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 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

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

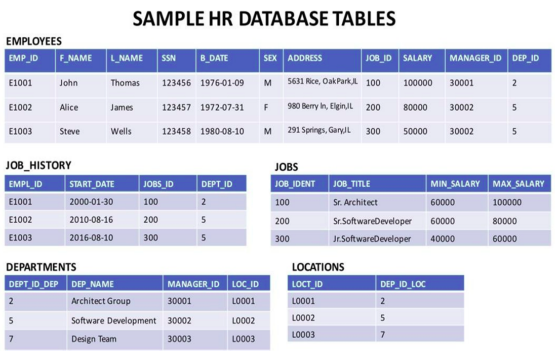
Software used in this lab

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:



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.
2. Create a blank database named HR. Use the script shared in the link below to create the required tables.
   Script_Create_Tables.sql
3. Download the files in the links below to your local machine (if not already done in previous labs).
   Departments. csv
   Jobs. csv
   JobsHistory.csv
   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
SELECT * FROM EMPLOYEES WHERE JOB_ID IN (SELECT JOB_IDENT FROM JOBS);
```

Copied!

The expected output would look as shown below.

+ Options

	EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID
Edit Copy Delete	E1001	John	Thomas	123456	1976-09-01	M	5631 Rice, OakPark,IL	100	100000.00	30001	2
Edit Copy Delete	E1002	Alice	James	123457	1972-07-31	F	980 Berry ln, Elgin,IL	200	80000.00	30002	5
Edit Copy Delete	E1003	Steve	Wells	123458	1980-10-08	M	291 Springs, Gary,IL	300	50000.00	30002	5
Edit Copy Delete	E1004	Santosh	Kumar	123459	1985-07-20	M	511 Aurora Av, Aurora,IL	400	60000.00	30004	5
Edit Copy Delete	E1005	Ahmed	Hussain	123410	1981-04-01	M	216 Oak Tree, Geneva,IL	500	70000.00	30001	2
Edit Copy Delete	E1006	Nancy	Allen	123411	1978-06-02	F	111 Green Pl, Elgin,IL	600	90000.00	30001	2
Edit Copy Delete	E1007	Mary	Thomas	123412	1975-05-05	F	100 Rose Pl, Gary,IL	650	65000.00	30003	7
Edit Copy Delete	E1008	Bharath	Gupta	123413	1985-06-05	M	145 Berry Ln, Naperville,IL	660	65000.00	30003	7
Edit Copy Delete	E1009	Andrea	Jones	123414	1990-09-07	F	120 Fall Creek, Gary,IL	234	70000.00	30003	7
Edit Copy Delete	E1010	Ann	Jacob	123415	1982-03-30	F	111 Britany Springs,Elgin,IL	220	70000.00	30004	5

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
2. FROM JOBS
3. WHERE JOB_IDENT IN (select JOB_ID from EMPLOYEES where SALARY > 70000 );
```

Copied!

The expected output would look as shown below.

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

+ Options

	JOB_TITLE	MIN_SALARY	MAX_SALARY	JOB_IDENT
Edit Copy Delete	Sr. Architect	60000.00	100000.00	100
Edit Copy Delete	Sr.Software Developer	60000.00	80000.00	200
Edit Copy Delete	Lead Architect	70000.00	100000.00	600

Check all With selected: Edit Copy Delete Export

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
2. 2
3. 3
```

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

Copied!

The expected output is shown below.

Showing rows 0 - 9 (10 total, Query took 0.0006 seconds.)

```
select * from EMPLOYEES, JOBS where EMPLOYEES.JOB_ID = JOBS.JOB_IDENT
```

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

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID	JOB_IDENT	JOB_TITLE	MIN_SALARY	MAX_SALARY
E1001	John	Thomas	123456	1976-09-01	M	5631 Rice, OakPark,IL	100	100000.00	30001	2	100	Sr. Architect	60000.00	100000.00
E1002	Alice	James	123457	1972-07-31	F	980 Berry ln, Elgin,IL	200	80000.00	30002	5	200	Sr.Software Developer	60000.00	80000.00
E1003	Steve	Wells	123458	1980-10-08	M	291 Springs, Gary,IL	300	50000.00	30002	5	300	Jr.Software Developer	40000.00	60000.00
E1004	Santosh	Kumar	123459	1985-07-20	M	511 Aurora Av, Aurora,IL	400	60000.00	30004	5	400	Jr.Software Developer	40000.00	60000.00
E1005	Ahmed	Hussain	123410	1981-04-01	M	216 Oak Tree, Geneva,IL	500	70000.00	30001	2	500	Jr. Architect	50000.00	70000.00
E1006	Nancy	Allen	123411	1978-06-02	F	111 Green Pl, Elgin,IL	600	90000.00	30001	2	600	Lead Architect	70000.00	100000.00
E1007	Mary	Thomas	123412	1975-05-05	F	100 Rose Pl, Gary,IL	650	65000.00	30003	7	650	Jr. Designer	60000.00	70000.00
E1008	Bharath	Gupta	123413	1985-06-05	M	145 Berry Ln, Naperville,IL	660	65000.00	30003	7	660	Jr. Designer	60000.00	70000.00
E1009	Andrea	Jones	123414	1990-09-07	F	120 Fall Creek,	234	70000.00	30003	7	234	Sr. Designer	70000.00	90000.00

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

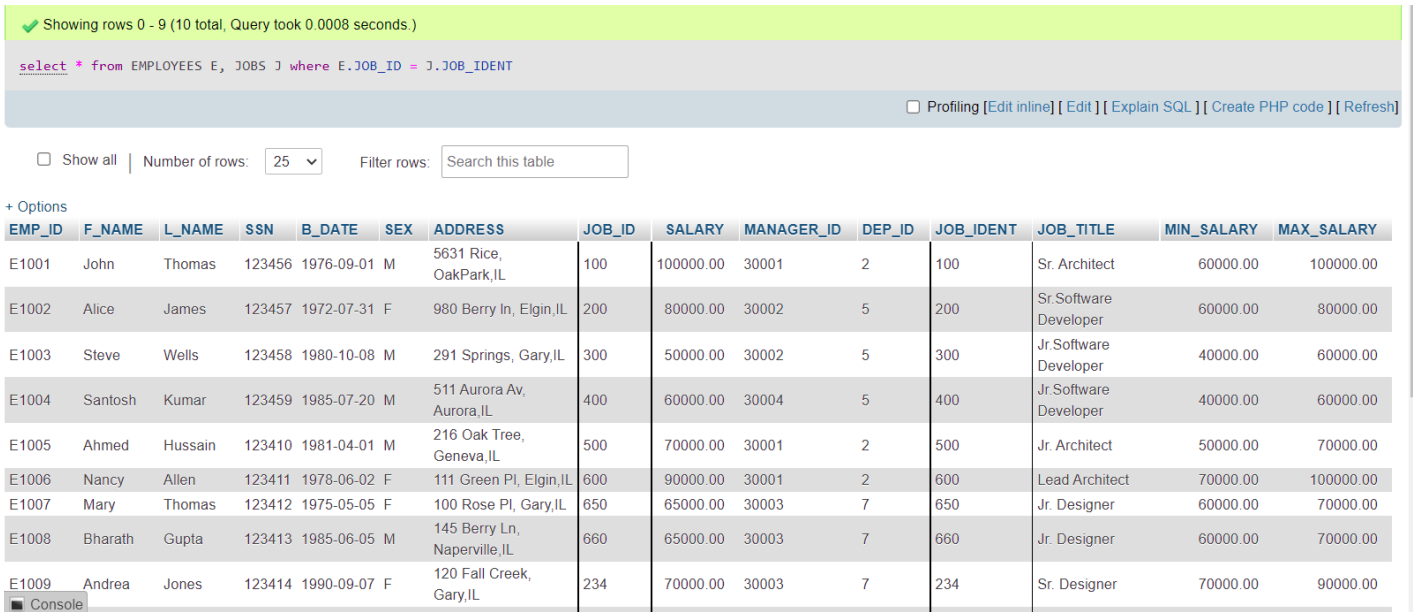
Note that the tables in question can be assigned shorter aliases. This is especially helpful in cases where specific columns are to be accessed from different tables. The query would be modified to:

```
1. 1
2. 2
3. 3
```

```
SELECT *
FROM EMPLOYEES E, JOBS J
WHERE E.JOB_ID = J.JOB_IDENT;
```

Copied!

The output would look like:



Showing rows 0 - 9 (10 total, Query took 0.0008 seconds.)

```
select * from EMPLOYEES E, JOBS J where E.JOB_ID = J.JOB_IDENT
```

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

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

EMP_ID	F_NAME	L_NAME	SSN	B_DATE	SEX	ADDRESS	JOB_ID	SALARY	MANAGER_ID	DEP_ID	JOB_IDENT	JOB_TITLE	MIN_SALARY	MAX_SALARY
E1001	John	Thomas	123456	1976-09-01	M	5631 Rice, OakPark,IL	100	100000.00	30001	2	100	Sr. Architect	60000.00	100000.00
E1002	Alice	James	123457	1972-07-31	F	980 Berry ln, Elgin,IL	200	80000.00	30002	5	200	Sr.Software Developer	60000.00	80000.00
E1003	Steve	Wells	123458	1980-10-08	M	291 Springs, Gary,IL	300	50000.00	30002	5	300	Jr.Software Developer	40000.00	60000.00
E1004	Santosh	Kumar	123459	1985-07-20	M	511 Aurora Av, Aurora,IL	400	60000.00	30004	5	400	Jr.Software Developer	40000.00	60000.00
E1005	Ahmed	Hussain	123410	1981-04-01	M	216 Oak Tree, Geneva,IL	500	70000.00	30001	2	500	Jr. Architect	50000.00	70000.00
E1006	Nancy	Allen	123411	1978-06-02	F	111 Green Pl, Elgin,IL	600	90000.00	30001	2	600	Lead Architect	70000.00	100000.00
E1007	Mary	Thomas	123412	1975-05-05	F	100 Rose Pl, Gary,IL	650	65000.00	30003	7	650	Jr. Designer	60000.00	70000.00
E1008	Bharath	Gupta	123413	1985-06-05	M	145 Berry Ln, Naperville,IL	660	65000.00	30003	7	660	Jr. Designer	60000.00	70000.00
E1009	Andrea	Jones	123414	1990-09-07	F	120 Fall Creek, Gary,IL	234	70000.00	30003	7	234	Sr. Designer	70000.00	90000.00

Console

Notice that the two queries are giving the same response.

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

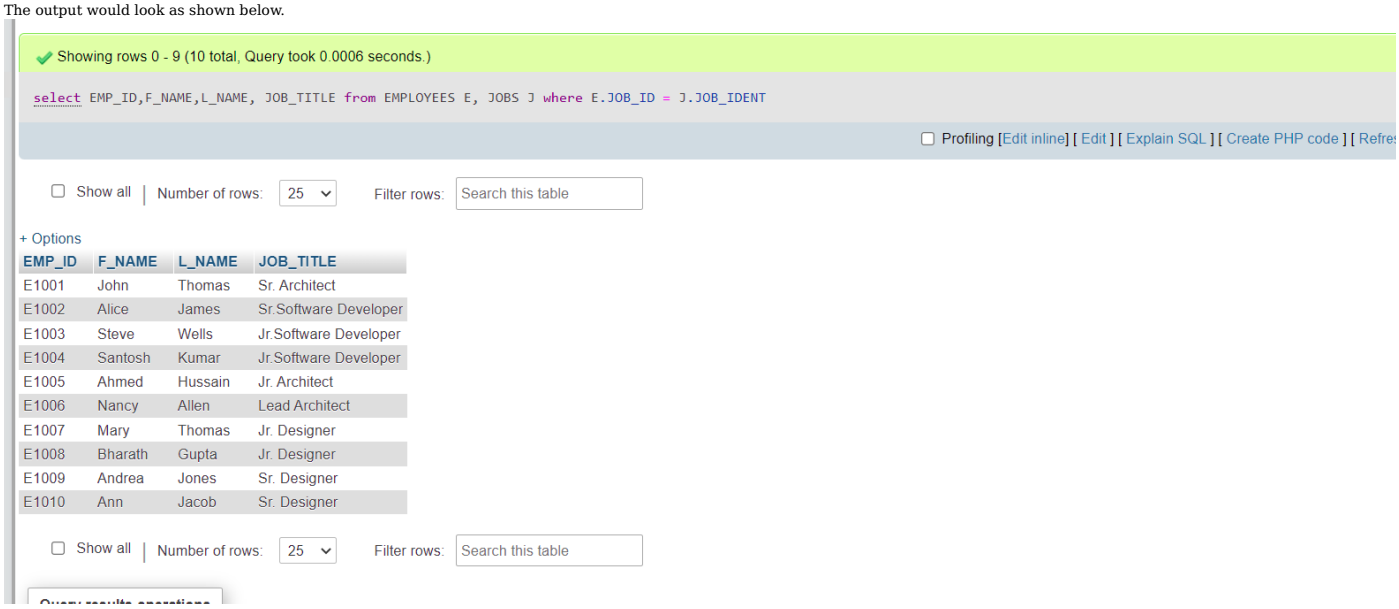
Notice that Job Title is a column of the JOBS table, and other details are coming from the EMPLOYEES table. The two tables will be joined on Job ID. The query would be as follows:

```
1
2
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.



Showing rows 0 - 9 (10 total, Query took 0.0006 seconds.)

```
select EMP_ID,F_NAME,L_NAME, JOB_TITLE from EMPLOYEES E, JOBS J where E.JOB_ID = J.JOB_IDENT
```

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

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

EMP_ID	F_NAME	L_NAME	JOB_TITLE
E1001	John	Thomas	Sr. Architect
E1002	Alice	James	Sr.Software Developer
E1003	Steve	Wells	Jr.Software Developer
E1004	Santosh	Kumar	Jr.Software Developer
E1005	Ahmed	Hussain	Jr. Architect
E1006	Nancy	Allen	Lead Architect
E1007	Mary	Thomas	Jr. Designer
E1008	Bharath	Gupta	Jr. Designer
E1009	Andrea	Jones	Sr. Designer
E1010	Ann	Jacob	Sr. Designer

Show all | Number of rows: 25 Filter rows: Search this table

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

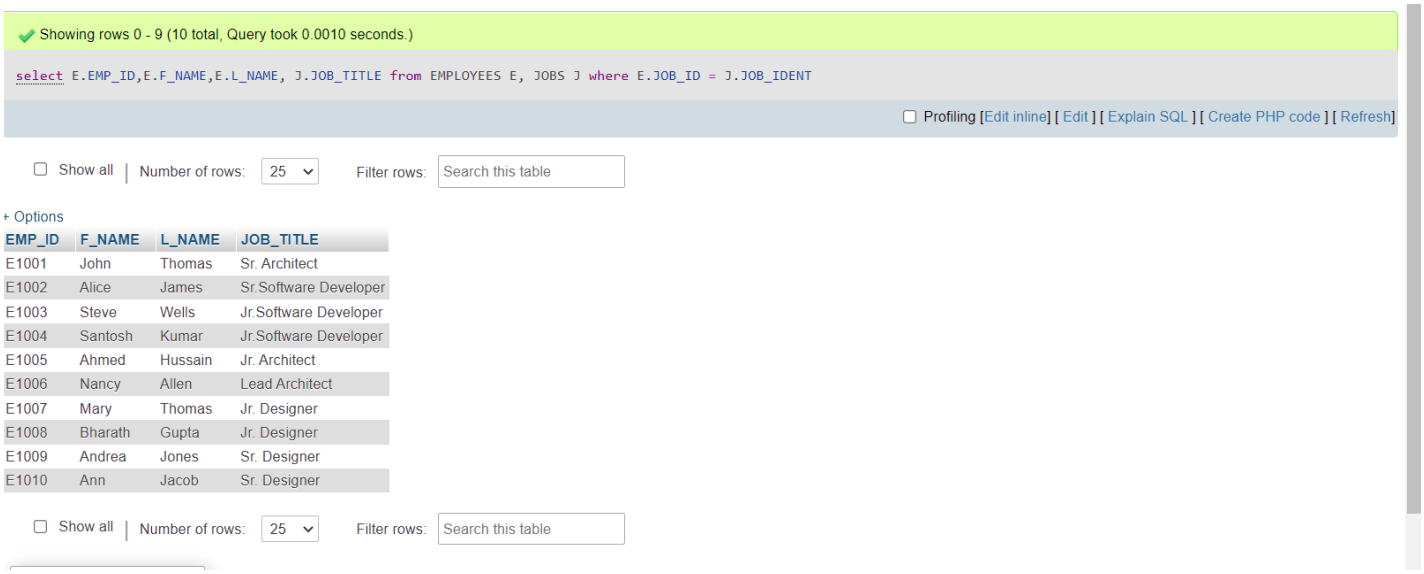
The column names can also be prefixed with table aliases to keep track of where each column is coming from. The above query will be modified as shown below.

```
1
2
3
```

```
SELECT E.EMP_ID, E.F_NAME, E.L_NAME, J.JOB_TITLE
FROM EMPLOYEES E, JOBS J
WHERE E.JOB_ID = J.JOB_IDENT;
```

Copied!

The expected output is:



Showing rows 0 - 9 (10 total, Query took 0.0010 seconds.)

```
select E.EMP_ID,E.F_NAME,E.L_NAME, J.JOB_TITLE from EMPLOYEES E, JOBS J where E.JOB_ID = J.JOB_IDENT
```

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

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

EMP_ID	F_NAME	L_NAME	JOB_TITLE
E1001	John	Thomas	Sr. Architect
E1002	Alice	James	Sr.Software Developer
E1003	Steve	Wells	Jr.Software Developer
E1004	Santosh	Kumar	Jr.Software Developer
E1005	Ahmed	Hussain	Jr. Architect
E1006	Nancy	Allen	Lead Architect
E1007	Mary	Thomas	Jr. Designer
E1008	Bharath	Gupta	Jr. Designer
E1009	Andrea	Jones	Sr. Designer
E1010	Ann	Jacob	Sr. Designer

Show all | Number of rows: 25 Filter rows: Search this table

Practice problems

1. Retrieve only the list of employees whose JOB_TITLE is Jr. Designer.

a. Using sub-queries

▾ Solution

```
1
2
3
4
5
```

```
SELECT *
FROM EMPLOYEES
WHERE JOB_ID IN (SELECT JOB_IDENT
                 FROM JOBS
                 WHERE JOB_TITLE= 'Jr. Designer');
```

Copied!

b. Using Implicit Joins

▾ Solution

```
1
2
3
```

```
SELECT *
FROM EMPLOYEES E, JOBS J
WHERE E.JOB_ID = J.JOB_IDENT AND J.JOB_TITLE= 'Jr. Designer';
```

Copied!

2. Retrieve JOB information and a list of employees whose birth year is after 1976.

a. Using sub-queries

▾ Solution

```
1
2
3
4
5
```

```
SELECT JOB_TITLE, MIN_SALARY, MAX_SALARY, JOB_IDENT
FROM JOBS
WHERE JOB_IDENT IN (SELECT JOB_ID
                    FROM EMPLOYEES
                    WHERE YEAR(B_DATE)>1976 );
```

Copied!

b. Using implicit join

▾ Solution

```
1
2
3
```

```
SELECT J.JOB_TITLE, J.MIN_SALARY, J.MAX_SALARY, J.JOB_IDENT
FROM JOBS J, EMPLOYEES E
WHERE E.JOB_ID = J.JOB_IDENT AND YEAR(E.B_DATE)>1976;
```

Copied!

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

Changelog

Date	Version	Changed by	Change Description
2023-10-13	0.6	Mary Stenberg	QA pass with edits
2023-10-13	0.5	Misty Taylor	ID Check
2023-10-10	0.4	Abhishek Gagneja	Instructions updated
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