Hands-on Lab: Working with Multiple Tables



Estimated time needed: 20 minutes

Objectives

After completing this lab, you will be able to:

- Write SQL queries that access more than one table
- Compose queries that access multiple tables using a nested statement in the WHERE clause
- Build gueries with multiple tables in the FROM clause
- Write Implicit Join queries with join criteria specified in the WHERE clause
- Specify aliases for table names and qualify column names with table aliases

In this lab, you will complete SQL practice problems that will provide hands-on experience with SQL queries that access multiple tables. You will be:

- · Accessing Multiple Tables with Sub-Queries
- · Accessing Multiple Tables with Implicit Joins

Software used in this lab

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



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 internal. 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

EMP_ID	F_NAME	L_NAM		SSN	B_DAT		SEX	ADDRESS		JOB_ID	SALAF	RY MANAG	ER_ID	DEP_ID	
E1001	John	Thomas		123456	1976-0	1-09	М	5631 Rice, OakPark,IL		100	10000	00 30001		2	
E1002	Alice	James		123457	1972-0	7-31	F	980 Berry In	, Elgin,IL	200 80000		30002		5	
E1003	Steve	Wells		123458	1980-0	8-10	М	291 Springs, Gary, IL		300	50000 30002			5	
JOB_HIST	ORY						J	OBS							
EMPL_ID	START_D	START_DATE		JOBS_ID		DEPT_ID		DB_IDENT	JOB_TIT	JOB_TITLE		MIN_SALARY	MA	X_SALAR	
E1001	2000-01	2000-01-30		100		2		00	Sr. Architect		60000	100	0000		
E1002	2010-08	2010-08-16		0 5		20		00 Sr.Softv		vareDeveloper		60000	800	80000	
E1003	2016-08	2016-08-10		00 5			300		Jr.SoftwareDeveloper		oper	40000	600	000	
DEPARTM	IENTS							LOCATIO	ONS						
DEPT_ID_D	EP DEP_NA	DEP_NAME		MANAGER_ID		LOC_ID		LOCT_ID		DEP	_ID_LOC				
2	Architec	Architect Group		30001		L0001		L0001		2					
5	Softwar	Software Development		30002		L0002		L0002		5					
7	Design 1	Design Team			30003			L0003		7					

Load the database

Using the skills acquired in the previous modules, you should first create the database in MySQL. Follow the steps below:

- 1. Open the phpMyAdmin interface from the Skills Network Toolbox in Cloud IDE.
- Create a blank database named HR. Use the script shared in the link below to create the required tables. <u>Script Create Tables.sql</u>

3. Download the files in the links below to your local machine (if not already done in previous labs).

Departments. csv

Jobs. csv

<u>JobsHistory.csv</u> Locations. csv

Employees. csv

4. Use these files to the interface as data for respective tables in the HR database.

Accessing multiple tables with sub-queries

Let us see some examples of queries requiring multiple table access using sub-queries.

1. Retrieve only the EMPLOYEES records corresponding to jobs in the JOBS table.

For such a question, you can implement the sub-query in the WHERE clause, such that the overlapping column of JOD ID can identify the required entries.

1

1. SELECT * FROM EMPLOYEES WHERE JOB_ID IN (SELECT JOB_IDENT FROM JOBS);

Copied!

The expected output would look as shown below.

+ Opt	ions											
←Τ	→		~	EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALA
	<i></i> € Edit	≩ € Copy	Delete	E1001	John	Thomas	123456	1976-09-01	М	5631 Rice, OakPark,IL	100	100000
	<i>❷</i> Edit	∄ € Copy	Delete	E1002	Alice	James	123457	1972-07-31	F	980 Berry In, Elgin,IL	200	80000
	Ø Edit	∄ € Copy	Delete	E1003	Steve	Wells	123458	1980-10-08	М	291 Springs, Gary,IL	300	50000
	Ø Edit	≩ € Copy	Delete	E1004	Santosh	Kumar	123459	1985-07-20	М	511 Aurora Av, Aurora,IL	400	60000
	🥒 Edit	≩ € Copy	Delete	E1005	Ahmed	Hussain	123410	1981-04-01	М	216 Oak Tree, Geneva,IL	500	70000
	Ø Edit	∄ € Copy	Delete	E1006	Nancy	Allen	123411	1978-06-02	F	111 Green PI, Elgin,IL	600	90000
	🥒 Edit	≩ € Copy	Delete	E1007	Mary	Thomas	123412	1975-05-05	F	100 Rose PI, Gary,IL	650	65000
	Ø Edit	≩ € Copy	Delete	E1008	Bharath	Gupta	123413	1985-06-05	М	145 Berry Ln, Naperville, IL	660	65000
	🥒 Edit	≩ € Copy	Delete	E1009	Andrea	Jones	123414	1990-09-07	F	120 Fall Creek, Gary,IL	234	70000
	<i>❷</i> Edit	∄ Copy	Delete	E1010	Ann	Jacob	123415	1982-03-30	F	111 Britany Springs, Elgin, IL	220	70000

2. Retrieve JOB information for employees earning over \$70,000.

For this example, retrieve the details from the JOBS table, which has common IDs with those available in the EMPLOYEES table, provided the salary in the EMPLOYEES table is greater than \$70,000. You can write the query as:

- 1. 1
- 2. 2 3. 3
- 1. SELECT JOB_TITLE, MIN_SALARY, MAX_SALARY, JOB_IDENT
- FROM JOBS
- WHERE JOB_IDENT IN (select JOB_ID from EMPLOYEES where SALARY > 70000);

Copied!

The expected output would look as shown below.



Accessing multiple tables with Implicit Joins

Let us see some examples of queries that require access of multiple tables using Implicit Joins.

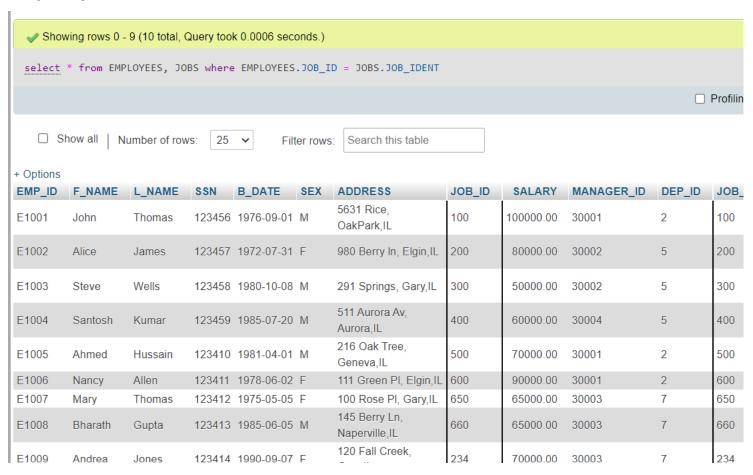
1. Retrieve only the EMPLOYEES records corresponding to jobs in the JOBS table.

The same question as before, but now we will use Implicit Join to retrieve the required information. For this, you will combine the tables based on job IDs. Using the following query for this:

- 1. 1
- 3. 3
- 1. SELECT *
- 2. FROM EMPLOYEES, JOBS
- 3. WHERE EMPLOYEES.JOB_ID = JOBS.JOB_IDENT;

Copied!

The expected output is shown below.



2. Redo the previous query using shorter aliases for table names.

Note that the tables in question can be assigned shorter aliases. This is especially helpful in cases where specific columns are to be accessed from different tables. The query would be modified to:

- 1. 1
- 3. 3
- SELECT *
- 2. FROM EMPLOYEES E, JOBS J
- 3. WHERE E.JOB_ID = J.JOB_IDENT;

Copied!

The output would look like:



+ Options

EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID	JOB_I
E1001	John	Thomas	123456	1976-09-01	M	5631 Rice, OakPark,IL	100	100000.00	30001	2	100
E1002	Alice	James	123457	1972-07-31	F	980 Berry In, Elgin,IL	200	80000.00	30002	5	200
E1003	Steve	Wells	123458	1980-10-08	M	291 Springs, Gary,IL	300	50000.00	30002	5	300
E1004	Santosh	Kumar	123459	1985-07-20	M	511 Aurora Av, Aurora,IL	400	60000.00	30004	5	400
E1005	Ahmed	Hussain	123410	1981-04-01	M	216 Oak Tree, Geneva,IL	500	70000.00	30001	2	500
E1006	Nancy	Allen	123411	1978-06-02	F	111 Green PI, Elgin,IL	600	90000.00	30001	2	600
E1007	Mary	Thomas	123412	1975-05-05	F	100 Rose PI, Gary,IL	650	65000.00	30003	7	650
E1008	Bharath	Gupta	123413	1985-06-05	M	145 Berry Ln, Naperville,IL	660	65000.00	30003	7	660
E1009 Consol	Andrea e	Jones	123414	1990-09-07	F	120 Fall Creek, Gary,IL	234	70000.00	30003	7	234

Notice that the two queries are giving the same response.

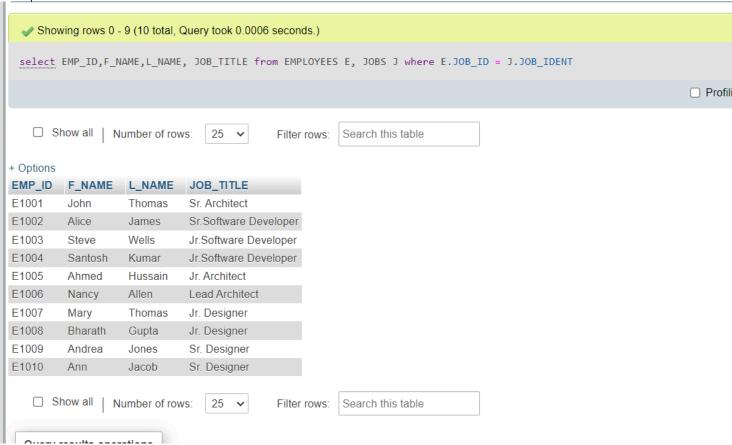
3. In the previous query, retrieve only the Employee ID, Name, and Job Title.

Notice that Job Title is a column of the JOBS table, and other details are coming from the EMPLOYEES table. The two tables will be joined on Job ID. The query would be as follows:

- 1. 1 2. 2 3. 3
- SELECT EMP_ID,F_NAME,L_NAME, JOB_TITLE
 FROM EMPLOYEES E, JOBS J
 WHERE E.JOB_ID = J.JOB_IDENT;

Copied!

The output would look as shown below.



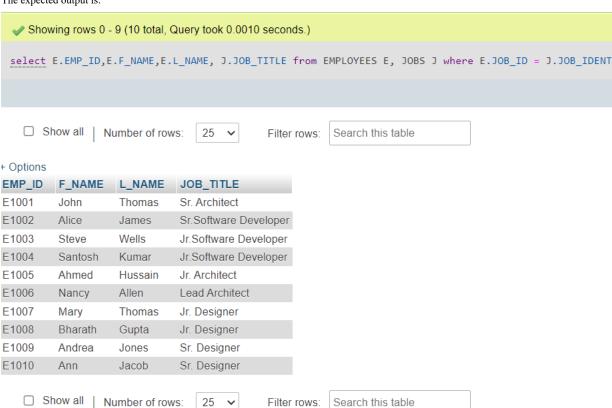
4. Redo the previous query, but specify the fully qualified column names with aliases in the SELECT clause.

The column names can also be prefixed with table aliases to keep track of where each column is coming from. The above query will be modified as shown below.

- 2. 2
- 1. SELECT E.EMP_ID, E.F_NAME, E.L_NAME, J.JOB_TITLE
- 2. FROM EMPLOYEES E, JOBS J
 3. WHERE E.JOB_ID = J.JOB_IDENT;

Copied!

The expected output is:



Profiling

Practice problems

- 1. Retrieve only the list of employees whose JOB_TITLE is Jr. Designer.
- a. Using sub-queries
- ► Solution
- b. Using Implicit Joins
- ► Solution
 - 2. Retrieve JOB information and a list of employees whose birth year is after 1976.
- a. Using sub-queries
- ► Solution
- b. Using implicit join
- ► Solution

Conclusion

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

At the end of this lab, you are now able to:

- Write SQL queries that access more than one table
- Compose queries that access multiple tables using a nested statement in the WHERE clause
- Build queries with multiple tables in the FROM clause
- Write Implicit Join queries with join criteria specified in the WHERE clause
- Specify aliases for table names and qualify column names with table aliases

Author(s)

Abhishek Gagneja

Lakshmi Holla

Malika Singla

