STAR TALENT
Milestone: Application (Python)

Group 14
Student 1: Pooja Laxmi Sankarakameswaran
Student 2: Sabarish Subramaniam Anandhi Vijayaragavan

(857)-423-0754
(617)-992-5508

sankarakameswaran.p@northeastern.edu
anandhivijayaragav.s@northeastern.edu

Percentage of Effort Contributed by Student1: 50
Percentage of Effort Contributed by Student2: 50

Signature of Student 1: [illegible]

Signature of Student 2: [illegible]

Submission Date: 26th November 2023

1. INTRODUCTION

The purpose of this milestone is to demonstrate the integration of MySQL with Python in a Jupyter Notebook environment. MySQL is a popular relational database management system, and Python is a versatile programming language. Combining these two technologies allows for efficient data manipulation, analysis, and visualization within the Jupyter Notebook interface.

2. MILESTONE OBJECTIVES

- Establish a connection between Python and MySQL.
- Execute SQL queries using Python in Jupyter Notebook.
- Fetch data from MySQL database tables.
- Visualize the data using Python libraries within the Jupyter environment.

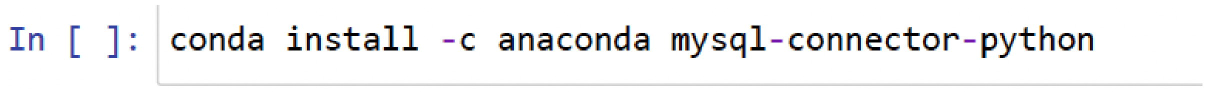
3. TECHNOLOGIES USED

- MySQL: As the relational database management system.
- Python: As the programming language for scripting and data manipulation.
- Jupyter Notebook: As the interactive computing environment for executing Python code.

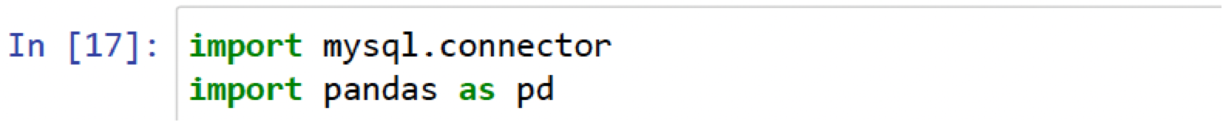
4. IMPLEMENTATION

4.1 Setting up the Environment

Install necessary libraries: `pandas` for data manipulation, `mysql-connector-python` for MySQL connectivity.



```
In [ ]: conda install -c anaconda mysql-connector-python
```



```
In [17]: import mysql.connector
         import pandas as pd
```

4.2 Connecting to MySQL

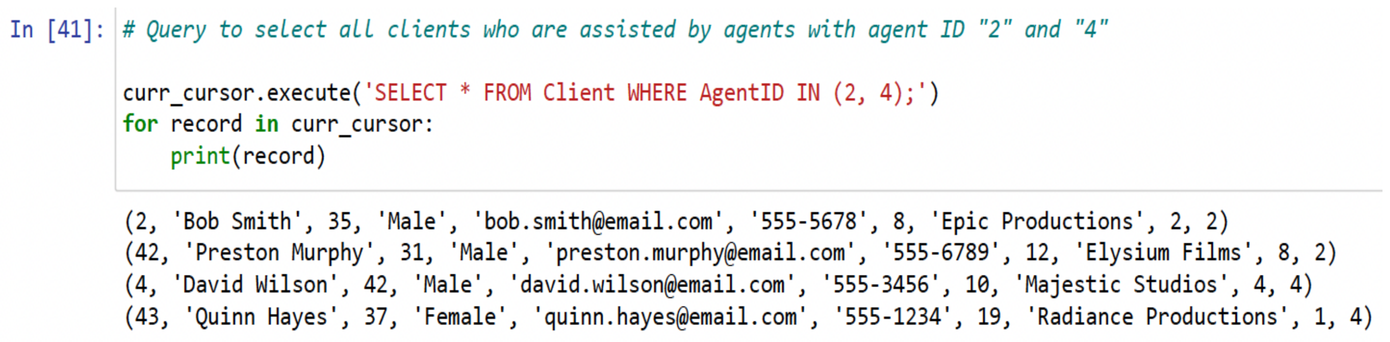
Import the required libraries and establish a connection to the MySQL database.

```
startalent_db = mysql.connector.connect(host = 'localhost', user = 'root', password
= 'admin12345', database = 'startalent')

curr_cursor = startalent_db.cursor()
```

4.3 Executing SQL Queries

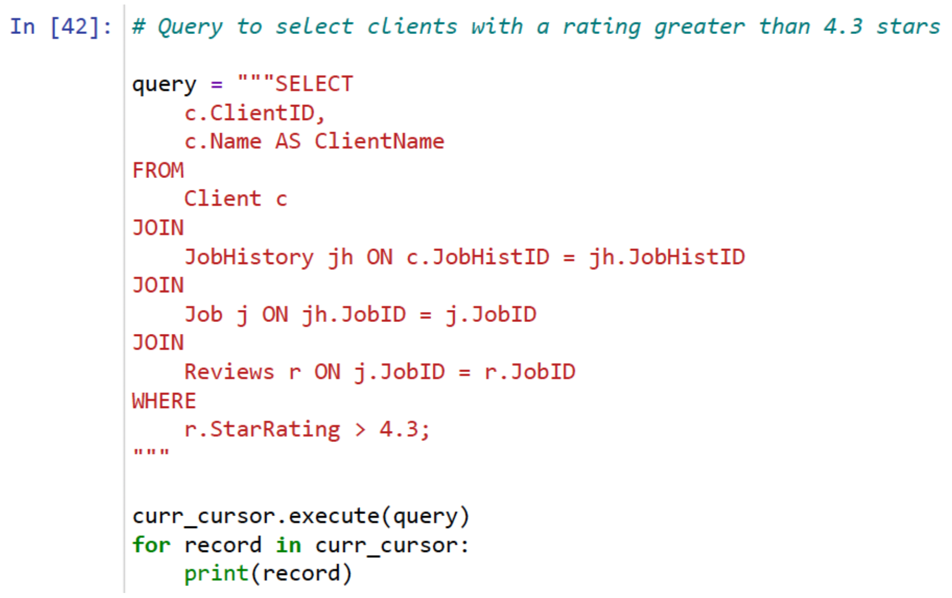
Use the connection to execute SQL queries.

The following SQL query selects all the clients who are assisted by agents with agent IDs "2" and "4".

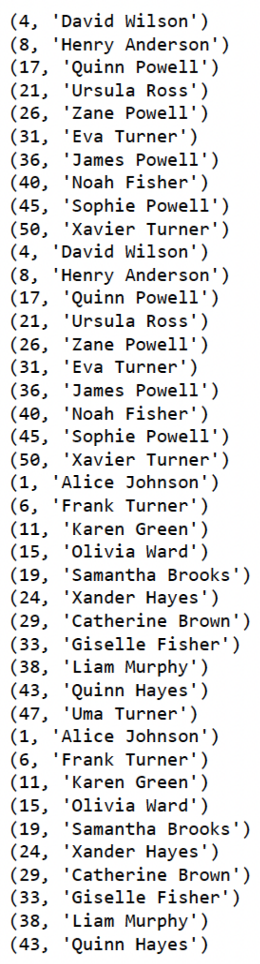


```
In [41]: # Query to select all clients who are assisted by agents with agent ID "2" and "4"

         curr_cursor.execute('SELECT * FROM Client WHERE AgentID IN (2, 4);')
         for record in curr_cursor:
             print(record)

         (2, 'Bob Smith', 35, 'Male', 'bob.smith@email.com', '555-5678', 8, 'Epic Productions', 2, 2)
         (42, 'Preston Murphy', 31, 'Male', 'preston.murphy@email.com', '555-6789', 12, 'Elysium Films', 8, 2)
         (4, 'David Wilson', 42, 'Male', 'david.wilson@email.com', '555-3456', 10, 'Majestic Studios', 4, 4)
         (43, 'Quinn Hayes', 37, 'Female', 'quinn.hayes@email.com', '555-1234', 19, 'Radiance Productions', 1, 4)
```

The following SQL query selects all the clients with greater than a 4.3-star rating.

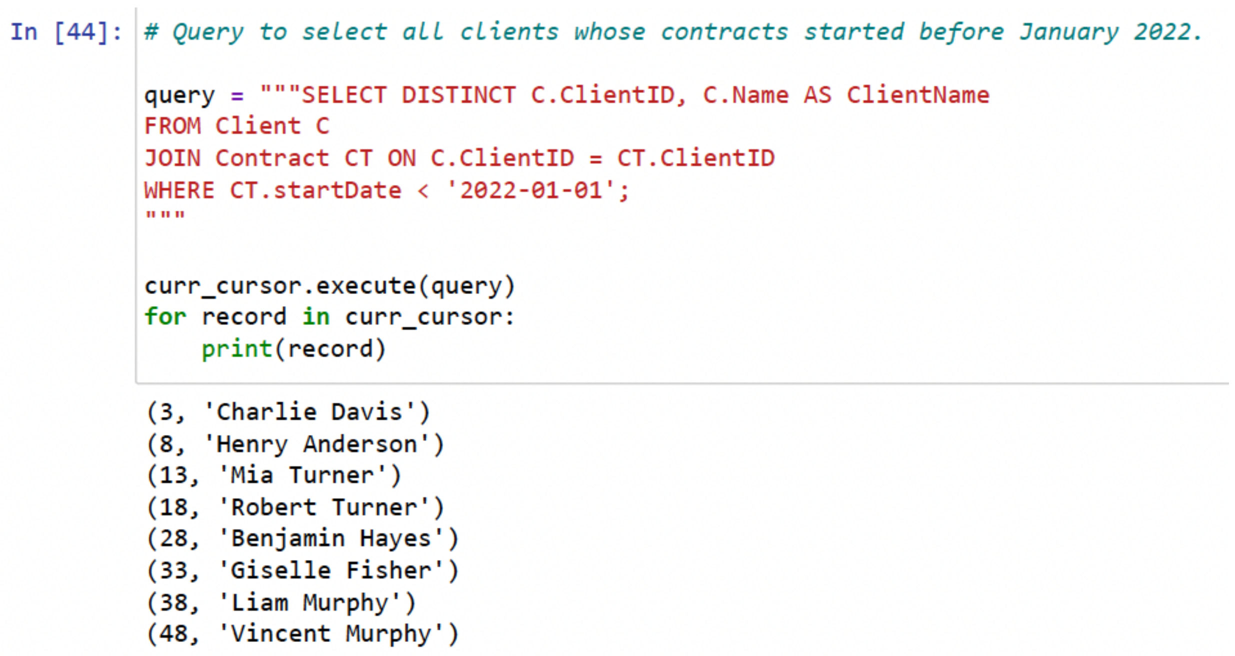


```
In [42]: # Query to select clients with a rating greater than 4.3 stars

         query = """SELECT
             c.ClientID,
             c.Name AS ClientName
         FROM
             Client c
         JOIN
             JobHistory jh ON c.JobHistID = jh.JobHistID
         JOIN
             Job j ON jh.JobID = j.JobID
         JOIN
             Reviews r ON j.JobID = r.JobID
         WHERE
             r.StarRating > 4.3;
         """

         curr_cursor.execute(query)
         for record in curr_cursor:
             print(record)
```



```
(4, 'David Wilson')
(8, 'Henry Anderson')
(17, 'Quinn Powell')
(21, 'Ursula Ross')
(26, 'Zane Powell')
(31, 'Eva Turner')
(36, 'James Powell')
(40, 'Noah Fisher')
(45, 'Sophie Powell')
(50, 'Xavier Turner')
(4, 'David Wilson')
(8, 'Henry Anderson')
(17, 'Quinn Powell')
(21, 'Ursula Ross')
(26, 'Zane Powell')
(31, 'Eva Turner')
(36, 'James Powell')
(40, 'Noah Fisher')
(45, 'Sophie Powell')
(50, 'Xavier Turner')
(1, 'Alice Johnson')
(6, 'Frank Turner')
(11, 'Karen Green')
(15, 'Olivia Ward')
(19, 'Samantha Brooks')
(24, 'Xander Hayes')
(29, 'Catherine Brown')
(33, 'Giselle Fisher')
(38, 'Liam Murphy')
(43, 'Quinn Hayes')
(47, 'Uma Turner')
(1, 'Alice Johnson')
(6, 'Frank Turner')
(11, 'Karen Green')
(15, 'Olivia Ward')
(19, 'Samantha Brooks')
(24, 'Xander Hayes')
(29, 'Catherine Brown')
(33, 'Giselle Fisher')
(38, 'Liam Murphy')
(43, 'Quinn Hayes')
```

The following query selects all the clients whose contracts started before January 2022.

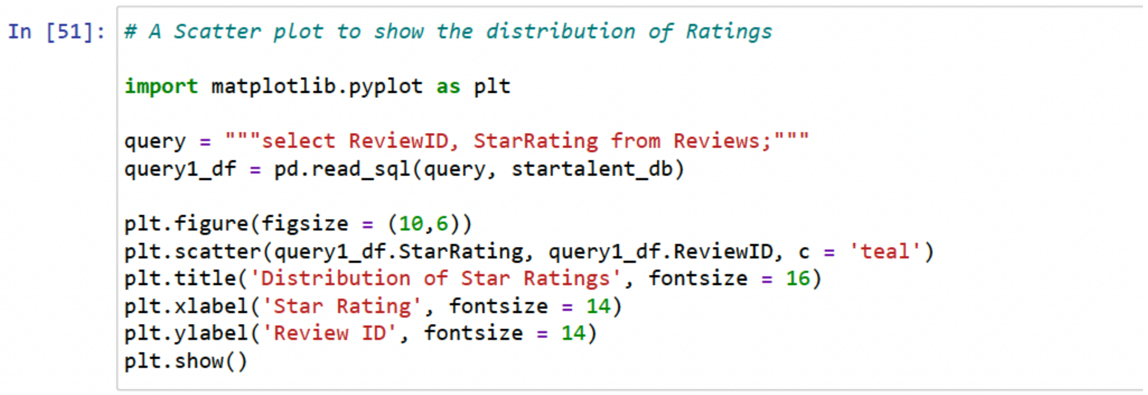


```
In [44]: # Query to select all clients whose contracts started before January 2022.

         query = """SELECT DISTINCT C.ClientID, C.Name AS ClientName
         FROM Client C
         JOIN Contract CT ON C.ClientID = CT.ClientID
         WHERE CT.startDate < '2022-01-01';
         """

         curr_cursor.execute(query)
         for record in curr_cursor:
             print(record)
```

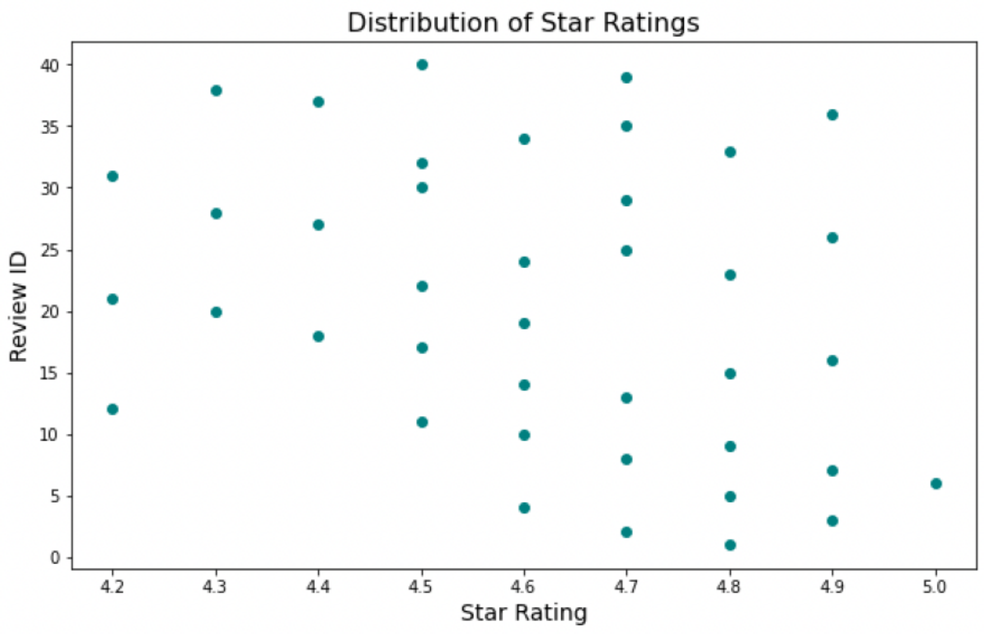
```
(3, 'Charlie Davis')
(8, 'Henry Anderson')
(13, 'Mia Turner')
(18, 'Robert Turner')
(28, 'Benjamin Hayes')
(33, 'Giselle Fisher')
(38, 'Liam Murphy')
(48, 'Vincent Murphy')
```

4.4 Data Visualization

This code will display a scatter plot to show the distribution of Star ratings in all the Reviews.



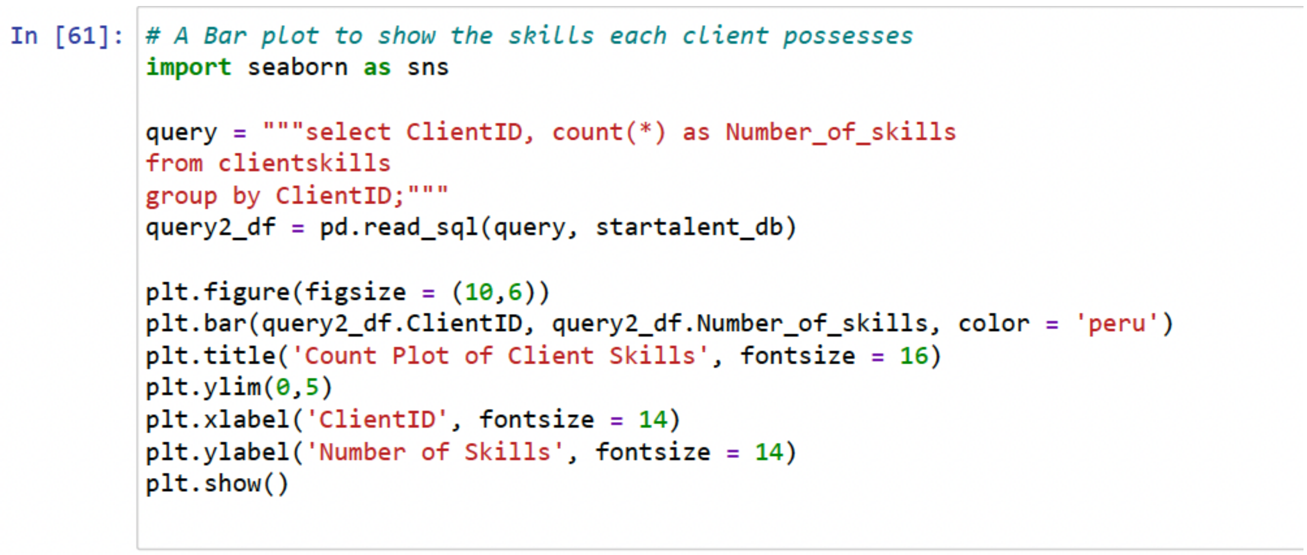
```
In [51]: # A Scatter plot to show the distribution of Ratings

         import matplotlib.pyplot as plt

         query = """select ReviewID, StarRating from Reviews;"""
         query1_df = pd.read_sql(query, startalent_db)

         plt.figure(figsize = (10,6))
         plt.scatter(query1_df.StarRating, query1_df.ReviewID, c = 'teal')
         plt.title('Distribution of Star Ratings', fontsize = 16)
         plt.xlabel('Star Rating', fontsize = 14)
         plt.ylabel('Review ID', fontsize = 14)
         plt.show()
```



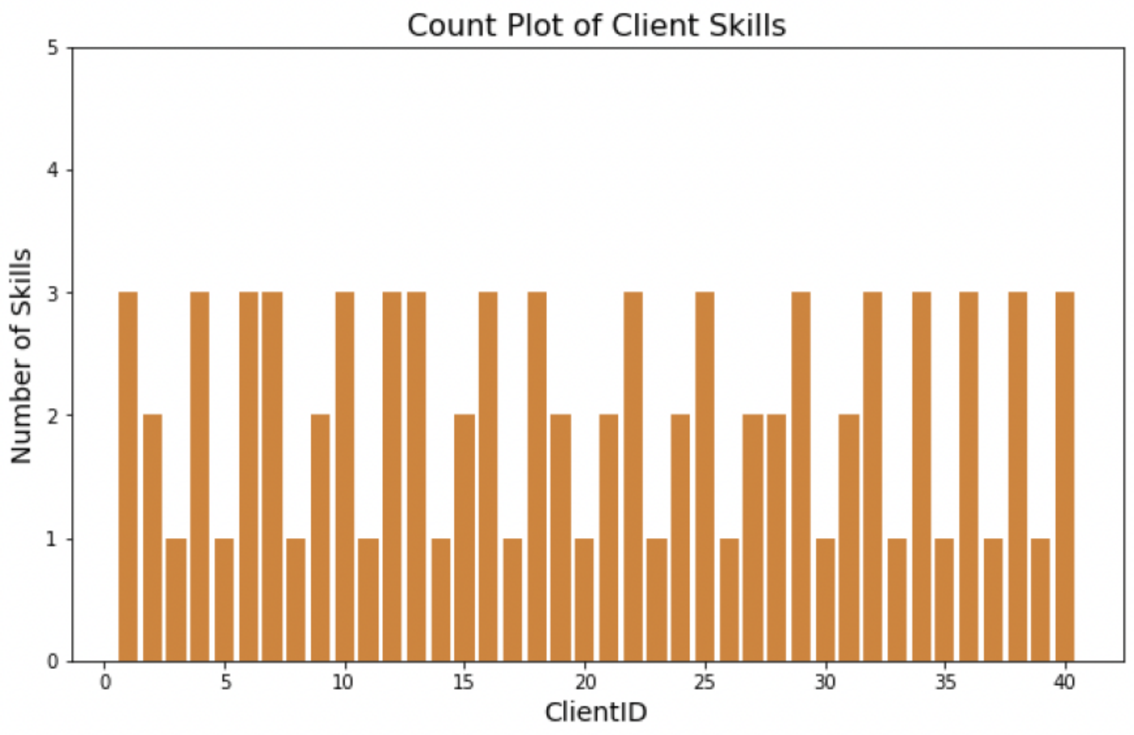
Inference:

The scatter plot shows that the ratings range between 4.2 and 5.0 for all the Clients and are regularly spaced out.

This code will display a bar plot visualizing the number of skills possessed by each client.



```
In [61]: # A Bar plot to show the skills each client possesses
         import seaborn as sns

         query = """select ClientID, count(*) as Number_of_skills
         from clientskills
         group by ClientID;"""
         query2_df = pd.read_sql(query, startalent_db)

         plt.figure(figsize = (10,6))
         plt.bar(query2_df.ClientID, query2_df.Number_of_skills, color = 'peru')
         plt.title('Count Plot of Client Skills', fontsize = 16)
         plt.ylim(0,5)
         plt.xlabel('ClientID', fontsize = 14)
         plt.ylabel('Number of Skills', fontsize = 14)
         plt.show()
```

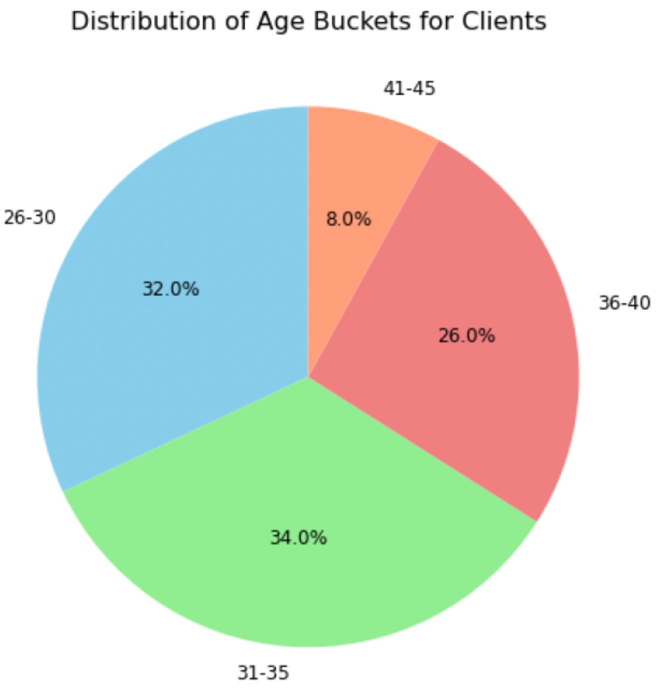


Inference:

The bar graph shows that most clients only have 1-3 skills and this makes them employable across various jobs.

This code will display a pie chart visualizing the distribution of ages among clients.

```
In [71]: # Pie chart to show the composition of Ages of Clients

         query = """select ClientID, Age
         from client;"""
         query3_df = pd.read_sql(query, startalent_db)

         age_buckets = {
             '26-30': range(26, 31),
             '31-35': range(31, 36),
             '36-40': range(36, 41),
             '41-45': range(41, 46),
         }

         # Categorize ages into buckets
         categorized_ages = {bucket: sum(1 for a in query3_df.Age if a in range_) for bucket, range_ in age_buckets.items()}

         # Create a pie chart
         plt.figure(figsize=(8, 8))
         plt.pie(categorized_ages.values(), labels=categorized_ages.keys(), autopct='%1.1f%%',\
                 startangle=90, colors=['skyblue', 'lightgreen', 'lightcoral', 'lightsalmon', 'gold'],\
                 textprops={'fontsize': 12})
         plt.title('Distribution of Age Buckets for Clients', fontsize = 16)
         plt.show()
```



Inference:

The distribution shows that 34% of the clients are aged 31-35 and the next populous age group is 26-30.

5. RESULTS

- Successful connection to the MySQL database.
- Execution of SQL queries and retrieval of data.
- Visualization of data within the Jupyter Notebook environment.

6. CONCLUSION

This milestone demonstrates the seamless integration of MySQL with Python in a Jupyter Notebook environment. The ability to execute SQL queries, fetch data, perform data manipulation, and visualize results within a single platform enhances the efficiency of data analysis workflows.