

CASE STUDY

EMPLOYEES ATTENDANCE MANAGEMENT SYSTEM

Project Overview :

Objective :

Design and implement an attendance management system to track employees' attendance, leave status, and working hours effectively.

Scope:

- Employees can log attendance.
- Managers can view attendance records.
- Admins can manage employees and attendance policies.

Requirements Analysis :

Features:

1. **Employee Module:**
 - Add new employees.
 - Update employee details.
 - Remove employees.
2. **Attendance Module:**
 - Log daily attendance (check-in/check-out time).
 - Mark absences and late arrivals.
 - Generate reports (daily, monthly, yearly).
3. **Leave Management:**
 - Track leave requests and approvals.
 - Calculate remaining leave days.
4. **Reports:**
 - Attendance summary.
 - Employee punctuality statistics.
 - Leave utilization.

Database Design :

Creating an Employee Attendance Management System using SQL can be an excellent project to showcase your database design, querying, and management skills. Here's a structured approach for your case study:

1. Project Overview

Objective:

Design and implement an attendance management system to track employees' attendance, leave status, and working hours effectively.

Scope:

- Employees can log attendance.
- Managers can view attendance records.
- Admins can manage employees and attendance policies.

2. Requirements Analysis

Features:

1. **Employee Module:**
 - Add new employees.
 - Update employee details.
 - Remove employees.
2. **Attendance Module:**
 - Log daily attendance (check-in/check-out time).
 - Mark absences and late arrivals.
 - Generate reports (daily, monthly, yearly).
3. **Leave Management:**
 - Track leave requests and approvals.
 - Calculate remaining leave days.
4. **Reports:**
 - Attendance summary.
 - Employee punctuality statistics.
 - Leave utilization.

3. Database Design

Entities and Relationships:

Tables:

1. **Employees:**

- EmployeeID (Primary Key)
- FirstName, LastName
- Department
- Position
- DateOfJoining
- ContactInfo

2. **Attendance:**

- AttendanceID (Primary Key)
- EmployeeID (Foreign Key)
- Date
- CheckInTime
- CheckOutTime
- Status (e.g., Present, Absent, Late)

3. **Leave:**

- LeaveID (Primary Key)
- EmployeeID (Foreign Key)
- LeaveDate
- LeaveType (e.g., Sick, Casual, Paid)
- Status (Pending, Approved, Rejected)

4. **Departments:**

- DepartmentID (Primary Key)
- DepartmentName

5. **Users** (for login):

- UserID (Primary Key)
- Username
- Password
- Role (Admin, Manager, Employee)

SQL Queries :

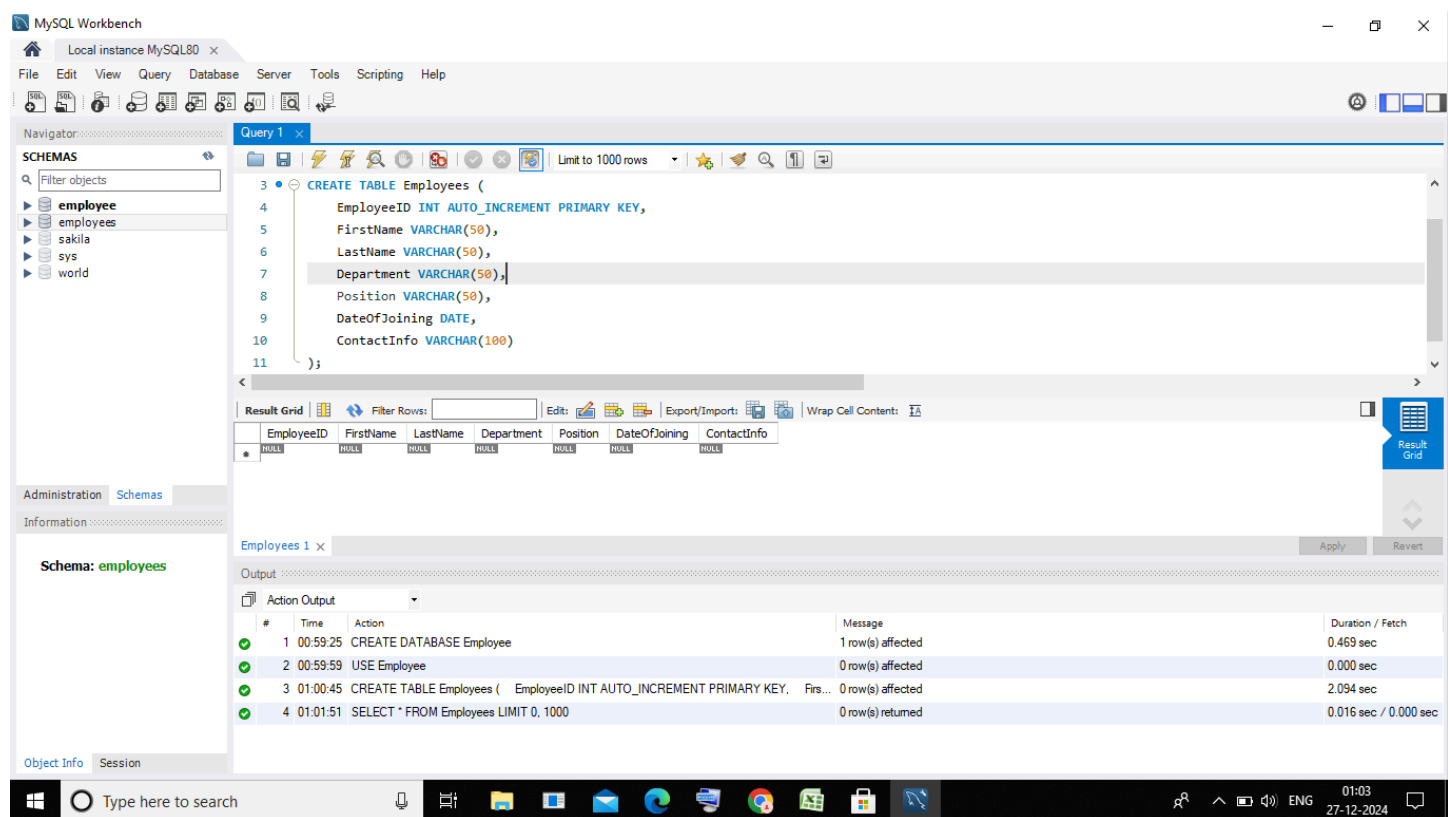
Create Tables:

Employees Table :

```
CREATE TABLE Employees (
    EmployeeID INT AUTO_INCREMENT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Department VARCHAR(50),
    Position VARCHAR(50),
    DateOfJoining DATE,
```

ContactInfo VARCHAR(100)

);



Attendance Table :

CREATE TABLE Attendance (

AttendanceID INT AUTO_INCREMENT PRIMARY KEY,

EmployeeID INT,

Date DATE,

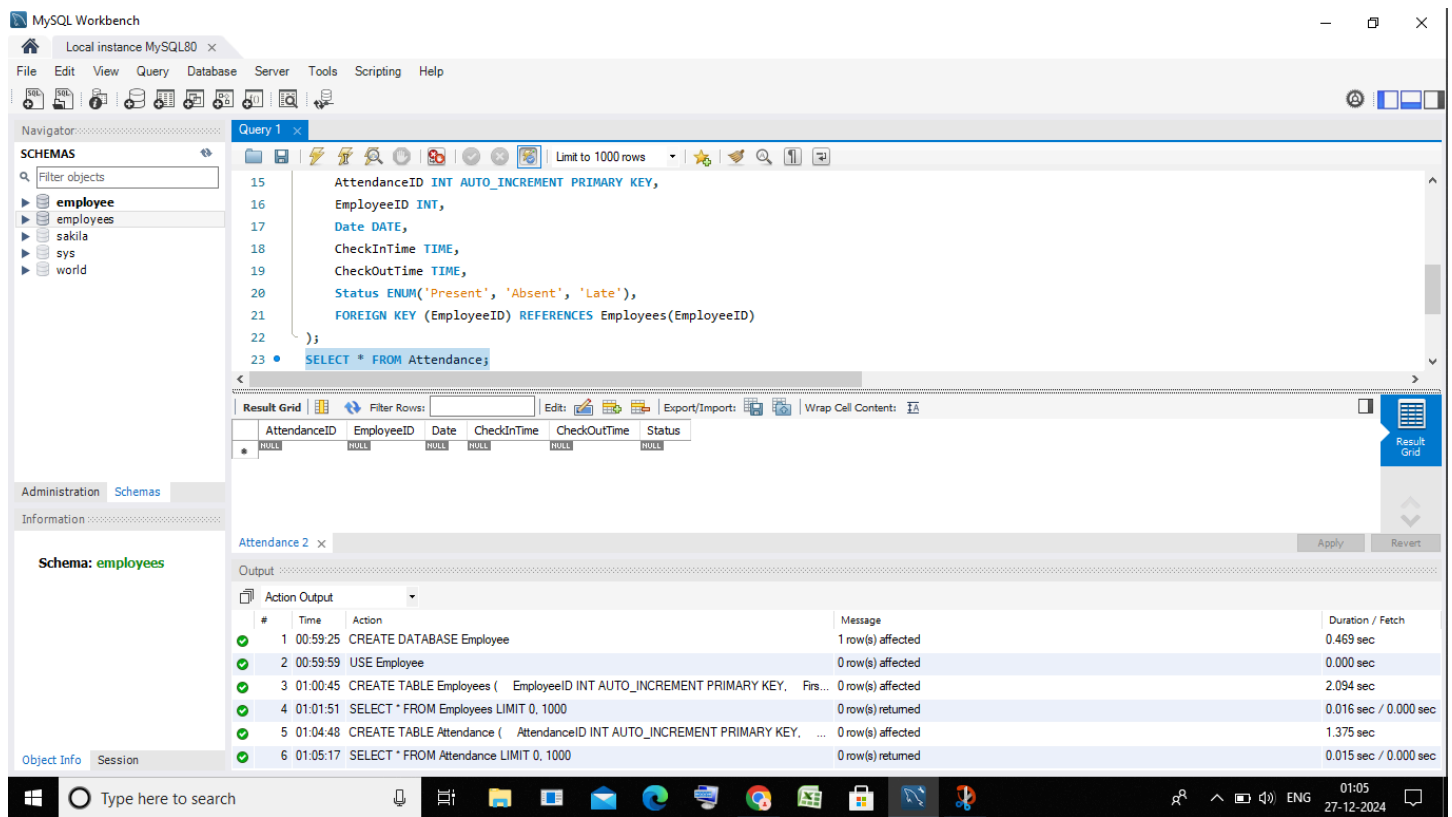
CheckInTime TIME,

CheckOutTime TIME,

Status ENUM('Present', 'Absent', 'Late'),

FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID)

);



Leave Table :

CREATE TABLE Leave _(

LeaveID INT AUTO_INCREMENT PRIMARY KEY,

EmployeeID INT,

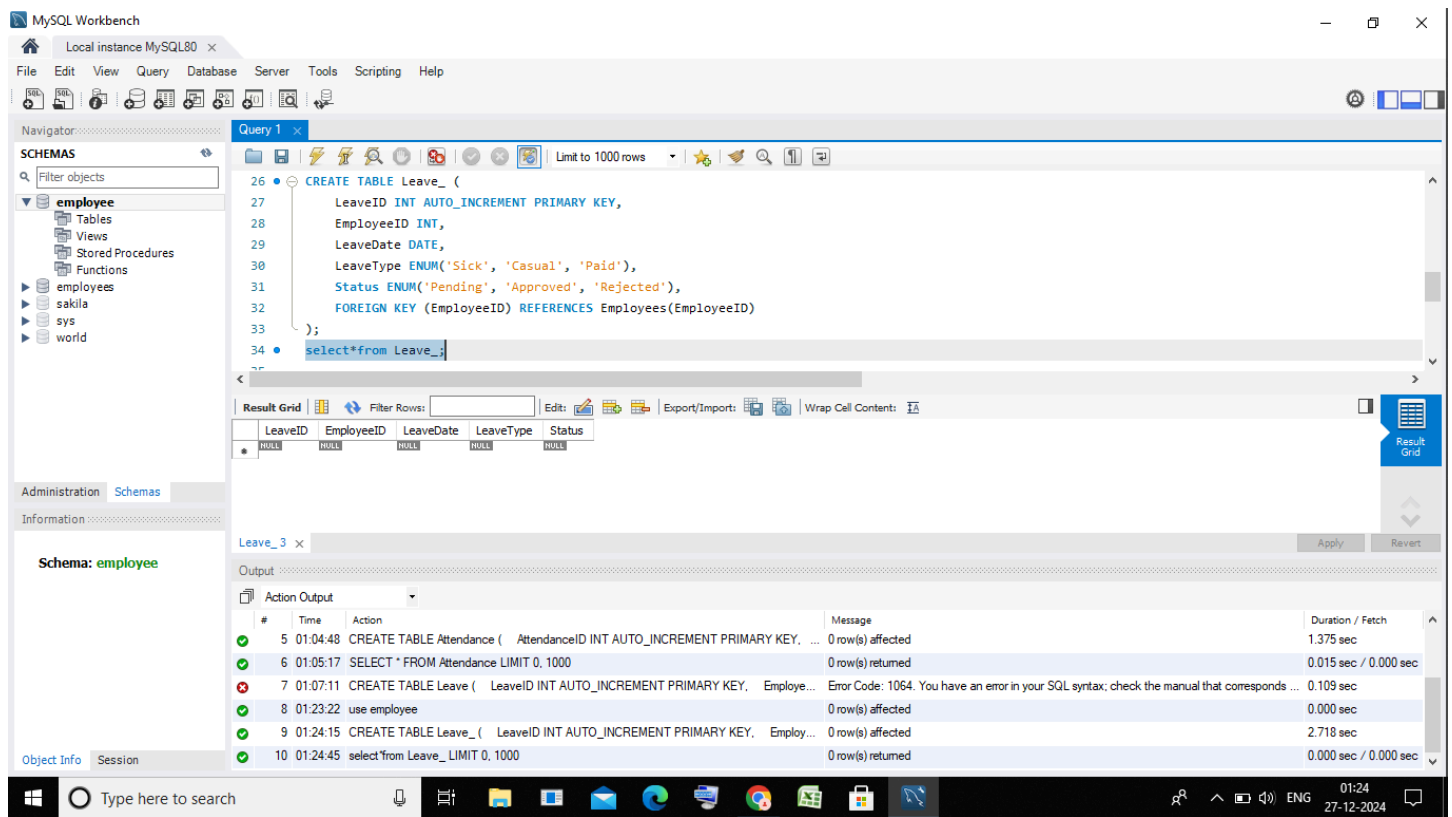
LeaveDate DATE,

LeaveType ENUM('Sick', 'Casual', 'Paid'),

Status ENUM('Pending', 'Approved', 'Rejected'),

FOREIGN KEY (EmployeeID) REFERENCES Employees(EmployeeID)

);



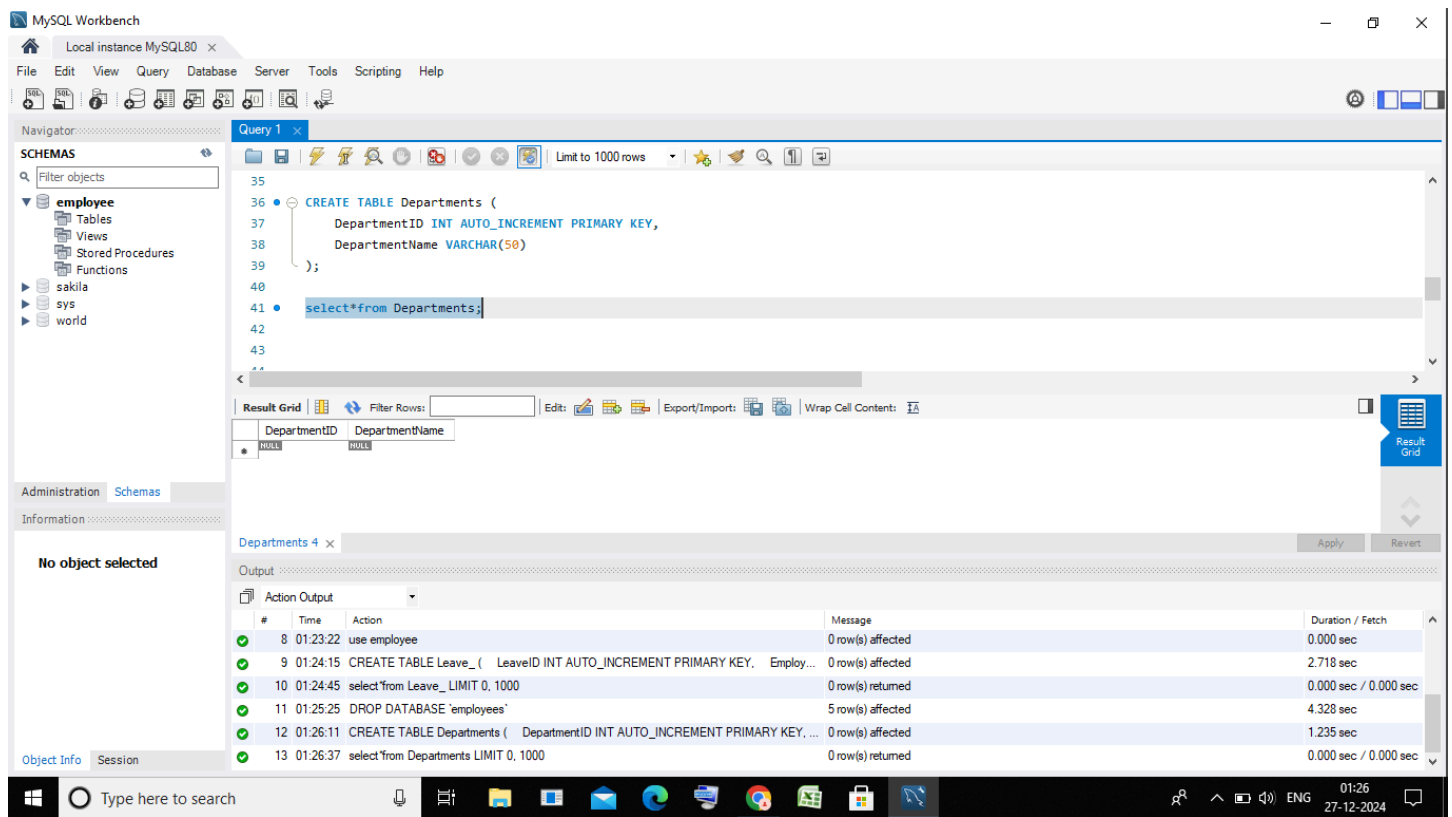
Department Table :

CREATE TABLE Departments (

DepartmentID INT AUTO_INCREMENT PRIMARY KEY,

DepartmentName VARCHAR(50)

);



User Table :

CREATE TABLE Users (

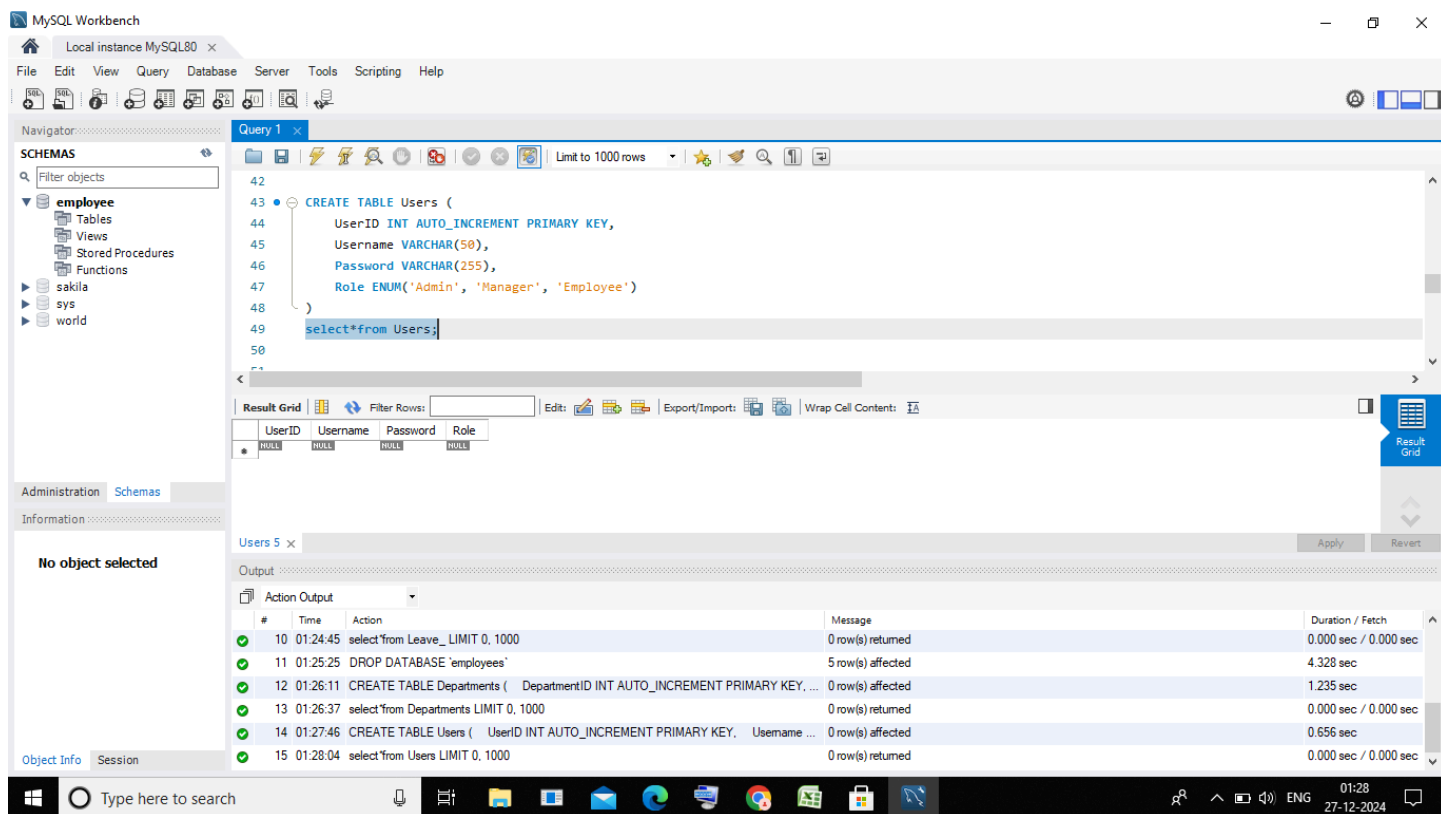
UserID INT AUTO_INCREMENT PRIMARY KEY,

Username VARCHAR(50),

Password VARCHAR(255),

Role ENUM('Admin', 'Manager', 'Employee')

);



Sample Queries:

Employees Details :

INSERT INTO Employees(FirstName, LastName, Department, Position, DateOfJoining, ContactInfo) VALUES

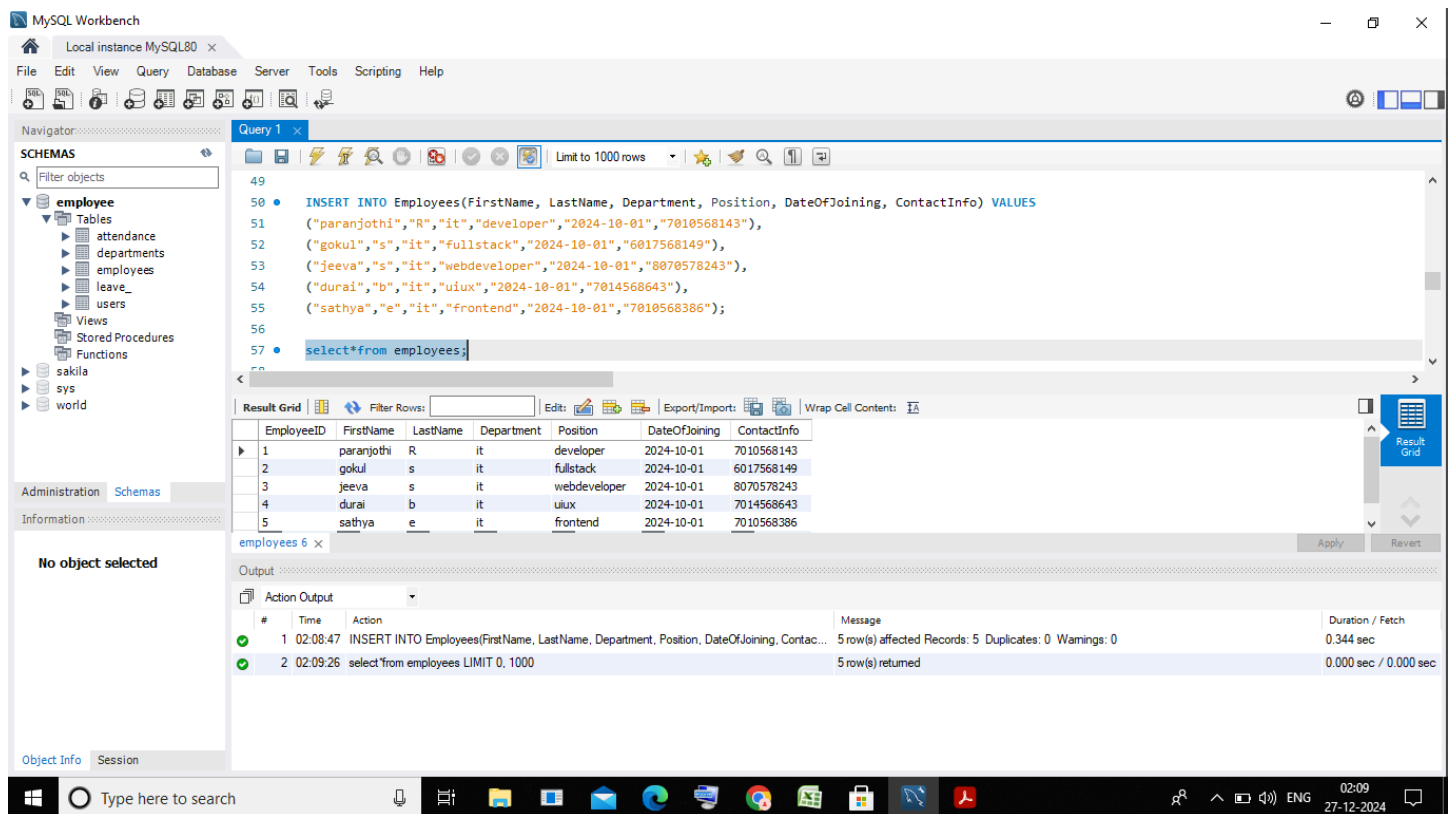
("paranjothi","R","it","developer","2024-10-01","7010568143"),

("gokul","s","it","fullstack","2024-10-01","6017568149"),

("jeeva","s","it","webdeveloper","2024-10-01","8070578243"),

("durai","b","it","uiux","2024-10-01","7014568643"),

("sathya","e","it","frontend","2024-10-01","7010568386");



Insert Attendance :

INSERT INTO Attendance (EmployeeID, Date, CheckInTime, Status)

VALUES (1, '2024-12-01', '09:00:00', 'Present'),

(2, '2024-12-01', '09:00:00', 'Present'),

(3, '2024-12-01', '09:00:00', 'Present'),

(4, '2024-12-01', '09:00:00', 'Present'),

(5, '2024-12-01', '09:00:00', 'Present'),

(1, '2024-12-02', '09:00:00', 'Present'),

(2, '2024-12-02', '09:00:00', 'Present'),

(3, '2024-12-02', '09:00:00', 'Present'),

(4, '2024-12-02', '09:00:00', 'Present'),

(5, '2024-12-02', '09:00:00', 'Present'),

(1, '2024-12-03', '09:00:00', 'Present'),

(2, '2024-12-03', '09:00:00', 'Absent'),

```
(3, '2024-12-03', '09:00:00', 'Present'),  
(4, '2024-12-03', '09:00:00', 'Present'),  
(5, '2024-12-03', '09:00:00', 'Absent');
```

The screenshot shows the MySQL Workbench interface. The 'Query' tab is active, displaying an SQL query that inserts attendance records for five employees on December 1st and 2nd, 2024. The query is as follows:

```
INSERT INTO Attendance (EmployeeID, Date, CheckInTime, Status)  
VALUES (1, '2024-12-01', '09:00:00', 'Present'),  
(2, '2024-12-01', '09:00:00', 'Present'),  
(3, '2024-12-01', '09:00:00', 'Present'),  
(4, '2024-12-01', '09:00:00', 'Present'),  
(5, '2024-12-01', '09:00:00', 'Present'),  
(1, '2024-12-02', '09:00:00', 'Present'),  
(2, '2024-12-02', '09:00:00', 'Present');
```

The 'Result Grid' shows the output of the query, displaying 15 rows of attendance records. The columns are AttendanceID, EmployeeID, Date, CheckInTime, CheckOutTime, and Status. The first 5 rows are for December 1st, and the next 5 rows are for December 2nd. The status for all records is 'Present'.

AttendanceID	EmployeeID	Date	CheckInTime	CheckOutTime	Status
1	1	2024-12-01	09:00:00	NULL	Present
2	2	2024-12-01	09:00:00	NULL	Present
3	3	2024-12-01	09:00:00	NULL	Present
4	4	2024-12-01	09:00:00	NULL	Present
5	5	2024-12-01	09:00:00	NULL	Present
6	1	2024-12-02	09:00:00	NULL	Present
7	2	2024-12-02	09:00:00	NULL	Present
8	3	2024-12-02	09:00:00	NULL	Present
9	4	2024-12-02	09:00:00	NULL	Present
10	5	2024-12-02	09:00:00	NULL	Present
11	1	2024-12-03	09:00:00	NULL	Present
12	2	2024-12-03	09:00:00	NULL	Present
13	3	2024-12-03	09:00:00	NULL	Present
14	4	2024-12-03	09:00:00	NULL	Present
15	5	2024-12-03	09:00:00	NULL	Absent

The 'Output' tab shows the execution log, indicating that 15 rows were affected and 15 rows were returned for the insert statement.

Insert Users :

```
INSERT INTO Users (Username, Password, Role) VALUES(  
"paranji","paranji0905","employee"),  
("gokul","gokul0905","employee"),  
("jeeva","jeeva0905","employee"),  
("durai","durai0905","employee"),  
("sathya","sathya0905","employee");
```

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

employee
sakila
sys
world

Query 1

Limit to 1000 rows

```

149 WHERE MONTH(Date) = 12 AND Status = 'Absent'
150 GROUP BY EmployeeID;
151
152 • select*from users;
153
154
155
156
157

```

Result Grid

UserID	Username	Password	Role
1	paranji	paranj0905	Employee
2	gokul	gokul0905	Admin
3	jeeva	jeeva0905	Employee
4	durai	durai0905	Manager
5	sathya	sathya0905	Employee

users 20

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	03:00:11	SELECT EmployeeID,COUNT(*) AS PresentDays FROM Attendance WHERE MONTH(Date) ...	2 row(s) returned	0.000 sec / 0.000 sec
2	03:39:21	SELECT*FROM Leave_ LIMIT 0, 1000	2 row(s) returned	0.016 sec / 0.000 sec
3	03:41:49	select*from users LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Type here to search

27-12-2024

Insert Leave_ :

insert into Leave_ (EmployeeID,LeaveDate,LeaveType)values(

2,"2024-12-03","paid"),

(5,"2024-12-03","paid");

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

employee
Tables
attendance
departments
employees
leave_
users
Views
Stored Procedures
Functions
sakila
sys
world

Query 1

Limit to 1000 rows

```

77
78 • insert into Leave_ (EmployeeID,LeaveDate,LeaveType)values(
79 2,"2024-12-03","paid"),
80 (5,"2024-12-03","paid");
81
82 • select*from Leave_;
83
84
85

```

Result Grid

LeaveID	EmployeeID	LeaveDate	LeaveType	Status
1	2	2024-12-03	Paid	NULL
2	5	2024-12-03	Paid	NULL
3	NULL	NULL	NULL	NULL

Leave_ 9

Output

Action Output

#	Time	Action	Message	Duration / Fetch
2	02:09:26	select*from employees LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
3	02:19:04	INSERT INTO Attendance (EmployeeID, Date, CheckInTime, Status) VALUES (1, '2024-12-...	15 row(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.297 sec
4	02:19:43	select*from Attendance LIMIT 0, 1000	15 row(s) returned	0.000 sec / 0.000 sec
5	02:25:20	select*from Leave_ LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
6	02:31:46	insert into Leave_ (EmployeeID,LeaveDate,LeaveType)values(2,"2024-12-03","paid"), (5,"...	2 row(s) affected Records: 2 Duplicates: 0 Warnings: 0	0.110 sec
7	02:32:12	select*from Leave_ LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Type here to search

27-12-2024

Insert Department :

INSERT INTO Departments(DepartmentName)VALUES("it");

The screenshot shows the MySQL Workbench interface. The 'Query 1' window contains the following SQL script:

```
86 WHERE LeaveID = 1;  
87  
88 UPDATE Leave_  
89 SET Status = 'Rejected'  
90 WHERE LeaveID = 2;  
91  
92 INSERT INTO Departments(DepartmentName)VALUES("it");  
93  
94 select*from departments;
```

The 'Result Grid' shows the output of the last query, displaying a single row in the 'departments' table:

DepartmentID	DepartmentName
1	it

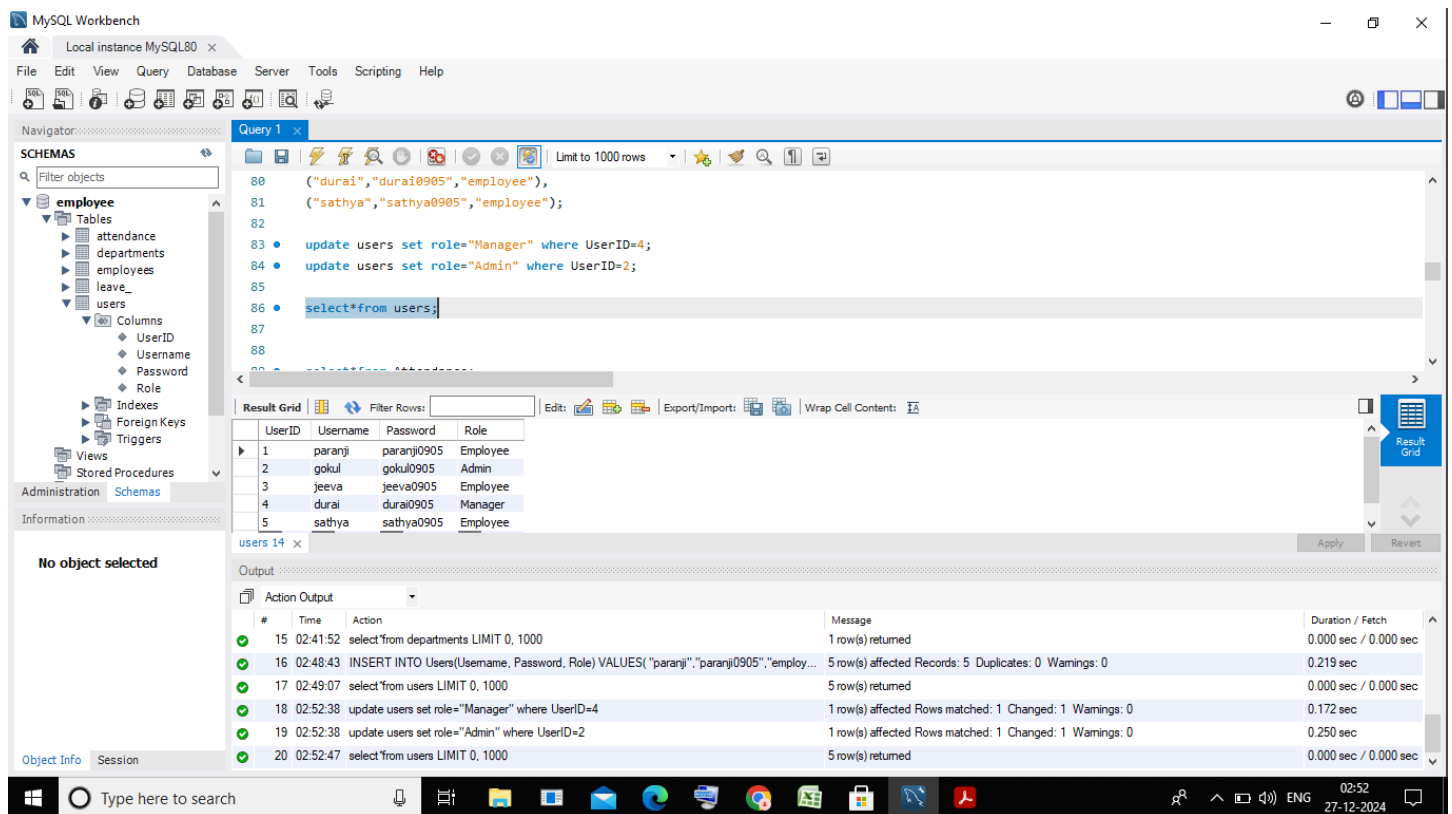
The 'Action Output' window shows the execution log for the queries:

#	Time	Action	Message	Duration / Fetch
10	02:35:05	select*from Leave_ LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
11	02:35:44	UPDATE Leave_ SET Status = 'Rejected' WHERE LeaveID = 2	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.281 sec
12	02:35:47	UPDATE Leave_ SET Status = 'Approved' WHERE LeaveID = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.125 sec
13	02:35:51	select*from Leave_ LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
14	02:41:22	INSERT INTO Departments(DepartmentName)VALUES("it")	1 row(s) affected	0.078 sec
15	02:41:52	select*from departments LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Update :

update users set role="Manager" where UserID=4;

update users set role="Admin" where UserID=2;



Update Leave :

UPDATE Leave_

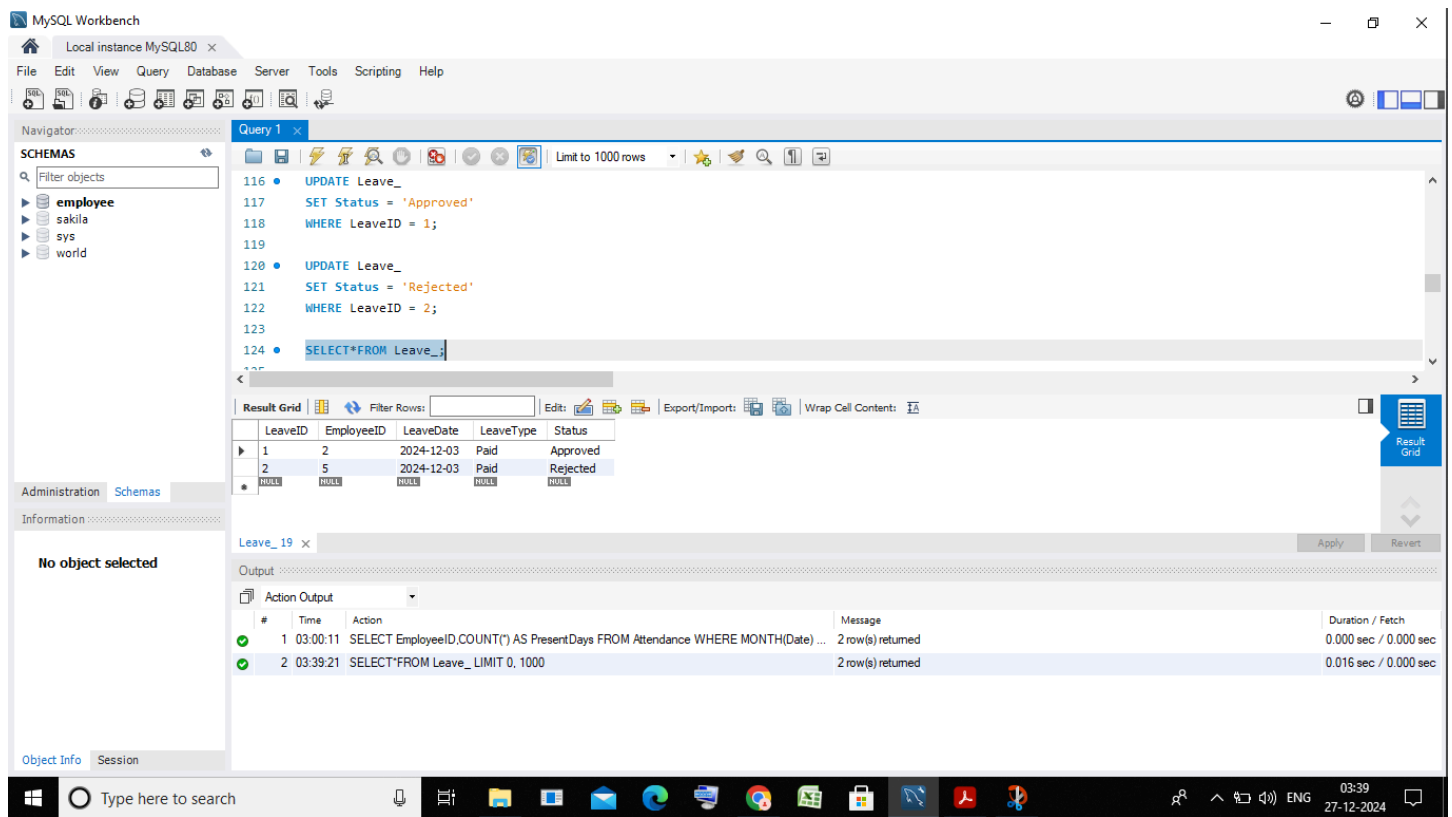
SET Status = 'Approved'

WHERE LeaveID = 1;

UPDATE Leave_

SET Status = 'Rejected'

WHERE LeaveID = 2;



Present days Count :

SELECT EmployeeID, COUNT(*) AS PresentDays

FROM Attendance

WHERE MONTH(Date) = 12 AND Status = 'Present'

GROUP BY EmployeeID;

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'employee' selected, showing tables like 'attendance', 'departments', 'employees', 'leave_', and 'users'. The main editor shows a query in 'Query 1' with the following SQL code:

```
105 • INSERT INTO Departments(DepartmentName)VALUES("it");
106
107 • select*from departments;
108
109
110 • SELECT EmployeeID, COUNT(*) AS PresentDays
111 FROM Attendance
112 WHERE MONTH(Date) = 12 AND Status = 'Present'
113 GROUP BY EmployeeID;
```

The 'Result Grid' shows the output of the query:

EmployeeID	PresentDays
1	3
2	2
3	3
4	3
5	2

The 'Action Output' pane at the bottom shows a list of executed queries and their results, including messages like '5 row(s) affected', '5 row(s) returned', and '1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0'.

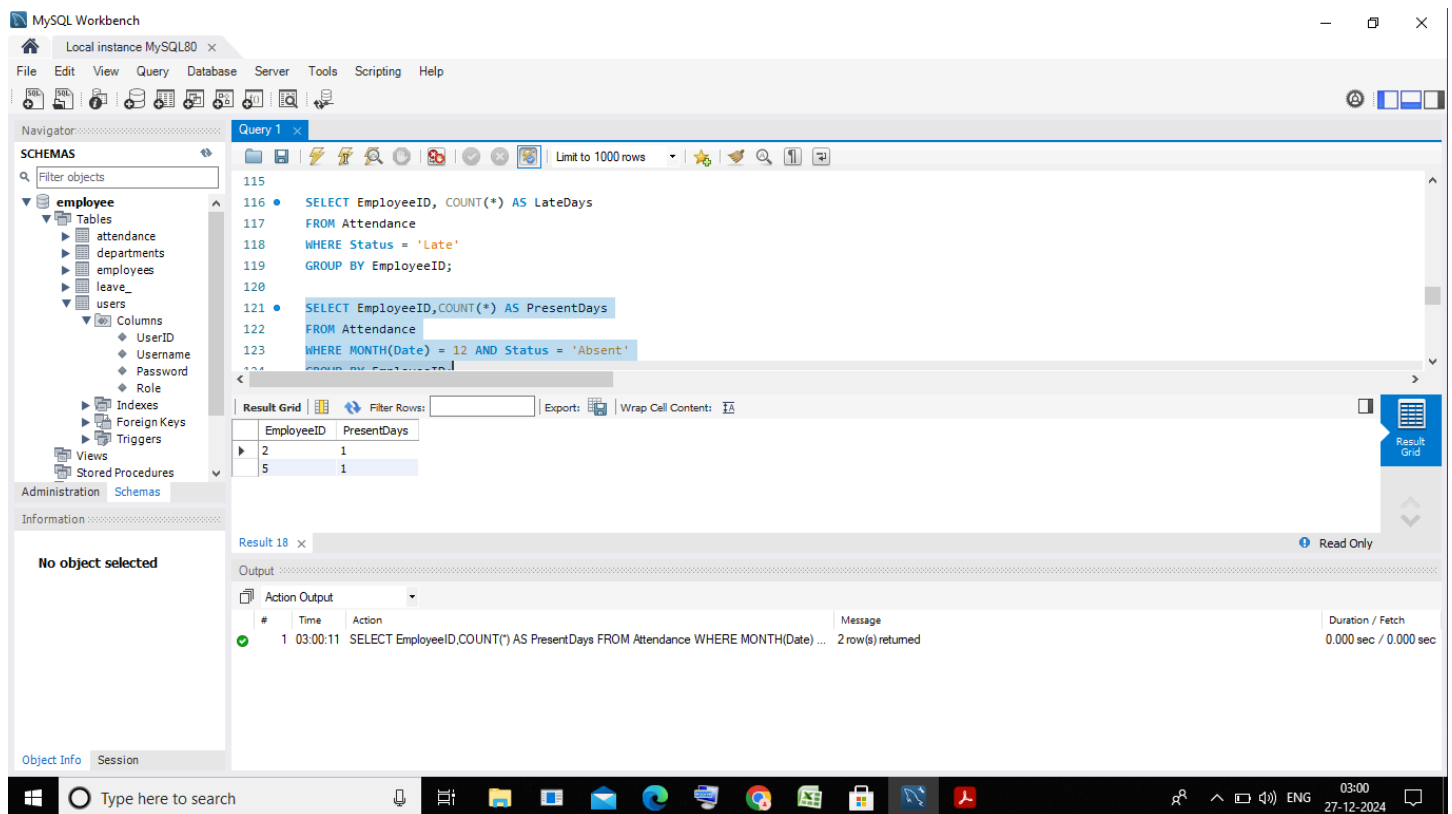
Absent days Count :

SELECT EmployeeID,COUNT(*) AS PresentDays

FROM Attendance

WHERE MONTH(Date) = 12 AND Status = 'Absent'

GROUP BY EmployeeID;



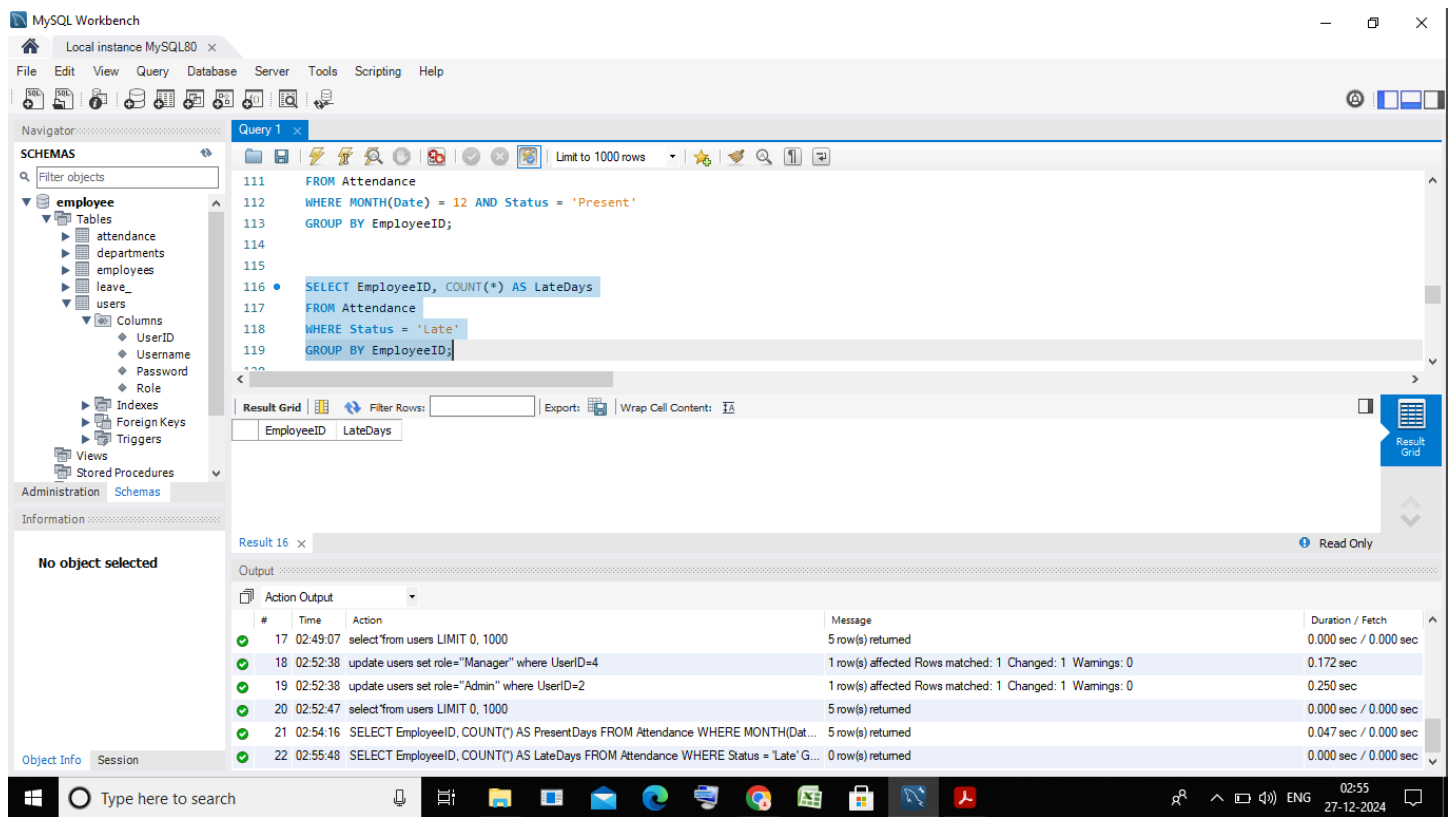
Late Count :

SELECT EmployeeID, COUNT(*) AS LateDays

FROM Attendance

WHERE Status = 'Late'

GROUP BY EmployeeID;



Additional Features

- Use **triggers** to auto-update employee leave balances.
- Add **stored procedures** for recurring reports.
- Implement role-based access control with **views**.