HR AND PAYROLL MANAGEMENT SYSTEM

- ANISHA SUNIL WAIKAR

BATCH-T341

HR AND PAYROLL MANAGEMENT SYSTEM

This project focuses on building a reliable and structured database system to manage the essential HR and payroll operations within an organization. It handles everything from storing employee details and tracking attendance to managing leaves and calculating salaries.

The system is developed using MySQL, where multiple interlinked tables are created to reflect real-world HR processes. It supports efficient data retrieval and management using advanced SQL queries, including joins, nested conditions, and grouping functions. The purpose of this project is to demonstrate how a relational database can support the day-to-day HR activities of a company, improve record-keeping, and eliminate manual payroll errors through automation

PROJECT AIM:

- **Employee Management:** Add, update, and maintain employee records including full name, department, designation, salary, joining date, and employment status.
- **Attendance Management:** Track daily attendance with check-in and check-out times. Store records for each employee to support payroll calculations and performance evaluation.
- **Leave Management:** Manage leave requests, approvals, and leave balances. Keep a history of leave taken and categorize types such as sick leave, casual leave, and earned leave.
- **Payroll Processing:** Calculate monthly salaries based on working days, leaves, and allowances. Include deductions such as taxes or absences and generate structured salary slips.
- **Report Generation:** Provide summarized reports on employee attendance, leave records, and payroll data for HR analysis and auditing purposes

OBJECTIVES:

1. **Set up the HR and Payroll Management Database:**
   Design and populate the database with tables for employees, departments, branches, attendance, leaves, and payroll records.

2. **CRUD Operations:**
   Perform Create, Read, Update, and Delete operations on employee data, attendance logs, leave records, and payroll entries.

3. **Advanced SQL Queries:**
   Develop complex SQL queries to calculate salaries, generate monthly payroll summaries, track employee attendance trends, and monitor leave balances.

ER Diagram for HR And PAYROLL Management System

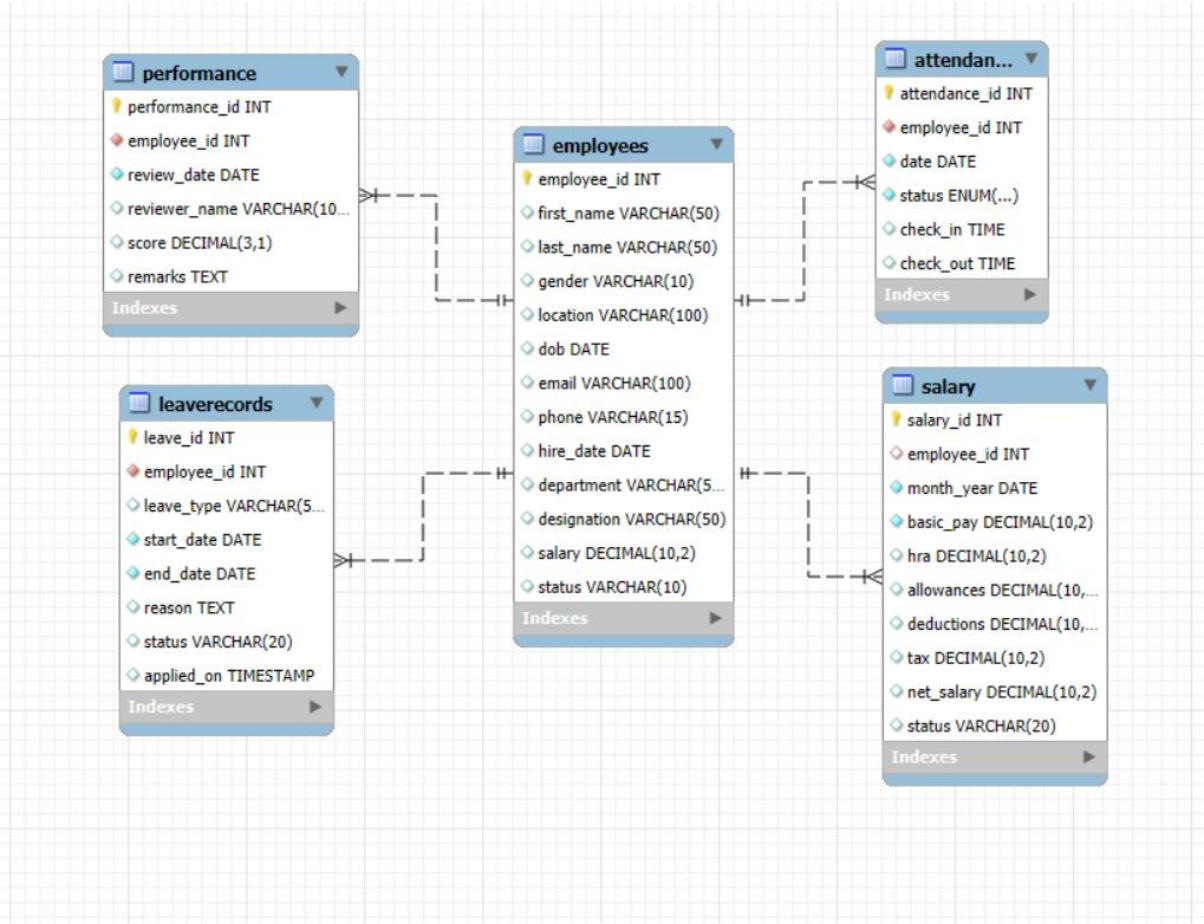

Table Description:

1)Employees

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
employee_id	int	NO	PRI	NULL	
first_name	varchar(50)	YES		NULL	
last_name	varchar(50)	YES		NULL	
gender	varchar(10)	YES		NULL	
location	varchar(100)	YES		NULL	
dob	date	YES		NULL	
email	varchar(100)	YES	UNI	NULL	
phone	varchar(15)	YES		NULL	
hire_date	date	YES		NULL	
department	varchar(50)	YES		NULL	
designation	varchar(50)	YES		NULL	
salary	decimal(10,2)	YES		NULL	
status	varchar(10)	YES		NULL	

2) Attendance

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
attendance_id	int	NO	PRI	NULL	
employee_id	int	NO	MUL	NULL	
date	date	NO		NULL	
status	enum('Present','Absent','Half-Day','Leave')	NO		NULL	
check_in	time	YES		NULL	
check_out	time	YES		NULL	

3) salary

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
salary_id	int	NO	PRI	NULL	auto_increment
employee_id	int	YES	MUL	NULL	
month_year	date	NO		NULL	
basic_pay	decimal(10,2)	NO		NULL	
hra	decimal(10,2)	YES		0.00	
allowances	decimal(10,2)	YES		0.00	
deductions	decimal(10,2)	YES		0.00	
tax	decimal(10,2)	YES		0.00	
net_salary	decimal(10,2)	YES		NULL	STORED GENERATED
status	varchar(20)	YES		NULL	

4) LeaveRecords

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
leave_id	int	NO	PRI	NULL	auto_increment
employee_id	int	NO	MUL	NULL	
leave_type	varchar(50)	YES		NULL	
start_date	date	NO		NULL	
end_date	date	NO		NULL	
reason	text	YES		NULL	
status	varchar(20)	YES		Pending	
applied_on	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED

4)Performance

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
performance_id	int	NO	PRI	NULL	auto_increment
employee_id	int	NO	MUL	NULL	
review_date	date	NO		NULL	
reviewer_name	varchar(100)	YES		NULL	
score	decimal(3,1)	YES		NULL	
remarks	text	YES		NULL	

CREATING DATABASE:

CREATE DATABASE hr_payroll_system;
use hr_payroll_system;

Table Creation & Insertion Commands:

1) Create table employee

CREATE TABLE Employees
(employee_id INT PRIMARY KEY,
first_name VARCHAR(50),
last_name VARCHAR(50),
gender VARCHAR(10),
location VARCHAR(100),
dob DATE,
email VARCHAR(100) UNIQUE,
phone VARCHAR(15),
hire_date DATE,
department VARCHAR(50),
designation VARCHAR(50),
salary DECIMAL(10,2),
status VARCHAR(10));

Inserting Values into employees:

INSERT INTO Employees
(employee_id, first_name, last_name, gender, location, dob, email, phone, hire_date, department, designation, salary, status)
VALUES
(101, 'Divya', 'Desai', 'Female', 'Andheri', '1991-07-02', 'divya.desai101@company.com', '9479608458', '2023-08-12', 'Operations', 'Assistant', 35053.14, 'Active'),
(102, 'Rohit', 'Chopra', 'Male', 'Kalbadevi', '1999-07-15', 'rohit.chopra102@company.com', '9891468251', '2023-07-24', 'Finance', 'Executive', 73000.84, 'Active'),
(103, 'Rohit', 'Naik', 'Male', 'Santacruz', '1990-12-02', 'rohit.naik103@company.com', '9317718930', '2023-09-13', 'Operations', 'Assistant', 33579.76, 'Active'),
(104, 'Sneha', 'Salvi', 'Female', 'Tilak Nagar', '1983-08-15', 'sneha.salvi104@company.com', '9184275390', '2024-11-23', 'Marketing', 'Analyst', 25364.3, 'Active'),
(105, 'Kavita', 'Patel', 'Female', 'Chembur', '1991-10-11', 'kavita.patel105@company.com', '9617857180', '2024-01-16', 'Operations', 'Assistant', 52078.72, 'Active',....);

SELECT * FROM employees;

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

employee_id	first_name	last_name	gender	location	dob	email	phone	hire_date	department	designation	salary	status
101	Divya	Desai	Female	Andheri	1991-07-02	divya.desai101@company.com	9479608458	2023-08-12	Operations	Assistant	35053.14	Active
102	Rohit	Chopra	Male	Kalbadevi	1999-07-15	rohit.chopra102@company.com	9891468251	2023-07-24	Finance	Executive	73000.84	Active
103	Rohit	Naik	Male	Santacruz	1990-12-02	rohit.naik103@company.com	9317718930	2023-09-13	Operations	Assistant	33579.76	Active
104	Sneha	Salvi	Female	Tilak Nagar	1983-08-15	sneha.salvi104@company.com	9184275390	2024-11-23	Marketing	Analyst	25364.30	Active
105	Kavita	Patel	Female	Chembur	1991-10-11	kavita.patel105@company.com	9617857180	2024-01-16	Operations	Assistant	52078.72	Active

2) Create table Attendance

CREATE TABLE Attendance
(attendance_id INT PRIMARY KEY,
employee_id INT NOT NULL,
date DATE NOT NULL,
status ENUM('Present', 'Absent', 'Half-Day', 'Leave') NOT NULL,
check_in TIME,
check_out TIME,
FOREIGN KEY (employee_id) REFERENCES Employee(employee_id));

Inserting Values into attendance:

INSERT INTO Attendance (attendance_id, employee_id, date, status, check_in, check_out)
VALUES
(1, 101, '2025-06-02', 'Present', '09:16:00', '17:10:00'),
(2, 101, '2025-06-03', 'Present', '09:17:00', '17:19:00'),
(3, 101, '2025-06-04', 'Present', '09:11:00', '17:28:00'),
(4, 101, '2025-06-05', 'Present', '09:10:00', '17:27:00'),
(5, 101, '2025-06-06', 'Present', '09:24:00', '17:20:00',.....);

SELECT * FROM attendance;

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import:

attendance_id	employee_id	date	status	check_in	check_out
1	101	2025-06-02	Present	09:16:00	17:10:00
2	101	2025-06-03	Present	09:17:00	17:19:00
3	101	2025-06-04	Present	09:11:00	17:28:00
4	101	2025-06-05	Present	09:10:00	17:27:00
5	101	2025-06-06	Present	09:24:00	17:20:00

3) Create Table Salary

CREATE TABLE Salary
(salary_id INT PRIMARY KEY AUTO_INCREMENT,
employee_id INT,
month_year DATE NOT NULL,
basic_pay DECIMAL(10, 2) NOT NULL,
hra DECIMAL(10, 2) DEFAULT 0,
allowances DECIMAL(10, 2) DEFAULT 0,
deductions DECIMAL(10, 2) DEFAULT 0,
tax DECIMAL(10, 2) DEFAULT 0,
net_salary DECIMAL(10, 2) GENERATED ALWAYS AS (
basic_pay + hra + allowances - deductions - tax) STORED,
status VARCHAR(20),
FOREIGN KEY (employee_id) REFERENCES Employee(employee_id));

Inserting Values into Salary:

INSERT INTO Salary
(employee_id, month_year, basic_pay, hra, allowances, deductions, tax, status)
VALUES
('101', '2025-07-01', 30000.00, 6000.00, 2500.00, 1000.00, 2000.00, 'Paid'),
('102', '2025-07-01', 65000.00, 8000.00, 3000.00, 2000.00, 3000.00, 'Paid'),
('103', '2025-07-01', 31000.00, 5000.00, 2500.00, 800.00, 1800.00, 'Paid'),
('104', '2025-07-01', 24000.00, 4000.00, 2000.00, 500.00, 1400.00, 'Paid'),
('105', '2025-07-01', 46000.00, 5000.00, 2500.00, 1000.00, 2000.00, 'Paid',.....);

SELECT * FROM salary;

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

salary_id	employee_id	month_year	basic_pay	hra	allowances	deductions	tax	net_salary	status
1	101	2025-07-01	30000.00	6000.00	2500.00	1000.00	2000.00	35500.00	Paid
2	102	2025-07-01	65000.00	8000.00	3000.00	2000.00	3000.00	71000.00	Paid
3	103	2025-07-01	31000.00	5000.00	2500.00	800.00	1800.00	35900.00	Paid
4	104	2025-07-01	24000.00	4000.00	2000.00	500.00	1400.00	28100.00	Paid
5	105	2025-07-01	46000.00	5000.00	2500.00	1000.00	2000.00	50500.00	Paid
6	106	2025-07-01	36000.00	5000.00	2000.00	1000.00	2000.00	40000.00	Paid
7	107	2025-07-01	37000.00	4000.00	2000.00	1500.00	2000.00	39500.00	Paid
8	108	2025-07-01	58000.00	6000.00	3000.00	1500.00	2000.00	63500.00	Paid
9	109	2025-07-01	45000.00	5000.00	2000.00	1000.00	1500.00	49500.00	Paid
10	110	2025-07-01	47000.00	6000.00	3000.00	1200.00	1800.00	53000.00	Paid

4) **create table LeaveRecords**

CREATE TABLE LeaveRecords
(leave_id INT PRIMARY KEY AUTO_INCREMENT,
employee_id INT NOT NULL,
leave_type VARCHAR(50),
start_date DATE NOT NULL,
end_date DATE NOT NULL,
reason TEXT,
status VARCHAR(20) DEFAULT 'Pending',
applied_on TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
FOREIGN KEY (employee_id) REFERENCES Employee(employee_id));

Inserting Values into LeaveRecords:

INSERT INTO LeaveRecords (employee_id, leave_type, start_date, end_date, reason, status)
VALUES
(101, 'Sick', '2025-07-02', '2025-07-03', 'Flu symptoms', 'Approved'),
(102, 'Casual', '2025-07-06', '2025-07-06', 'Bank work', 'Approved'),
(103, 'Earned', '2025-07-16', '2025-07-18', 'Family event', 'Pending'),
(104, 'Sick', '2025-07-04', '2025-07-04', 'Fever', 'Approved'),
(105, 'Casual', '2025-07-07', '2025-07-07', 'Personal work', 'Approved',....);

SELECT * FROM leaveRecords;

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

leave_id	employee_id	leave_type	start_date	end_date	reason	status	applied_on
1	101	Sick	2025-07-02	2025-07-03	Flu symptoms	Approved	2025-07-09 22:40:12
2	102	Casual	2025-07-06	2025-07-06	Bank work	Approved	2025-07-09 22:40:12
3	103	Earned	2025-07-16	2025-07-18	Family event	Pending	2025-07-09 22:40:12
4	104	Sick	2025-07-04	2025-07-04	Fever	Approved	2025-07-09 22:40:12
5	105	Casual	2025-07-07	2025-07-07	Personal work	Approved	2025-07-09 22:40:12
6	106	Maternity	2025-07-01	2025-07-10	Maternity phase	Approved	2025-07-09 22:40:12
7	107	Earned	2025-07-17	2025-07-19	Annual leave	Pending	2025-07-09 22:40:12
8	108	Sick	2025-07-03	2025-07-04	Cold and cough	Approved	2025-07-09 22:40:12
9	109	Casual	2025-07-08	2025-07-08	Home function	Approved	2025-07-09 22:40:12
10	110	Earned	2025-07-18	2025-07-20	Pilgrimage	Pending	2025-07-09 22:40:12
11	111	Sick	2025-07-05	2025-07-06	Flu	Approved	2025-07-09 22:40:12
12	112	Casual	2025-07-09	2025-07-09	Family errand	Approved	2025-07-09 22:40:12

5) **create table Performance**

CREATE TABLE Performance
(performance_id INT PRIMARY KEY AUTO_INCREMENT,
employee_id INT NOT NULL,
review_date DATE NOT NULL,
reviewer_name VARCHAR(100),
score DECIMAL(3,1) CHECK (score BETWEEN 0 AND 5),
remarks TEXT,
FOREIGN KEY (employee_id) REFERENCES Employee(employee_id));

Inserting Values into Performance:

INSERT INTO Performance (employee_id, review_date, reviewer_name, score, remarks)
VALUES
(101, '2025-06-30', 'Neha Sharma', 4.5, 'Excellent team collaboration and consistency.'),
(102, '2025-06-30', 'Amit Kapoor', 3.8, 'Meets expectations, reliable work ethic.'),
(103, '2025-06-30', 'Neha Sharma', 4.1, 'Shows initiative and learns quickly.'),
(104, '2025-06-30', 'Raj Mehta', 3.2, 'Performance needs improvement in deadlines.'),
(105, '2025-06-30', 'Sneha Salvi', 4.3, 'Strong ownership of tasks and timely delivery.'),
(106, '2025-06-30', 'Anjali Rana', 4.0, 'Technically sound with attention to detail.'),
(107, '2025-06-30', 'Rohit Mehta', 2.9, 'Inconsistent attendance affected delivery.',....);

SELECT * FROM Performance;

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

performance_id	employee_id	review_date	reviewer_name	score	remarks
1	101	2025-06-30	Neha Sharma	4.5	Excellent team collaboration and consistency.
2	102	2025-06-30	Amit Kapoor	3.8	Meets expectations, reliable work ethic.
3	103	2025-06-30	Neha Sharma	4.1	Shows initiative and learns quickly.
4	104	2025-06-30	Raj Mehta	3.2	Performance needs improvement in deadlines.
5	105	2025-06-30	Sneha Salvi	4.3	Strong ownership of tasks and timely delivery.
6	106	2025-06-30	Anjali Rana	4.0	Technically sound with attention to detail.
7	107	2025-06-30	Rohit Mehta	2.9	Inconsistent attendance affected delivery.
8	108	2025-06-30	Meena Joshi	3.7	Supportive to team members and responsive.
9	109	2025-06-30	Rohit Naik	3.4	Good at planning, needs faster execution.
10	110	2025-06-30	Divya Desai	4.2	Reliable and disciplined throughout the year.

BASIC QUESTIONS

1.Which departments exist in the Employee table?

SELECT DISTINCT department **FROM** Employees;

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department
Operations
Finance
Marketing
IT
HR

2.Can you list all salaries in descending order without showing duplicate values?

SELECT DISTINCT net_salary **FROM** salary
ORDER BY net_salary **DESC;**

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

net_salary
78000.00
76000.00
75000.00
73500.00
72500.00
71800.00
71000.00
68000.00
67500.00
67000.00
66500.00
64700.00
63500.00
63000.00

3. List all the days an employee was marked present in June 2025.

SELECT * FROM Attendance
WHERE employee_id = 102 **AND** status = 'Present'**AND** date **BETWEEN** '2025-06-01'
AND '2025-06-30'**;**

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

attendance_id	employee_id	date	status	check_in	check_out
22	102	2025-06-03	Present	09:18:00	17:29:00
23	102	2025-06-04	Present	09:03:00	17:01:00
24	102	2025-06-05	Present	09:04:00	17:04:00
25	102	2025-06-06	Present	09:28:00	17:18:00
26	102	2025-06-09	Present	09:08:00	17:25:00
27	102	2025-06-10	Present	09:18:00	17:12:00
28	102	2025-06-11	Present	09:06:00	17:23:00
29	102	2025-06-12	Present	09:10:00	17:25:00
30	102	2025-06-13	Present	09:15:00	17:26:00
34	102	2025-06-19	Present	09:20:00	17:09:00
35	102	2025-06-20	Present	09:25:00	17:18:00

4.find out which employees have never applied for or taken leave, based on attendance data?

SELECT employee_id, first_name, last_name
FROM Employee
WHERE employee_id **NOT IN (SELECT DISTINCT** employee_id
FROM Attendance
WHERE status = 'Leave'**);**

OUTPUT:

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

employee_id	first_name	last_name
141	Amit	Bose
142	Raj	Mehta
143	Divya	Kapoor
144	Meena	Sharma
145	Sneha	Sharma
NULL	NULL	NULL

5.identify the number of days each employee was unavailable for work, based on attendance records?

```
SELECT employee_id, COUNT(*) AS total_absent_days
FROM Attendance
WHERE status = 'Absent'
GROUP BY employee_id;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	total_absent_days
101	3
102	2
103	3
104	2
105	1
106	1
107	2
108	3
109	3
110	5
111	3

6. Find the dates on which employee 103 was either present or marked half day.

```
SELECT date, status
FROM Attendance
WHERE employee_id = 103 AND status IN ('Present', 'Half Day');
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

date	status
2025-06-05	Present
2025-06-06	Present
2025-06-09	Present
2025-06-10	Present
2025-06-11	Present
2025-06-17	Present
2025-06-18	Present

7. How can you check which employees have any kind of attendance record on July 5, 2025 — without caring whether they were present, absent, or on leave?

```
SELECT DISTINCT employee_id
FROM Attendance
WHERE date = '2025-06-05';
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id
101
102
103
104
105
106
107
108
109
110
111

SUB-QUERIES

1.Detects underpaid employees within each department — useful for HR reviews or pay parity checks

```
SELECT employee_id, CONCAT(first_name, ' ', last_name) AS employee_name,
department, salary
FROM Employees
WHERE salary < (SELECT AVG(salary)
FROM Employees AS dept_avg
WHERE dept_avg.department = Employees.department);
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	employee_name	department	salary
101	Divya Desai	Operations	35053.14
103	Rohit Naik	Operations	33579.76
104	Sneha Salvi	Marketing	25364.30
106	Divya Bose	IT	38939.62
107	Anjali Rana	Finance	40201.32
109	Meena Rana	Marketing	50823.53
110	Anjali Kapoor	Finance	53069.75
114	Meena Joshi	HR	41349.83
115	Divya Joshi	Operations	28174.14
117	Neha Kapoor	Finance	51669.64
121	Meena Desai	Marketing	52242.36
122	Meena Mehta	Marketing	39146.54
128	Meena Joshi	IT	35280.14

2. Show the names of employees who earn the highest salary from the salary table

```
SELECT first_name, last_name
FROM Employees
WHERE employee_id IN (SELECT employee_id
FROM salary
WHERE net_salary = (SELECT MAX(net_salary) FROM salary));
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name
Raj	Nair

3.List departments where at least one employee has a salary above ₹60,000.

```
SELECT DISTINCT department,salary
FROM Employees
WHERE employee_id IN (
SELECT employee_id
FROM salary
WHERE net_salary > 60000);
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department	salary
IT	66273.81
Operations	67502.90
Finance	78622.39
Marketing	69531.14
Operations	67681.28
IT	77350.81
HR	71827.56
Finance	74807.32
Marketing	66429.99
Marketing	66480.48
HR	67472.32
IT	62108.58
Operations	63259.89
HR	72224.56

4.Fetch employees whose basic pay is more than the average basic pay

```
SELECT employee_id, basic_pay
FROM salary
WHERE basic_pay >
(SELECT AVG(basic_pay) FROM salary);
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	basic_pay
102	65000.00
108	58000.00
111	51000.00
112	60000.00
113	50000.00
116	70000.00
118	74000.00
120	61000.00

58000.00

5.Find employees who took leave on the day when the maximum number of leaves were recorded.

```
SELECT employee_id, status FROM Attendance
WHERE status = 'Leave' AND date = (SELECT date FROM Attendance
WHERE status = 'Leave'
GROUP BY date
ORDER BY COUNT(*) DESC
LIMIT 1);
```

<u>OUTPUT:</u>

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	status
102	Leave
107	Leave
108	Leave
112	Leave
115	Leave
117	Leave
120	Leave
122	Leave

JOINS

1. Identifies employees who are absent too often and underperforming — useful for HR interventions.

```
SELECT e.employee_id,
CONCAT(e.first_name, ' ', e.last_name) AS employee_name,
COUNT(a.attendance_id) AS total_absents, p.score AS performance_score
FROM Employees e
JOIN Attendance a ON e.employee_id = a.employee_id
JOIN Performance p ON e.employee_id = p.employee_id
WHERE a.status = 'Absent'
GROUP BY e.employee_id, employee_name, p.score
HAVING total_absents > 2 AND p.score < 3.5
ORDER BY total_absents DESC;
```

<u>OUTPUT:</u>

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	employee_name	total_absents	performance_score
109	Meena Rana	3	3.4
121	Meena Desai	3	3.2
127	Rohit Kapoor	3	3.0
134	Raj Rana	3	3.1
140	Karan Iyer	3	2.7
146	Anjali Bose	3	3.2
149	Divya Nair	3	2.8

2. Highlights departments where salary generation is missing — useful for payroll checks.

```
SELECT e.department,
COUNT(e.employee_id) AS total_employees,
COUNT(s.salary_id) AS paid_salaries
FROM Employees e
LEFT JOIN MonthlySalary s ON e.employee_id = s.employee_id AND s.month_year = '2025-07-01'
GROUP BY e.department;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department	total_employees	paid_salaries
Operations	11	11
Finance	11	11
Marketing	12	12
IT	8	8
HR	8	8

3. leave types most commonly taken by each department

```
SELECT e.department, l.leave_type,
COUNT(*) AS total_leaves
FROM LeaveRecords l
JOIN Employees e ON l.employee_id = e.employee_id
GROUP BY e.department, l.leave_type
ORDER BY e.department, total_leaves DESC;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department	leave_type	total_leaves
Finance	Earned	5
Finance	Casual	3
Finance	Sick	3
HR	Casual	4
HR	Sick	3
HR	Earned	1
IT	Sick	5
IT	Earned	2
IT	Maternity	1
Marketing	Sick	4
Marketing	Casual	4
Marketing	Earned	3
Marketing	Maternity	1
Operations	Casual	5
Operations	Earned	4

4. Find department-wise average performance and salary --

```
SELECT e.department,
ROUND(AVG(p.score), 2) AS avg_performance_score,
ROUND(AVG(e.salary), 2) AS avg_salary
FROM Employee e
JOIN Performance p ON e.employee_id = p.employee_id
GROUP BY e.department
ORDER BY avg_performance_score DESC;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

department	avg_performance_score	avg_salary
Operations	3.80	51191.50
Marketing	3.76	58133.97
IT	3.70	53009.33
HR	3.70	54538.43
Finance	3.54	53985.60

5. Find employees who got high performance ratings but low salary --

```
SELECT e.employee_id,
CONCAT(e.first_name, ' ', e.last_name) AS employee_name, e.department,
ROUND(p.score, 1) AS performance_score, s.basic_pay, s.net_salary
FROM Employees e
JOIN Performance p ON e.employee_id = p.employee_id
JOIN Salary s  ON e.employee_id = s.employee_id
WHERE p.score >= 4.2AND s.net_salary < 40000 AND s.month_year = '2025-07-01'
ORDER BY p.score DESC, s.net_salary ASC;
```

OUTPUT:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_id	employee_name	department	performance_score	basic_pay	net_salary
101	Divya Desai	Operations	4.5	30000.00	35500.00
141	Amit Bose	HR	4.2	32000.00	35500.00
128	Meena Joshi	IT	4.2	33000.00	36500.00

CONCLUSION

The HR and Payroll Management System project successfully streamlines essential HR functions, such as employee data management, attendance tracking, leave processing, and automated payroll calculation. By integrating structured database design and SQL queries, the system ensures accurate, real-time data handling and enhances overall operational efficiency.

This project not only reduces manual errors and administrative workload but also enables data-driven decision-making through insightful reporting. The implementation of this system demonstrates a practical application of relational database concepts and showcases the potential for scalable, automated HR solutions in real-world business environments.