

# Project Title

## Employee Payroll & Attendance Management System (SQL Project)

*SQL Project using MySQL & Workbench*

Presented By

Viraj Patil

KJCOEMR

[GitHub Link](#)

## Project Overview

This project is a relational database management system built using MySQL, designed to manage employees, departments, attendance, and payroll efficiently. It demonstrates core database concepts such as tables, relationships, foreign keys, constraints, stored procedures, triggers, and views.

## Features

- Employee management with full details
- Department management
- Attendance tracking (Present / Absent / Leave)
- Payroll management with auto-calculated net salary
- Reports and queries (department-wise salary summaries, attendance summaries, highest- paid employees, monthly payroll reports)
- Stored procedures for automation
- Triggers for automatic calculations
- Views for simplified reporting
- Optional read-only HR users

## Database Schema

Tables:

- Departments – Department details
- Employees – Employee details linked to Departments
- Attendance – Daily attendance linked to Employees
- Payroll – Monthly payroll linked to Employees

Relationships:

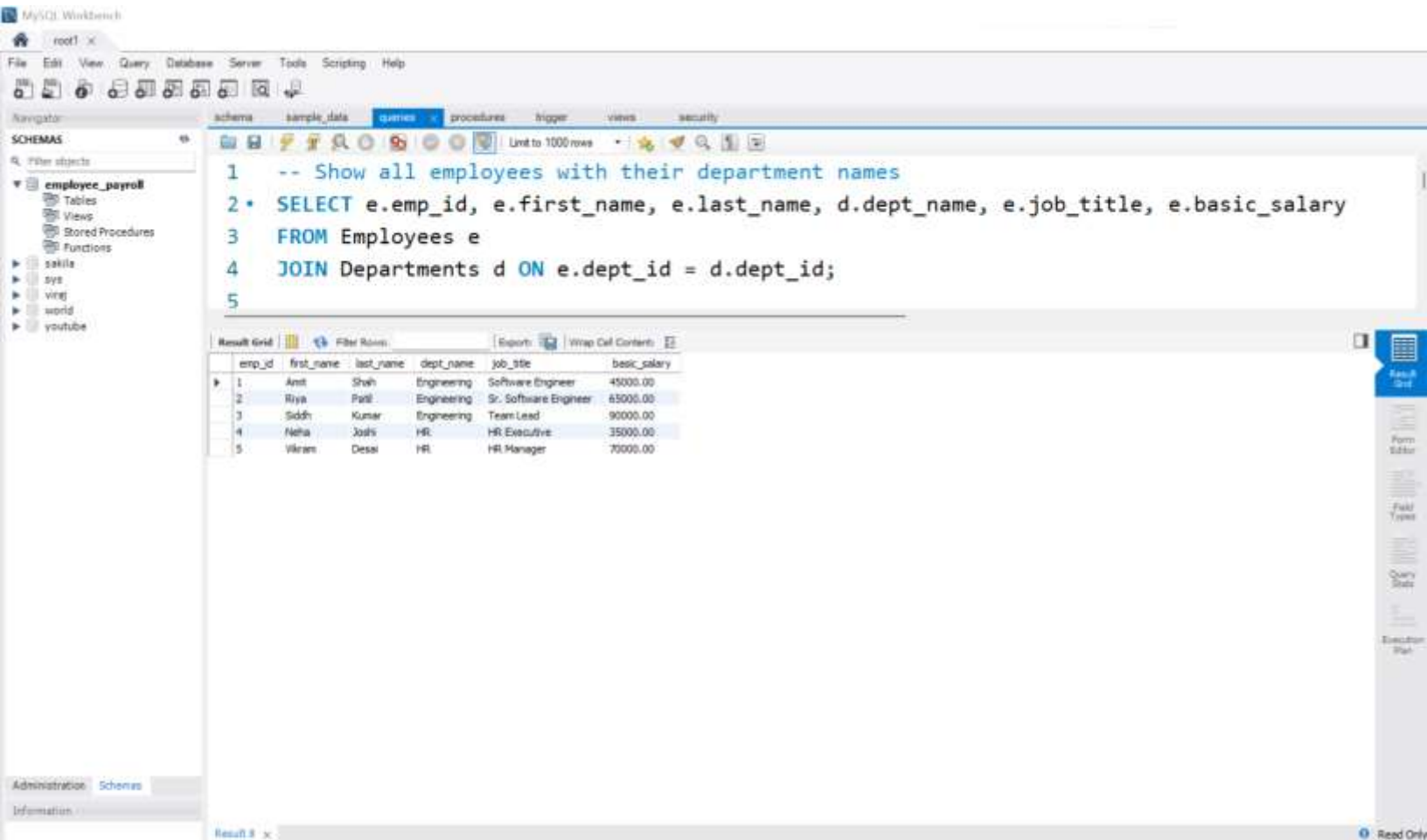
- One Department → Many Employees
- One Employee → Many Attendance Records
- One Employee → Many Payroll Records

## Setup & Installation

1. Install MySQL 8.0 and MySQL Workbench
2. Create database:  
`CREATE DATABASE employee_payroll; USE employee_payroll;`
3. Create all tables (Departments, Employees, Attendance, Payroll) with foreign keys
4. Insert sample data.
5. Create stored procedures, triggers, and views.
6. Run queries to test reports.

# 1.SQL Basic Queries

1. Show all employees with their department names :-



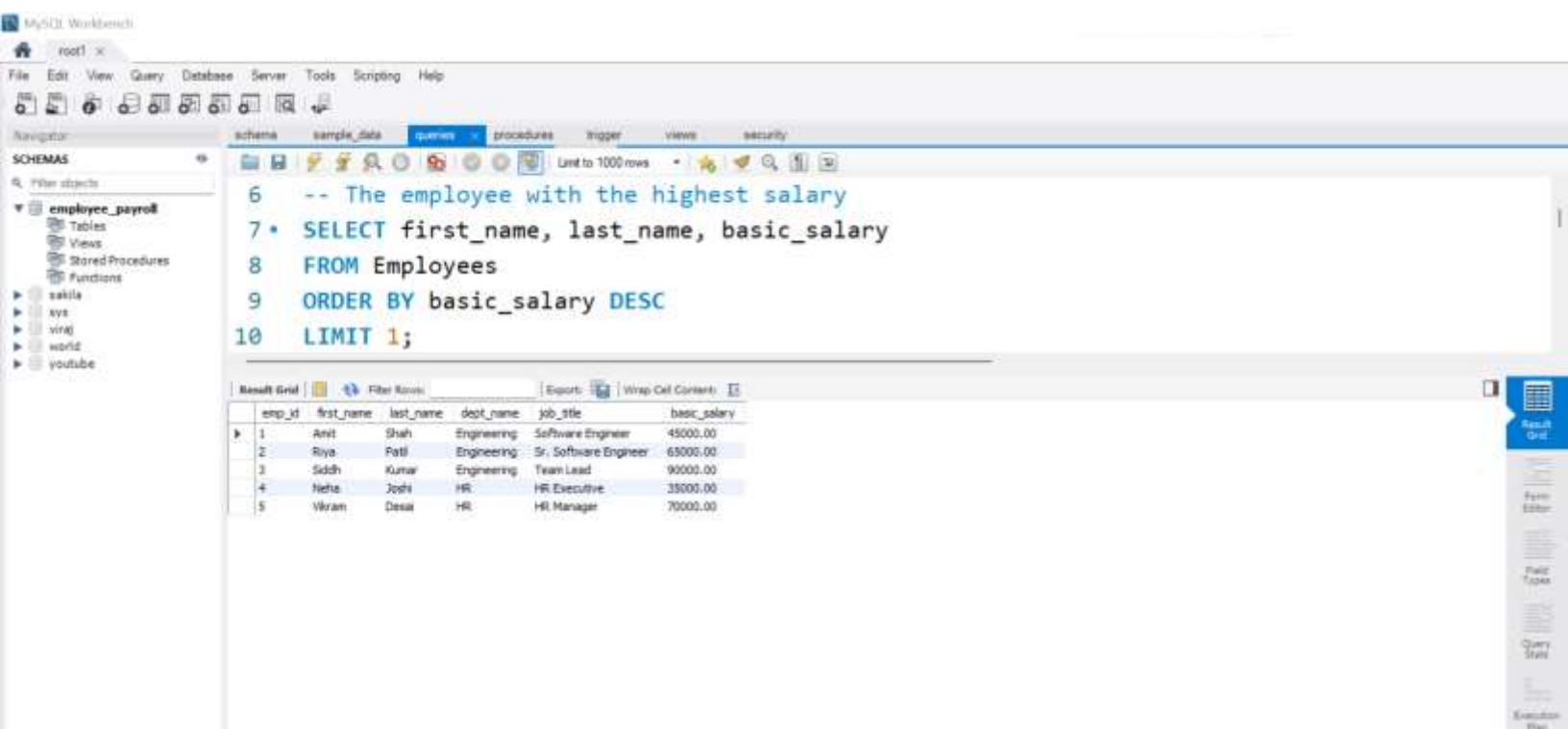
The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```
1 -- Show all employees with their department names
2 * SELECT e.emp_id, e.first_name, e.last_name, d.dept_name, e.job_title, e.basic_salary
3 FROM Employees e
4 JOIN Departments d ON e.dept_id = d.dept_id;
5
```

The result grid displays the following data:

emp_id	first_name	last_name	dept_name	job_title	basic_salary
1	Amit	Shah	Engineering	Software Engineer	45000.00
2	Riya	Patil	Engineering	Sr. Software Engineer	65000.00
3	Siddh	Kumar	Engineering	Team Lead	90000.00
4	Neha	Joshi	HR	HR Executive	35000.00
5	Vikram	Desai	HR	HR Manager	70000.00

2. The employee with the highest salary :-



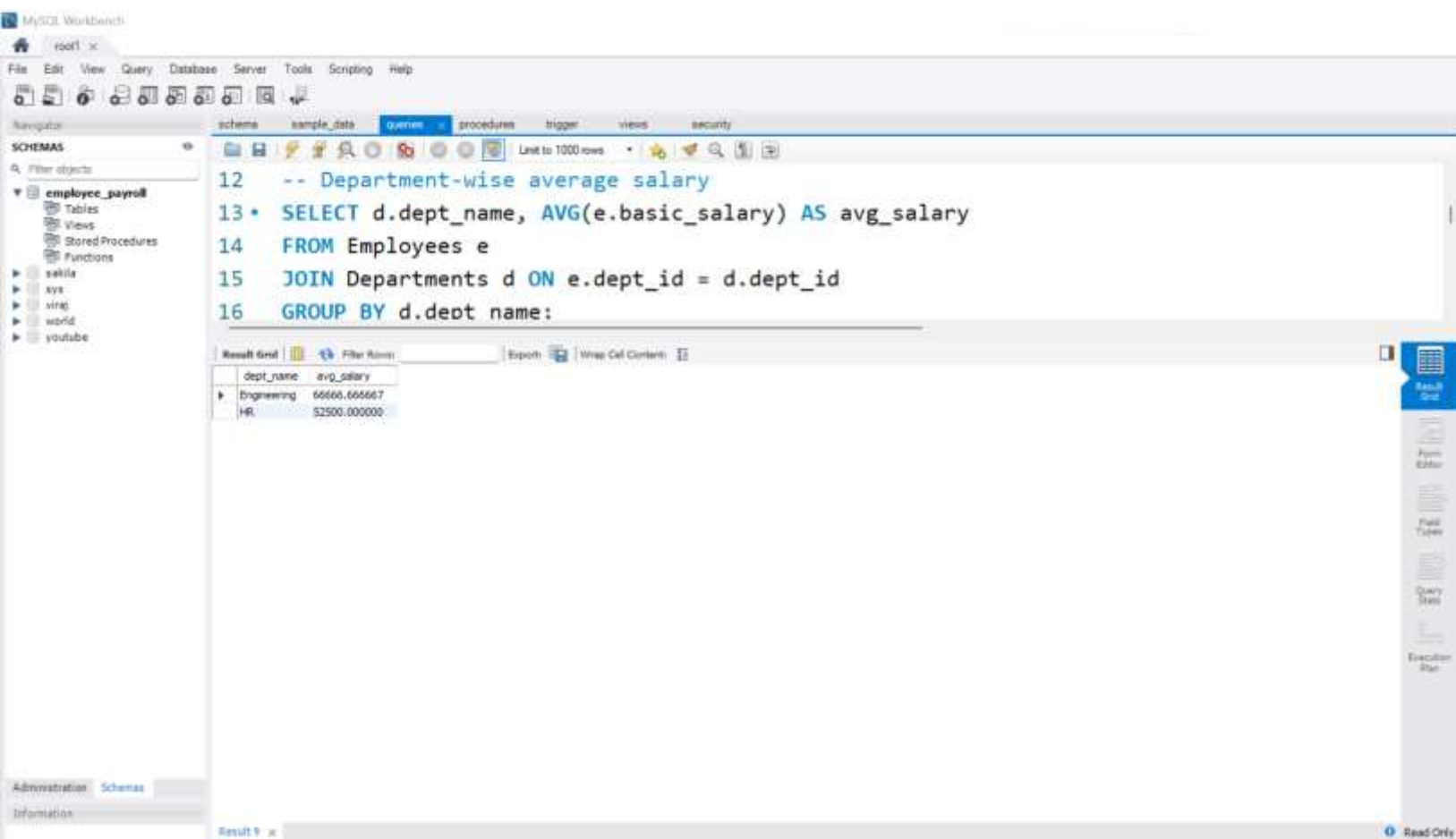
The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```
6 -- The employee with the highest salary
7 * SELECT first_name, last_name, basic_salary
8 FROM Employees
9 ORDER BY basic_salary DESC
10 LIMIT 1;
```

The result grid displays the following data:

emp_id	first_name	last_name	dept_name	job_title	basic_salary
1	Amit	Shah	Engineering	Software Engineer	45000.00
2	Riya	Patil	Engineering	Sr. Software Engineer	65000.00
3	Siddh	Kumar	Engineering	Team Lead	90000.00
4	Neha	Joshi	HR	HR Executive	35000.00
5	Vikram	Desai	HR	HR Manager	70000.00

### 3. Department wise average salary :-



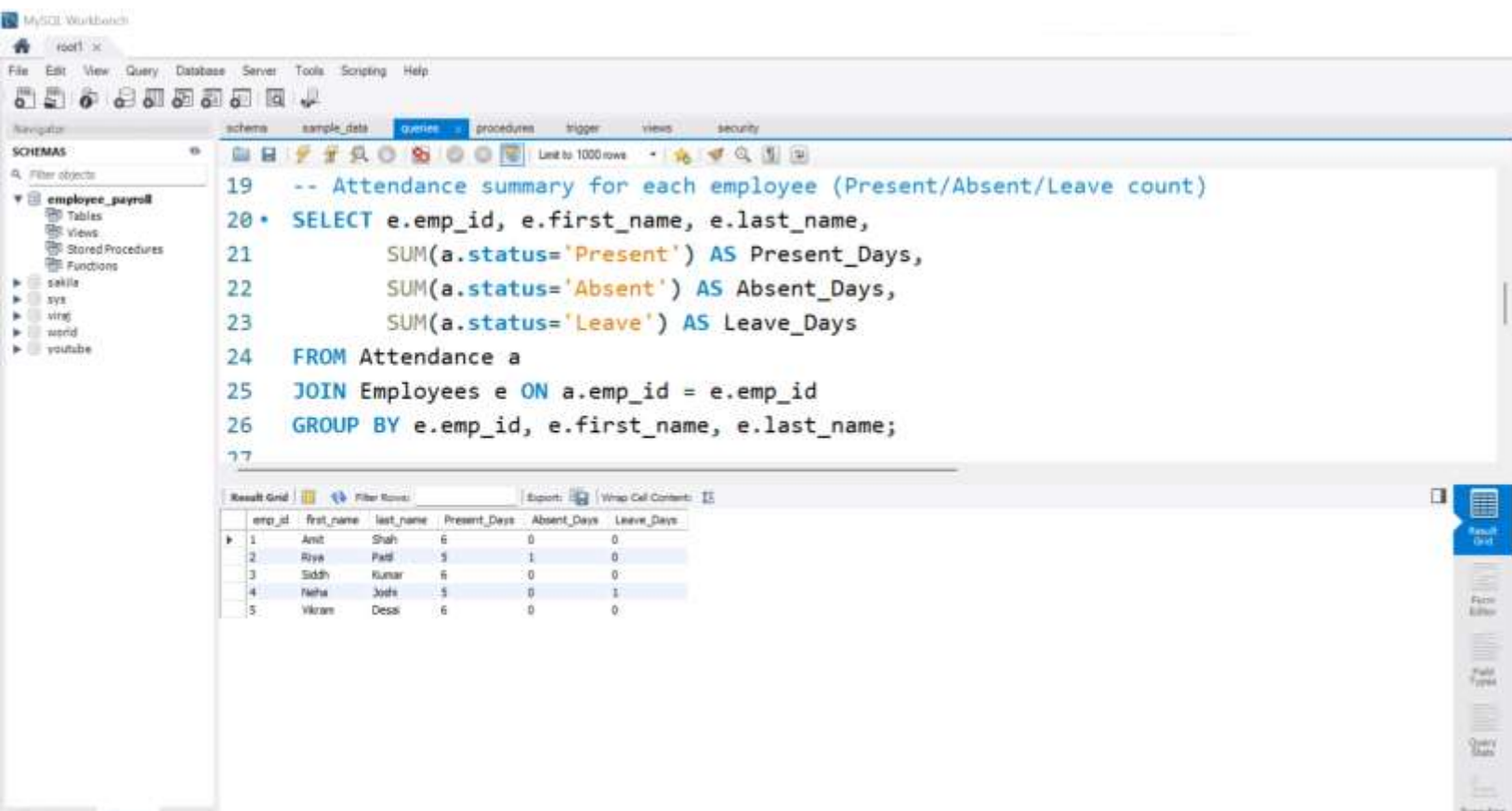
The screenshot shows the MySQL Workbench interface. The 'Queries' tab is active, displaying a SQL query to calculate the average salary for each department. The query is as follows:

```
12 -- Department-wise average salary
13 * SELECT d.dept_name, AVG(e.basic_salary) AS avg_salary
14 FROM Employees e
15 JOIN Departments d ON e.dept_id = d.dept_id
16 GROUP BY d.dept name;
```

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

dept_name	avg_salary
Engineering	66666.666667
HR	52500.000000

### 4. Attendance summary for each employee :-



The screenshot shows the MySQL Workbench interface. The 'Queries' tab is active, displaying a SQL query to generate an attendance summary for each employee. The query is as follows:

```
19 -- Attendance summary for each employee (Present/Absent/Leave count)
20 * SELECT e.emp_id, e.first_name, e.last_name,
21         SUM(a.status='Present') AS Present_Days,
22         SUM(a.status='Absent') AS Absent_Days,
23         SUM(a.status='Leave') AS Leave_Days
24 FROM Attendance a
25 JOIN Employees e ON a.emp_id = e.emp_id
26 GROUP BY e.emp_id, e.first_name, e.last_name;
```

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

emp_id	first_name	last_name	Present_Days	Absent_Days	Leave_Days
1	Anil	Shah	6	0	0
2	Riya	Patil	5	1	0
3	Siddh	Kumar	6	0	0
4	Neha	Joshi	5	0	1
5	Vikram	Desai	6	0	0

5. Payroll report for July 2025 :-

MySQL Workbench

root1 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

employee\_payroll

Tables

Views

Stored Procedures

Functions

sakila

sys

viraj

world

youtube

schema sample\_data queries procedures trigger views security

Limit to 1000 rows

```
27
28 -- Payroll report for July 2025 (with employee names)
29 * SELECT e.first_name, e.last_name, p.pay_month,
30        p.basic_salary, p.deductions, p.net_salary
31 FROM Payroll p
32 JOIN Employees e ON p.emp_id = e.emp_id
33 WHERE p.pay_month = '2025-07';
34
```

Result Grid

first_name	last_name	pay_month	basic_salary	deductions	net_salary
Amit	Shah	2025-07	45000.00	2000.00	43000.00
Riya	Patil	2025-07	65000.00	5000.00	60000.00
Siddh	Kumar	2025-07	90000.00	7000.00	83000.00
Neha	Joshi	2025-07	35000.00	1000.00	34000.00
Vikram	Desai	2025-07	70000.00	4000.00	66000.00

Administration Schemas

Information

Result 11 x

Read Only

6. Top 3 highest paid employees :-

MySQL Workbench

root1 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

employee\_payroll

Tables

Views

Stored Procedures

Functions

sakila

sys

viraj

world

youtube

schema sample\_data queries procedures trigger views security

Limit to 1000 rows

```
34
35 -- Top 3 highest-paid employees
36 * SELECT first_name, last_name, job_title, basic_salary
37 FROM Employees
38 ORDER BY basic_salary DESC
39 LIMIT 3;
```

Result Grid

first_name	last_name	job_title	basic_salary
Siddh	Kumar	Team Lead	90000.00
Vikram	Desai	HR Manager	70000.00
Riya	Patil	Sr. Software Engineer	65000.00

Administration Schemas

Information

Result 11 x

Read Only

## 7. Employees who took at least one leave :-

MySQL Workbench

root1 x

File Edit View Query Database Server Tools Scripting Help

schema sample\_data queries procedures trigger views security

Limit to 1000 rows

SCHEMAS

Filter objects

employee\_payroll

Tables

Views

Stored Procedures

Functions

sakila

sys

world

youtube

```
40
41 -- Employees who took at least one Leave
42 * SELECT DISTINCT e.emp_id, e.first_name, e.last_name
43 FROM Attendance a
44 JOIN Employees e ON a.emp_id = e.emp_id
45 WHERE a.status = 'Leave';
```

Result Grid

Filter Rows:

Export:

Wrap Cell Contents:

emp_id	first_name	last_name
4	Heba	Josh

Administration Schemas

Information

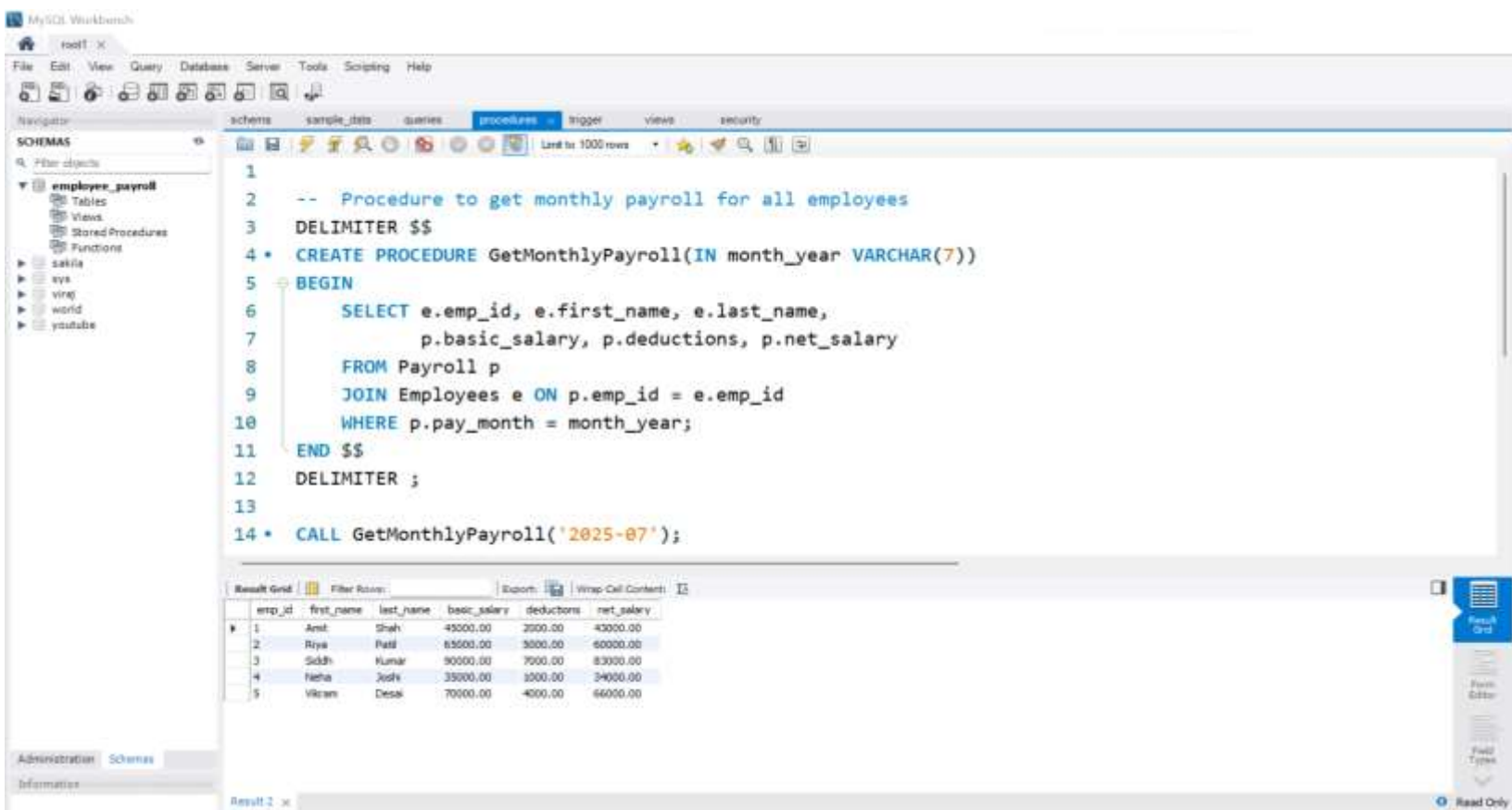
Result 13 x

Read Only



# 2.PROCEDURES

## 1.Procedure to get monthly payroll for all employees :-



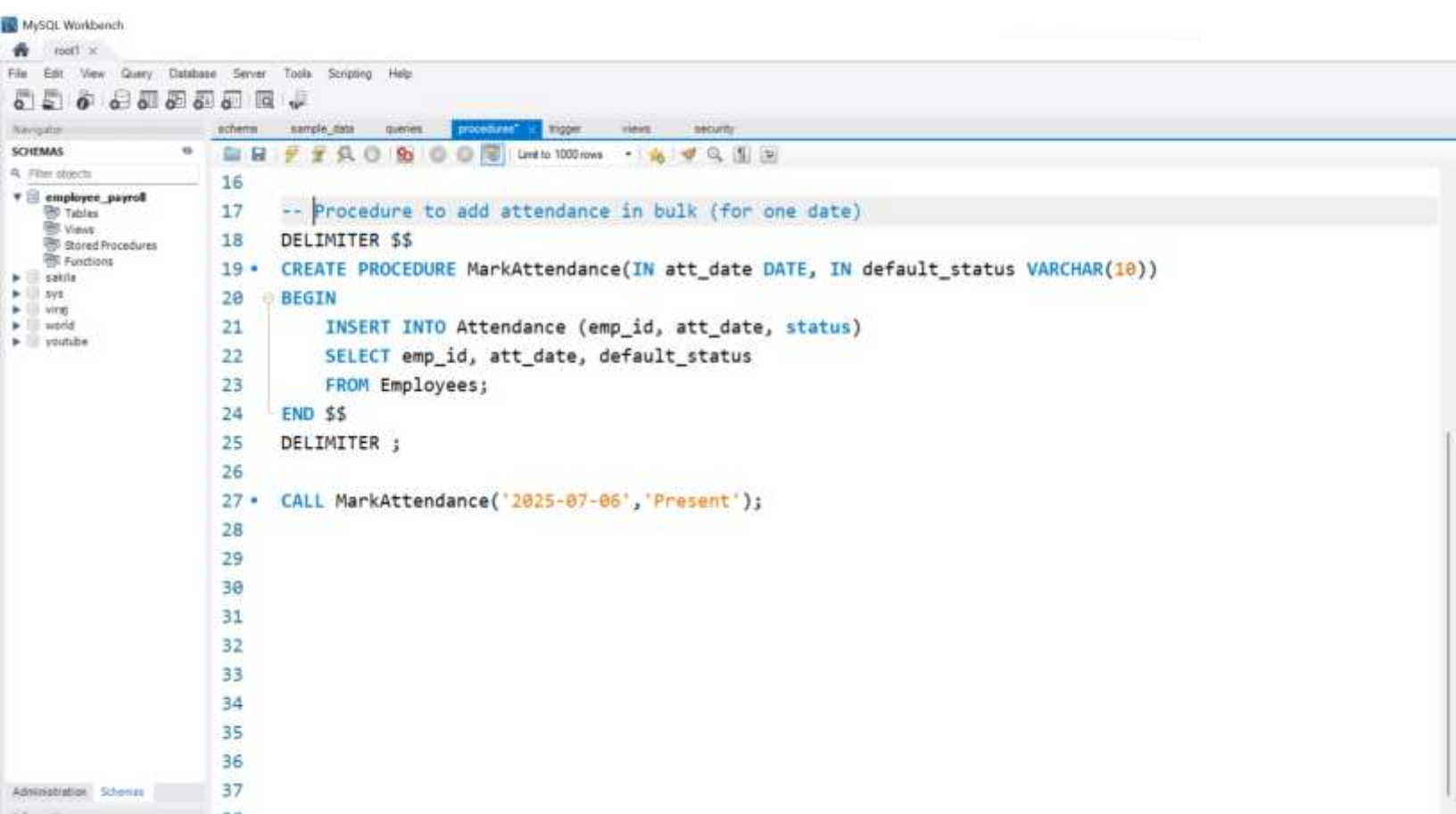
The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'employee\_payroll' selected. The main editor window shows the following SQL code:

```
1
2  -- Procedure to get monthly payroll for all employees
3  DELIMITER $$
4  * CREATE PROCEDURE GetMonthlyPayroll(IN month_year VARCHAR(7))
5  BEGIN
6      SELECT e.emp_id, e.first_name, e.last_name,
7             p.basic_salary, p.deductions, p.net_salary
8      FROM Payroll p
9      JOIN Employees e ON p.emp_id = e.emp_id
10     WHERE p.pay_month = month_year;
11 END $$
12 DELIMITER ;
13
14 * CALL GetMonthlyPayroll('2025-07');
```

Below the code editor, the 'Result Grid' shows the output of the procedure call:

emp_id	first_name	last_name	basic_salary	deductions	net_salary
1	Arshi	Shah	45000.00	2000.00	43000.00
2	Riya	Patil	65000.00	3000.00	62000.00
3	Siddh	Kumar	90000.00	7000.00	83000.00
4	Neha	Joshi	35000.00	1000.00	34000.00
5	Vikram	Desai	70000.00	4000.00	66000.00

## 2.Procedure to add attendance in bulk :-



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'employee\_payroll' selected. The main editor window shows the following SQL code:

```
16
17  -- Procedure to add attendance in bulk (for one date)
18  DELIMITER $$
19  * CREATE PROCEDURE MarkAttendance(IN att_date DATE, IN default_status VARCHAR(10))
20  BEGIN
21      INSERT INTO Attendance (emp_id, att_date, status)
22      SELECT emp_id, att_date, default_status
23      FROM Employees;
24  END $$
25  DELIMITER ;
26
27  * CALL MarkAttendance('2025-07-06', 'Present');
```

# 3.TRIGGER

1.Create a trigger to auto calculate net salary in payroll :-

MySQL Workbench

root1 x

File Edit View Query Database Server Tools Scripting Help

Navigation: schemas sample\_data queries procedures trigger views security

SCHEMAS

Filter objects

employee\_payroll

Tables

Views

Stored Procedures

Functions

sakila

sys

vinyl

world

youtube

```
1
2  -- Auto-calculate net salary in Payroll
3
4  DELIMITER $$
5  * CREATE TRIGGER before_payroll_insert
6  BEFORE INSERT ON Payroll
7  FOR EACH ROW
8  BEGIN
9      SET NEW.net_salary = NEW.basic_salary - NEW.deductions;
10 END $$
11 DELIMITER ;
12
13 * SELECT * FROM Payroll WHERE emp_id = 1;
```

Result grid

Filter Rows:

Grid: Edit: Export/Import: Wrap Cell Content:

pay_id	emp_id	pay_month	basic_salary	bonus	deductions	net_salary	created_at
1	1	2025-07	45000.00	0.00	2000.00	43000.00	2025-09-08 12:15:00

Administration Schemas

Information:

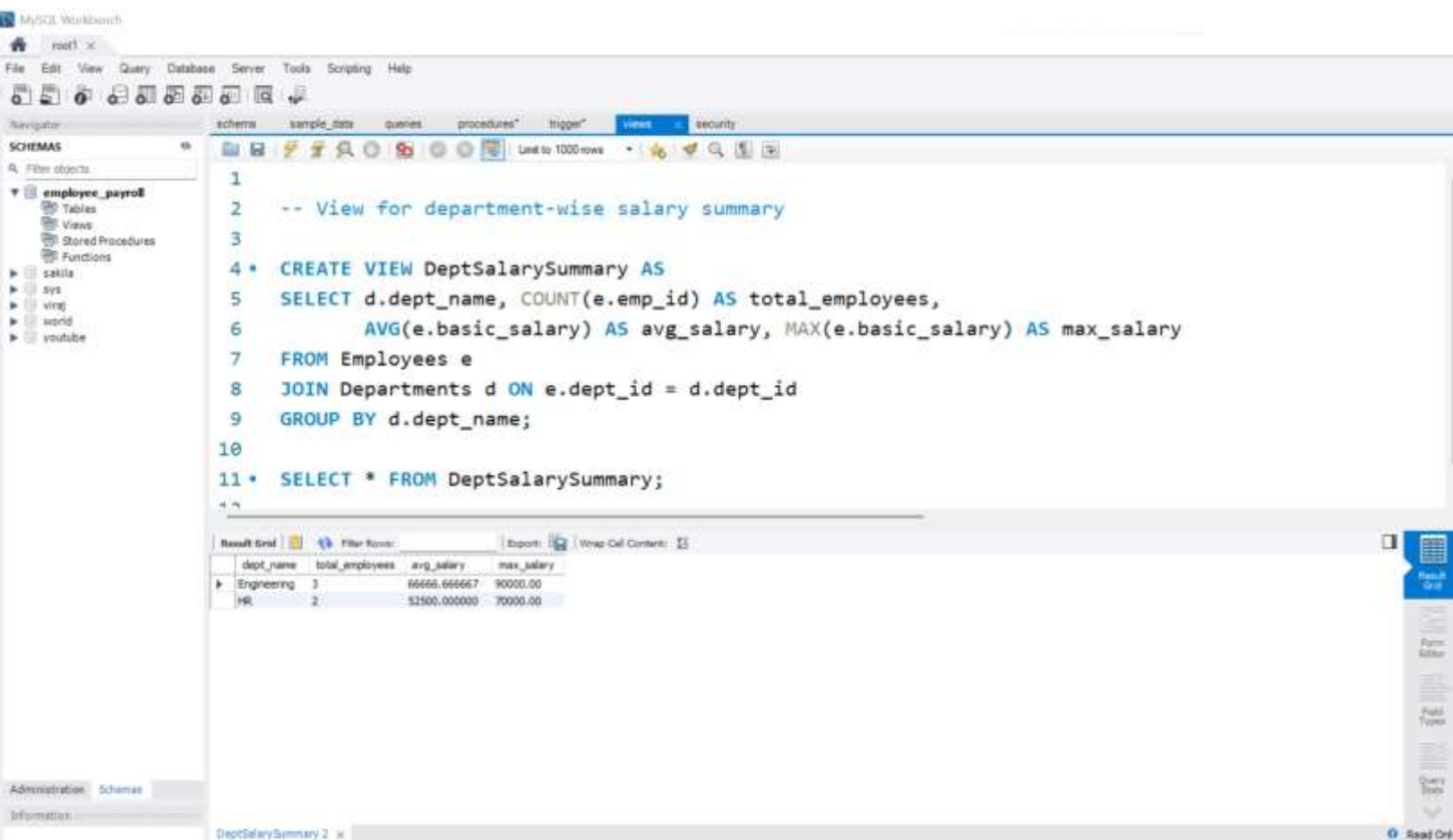
Payroll 1 x

Apply Reset



## 4.VIEW

View for department wise salary summary :-



The screenshot shows the MySQL Workbench interface. The 'Views' tab is active, displaying the following SQL code:

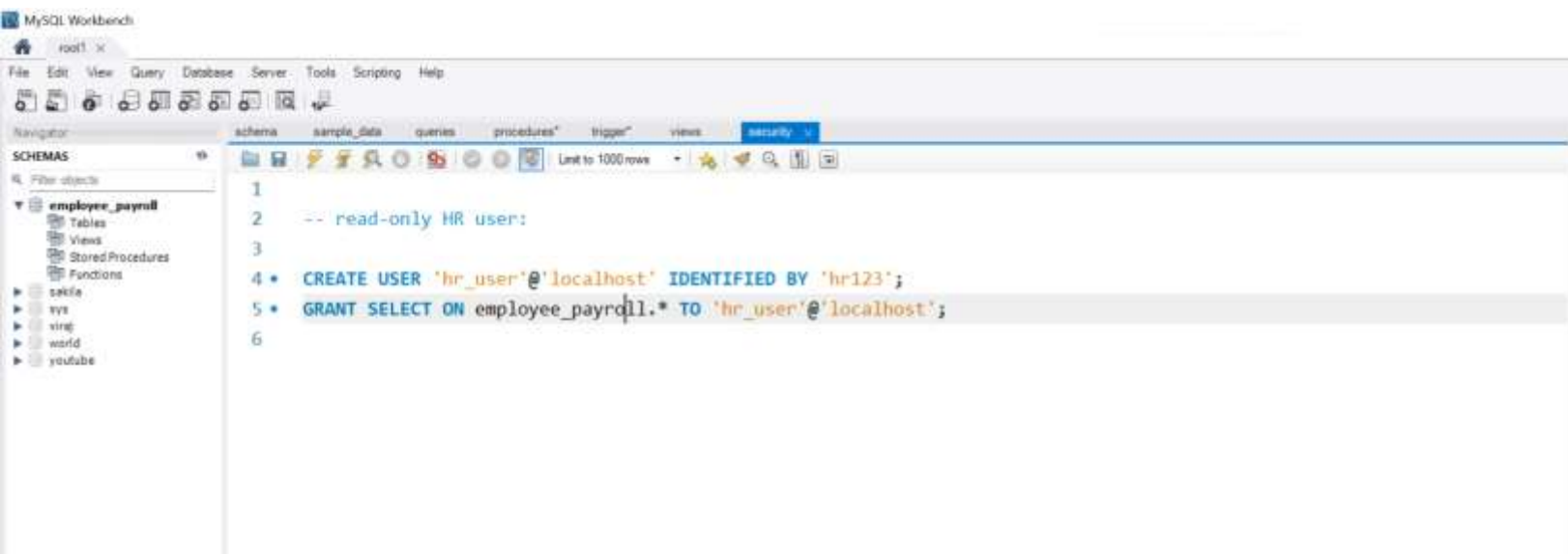
```
1
2  -- View for department-wise salary summary
3
4 * CREATE VIEW DeptSalarySummary AS
5 SELECT d.dept_name, COUNT(e.emp_id) AS total_employees,
6        AVG(e.basic_salary) AS avg_salary, MAX(e.basic_salary) AS max_salary
7 FROM Employees e
8 JOIN Departments d ON e.dept_id = d.dept_id
9 GROUP BY d.dept_name;
10
11 * SELECT * FROM DeptSalarySummary;
```

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

dept_name	total_employees	avg_salary	max_salary
Engineering	3	66666.666667	90000.00
HR	2	52500.000000	70000.00

## 5.SECURITY

Create a read only HR user :-



The screenshot shows the MySQL Workbench interface. The 'Security' tab is active, displaying the following SQL code:

```
1
2  -- read-only HR user:
3
4 * CREATE USER 'hr_user'@'localhost' IDENTIFIED BY 'hr123';
5 * GRANT SELECT ON employee_payroll.* TO 'hr_user'@'localhost';
6
```

## Sample Queries

List employees with department:

```
SELECT e.first_name, e.last_name, d.dept_name FROM Employees e  
JOIN Departments d ON e.dept_id = d.dept_id;
```

Department-wise average salary:

```
SELECT d.dept_name, AVG(e.basic_salary) AS avg_salary FROM Employees e  
JOIN Departments d ON e.dept_id = d.dept_id GROUP BY  
d.dept_name;
```

Attendance summary:

```
SELECT e.first_name, e.last_name, SUM(status='Present') AS  
    Present_Days, SUM(status='Absent') AS Absent_Days  
FROM Attendance a  
JOIN Employees e ON a.emp_id = e.emp_id GROUP BY e.emp_id;
```

Monthly payroll report (stored procedure):

```
CALL GetMonthlyPayroll('2025-07');
```

## Advanced Features

- Stored Procedures: Automate bulk attendance marking and payroll generation
- Triggers: Auto-calculate net salary
- Views: Department salary summary
- Referential Integrity: Foreign keys prevent invalid data

## Tools & Technologies

- MySQL 8.0
- MySQL Workbench
- SQL scripts for table creation, sample data, stored procedures, triggers, and views

## Conclusion

The Employee Payroll & Attendance Management System successfully streamlines HR operations by integrating employee, department, attendance, and payroll management into a single MySQL database. With stored procedures, triggers, and views, it ensures automation, accuracy, and efficiency.