Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Fall,Year:2024),B.Sc.in CSE (Day)

LAB REPORT NO - 02

Course Title: Database System

Course Code: CSE210 Section:222-D3

Lab Experiment Name : Implement Querying and Filtering by following Lab manual 5 and 6.

Student Details

	Name	ID
1.	MD.SHAJALAL	223002088

Lab Date : **02 - 10 - 2024**

Submission Date : **09 – 10 – 2024**

Course Teacher's Name : Farhana Akter Sunny

Lab Report Status

Marks: …… **Signature:.....**

Comments:..... **Date:.....**

1. **TITLE OF THE LAB REPOT EXPERIMENT**

   Implement Querying and Filtering by following Lab manual 5 and 6.

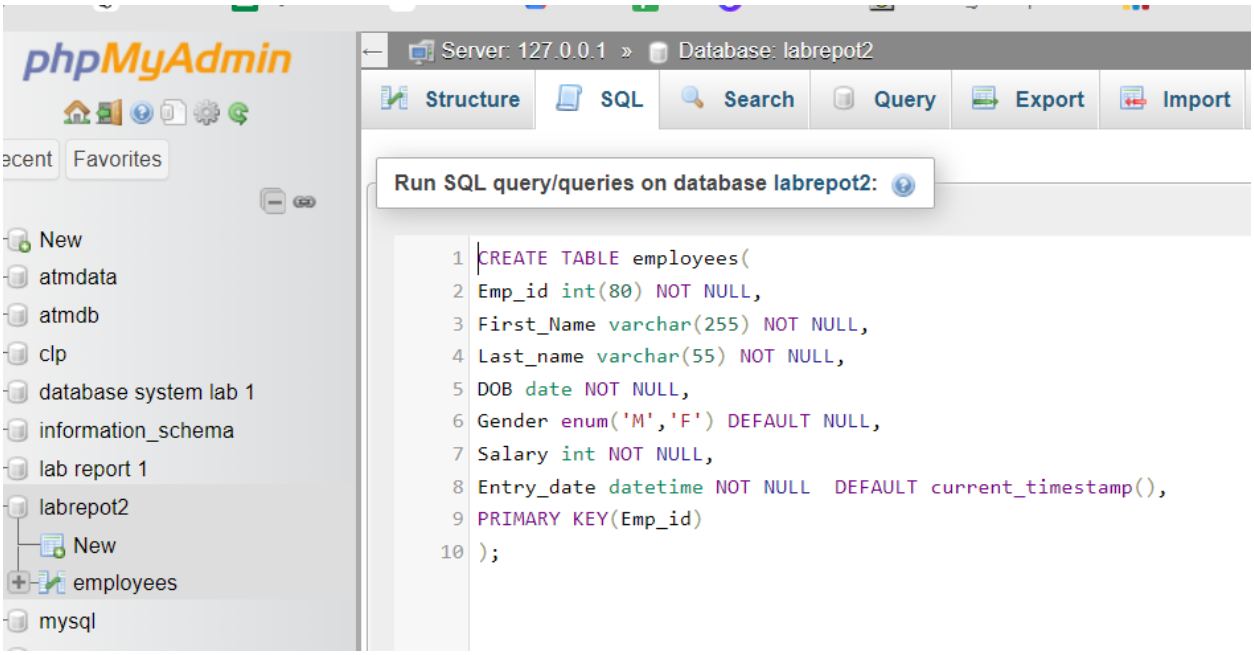
2. **OBJECTIVES**

   - To explore SQL operations such as DISTINCT, WHERE, and comparison operators (<, >, <=, >=, <>) on the employees table to retrieve specific data and eliminate duplicates.

   - The primary objective of this lab is to explore the implementation of SQL queries using logical operators (AND, OR, NOT), pattern matching with LIKE, filtering data with BETWEEN, IN, and LIMIT clauses, and handling NULL values in MySQL databases.

3.IMPLEMENTATION

1. Using MySQL SELECT statement to query data(Create a employees table):



Insert Multiple VALUES at a time:

```
INSERT INTO employees (Emp_id, First_Name, Last_name,DOB, Gender, Salary)
VALUES (1, ‘Md.Shajalal’, ‘sojib’,‘1998-08-02’, ‘M’, 50000),
(2, ‘Sakib’, ‘Hasan’,‘1998-08-02’, ‘M’, 20000),
(3, ‘Ananna’, ‘Rahman’,‘1998-08-02’, ‘F’, 40000),
(4, ‘Jannat’, ‘Hasan’,‘1998-08-02’, ‘F’, 45000),
(5, ‘Sabbir ’, ‘Rahman’,‘1998-08-02’, ‘M’, 30000);
```

Output:

Emp_id	First_Name	Last_name	DOB	Gender	Salary	Entry_date
1	Md.Shajalal	sojib	1998-08-02	M	50000	2024-10-09 19:34:24
2	Sakib	Hasan	1998-08-02	M	20000	2024-10-09 19:34:24
3	Ananna	Rahman	1998-08-02	F	40000	2024-10-09 19:34:24
4	Jannat	Hasan	1998-08-02	F	45000	2024-10-09 19:34:24
5	Sabbir	Rahman	1998-08-02	M	30000	2024-10-09 19:34:24

- **Eliminating Duplicate Rows with DISTINCT:**

This query eliminates duplicates by selecting unique combinations of First_Name and Last_name. The result will show distinct rows, ignoring any repeated values.

```
SELECT DISTINCT First_Name, Last_name

FROM employees;
```

Output:

			First_Name	Last_name
dit	Copy	Delete	Md.Shajalal	sojib
dit	Copy	Delete	Sakib	Hasan
dit	Copy	Delete	Ananna	Rahman
dit	Copy	Delete	Jannat	Hasan
dit	Copy	Delete	Sabbir	Rahman

- **Filtering Rows Using WHERE:**

a. Using WHERE Clause for Integer Value:

This query retrieves the First_Name, Last_name, and Salary of the employee whose Emp_id is equal to 2.

```
SELECT First_Name, Last_name, Salary
FROM employees
WHERE Emp_id = 2;
```

Output:

	First_Name	Last_name	Salary
Edit Copy Delete	Sakib	Hasan	20000

b. Using WHERE Clause for String Value:

This query retrieves the details (Emp_id, Last_name, DOB, Salary, and Entry_date) of employees whose first name is 'Sabbir'.

```
SELECT Emp_id, Last_name, DOB, Salary, Entry_date
FROM employees
WHERE First_Name = 'Sabbir';
```

Output:

Emp_id	First_Name	Last_name	DOB	Gender	Salary	Entry_date
1	Md.Shajalal	sojib	1998-08-02	M	50000	2024-10-09 19:34:24
2	Sakib	Hasan	1998-08-02	M	20000	2024-10-09 19:34:24
3	Ananna	Rahman	1998-08-02	F	40000	2024-10-09 19:34:24
4	Jannat	Hasan	1998-08-02	F	45000	2024-10-09 19:34:24
5	Sabbir	Rahman	1998-08-02	M	30000	2024-10-09 19:34:24

➢ Using Comparison Operators:

Example 1: Using >= (Greater Than or Equal To):

This query fetches the Emp_id, First_Name, and Last_name of all employees whose Salary is greater than or equal to 40,000.

```
SELECT Emp_id, First_Name, Last_name
FROM employees
WHERE Salary >= 40000;
```

Output:

Emp_id	First_Name	Last_name
1	Md.Shajalal	sojib
3	Ananna	Rahman
4	Jannat	Hasan

Example 2: Using <> (Not Equal To):

This query retrieves the Emp_id, First_Name, and Last_name of all employees whose Salary is not equal to 30,000.

```
SELECT Emp_id, First_Name, Last_name

FROM employees

WHERE Salary <> 30000;
```

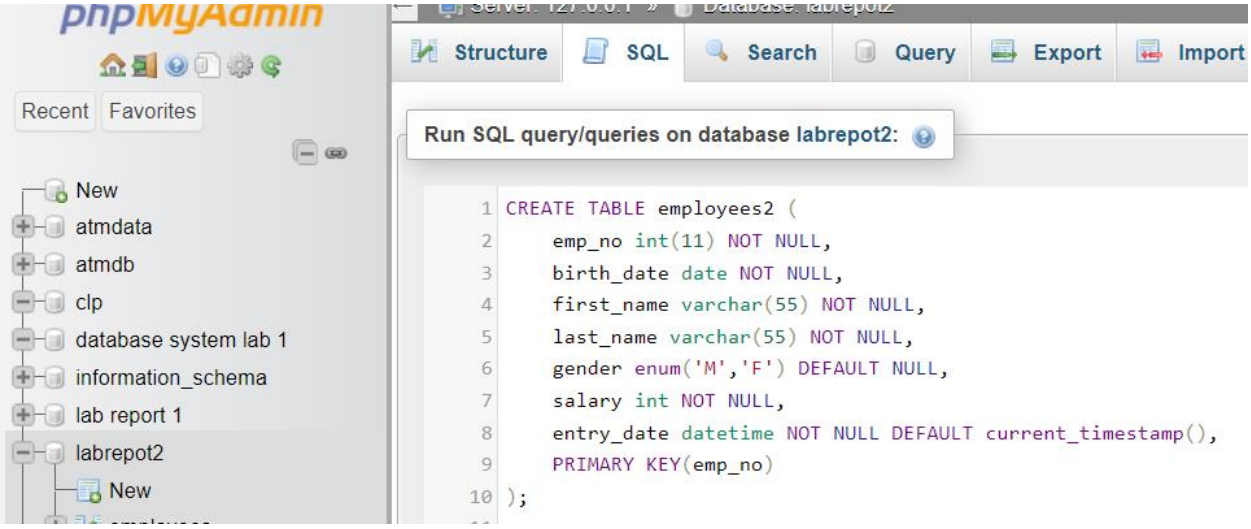
Output:

	Emp_id	First_Name	Last_name
e	1	Md.Shajalal	sojib
e	2	Sakib	Hasan
e	3	Ananna	Rahman
e	4	Jannat	Hasan

Lab-6:

Create employees Table:

The table is created with columns like emp_no, first_name, last_name, gender, salary, and entry_date.



Insert Multiple Records:

```
INSERT INTO employees2 (emp_no, birth_date, first_name, last_name, gender, salary)

VALUES

(1015312001, '1989-08-28', 'Rina', 'Khanam', 'F', 45000),

(1015312002, '1988-07-19', 'Sakib', 'Hasan', 'M', 67000),

(1015312003, '1991-05-23', 'Sabbir', 'Rahman', 'M', 32000);
```

Output:

emp_no	birth_date	first_name	last_name	gender	salary	entry_date
1015312001	1989-08-28	Rina	Khanam	F	45000	2024-10-09 21:14:44
1015312002	1988-07-19	Sakib	Hasan	M	67000	2024-10-09 21:14:44
1015312003	1991-05-23	Sabbir	Rahman	M	32000	2024-10-09 21:14:44

Insert Single Records:

```
INSERT INTO employees2

VALUES (1015312008, '1991-05-23', 'Sabbir', 'Rahman', 'M', 24000, '2017-11-11');

INSERT INTO employees2

VALUES (1015312009, '1991-05-23', 'Sabbir', 'Rahman', 'M', 25600, '2017-11-11 21:44:35');
```

Output:

emp_no	birth_date	first_name	last_name	gender	salary	entry_date
1015312001	1989-08-28	Rina	Khanam	F	45000	2024-10-09 21:14:44
1015312002	1988-07-19	Sakib	Hasan	M	67000	2024-10-09 21:14:44
1015312003	1991-05-23	Sabbir	Rahman	M	32000	2024-10-09 21:14:44

Using Logical Operators (AND, OR, NOT)

- Using AND Operator:

This query retrieves the record where the first_name is "Rina" **and** the last_name is "Khanam".

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE first_name = 'Rina' AND last_name = 'Khanam';
```

Output:

emp_no	first_name	last_name	salary	entry_date
1015312001	Rina	Khanam	45000	2024-10-09 21:14:44

- Using OR Operator:

This query retrieves records where the first_name is "Rina" **or** the last_name is "Khan"

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE first_name = 'Rina' OR last_name = 'Khan';
```

Output:

emp_no	first_name	last_name	salary	entry_date
1015312001	Rina	Khanam	45000	2024-10-09 21:14:44

cted: Edit Copy Delete Export

- **Using Precedence with AND and OR:**

MySQL evaluates AND before OR. This query retrieves employees whose last name is "Rahman" and salary is less than or equal to 40,000, **or** whose first name is "Rina."

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE first_name = 'Rina' OR last_name = 'Rahman' AND salary <= 40000;
```

Output:

emp_no	first_name	last_name	salary	entry_date
1015312001	Rina	Khanam	45000	2024-10-09 21:14:44
1015312003	Sabbir	Rahman	32000	2024-10-09 21:14:44

Changing Precedence with Parentheses:

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE (first_name = 'Rina' OR last_name = 'Rahman') AND salary <= 40000;
```

Output:

emp_no	first_name	last_name	salary	entry_date
1015312003	Sabbir	Rahman	32000	2024-10-09 21:14:44

cted: Edit Copy Delete Export

Using LIMIT (with ORDER BY, ASC, DESC)

- **Select First 3 Employees:**

```
SELECT emp_no, first_name, last_name, salary

FROM employees2

LIMIT 3;
```

Output:

emp_no	first_name	last_name	salary
1015312001	Rina	Khanam	45000
1015312002	Sakib	Hasan	67000
1015312003	Sabbir	Rahman	32000

- **Skip First 2 Records and Retrieve Next 4:**

```
SELECT emp_no, first_name, last_name, salary

FROM employees2

LIMIT 2, 4;
```

Output:

emp_no	first_name	last_name	salary
1015312003	Sabbir	Rahman	32000

ected: Edit Copy Delete Export

- **Retrieve Top 3 Salaries:**

```
SELECT emp_no, first_name, last_name, salary

FROM employees2

ORDER BY salary DESC

LIMIT 3;
```

Output:

emp_no	first_name	last_name	salary 1
1015312002	Sakib	Hasan	67000
1015312001	Rina	Khanam	45000
1015312003	Sabbir	Rahman	32000

Using BETWEEN, IN, and NOT IN

- **Using IN (Similar to OR):**

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE salary IN (32000, 40000);
```

Output:

emp_no	first_name	last_name	salary	entry_date
1015312003	Sabbir	Rahman	32000	2024-10-09 21:14:44

- **Using NOT IN:**

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE salary NOT IN (32000, 45000, 25600);
```

Output:

emp_no	first_name	last_name	salary	entry_date
015312002	Sakib	Hasan	67000	2024-10-09 21:14:44

fed: Edit Copy Delete Export

Using LIKE Operator for Pattern Matching:

- **Find Employees with First Name Starting with 'm':**

```
SELECT emp_no, first_name, last_name, salary, entry_date

FROM employees2

WHERE first_name LIKE 'm%';
```

Output:

emp_no	first_name	last_name	salary	entry_date

- **Find Employees with First Name Ending in 'r':**

```
SELECT emp_no, first_name, last_name, salary, entry_date
FROM employees2
WHERE first_name LIKE '%r';
```

Output:

emp_no	first_name	last_name	salary	entry_date
1015312003	Sabbir	Rahman	32000	2024-10-09 21:14:44

ected: Edit Copy Delete Export

Checking for NULL Values:

- **Check for NULL Values in Gender:**

Retrieves employees whose gender field is NULL.

```
SELECT * FROM employees2

WHERE gender IS NULL;
```

Output:

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0002 seconds.)

```
SELECT * FROM employees2 WHERE gender IS NULL;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

emp_no	birth_date	first_name	last_name	gender	salary	entry_date

Query results operations

Create view

Conclusion:

This lab report demonstrates the effective use of essential SQL commands for data manipulation and retrieval. Logical operators (AND, OR, NOT), pattern matching with LIKE, filtering with BETWEEN and IN, and handling NULL values were used to query and filter data efficiently. Additionally, the DISTINCT operator eliminated duplicate entries, while the WHERE clause and comparison operators provided precise control over record selection. These operations are crucial for managing and retrieving data in any relational database system, with all queries executed successfully to meet the desired outputs.