

lab_assignment-8

lab9-SQL-ANP-C7281-Inner-join

PEDDA JAGADEESH

AFO366969

ANP-C7281

lab9_SQL_ANP_C7281_Inner_join

Lab: Use the Student Management System Database and table from previous

lab. Perform the following commands on the table Student and Enrollment.

1. Let's consider a scenario where you have a database tracking student enrollments and some students may not be enrolled in any courses.

John Doe (StudentID: 1) is enrolled in courses with EnrollmentIDs 101 and 102.

Jane Smith (StudentID: 2) is enrolled in courses with EnrollmentIDs 103 and 104.

Bob Johnson (StudentID: 3) is not enrolled in any courses.

Now, run RIGHT OUTER JOIN query to retrieve data.

2. Assume a university where students can enroll in various courses. Here are some fictional details:

Student Information:

Student with ID 1: John, email: john@email.com

Student with ID 2: Jane, email: jane@email.com

Student with ID 3: Bob, email: bob@email.com

Enrollment Information:

Enrollment with ID 101: John (StudentID: 1) enrolls in Math (CourseID: MATH101).

Enrollment with ID 102: John (StudentID: 1) enrolls in History (CourseID: HIST201).

Enrollment with ID 103: Jane (StudentID: 2) enrolls in Physics (CourseID: PHYS301).

Enrollment with ID 104: Bob (StudentID: 3) enrolls in Chemistry (CourseID: CHEM401).

Enrollment with ID 105: Alice (StudentID: 4) enrolls in English (CourseID: ENG501).

Now, write a LEFT JOIN query to retrieve the data.

Submission:

Create an SQL script file containing your solutions for all tasks (queries). Name the file "lab_assignment1.sql" Provide comments above each query to indicate the query's purpose.

ChatGPT Exercise

Using ChatGPT generates SQL queries of the below problem .

Scenario 1: You have two tables, employees and departments. Retrieve a list of employees along with their department names using an inner join.

Scenario 2: In an employee database, join the employees table with itself to display each employee along with their manager, including employees without managers, using

a left join.

We have an "Employee" table with the following columns:

EmployeeID, EmployeeName, ManagerID (Foreign Key) and "Manager" table with following columns: ManagerID, ManagerName. You want to retrieve each employee along

with your manager. Generate a chatGPT prompt for the scenario.

Lab: Use the Student Management System Database and table from previous

lab. Perform the following commands on the table Student and Enrollment.

1. Let's consider a scenario where you have a database tracking student enrollments and some students may not be enrolled in any courses.

John Doe (StudentID: 1) is enrolled in courses with EnrollmentIDs 101 and 102.

Jane Smith (StudentID: 2) is enrolled in courses with EnrollmentIDs 103 and 104.

Bob Johnson (StudentID: 3) is not enrolled in any courses.

Code:

```
CREATE TABLE Student_data_1 (  
  
StudentID INT PRIMARY KEY,  
  
StudentName VARCHAR(30)  
  
);
```

Output:

```
mysql> CREATE TABLE Student_data_1 (  
-> StudentID INT PRIMARY KEY,  
-> StudentName VARCHAR(30)  
-> );  
Query OK, 0 rows affected (0.05 sec)
```

```
INSERT INTO Student_data_1 (StudentID, StudentName) VALUES  
  
(1, 'John Doe'),  
  
(2, 'Jane Smith'),  
  
(3, 'Bob Johnson');
```

Output:

```
mysql> select * from Student_data_1;  
+-----+-----+  
| StudentID | StudentName |  
+-----+-----+  
|          1 | John Doe    |  
|          2 | Jane Smith  |  
|          3 | Bob Johnson |  
+-----+-----+  
3 rows in set (0.00 sec)
```

```
CREATE TABLE Enrollment (  
  
EnrollmentID INT PRIMARY KEY,  
  
StudentID INT,  
  
CourseID VARCHAR(50)
```

);

Outout:

```
mysql> CREATE TABLE Enrollment (
-> EnrollmentID INT PRIMARY KEY,
-> StudentID INT,
-> CourseID VARCHAR(50)
-> );
Query OK, 0 rows affected (0.05 sec)

mysql> INSERT INTO Enrollment (EnrollmentID, StudentID, CourseID) VALUES
-> (101, 1, 'CourseA'),
-> (102, 1, 'CourseB'),
-> (103, 2, 'CourseC'),
-> (104, 2, 'CourseD');
Query OK, 4 rows affected (0.01 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> select * from Enrollment;
```

EnrollmentID	StudentID	CourseID
101	1	CourseA
102	1	CourseB
103	2	CourseC
104	2	CourseD

```
4 rows in set (0.00 sec)
```

SELECT s.StudentID, s.StudentName, e.EnrollmentID, e.CourseID

FROM Student_data_1 s

RIGHT OUTER JOIN Enrollment e ON s.StudentID = e.StudentID;

Output:

```
mysql> SELECT s.StudentID, s.StudentName, e.EnrollmentID, e.CourseID
-> FROM Student_data_1 s
-> RIGHT OUTER JOIN Enrollment e ON s.StudentID = e.StudentID;
```

StudentID	StudentName	EnrollmentID	CourseID
1	John Doe	101	CourseA
1	John Doe	102	CourseB
2	Jane Smith	103	CourseC
2	Jane Smith	104	CourseD

```
4 rows in set (0.00 sec)
```

Now,run RIGHT OUTER JOIN query to retrieve data.

2. Assume a university where students can enroll in various courses. Here are some

fictional details:

Student Information:

Student with ID 1: John, email: john@email.com

Student with ID 2: Jane, email: jane@email.com

Student with ID 3: Bob, email: bob@email.com

Code:

```
CREATE TABLE Student_data_2 (  
  
StudentID INT PRIMARY KEY,  
  
StudentName VARCHAR(100),  
  
Email VARCHAR(100)  
  
);  
  
INSERT INTO Student_data_2 (StudentID, StudentName, Email) VALUES  
  
(1, 'John', 'john@email.com'),  
  
(2, 'Jane', 'jane@email.com'),  
  
(3, 'Bob', 'bob@email.com'),  
  
(4, 'Alice', 'alice@email.com');
```

Output:

```
mysql> CREATE TABLE Student_data_2 (
->     StudentID INT PRIMARY KEY,
->     StudentName VARCHAR(100),
->     Email VARCHAR(100)
-> );
Query OK, 0 rows affected (0.05 sec)

mysql>
mysql> INSERT INTO Student_data_2 (StudentID, StudentName, Email) VALUES
-> (1, 'John', 'john@email.com'),
-> (2, 'Jane', 'jane@email.com'),
-> (3, 'Bob', 'bob@email.com'),
-> (4, 'Alice', 'alice@email.com');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from Student_data_2;
```

StudentID	StudentName	Email
1	John	john@email.com
2	Jane	jane@email.com
3	Bob	bob@email.com
4	Alice	alice@email.com

```
4 rows in set (0.00 sec)
```

Enrollment Information:

Enrollment with ID 101: John (StudentID: 1) enrolls in Math (CourseID: MATH101).

Enrollment with ID 102: John (StudentID: 1) enrolls in History (CourseID: HIST201).

Enrollment with ID 103: Jane (StudentID: 2) enrolls in Physics (CourseID: PHYS301).

Enrollment with ID 104: Bob (StudentID: 3) enrolls in Chemistry (CourseID: CHEM401).

Enrollment with ID 105: Alice (StudentID: 4) enrolls in English (CourseID: ENG501).

Now, write a LEFT JOIN query to retrieve the data.

Code:

```
CREATE TABLE Enrollment (

EnrollmentID INT PRIMARY KEY,

StudentID INT,

CourseID VARCHAR(50),

FOREIGN KEY (StudentID) REFERENCES Student_data_2(StudentID)

);
```

```
INSERT INTO Enrollment (EnrollmentID, StudentID, CourseID) VALUES
```

```
(101, 1, 'MATH101'),
```

```
(102, 1, 'HIST201'),
```

```
(103, 2, 'PHYS301'),
```

```
(104, 2, 'CHEM401'),
```

```
(105, 4, 'ENG501');
```

```
SELECT s.StudentID, s.StudentName, s.Email, e.EnrollmentID, e.CourseID
```

```
FROM Student_data_2 s
```

```
LEFT JOIN Enrollment e ON s.StudentID = e.StudentID;
```

Output:

```
mysql> select * from Enrollment;
```

EnrollmentID	StudentID	CourseID
101	1	MATH101
102	1	HIST201
103	2	PHYS301
104	2	CHEM401
105	4	ENG501

```
5 rows in set (0.00 sec)
```

```
mysql> SELECT s.StudentID, s.StudentName, s.Email, e.EnrollmentID, e.CourseID  
-> FROM Student_data_2 s  
-> LEFT JOIN Enrollment e ON s.StudentID = e.StudentID;
```

StudentID	StudentName	Email	EnrollmentID	CourseID
1	John	john@email.com	101	MATH101
1	John	john@email.com	102	HIST201
2	Jane	jane@email.com	103	PHYS301
2	Jane	jane@email.com	104	CHEM401
3	Bob	bob@email.com	NULL	NULL
4	Alice	alice@email.com	105	ENG501

```
6 rows in set (0.00 sec)
```

Submission:

Create an SQL script file containing your solutions for all tasks (queries). Name

the file "lab_assignment1.sql" Provide comments above each query to

indicate the query's purpose.

ChatGPT Exercise

Using ChatGPT generates SQL queries of the below problem .

Scenario 1: You have two tables, employees and departments. Retrieve a list of employees along with their department names using an inner join.

Scenario 2: In an employee database, join the employees table with itself to display each employee along with their manager, including employees without managers, using a left join.

We have an "Employee" table with the following columns:

EmployeeID, EmployeeName, ManagerID (Foreign Key) and "Manager" table with following columns: ManagerID, ManagerName. You want to retrieve each employee along

with your manager. Generate a chatGPT prompt for the scenario.

-- Create Employees table

```
CREATE TABLE Employees (  
EmployeeID INT PRIMARY KEY,  
EmployeeName VARCHAR(100),  
DepartmentID INT);
```

-- Create Departments table

```
CREATE TABLE Departments (  
DepartmentID INT PRIMARY KEY,  
DepartmentName VARCHAR(100));
```

-- Insert sample data into Employees table

```
INSERT INTO Employees (EmployeeID, EmployeeName, DepartmentID)
```

```
VALUES
```

```
(1, 'John Doe', 1),
```

```
(2, 'Jane Smith', 2),
```

```
(3, 'Bob Johnson', 1),
```

```
(4, 'Alice Brown', 2);
```

-- Insert sample data into Departments table

```
INSERT INTO Departments (DepartmentID, DepartmentName)
```

```
VALUES
```

```
(1, 'Engineering'),
```

```
(2, 'Marketing');
```

-- Query to retrieve employees with their department names using INNER JOIN

```
SELECT e.EmployeeID, e.EmployeeName, d.DepartmentName
```

```
FROM Employees e
```

```
INNER JOIN Departments d ON e.DepartmentID = d.DepartmentID;
```

Output:

```

mysql> CREATE TABLE Employees (
  -> EmployeeID INT PRIMARY KEY,
  -> EmployeeName VARCHAR(100),
  -> DepartmentID INT);
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE Departments (
  -> DepartmentID INT PRIMARY KEY,
  -> DepartmentName VARCHAR(100));
Query OK, 0 rows affected (0.04 sec)

mysql> INSERT INTO Employees (EmployeeID, EmployeeName, DepartmentID)
  -> VALUES
  -> (1, 'John Doe', 1),
  -> (2, 'Jane Smith', 2),
  -> (3, 'Bob Johnson', 1),
  -> (4, 'Alice Brown', 2);
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Departments (DepartmentID, DepartmentName)
  -> VALUES
  -> (1, 'Engineering'),
  -> (2, 'Marketing');
Query OK, 2 rows affected (0.21 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> SELECT e.EmployeeID, e.EmployeeName, d.DepartmentName
  -> FROM Employees e
  -> INNER JOIN Departments d ON e.DepartmentID = d.DepartmentID;
+-----+-----+-----+
| EmployeeID | EmployeeName | DepartmentName |
+-----+-----+-----+
| 1 | John Doe | Engineering |
| 2 | Jane Smith | Marketing |
| 3 | Bob Johnson | Engineering |
| 4 | Alice Brown | Marketing |
+-----+-----+-----+

```

Thank You, sir