University Management System

-Saheel Ramji

-Rutuja Doiphode

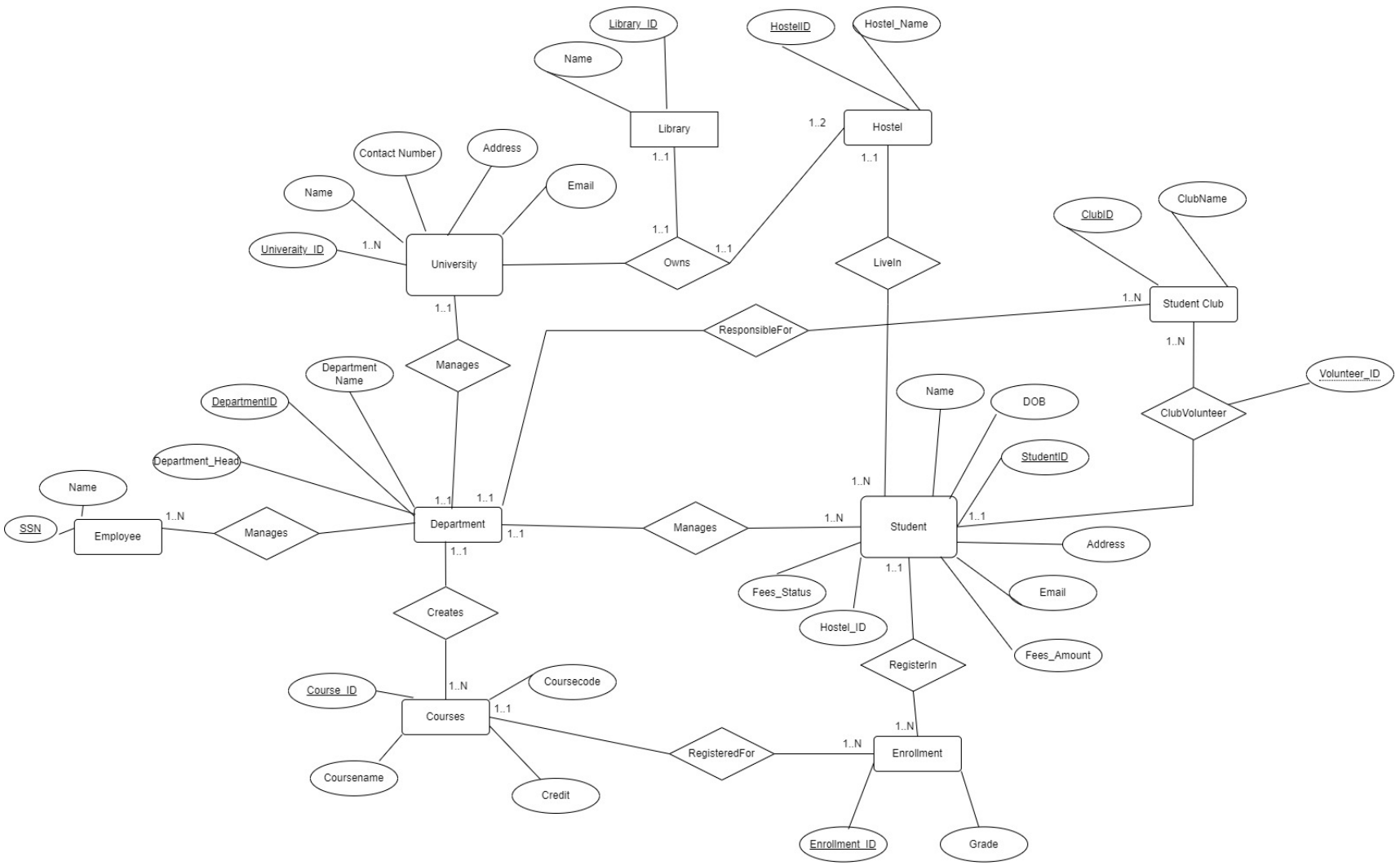
(Group 15)

Problem Statement:

"Revamp university management: Unify operations, automate tasks, and enhance student experience with an integrated system for academic excellence.“

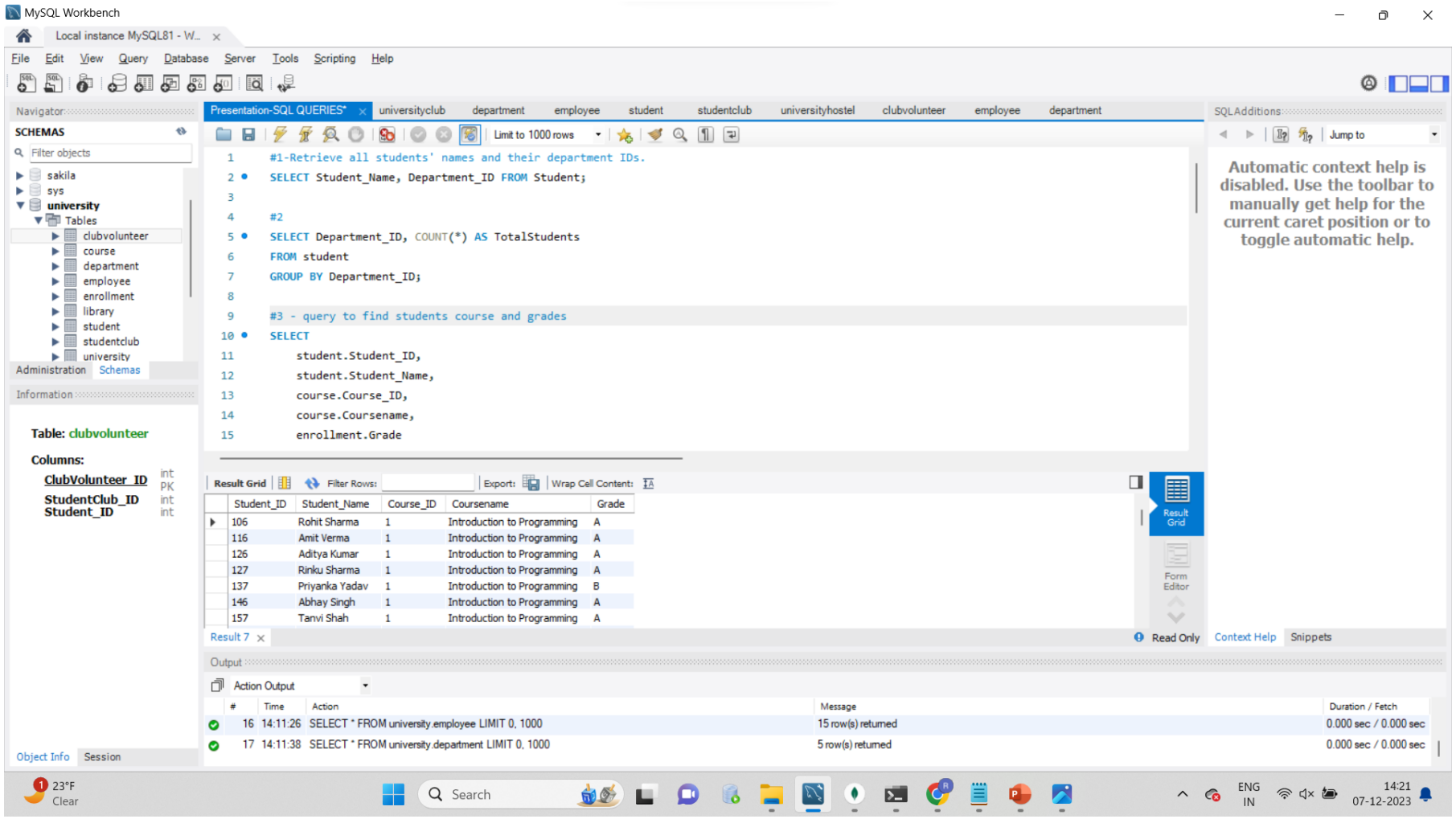
"Current university management systems suffer from inefficiencies, relying on manual processes and disparate tools, leading to delays, errors, and suboptimal resource use. Student enrollment, course management, and administrative tasks lack integration, hindering productivity and student experience. A pressing issue calls for a cohesive University Management System, automating tasks, enhancing data accuracy, and improving overall efficiency. The solution aims to optimize resource allocation, streamline communication, and elevate the university's competitiveness in the education sector."

EER MODEL:


Library_ID
Name
Library
HostelID
Hostel_Name
Hostel
1..2
1..1
1..1
Contact Number
Address
Email
Name
Univeraity_ID
1..N
University
1..1
Owns
1..1
LiveIn
ClubID
ClubName
Student Club
1..N
1..N
ResponsibleFor
1..1
Manages
Department Name
DepartmentID
Department_Head
Name
SSN
Employee
1..N
Manages
1..1
1..1
Department
1..1
Manages
1..N
Student
1..N
Name
DOB
StudentID
1..1
Volunteer_ID
ClubVolunteer
Address
Email
Fees_Amount
Fees_Status
Hostel_ID
1..1
RegisterIn
1..N
Enrollment
Enrollment_ID
Grade
1..N
Creates
1..N
Course_ID
Coursecode
Courses
1..1
Coursename
Credit
RegisteredFor
1..N

RELATIONAL MODEL:

MySQL demonstration:



QUERY 1:

```sql
#1-Retrieve all students' names and their department IDs.
SELECT Student_Name, Department_ID FROM Student;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Student_Name	Department_ID
Rohit Sharma	1
Ananya Desai	2
Karan Patel	3
Simran Singh	4
Aryan Verma	5
Neha Gupta	1
Rahul Shah	2
Anita Kumar	3
Vikas Singh	4
Priya Patel	5
Amit Verma	1
Pooja Jain	2
Manish Agarwal	3
Deepa Sharma	4
Anand Mishra	5
Swati Yadav	1
Vijay Singh	2

Student 8

Query 2:

Presentation-SQL QUERIES* | universityclub | department | employee

Limit to 1000 rows

```sql
#2
SELECT Department_ID, COUNT(*) AS TotalStudents
FROM student
GROUP BY Department_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

Department_ID	TotalStudents
1	33
2	20
3	19
4	24
5	19

QUERY 3:

```sql
#3 - query to find students course and grades
SELECT
    student.Student_ID,
    student.Student_Name,
    course.Course_ID,
    course.Coursename,
    enrollment.Grade
FROM
    student
INNER JOIN enrollment ON student.Student_ID = enrollment.Student_ID
INNER JOIN course ON enrollment.Course_ID = course.Course_ID;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Student_ID	Student_Name	Course_ID	Coursename	Grade
106	Rohit Sharma	1	Introduction to Programming	A
116	Amit Verma	1	Introduction to Programming	A
126	Aditya Kumar	1	Introduction to Programming	A
127	Rinku Sharma	1	Introduction to Programming	A
137	Priyanka Yadav	1	Introduction to Programming	B
146	Abhay Singh	1	Introduction to Programming	A
157	Tanvi Shah	1	Introduction to Programming	A
167	Tanvi Singh	1	Introduction to Programming	A
177	Kritika Patel	1	Introduction to Programming	A

Result 10

Query 4:

Presentation-SQL QUERIES* | universityclub | department | employee | university | universityhostel | department | universityhostel | student

Limit to 1000 rows

```sql
#4 - Nested Query Write a query for list of student names who are in hostel richards hostel and who are in department 4

SELECT Student_Name
FROM Student
WHERE Hostel_ID = (
    SELECT Hostel_ID
    FROM universityhostel
    WHERE Hostel_ID = 1
)
AND Department_ID = 4;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Student_Name
Deepa Sharma
Arjun Verma
Sneha Agarwal
Kavita Verma
Ruchi Shah
Mansi Agarwal
Neha Shah
Anjali Shah
Saurabh Shah

Student 12

QUERY 5:

Presentation-SQL QUERIES* | universityclub | department | employee | student | studentclub

Limit to 1000 rows

```sql
#5 - Correlated query to find department with avg credit more then 3
SELECT Department_ID, Department_Name
FROM department d
WHERE (
    SELECT AVG(Credit)
    FROM course c
    WHERE c.Department_ID = d.Department_ID
) > 3;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell

Department_ID	Department_Name
1	Computer Science
2	IT
3	ELEX
4	Cyber Security
NULL	NULL

QUERY 6:

Presentation-SQL QUERIES* | universityclub | department | employee | university | universityhostel | department

Limit to 1000 rows

```sql
#6 - uses the EXISTS clause to filter students based on the specified conditions.
SELECT s.Student_ID, s.Student_Name, s.Fees_status, e.Grade
FROM student s
JOIN enrollment e ON s.Student_ID = e.Student_ID
WHERE EXISTS (
    SELECT 1
    FROM enrollment
    WHERE Student_ID = s.Student_ID
    AND Grade = 'F'
)
AND s.Fees_status = 'Pending';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Student_ID	Student_Name	Fees_status	Grade
140	Amitabh Shah	pending	F
150	Nitin Mishra	pending	F
160	Rakesh Sharma	pending	F
180	Aditi Yadav	pending	F
200	Jaspreet Verma	pending	F

Presentation-SQL QUERIES* | universityclub | department | employee | university | universityhostel

Limit to 1000 rows

```sql
#7 - Total Number of volunteers in each club using Select From
SELECT
    sc.StudentClub_ID,
    sc.StudentClub_Name,
    (
        SELECT COUNT(*)
        FROM clubvolunteer
        WHERE StudentClub_ID = sc.StudentClub_ID
    ) AS Total_Volunteers
FROM
    studentclub sc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

StudentClub_ID	StudentClub_Name	Total_Volunteers
1	Coding Club	7
2	IT Innovators	7
3	Electronics Enthusiasts	6
4	Cyber Security Task Force	5
5	Robotics Wizards	5
6	Mathematics Enthusiasts	2
7	Physics Club	3
8	Literary Society	1
9	Artists Guild	2

Result 14

Output

Action Output

Presentation-SQL QUERIES* | universityclub | department | employee | university | universityhostel

Limit to 1000 rows

```sql
#8
-- Select students in Coding Club
SELECT s.Student_ID, s.Student_Name
FROM clubvolunteer cv
JOIN studentclub sc ON cv.StudentClub_ID = sc.StudentClub_ID
JOIN student s ON cv.Student_ID = s.Student_ID
WHERE sc.StudentClub_Name = 'Coding Club'

UNION

-- Select students in IT Innovators
SELECT s.Student_ID, s.Student_Name
FROM clubvolunteer cv
JOIN studentclub sc ON cv.StudentClub_ID = sc.StudentClub_ID
JOIN student s ON cv.Student_ID = s.Student_ID
WHERE sc.StudentClub_Name = 'IT Innovators';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Student_ID	Student_Name
111	Neha Gupta
112	Rahul Shah
116	Amit Verma
117	Pooja Jain
118	Manish Agarwal
119	Deepa Sharma
120	Anand Mishra
113	Anita Kumar
114	Vikas Singh

Result 15

NoSQL demonstration

In [29]:
```
#3
#Calculate the average fees paid by students in each department
result = db.Fees.aggregate([
    {"$group": {"_id": "$Couse_name", "average_fees": {"$avg": "$Fees_Paid"}}}
])

for avg_fees in result:
    print(avg_fees)
```

```
{'_id': 'IT', 'average_fees': 3454.1666666666665}
{'_id': 'Robotics', 'average_fees': 3666.6666666666665}
{'_id': 'Computer Science', 'average_fees': 3141.6666666666665}
{'_id': 'ELEX', 'average_fees': 3725.0}
{'_id': 'Cyber Security', 'average_fees': 3254.1666666666665}
```

In [30]:

```python
#4
#counts the number of students in each hostel, including those not assigned to a hostel (null value)
result = db.Student.aggregate([
    {"$group": {"_id": "$Hostel_ID", "student_count": {"$sum": 1}}},
    {"$lookup": {"from": "Hostel", "localField": "_id", "foreignField": "Hostel_ID", "as": "hostel_info"}},
    {"$project": {"hostel_name": "$hostel_info.Hostel_Name", "student_count": 1, "_id": 0}}
])

for student_count in result:
    print(student_count)
```

```
{'student_count': 48, 'hostel_name': ['Snell Hostel']}
{'student_count': 50, 'hostel_name': ['Richards Hostel']}
{'student_count': 17, 'hostel_name': []}
```

In [47]:

```python
# Count total fees status for each department, categorizing by 'Pending' and 'Complete' statuses - Aggergate Query
pipeline_total_fees_status = [
    {
        '$group': {
            '_id': {
                'Department_ID': '$Department_ID',
                'Fees_Status': '$Fees_Status'
            },
            'total_count': {'$sum': 1}
        }
    },
    {
        '$group': {
            '_id': '$_id.Department_ID',
            'fees_status_counts': {
                '$push': {
                    'Fees_Status': '$_id.Fees_Status',
                    'total_count': '$total_count'
                }
            }
        }
    },
    {
        '$project': {
            'Department_ID': '$_id',
            'fees_status_counts': 1,
            '_id': 0
        }
    }
]

# Assuming you've migrated 'Fees' to 'Student' collection, update the collection name
result_total_fees_status = list(db.Student.aggregate(pipeline_total_fees_status))
print(result_total_fees_status)
```

```
[{'fees_status_counts': [{'Fees_Status': 'complete', 'total_count': 3}, {'Fees_Status': 'pending', 'total_count': 30}], 'Depart
ment_ID': 1}, {'fees_status_counts': [{'Fees_Status': 'pending', 'total_count': 21}, {'Fees_Status': 'complete', 'total_count':
3}], 'Department_ID': 4}, {'fees_status_counts': [{'Fees_Status': 'complete', 'total_count': 3}, {'Fees_Status': 'pending', 'to
tal_count': 16}], 'Department_ID': 5}, {'fees_status_counts': [{'Fees_Status': 'complete', 'total_count': 3}, {'Fees_Status':
'pending', 'total_count': 17}], 'Department_ID': 2}, {'fees_status_counts': [{'Fees_Status': 'pending', 'total_count': 13}, {'F
ees_Status': 'complete', 'total_count': 6}], 'Department_ID': 3}]
```

Part IV: Application demonstration

In [1]:
```
!pip install mysql-connector-python
```

```
Requirement already satisfied: mysql-connector-python in c:\programdata\anaconda3\lib\site-packages (8.2.0)
Requirement already satisfied: protobuf<=4.21.12,>=4.21.1 in c:\programdata\anaconda3\lib\site-packages (from mysql-connector-p
ython) (4.21.12)
```

In [ ]:

In [6]:
```
import mysql.connector
from mysql.connector import Error

# Replace these values with your actual database credentials
host = "localhost"
user = "root"
password = "Rrutuja@094"
database = "university"
port=3306
try:
    connection = mysql.connector.connect(
        host=host,
        user=user,
        password=password,
        database=database
    )

    if connection.is_connected():
        print("Connected to MySQL database")

except Error as e:
    print(f"Error: {e}")
```

```
Connected to MySQL database
```

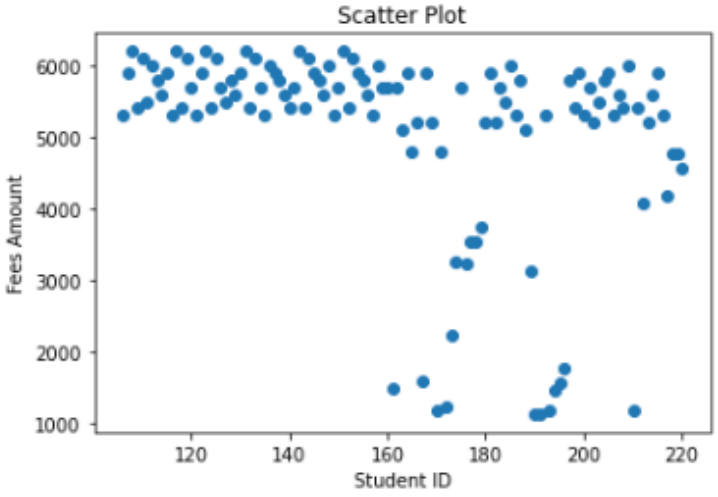
```python
# Retrieve employees in Department 3
employee_query = "SELECT Emp_SSN, Emp_Name FROM employee WHERE Emp_Department_ID = 3;"
cursor.execute(employee_query)
employees = cursor.fetchall()
print("Employees in Department 3:")
for employee in employees:
    print(employee)

# Retrieve students in Department 3 who have paid full fees
student_query = """
SELECT Student_ID, Student_Name, Student_Email, Student_Address, Student_DOB, Department_ID, Fees_Amount, Hostel_ID, Fees_Status
FROM student
WHERE Department_ID = 3
AND Student_ID IN (
    SELECT Student_ID
    FROM student
    WHERE Fees_Status = 'Complete'
);
"""
cursor.execute(student_query)
students = cursor.fetchall()
print("\nStudents in Department 3 who paid full fees with fee status complete:")
for student in students:
    print(student)
```

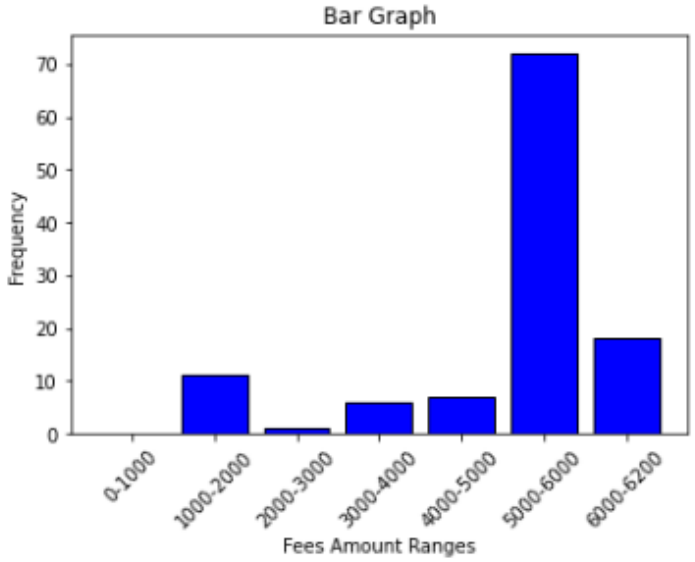
```
Employees in Department 3:
(1003, 'Rahul Singh')
(1006, 'Karan Patel')
(1014, 'Raj Patel')

Students in Department 3 who paid full fees with fee status complete:
(108, 'Karan Patel', 'karan.patel@example.com', '543 NOP Street', 19991025, 3, 6200, None, 'complete')
(123, 'Nisha Patel', 'nisha.patel@example.com', '543 NOP Street', 19991018, 3, 6200, 2, 'complete')
(133, 'Anu Sharma', 'anu.sharma@example.com', '543 ABC Lane', 19990120, 3, 6100, 1, 'complete')
(148, 'Rahul Patel', 'rahul.patel@example.com', '876 ABC Lane', 19990825, 3, 6000, 2, 'complete')
(153, 'Ankita Sharma', 'ankita.sharma@example.com', '543 RST Road', 19990915, 3, 6100, 1, 'complete')
(158, 'Sumit Patel', 'sumit.patel@example.com', '876 JKL Junction', 19980422, 3, 6000, 2, 'complete')
```

```python
#A correlated SQL query to retrieve the number of volunteers in studentclub, except for the two studentclubs with the lowest numb
# Create a cursor
cursor = connection.cursor()

# SQL query to retrieve clubvolunteers' information
query = """
SELECT sc.StudentClub_Name, COUNT(cv.Student_ID) AS Student_Count
FROM studentclub sc
LEFT JOIN clubvolunteer cv ON sc.StudentClub_ID = cv.StudentClub_ID
WHERE sc.StudentClub_ID NOT IN (
    SELECT StudentClub_ID
    FROM (
        SELECT StudentClub_ID, ROW_NUMBER() OVER (ORDER BY COUNT(Student_ID) ASC) AS ClubRank
        FROM clubvolunteer
        GROUP BY StudentClub_ID
    ) AS ClubRanking
    WHERE ClubRank <= 3
)
GROUP BY sc.StudentClub_Name;
"""


# Execute the query
cursor.execute(query)

# Fetch and print the result
result = cursor.fetchall()
for row in result:
    print(row)
```

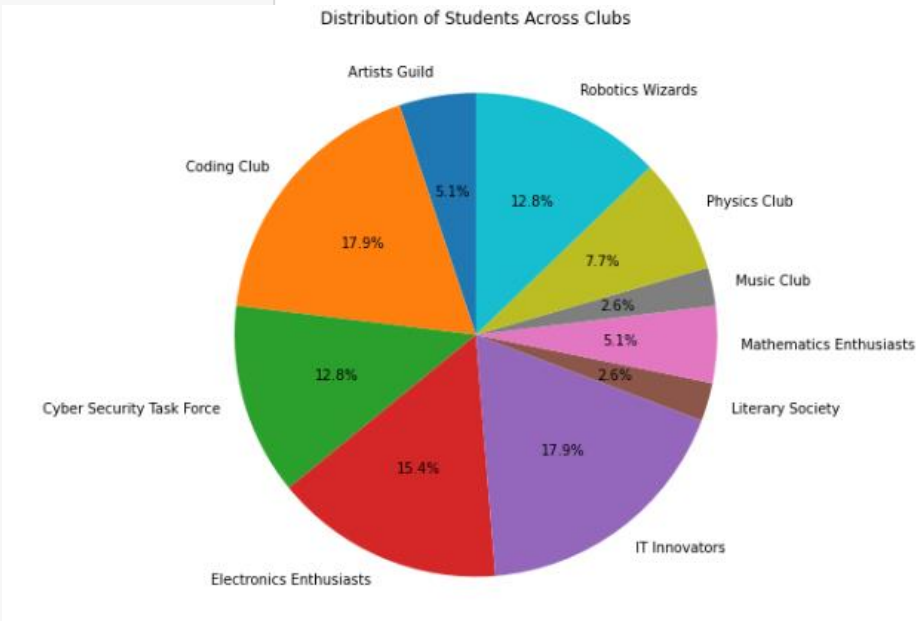
```
('Coding Club', 7)
('IT Innovators', 7)
('Electronics Enthusiasts', 6)
('Cyber Security Task Force', 5)
('Robotics Wizards', 5)
('Physics Club', 3)
('Artists Guild', 2)
```

```python
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns



# Connect to MySQL database
mysql_connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="Rrutuja@094",
        database="university"
)

# Query data from MySQL
mysql_query = "SELECT Student_ID, Student_Name, Fees_Amount FROM student"
mysql_data = pd.read_sql(mysql_query, mysql_connection)

# Close MySQL connection
mysql_connection.close()

# Plot Scatter Plot
plt.scatter(mysql_data['Student_ID'], mysql_data['Fees_Amount'])
plt.title('Scatter Plot')
plt.xlabel('Student ID')
plt.ylabel('Fees Amount')
plt.show()


import numpy as np

# Define the fee ranges
fee_ranges = [0, 1000, 2000, 3000, 4000, 5000, 6000, 6200]

# Create histogram data with specified bins
hist, bins = np.histogram(mysql_data['Fees_Amount'], bins=fee_ranges)

# Plot Bar Graph
plt.bar(range(len(hist)), hist, align='center', color='blue', edgecolor='black', width=0.8)
plt.xticks(range(len(bins) - 1), [f"{bins[i]}-{bins[i + 1]}" for i in range(len(bins) - 1)], rotation=45)
plt.title('Bar Graph')
plt.xlabel('Fees Amount Ranges')
plt.ylabel('Frequency')
plt.show()
```

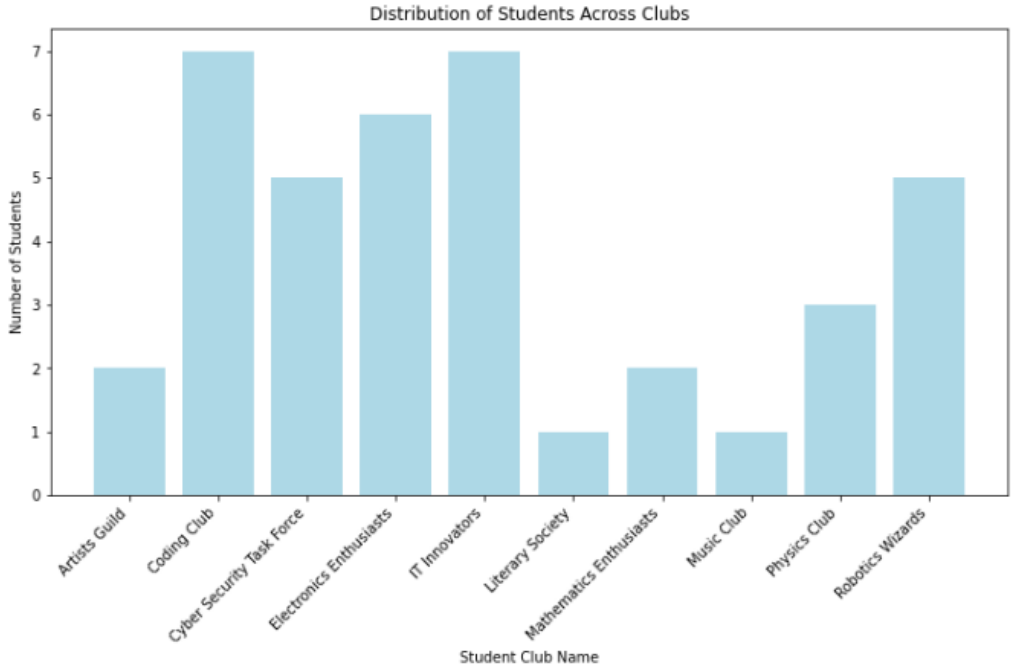




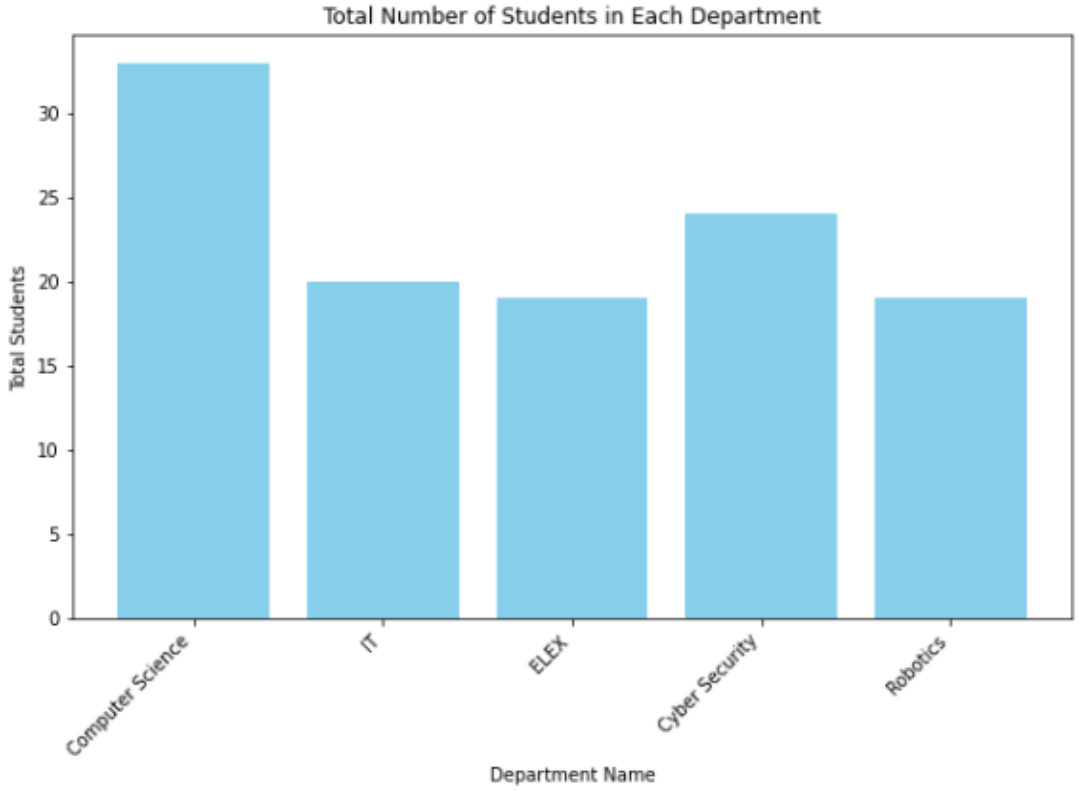
```python
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt

# Connect to MySQL database
mysql_connection = mysql.connector.connect(
    host="localhost",
    user="root",
    password="Rrutuja@094",
    database="university"
)

# Query data from MySQL
mysql_query = """
    SELECT sc.StudentClub_Name, COUNT(DISTINCT cv.Student_ID) as Total_Students
    FROM clubvolunteer cv
    JOIN studentclub sc ON cv.StudentClub_ID = sc.StudentClub_ID
    GROUP BY sc.StudentClub_Name;
"""
club_students_data = pd.read_sql(mysql_query, mysql_connection)

# Close MySQL connection
mysql_connection.close()

#3
# Plot Pie Chart
plt.figure(figsize=(10, 8))
plt.pie(club_students_data['Total_Students'], labels=club_students_data['StudentClub_Name'], autopct='%1.1f%%', startangle=90)
plt.title('Distribution of Students Across Clubs')
plt.show()
```



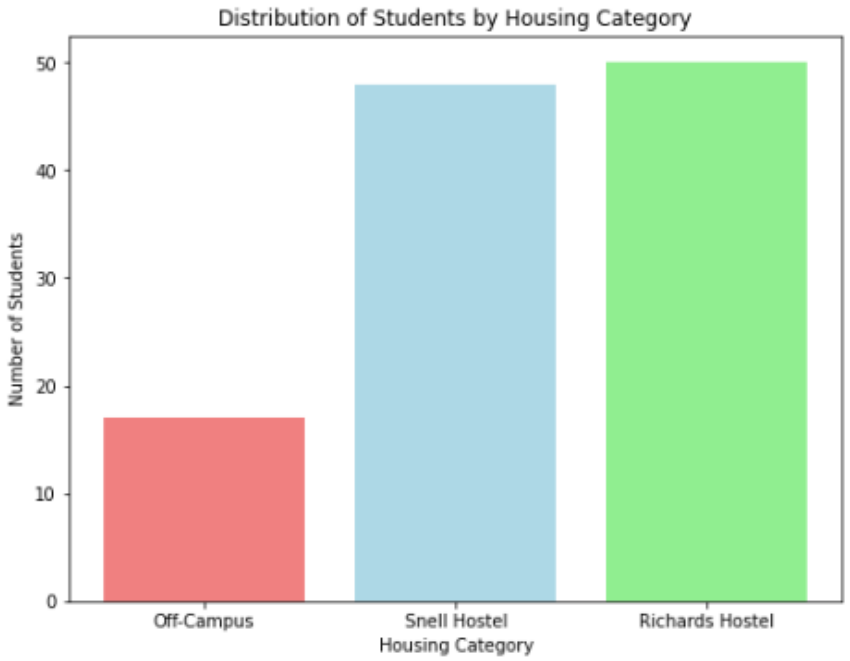
```python
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt

# Connect to MySQL database
mysql_connection = mysql.connector.connect(
    host="localhost",
    user="root",
    password="Rrutuja@094",
    database="university"
)

# Query data from MySQL
mysql_query = """
    SELECT sc.StudentClub_Name, COUNT(DISTINCT cv.Student_ID) as Total_Students
    FROM clubvolunteer cv
    JOIN studentclub sc ON cv.StudentClub_ID = sc.StudentClub_ID
    GROUP BY sc.StudentClub_Name;
"""
club_students_data = pd.read_sql(mysql_query, mysql_connection)

# Close MySQL connection
mysql_connection.close()


#3
# Plot Bar Chart
plt.figure(figsize=(12, 6))
plt.bar(club_students_data['StudentClub_Name'], club_students_data['Total_Students'], color='lightblue')
plt.title('Distribution of Students Across Clubs')
plt.xlabel('Student Club Name')
plt.ylabel('Number of Students')
plt.xticks(rotation=45, ha='right')
plt.show()
```



```
#4
# Plot Bar Chart
plt.figure(figsize=(10, 6))
plt.bar(department_data['Department_Name'], department_data['Total_Students'], color='skyblue')
plt.title('Total Number of Students in Each Department')
plt.xlabel('Department Name')
plt.ylabel('Total Students')
plt.xticks(rotation=45, ha='right')
plt.show()
```



```python
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt

# Connect to MySQL database
mysql_connection = mysql.connector.connect(
    host="localhost",
    user="root",
    password="Rrutuja@094",
    database="university"
)

# Query data from MySQL
mysql_query = "SELECT COUNT(CASE WHEN Hostel_ID IS NULL THEN 1 END) as Off_Campus, " \
              "COUNT(CASE WHEN Hostel_ID = 1 THEN 1 END) as Snell_Hostel, " \
              "COUNT(CASE WHEN Hostel_ID = 2 THEN 1 END) as Richards_Hostel " \
              "FROM student;"
housing_data = pd.read_sql(mysql_query, mysql_connection)

# Close MySQL connection
mysql_connection.close()


#2
# Plot Bar Graph
plt.figure(figsize=(8, 6))
categories = ['Off-Campus', 'Snell Hostel', 'Richards Hostel']
plt.bar(categories, housing_data.iloc[0], color=['lightcoral', 'lightblue', 'lightgreen'])
plt.title('Distribution of Students by Housing Category')
plt.xlabel('Housing Category')
plt.ylabel('Number of Students')
plt.show()
```



```python
import pandas as pd
import matplotlib.pyplot as plt
from sqlalchemy import create_engine

# Replace with your MySQL connection details
host = 'localhost'
user = 'root'
password = 'Rrutuja@094'
database = 'university'

# Create a MySQL connection
engine = create_engine(f"mysql+mysqlconnector://{user}:{password}@{host}/{database}")

# Replace with your actual SQL query to fetch grade distribution data
query = "SELECT Grade, COUNT(*) as Count FROM enrollment GROUP BY Grade"

# Fetch data from MySQL and create a DataFrame
df = pd.read_sql_query(query, engine)

# Plotting the bar chart
plt.bar(df['Grade'], df['Count'], color=['green', 'blue', 'orange', 'red'])
plt.xlabel('Grades')
plt.ylabel('Number of Students')
plt.title('Distribution of Grades')
plt.show()
```