Hands-on Lab: String Patterns, Sorting and Grouping in MySQL using phpMyAdmin

Estimated time needed: 20 minutes

In this lab, you will learn how to create tables and load data in the MySQL database service using the phpMyAdmin graphical user interface (GUI) tool.

Software Used in this Lab

In this lab, you will use MySQL. MySQL is a Relational Database Management System (RDBMS) designed to efficiently store, manipulate, and retrieve data.

To complete this lab you will utilize MySQL relational database service available as part of IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Database Used in this Lab

The database used in this lab is an internal database. You will be working on a sample HR database. This HR database schema consists of 5 tables called **EMPLOYEES**, **JOB_HISTORY**, **JOBS**, **DEPARTMENTS** and **LOCATIONS**. Each table has a few rows of sample data. The following diagram shows the tables for the HR database:

SAMPLE HR DATABASE TABLES

EMPLOYEES

EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID
E1001	John	Thomas	123456	1976-01-09	M	5631 Rice, OakPark,IL	100	100000	30001	2
E1002	Alice	James	123457	1972-07-31	F	980 Berry ln, Elgin,IL	200	80000	30002	5
E1003	Steve	Wells	123458	1980-08-10	M	291 Springs, Gary,IL	300	50000	30002	5

JOB_HISTORY

EMPL_ID	START_DATE	JOBS_ID	DEPT_ID
E1001	2000-01-30	100	2
E1002	2010-08-16	200	5
E1003	2016-08-10	300	5

JOBS

JOB_IDENT	JOB_TITLE	MIN_SALARY	MAX_SALARY
100	Sr. Architect	60000	100000
200	Sr.SoftwareDeveloper	60000	80000
300	Jr.SoftwareDeveloper	40000	60000

DEPARTMENTS

DEPT_ID_DEP	DEP_NAME	MANAGER_ID	LOC_ID
2	Architect Group	30001	L0001
5	Software Development	30002	L0002
7	Design Team	30003	L0003

LOCATIONS

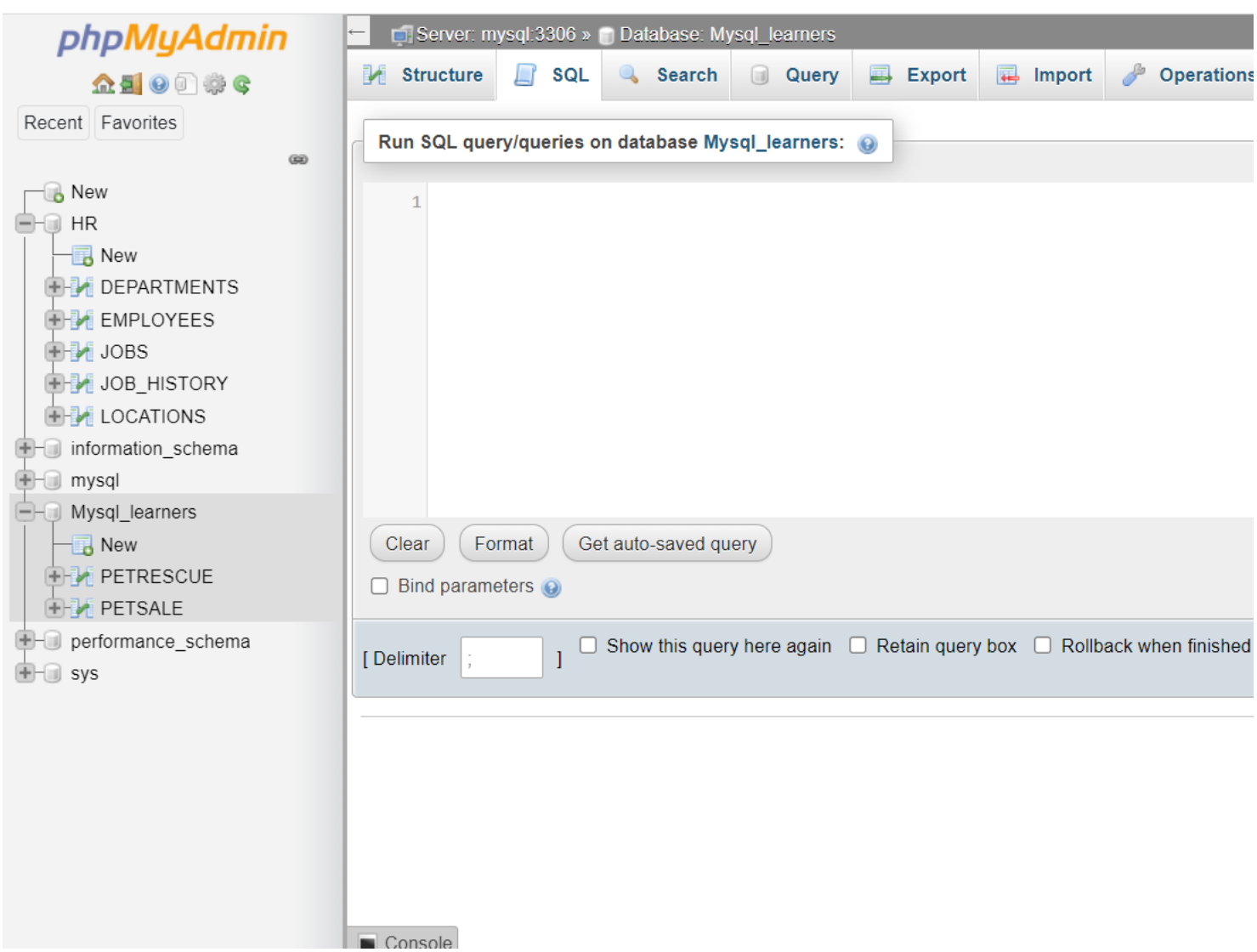
LOCT_ID	DEP_ID_LOC
L0001	2
L0002	5
L0003	7

Objectives

After completing this lab, you will be able to:

- Simplify a SELECT statement by using string patterns, ranges, or sets of values
- Sort the result set in either ascending or descending order and identify which column to use for the sorting order
- Eliminate duplicates from a result set and further restrict a result set

Once the tables are loaded open the sql editor to start executing the functions.



Exercise 1: String Patterns

In this exercise, you will go through some SQL problems on String Patterns.

1. Problem:

   Retrieve all employees whose address is in Elgin,IL.

   ▶ Hint
   ▼ Solution

```
1. 1
2. 2
3. 3
1. SELECT F_NAME , L_NAME
2. FROM EMPLOYEES
3. WHERE ADDRESS LIKE '%Elgin,IL%';
```

   Copied!

   ▶ Output

2. Problem:

   Retrieve all employees who were born during the 1970's.

   ▼ Hint

   Use the LIKE operator to find similar strings.

   ▼ Solution

```
1. 1
2. 2
3. 3
1. SELECT F_NAME , L_NAME
2. FROM EMPLOYEES
```

```
3. WHERE B_DATE LIKE '197%';
```

Copied!

▶ Output

3. Problem:

   Retrieve all employees in department 5 whose salary is between 60000 and 70000.

▶ Hint

▼ Solution

```
1. 1
2. 2
3. 3
1. SELECT *
2. FROM EMPLOYEES
3. WHERE (SALARY BETWEEN 60000 AND 70000) AND DEP_ID = 5;
```

Copied!

▶ Output

Exercise 2: Sorting

In this exercise, you will go through some SQL problems on Sorting.

1. Problem:

   Retrieve a list of employees ordered by department ID.

   ▼ Hint

   Use the ORDER BY clause for this SQL problem. By default, the ORDER BY clause sorts the records in ascending order.

   ▼ Solution

```
1. 1
2. 2
3. 3
1. SELECT F_NAME, L_NAME, DEP_ID
2. FROM EMPLOYEES
3. ORDER BY DEP_ID;
```

Copied!

▶ Output

2. Problem:

   Retrieve a list of employees ordered in descending order by department ID and within each department ordered alphabetically in descending order by last name.

   ▼ Hint

   Use the ORDER BY clause with DESC for this SQL problem.

   ▼ Solution

```
1. 1
2. 2
3. 3
1. SELECT F_NAME, L_NAME, DEP_ID
2. FROM EMPLOYEES
3. ORDER BY DEP_ID DESC, L_NAME DESC;
```

Copied!

▶ Output

3. (Optional) Problem:

   In SQL problem 2 (Exercise 2 Problem 2), use department name instead of department ID. Retrieve a list of employees ordered by department name, and within each department ordered alphabetically in descending order by last name.

   ▼ Hint

   Department name is in the DEPARTMENTS table. So your query will need to retrieve data from more than one table. Donâ€™t worry if you are not able to figure this SQL problem out. Weâ€™ll cover working with multiple tables in the lecture **Working with Multiple Tables**.

   ▼ Solution

```
1. 1
2. 2
3. 3
4. 4
1. SELECT D.DEP_NAME , E.F_NAME, E.L_NAME
2. FROM EMPLOYEES as E, DEPARTMENTS as D
3. WHERE E.DEP_ID = D.DEPT_ID_DEP
4. ORDER BY D.DEP_NAME, E.L_NAME DESC;
```

Copied!

In the SQL Query above, `D` and `E` are aliases for the table names. Once you define an alias like `D` in your query, you can simply write `D.COLUMN_NAME` rather than the full form `DEPARTMENTS.COLUMN_NAME`.

▶ Output

Exercise 3: Grouping

In this exercise, you will go through some SQL problems on Grouping.

NOTE: The SQL problems in this exercise involve usage of SQL Aggregate functions AVG and COUNT. COUNT has been covered earlier. AVG is a function that can be used to calculate the Average or Mean of all values of a specified column in the result set. For example, to retrieve the average salary for all employees in the EMPLOYEES table, issue the query: `SELECT AVG(SALARY) FROM EMPLOYEES;`. You will learn more about AVG and other aggregate functions later in the lecture **Built-in Database Functions**.

1. Problem:

   For each department ID retrieve the number of employees in the department.

   ▼ Hint

   Use COUNT(*) to retrieve the total count of a column, and then GROUP BY.

   ▼ Solution

   ```
   1. 1
   2. 2
   3. 3
   1. SELECT DEP_ID, COUNT(*)
   2. FROM EMPLOYEES
   3. GROUP BY DEP_ID;
   ```
   Copied!

   ▶ Output

2. Problem:

   For each department retrieve the number of employees in the department, and the average employee salary in the department..

   ▼ Hint

   Use COUNT(*) to retrieve the total count of a column, and AVG() function to compute average salaries, and then GROUP BY.

   ▼ Solution

   ```
   1. 1
   2. 2
   3. 3
   1. SELECT DEP_ID, COUNT(*), AVG(SALARY)
   2. FROM EMPLOYEES
   3. GROUP BY DEP_ID;
   ```
   Copied!

   ▶ Output

3. Problem:

   Label the computed columns in the result set of SQL problem 2 (Exercise 3 Problem 2) as NUM_EMPLOYEES and AVG_SALARY.

   ▼ Hint

   Use SQL Aliases: `column_name AS alias_name`. For example, AVG(SALARY) AS “AVG_SALARY”.

   ▼ Solution

   ```
   1. 1
   2. 2
   3. 3
   1. SELECT DEP_ID, COUNT(*) AS "NUM_EMPLOYEES", AVG(SALARY) AS "AVG_SALARY"
   2. FROM EMPLOYEES
   3. GROUP BY DEP_ID;
   ```
   Copied!

   ▶ Output

4. Problem:

   In SQL problem 3 (Exercise 3 Problem 3), order the result set by Average Salary..

   ▼ Hint

   Use ORDER BY after the GROUP BY.

   ▼ Solution

   ```
   1. 1
   2. 2
   ```

```
3. 3
4. 4
1. SELECT DEP_ID, COUNT(*) AS "NUM_EMPLOYEES", AVG(SALARY) AS "AVG_SALARY"
2. FROM EMPLOYEES
3. GROUP BY DEP_ID
4. ORDER BY AVG_SALARY;
```

Copied!

▶ Output

5. Problem:

   In SQL problem 4 (Exercise 3 Problem 4), limit the result to departments with fewer than 4 employees.

▼ Hint

Use HAVING after the GROUP BY, and use the count() function in the HAVING clause instead of the column label.

▼ Solution

```
1. 1
2. 2
3. 3
4. 4
5. 5
1. SELECT DEP_ID, COUNT(*) AS "NUM_EMPLOYEES", AVG(SALARY) AS "AVG_SALARY"
2. FROM EMPLOYEES
3. GROUP BY DEP_ID
4. HAVING count(*) < 4
5. ORDER BY AVG_SALARY;
```

Copied!

▶ Output

Solution Script

If you would like to run all the solution queries of the SQL problems of this lab with a script, download the script below.Import the script to phpadmin mysql interface and run. Follow Hands-on Lab : Create tables using SQL scripts and Load data into tables on how to upload a script to phpmyadmin console and run it.

- StringPattern-Sorting-Grouping_Solution_Script.sql

Congratulations! You have completed this lab, and you are ready for the next topic.

Author(s)

Lakshmi Holla

Malika Singla

Changelog

Date	Version	Changed by	Change Description
2023-05-10	0.3	Eric Hao & Vladislav Boyko	Updated Page Frames
2023-05-04	0.2	Rahul Jaideep	Updated Markdown file
2021-11-01	0.1	Lakshmi Holla, Malika Singla	Initial Version