visualizing relationships in data

Scatter Plots

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

In [18]:
```
# Query to get 'Age' and 'WorkWeekHrs'
query = "SELECT Age, WorkWeekHrs FROM master"
df = pd.read_sql_query(query, conn)

# Plot the scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(df['Age'], df['WorkWeekHrs'], alpha=0.6)
plt.title('Scatter Plot of Age vs WorkWeekHrs')
plt.xlabel('Age')
plt.ylabel('WorkWeekHrs')
plt.grid(alpha=0.3)
plt.show()# your code goes here
```

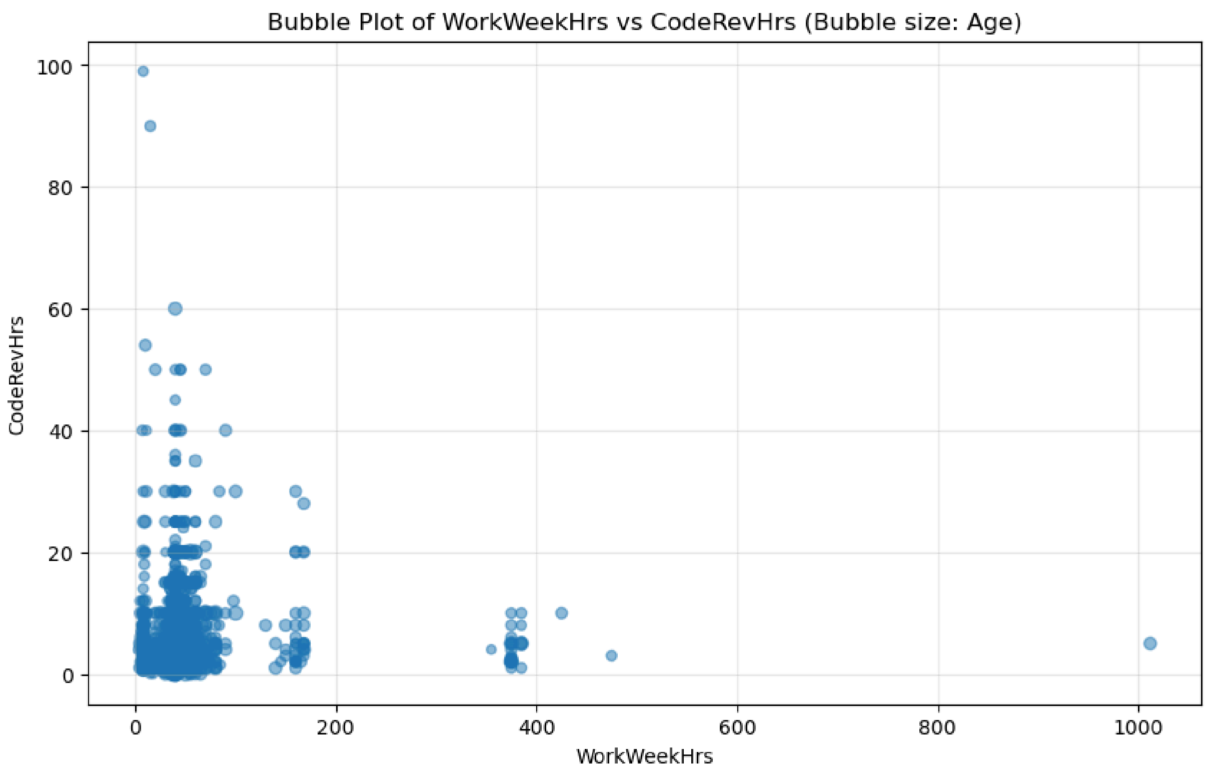


Bubble Plots

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

In [19]:
```
# Query to get 'WorkWeekHrs', 'CodeRevHrs', and 'Age'
query = "SELECT WorkWeekHrs, CodeRevHrs, Age FROM master"
df = pd.read_sql_query(query, conn)

# Plot the bubble plot
plt.figure(figsize=(10, 6))
plt.scatter(df['WorkWeekHrs'], df['CodeRevHrs'], s=df['Age'], alpha=0.5)
plt.title('Bubble Plot of WorkWeekHrs vs CodeRevHrs (Bubble size: Age)')
plt.xlabel('WorkWeekHrs')
plt.ylabel('CodeRevHrs')
plt.grid(alpha=0.3)
plt.show()# your code goes here
```



Visualizing composition of data

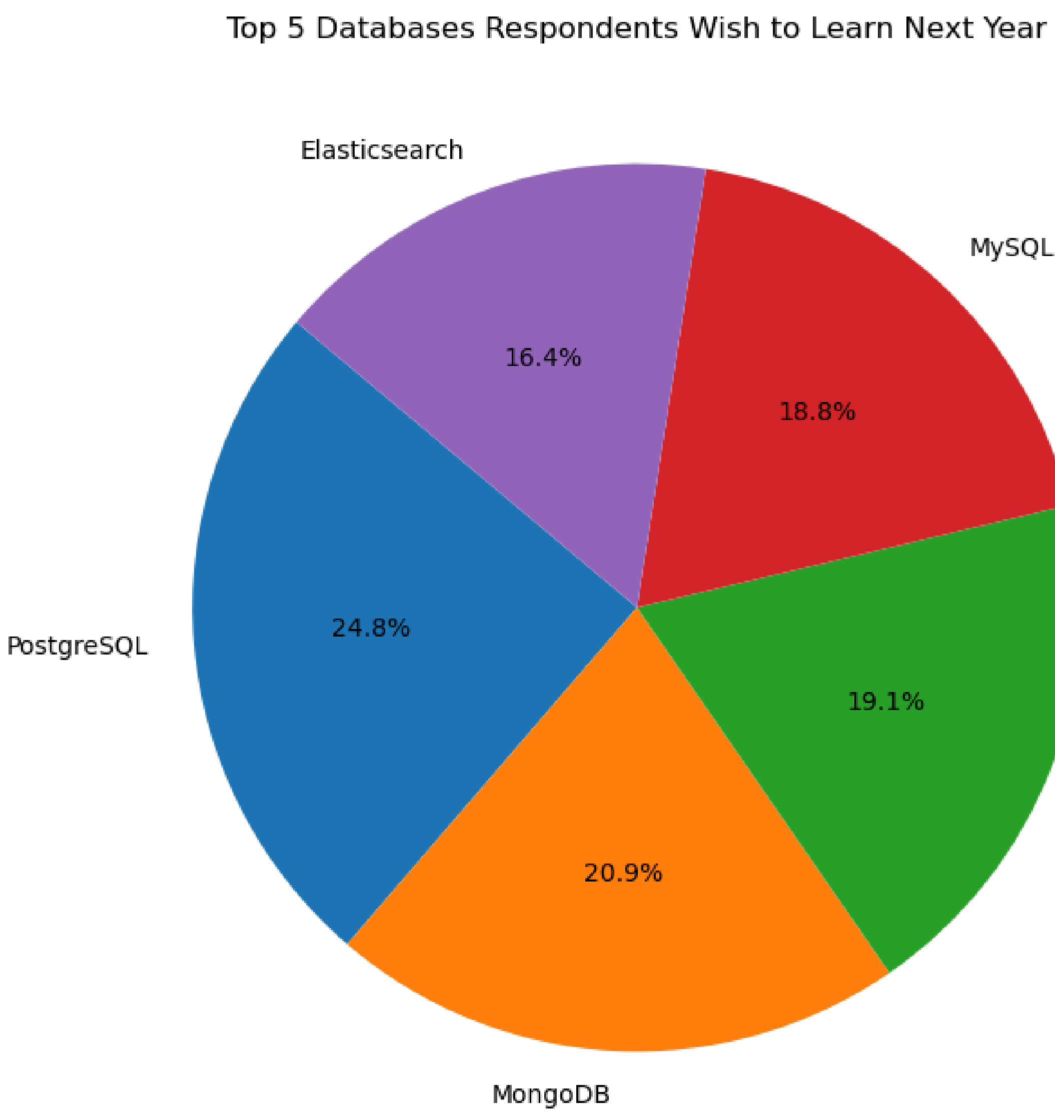
Pie Charts

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 [20]:
```
# Query to get the database respondents wish to learn next year
query = "SELECT DatabaseDesireNextYear FROM DatabaseDesireNextYear"
df = pd.read_sql_query(query, conn)

# Get the top 5 databases
top_databases = df['DatabaseDesireNextYear'].value_counts().head(5)

# Plot the pie chart
plt.figure(figsize=(8, 8))
plt.pie(top_databases, labels=top_databases.index, autopct='%1.1f%%', startangle=14
plt.title('Top 5 Databases Respondents Wish to Learn Next Year')
plt.show()# your code goes here
```



Stacked Charts

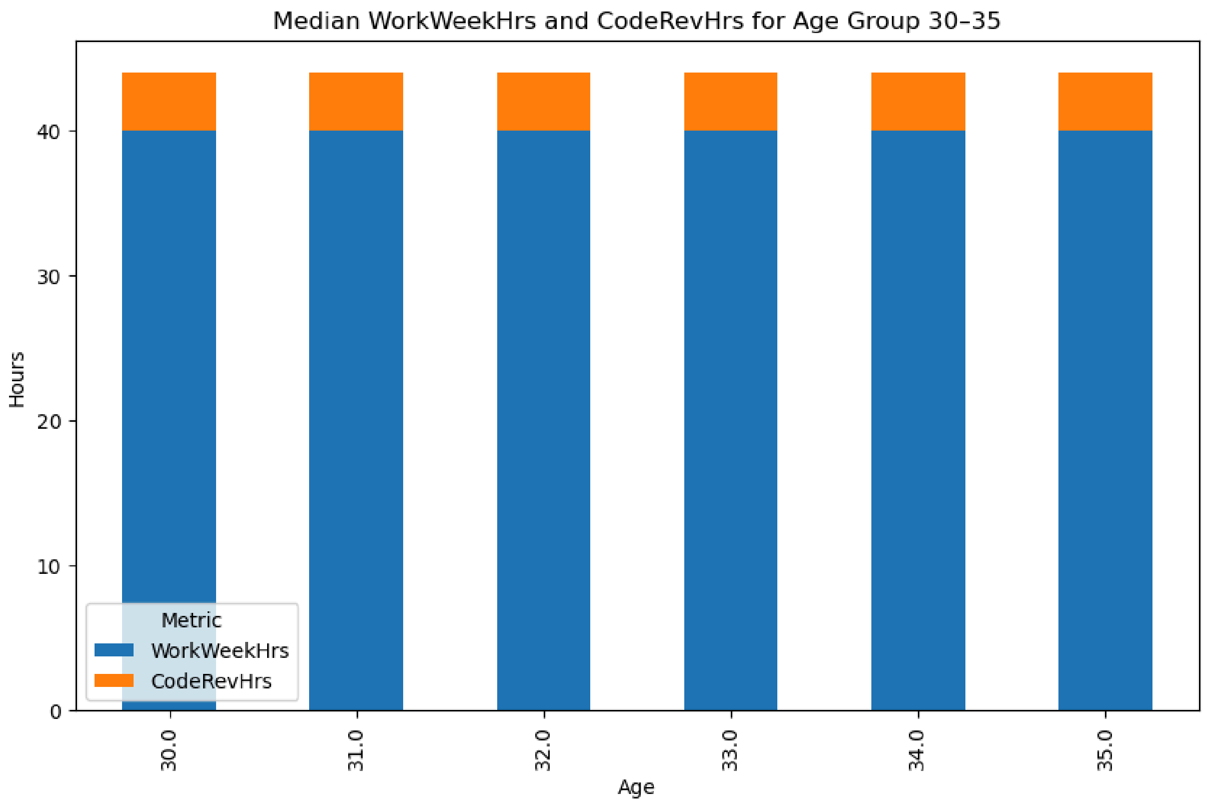
Create a stacked chart of median `WorkWeekHrs` and `CodeRevHrs` for the age group 30 to 35.

In [21]:
```
# Query to get the relevant columns
query = """
SELECT Age, WorkWeekHrs, CodeRevHrs
FROM master
WHERE Age BETWEEN 30 AND 35
"""
df = pd.read_sql_query(query, conn)

# Calculate the median
median_values = df.groupby('Age')[['WorkWeekHrs', 'CodeRevHrs']].median()

# Plot the stacked bar chart
median_values.plot(kind='bar', stacked=True, figsize=(10, 6))
```

```
plt.title('Median WorkWeekHrs and CodeRevHrs for Age Group 30-35')
plt.ylabel('Hours')
plt.xlabel('Age')
plt.legend(title='Metric')
plt.show()# your code goes here
```



Visualizing comparison of data

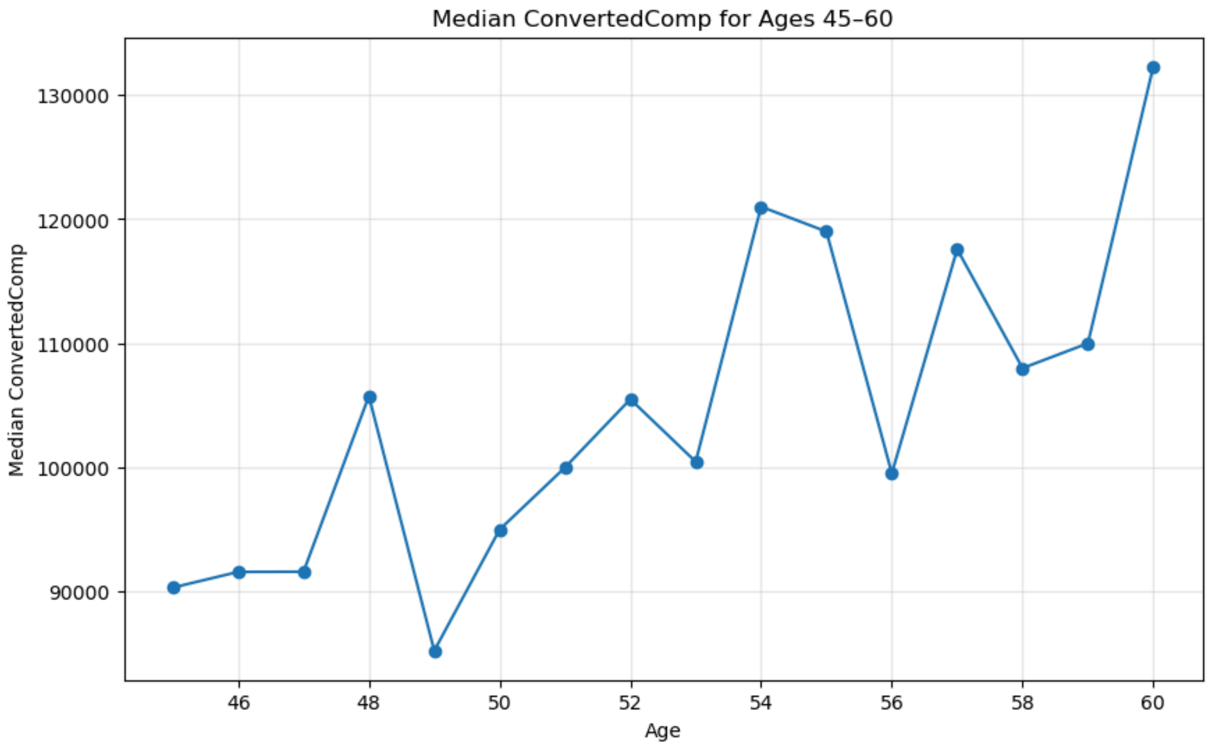
Line Chart

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

In [22]:
```
# Query to get the relevant columns
query = """
SELECT Age, ConvertedComp
FROM master
WHERE Age BETWEEN 45 AND 60
"""
df = pd.read_sql_query(query, conn)

# Calculate the median
median_comp = df.groupby('Age')['ConvertedComp'].median()

# Plot the line chart
plt.figure(figsize=(10, 6))
plt.plot(median_comp.index, median_comp.values, marker='o')
plt.title('Median ConvertedComp for Ages 45-60')
plt.xlabel('Age')
plt.ylabel('Median ConvertedComp')
plt.grid(alpha=0.3)
plt.show()# your code goes here
```



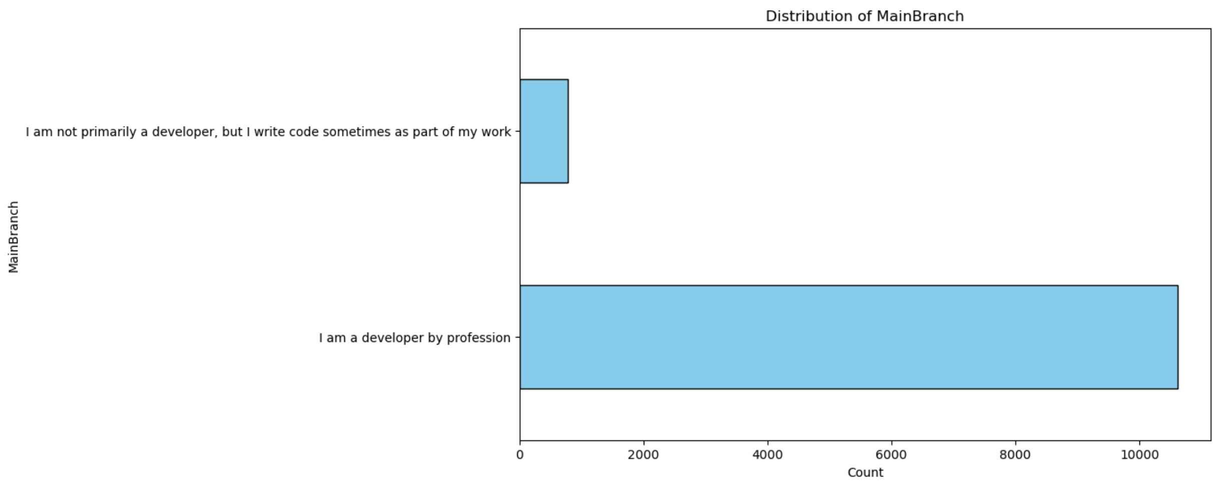
Bar Chart

Create a horizontal bar chart using column `MainBranch.`

In [23]:
```
# Query to get the 'MainBranch' column
query = "SELECT MainBranch FROM master"
df = pd.read_sql_query(query, conn)

# Count values in 'MainBranch'
main_branch_counts = df['MainBranch'].value_counts()

# Plot the horizontal bar chart
main_branch_counts.plot(kind='barh', figsize=(10, 6), color='skyblue', edgecolor='b
plt.title('Distribution of MainBranch')
plt.xlabel('Count')
plt.ylabel('MainBranch')
plt.show()# your code goes here
```



Close the database connection.

```
In [32]: # Import necessary libraries
import pandas as pd
import sqlite3

# Connect to the database
conn = sqlite3.connect("m4_survey_data.sqlite")

# SQL query to count occurrences of each developer type
query = """
SELECT DevType, COUNT(*) AS Count
FROM DevType
GROUP BY DevType
ORDER BY Count DESC
"""

# Execute the query and load the results into a DataFrame
df = pd.read_sql_query(query, conn)

# Get the majority developer type
majority_devtype = df.iloc[0]  # The first row contains the most frequent developer

# Print the result
print(f"The majority developer type is: {majority_devtype['DevType']} with {majorit

# Close the database connection
conn.close()
```

```
The majority developer type is: Developer, full-stack with 6928 respondents.
```

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 |--!>