WorkForcePro – Employee Onboarding & Performance Management System

---

1. Introduction

Human Resource Management is an important part of any organization. Companies need a system to manage employees, attendance, salary, performance, and training in an organized way.

WorkForcePro is a database-driven HR system designed to handle:

- Employee onboarding
- Department and role assignment
- Attendance tracking
- Performance evaluation
- Training programs
- Payroll processing
- Employee exit management

This project applies DBMS concepts such as **ER modeling, normalization, SQL queries, views, constraints, and relationships**.

---

2. System Overview

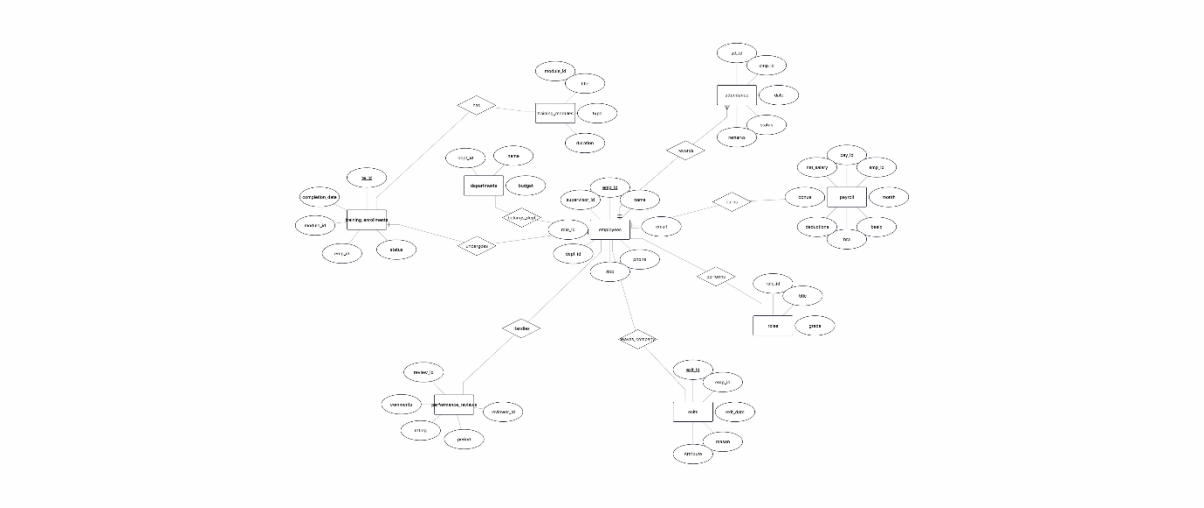
WorkForcePro stores all employee-related data in a centralized database. Each employee is linked to:

- One department
- One role
- One supervisor
- Multiple attendance records
- Multiple payroll records
- Performance reviews
- Training modules

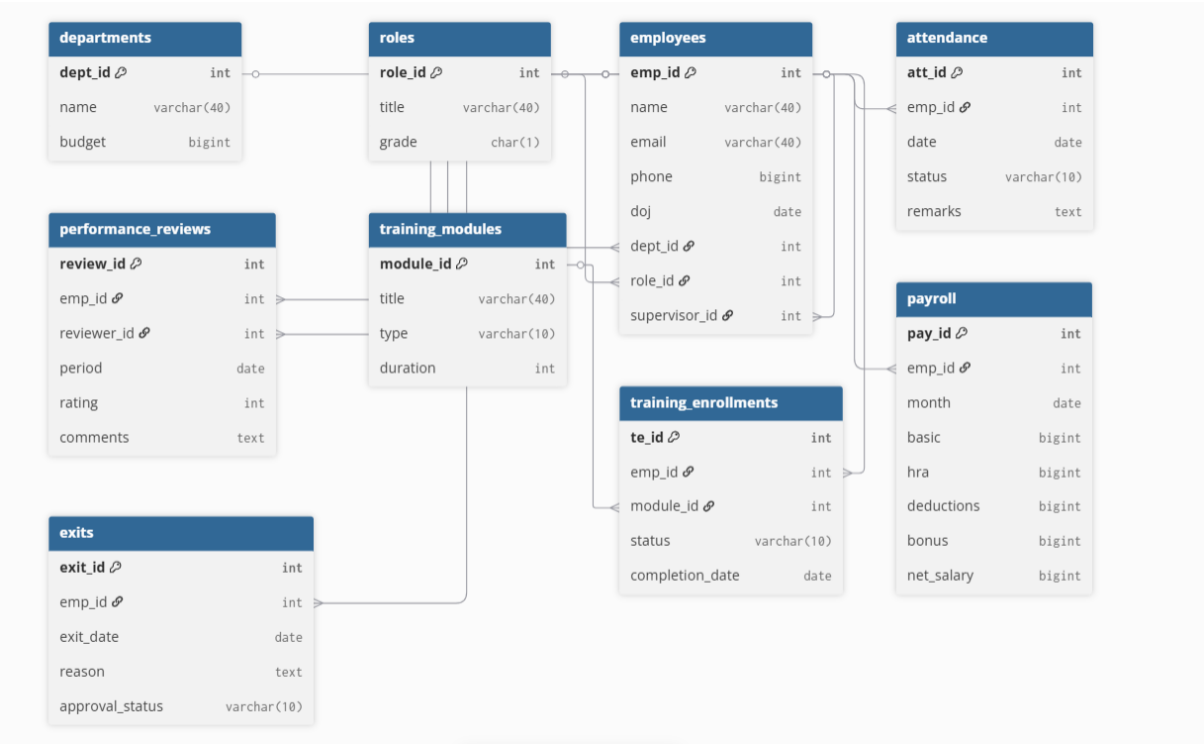
The system helps HR and management take better decisions using reports and queries.

---

3. ER Diagram



The ER diagram shows entities like Employees, Departments, Roles, Attendance, Payroll, Training, Reviews, and Exits.



This diagram represents how entities are converted into tables using primary keys and foreign keys.

---

4. Entity Description

4.0 SQL Code

```
CREATE DATABASE WorkForcePro;
use WorkForcePro;

CREATE TABLE departments(
   dept_id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(40),
   budget BIGINT
);

CREATE TABLE roles(
   role_id INT PRIMARY KEY AUTO_INCREMENT,
   title VARCHAR(40),
   grade CHAR(1)
);

CREATE TABLE employees (
   emp_id INT PRIMARY KEY AUTO_INCREMENT,
   name VARCHAR(40),
   email VARCHAR(40) UNIQUE,
   phone BIGINT,
   doj DATE,
   dept_id  INT,
   role_id INT,
```

```
    supervisor_id INT,

    CONSTRAINT fk_emp_dept FOREIGN KEY (dept_id)
    REFERENCES departments(dept_id),

    CONSTRAINT fk_emp_role FOREIGN KEY (role_id)
    REFERENCES roles(role_id),

    CONSTRAINT fk_emp_spr FOREIGN KEY (supervisor_id)
    REFERENCES employees(emp_id)
);

CREATE TABLE attendance (
    att_id INT PRIMARY KEY AUTO_INCREMENT,
    emp_id INT,
    date DATE,
    status ENUM("present","absent","leave","wfh"),
    remarks TEXT,

    CONSTRAINT fk_emp_att FOREIGN KEY (emp_id)
    REFERENCES employees(emp_id)
);

CREATE TABLE performance_reviews(
    review_id INT PRIMARY KEY AUTO_INCREMENT,
    emp_id INT,
```

```
    reviewer_id INT,
    period DATE,
    rating INT CHECK(rating BETWEEN 1 AND 5),
    comments TEXT,

    CONSTRAINT fk_emp_per FOREIGN KEY (emp_id)
    REFERENCES employees(emp_id),

    CONSTRAINT fk_emp_rev FOREIGN KEY (reviewer_id)
    REFERENCES employees(emp_id)
);

CREATE TABLE training_modules(
    module_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(40),
    type ENUM('completed','pending','failed'),
    duration INT
);

CREATE TABLE training_enrollments(
    te_id INT PRIMARY KEY AUTO_INCREMENT,
    emp_id INT,
    module_id INT,
    status ENUM("mandatory","optional"),
    completion_date DATE,
```

```
    CONSTRAINT fk_emp_twr FOREIGN KEY (emp_id)
    REFERENCES employees(emp_id),

    CONSTRAINT fk_mod_twr FOREIGN KEY (module_id)
    REFERENCES training_modules(module_id)
);

CREATE TABLE payroll(
    pay_id INT PRIMARY KEY AUTO_INCREMENT,
    emp_id INT,
    month DATE,
    basic BIGINT,
    hra BIGINT,
    deductions BIGINT,
    bonus BIGINT,
    net_salary BIGINT,

    CONSTRAINT fk_emp_pay FOREIGN KEY (emp_id)
    REFERENCES employees(emp_id)
);

CREATE TABLE exits(
    exit_id INT PRIMARY KEY AUTO_INCREMENT,
    emp_id INT,
    exit_date DATE,
    reason TEXT,
```

```
    approval_status ENUM('completed','pending','failed'),


    CONSTRAINT fk_emp_exits FOREIGN KEY (emp_id)
    REFERENCES employees(emp_id)
);
```

4.1 Employees

This is the main table.
It stores employee personal and organizational details.

Each employee belongs to a department, has a role, and may have a supervisor.

4.2 Departments

Stores department name and budget.
One department can have many employees.

4.3 Roles

Stores job title and grade.
Roles help in payroll and hierarchy.

4.4 Attendance

Attendance is recorded daily for each employee.
Status includes Present, Absent, Leave, and Work From Home.

4.5 Performance Reviews

Performance reviews are done periodically by supervisors.
Ratings range from 1 to 5.

4.6 Training Modules & Enrollments

Training modules store course details.
Training enrollments link employees with modules and show whether training is mandatory or optional.

4.7 Payroll

Payroll stores salary details month-wise.
Net salary is calculated after adding bonus and subtracting deductions.

4.8 Exits

Exit table stores details of employees leaving the organization.

---

5. Normalization

All tables are normalized up to **BCNF**.

Reasons:

- No repeating attributes
- No partial dependency
- No transitive dependency
- Each table represents a single concept

This reduces redundancy and improves data integrity.

---

6. SQL Schema Implementation

All tables were created using:

- PRIMARY KEY
- FOREIGN KEY
- UNIQUE
- CHECK
- ENUM

This ensures valid and consistent data.

---

7. SQL Queries and Explanation

Query 1: Employees who joined in 2024

This query filters employees based on joining date.

```
SELECT * FROM employees
WHERE doj BETWEEN '2024-01-01' AND '2024-12-31';
```

emp_id	name	email	phone	doj	dept_id	role_id	supervisor_id
23	Tarun Bajaj	tarun.bajaj@wfp.com	9000000023	2024-01-08	1	6	13
24	Swati Mishra	swati.mishra@wfp.com	9000000024	2024-02-14	1	6	4
25	Nikhil Bhatt	nikhil.bhatt@wfp.com	9000000025	2024-03-20	1	5	4
29	Rohit Soni	rohit.soni@wfp.com	9000000029	2024-01-15	2	8	5
30	Tanya Khanna	tanya.khanna@wfp.com	9000000030	2024-04-10	2	8	5
35	Manish Goel	manish.goel@wfp.com	9000000035	2024-01-22	3	10	6
39	Kiran Rao	kiran.rao@wfp.com	9000000039	2024-02-28	4	10	7
58	Manu Krishnan	manu.krishnan@wfp.com	9000000058	2024-01-05	10	6	12
59	Siddharth Saha	siddharth.saha@wfp.com	9000000059	2024-05-20	10	6	12
60	Jayanthi Balan	jayanthi.balan@wfp.com	9000000060	2024-06-10	10	6	12
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query 2: Managers with no performance reviews

This query finds employees who never acted as reviewers.

```
SELECT e.emp_id, e.name FROM employees e
WHERE e.emp_id NOT IN (
   SELECT DISTINCT reviewer_id
   FROM performance_reviews
);
```

emp_id	name
1	Arjun Mehta
2	Priya Sharma
9	Suresh Menon
10	Kavya Reddy
15	Karthik Balaji
16	Amit Verma
17	Ritu Ghosh
18	Sanjay Patel
19	Deepika Nanda
20	Mohit Saxena
21	Harsha Reddy
22	Lavanya Krish...
23	Tarun Bajaj
24	Swati Mishra
25	Nikhil Bhatt
26	Preethi Nair
27	Gaurav Sharma
28	Shalini Gupta
29	Rohit Soni
30	Tanya Khanna

Query 3: Pending mandatory training modules

This query finds training modules that are mandatory and still pending.

```
SELECT * FROM training_modules
WHERE type = 'pending'
AND module_id IN (
   SELECT module_id
   FROM training_enrollments
   WHERE status = 'mandatory'
);
```

module_id	title	type	duration
6	Cloud Architecture (AWS)	pending	20
10	Sales Techniques Advanced	pending	8
14	Project Management (PMP Prep)	pending	40
NULL	NULL	NULL	NULL

Query 4: Employees with more than 5 leaves in a month

This query counts leave records per employee per month.

```
SELECT emp_id, YEAR(date), MONTH(date), COUNT(*) AS leaves FROM
attendance
WHERE status = 'leave'
GROUP BY emp_id, YEAR(date), MONTH(date)
HAVING COUNT(*) > 5;
```

emp_id	YEAR(date)	MONTH(date)	leaves
2	2024	1	6
16	2024	1	6
18	2024	1	6
1	2024	2	6
19	2024	4	6
20	2024	3	6

Query 5: Departments with more than 20 employees

Used to identify large departments.

```
SELECT dept_id, COUNT(*) AS emp_count
FROM employees
GROUP BY dept_id
HAVING COUNT(*) > 20;
```

dept_id	emp_count
1	14

Query 6: Employee details with supervisor and department

This query displays employee name, department, and supervisor.

```
SELECT e.emp_id, e.name AS employee, d.name AS department, s.name AS
supervisor
FROM employees e
JOIN departments d ON e.dept_id = d.dept_id
LEFT JOIN employees s ON e.supervisor_id = s.emp_id;
```

emp_id	employee	department	supervisor
1	Arjun Mehta	Engineering	NULL
2	Priya Sharma	Engineering	NULL
4	Sneha Iyer	Engineering	Priya Sharma
13	Rahul Tiwari	Engineering	Priya Sharma
16	Amit Verma	Engineering	Sneha Iyer
17	Ritu Ghosh	Engineering	Sneha Iyer
18	Sanjay Patel	Engineering	Sneha Iyer
19	Deepika Nanda	Engineering	Sneha Iyer
20	Mohit Saxena	Engineering	Rahul Tiwari
21	Harsha Reddy	Engineering	Rahul Tiwari
22	Lavanya Krish...	Engineering	Rahul Tiwari
23	Tarun Bajaj	Engineering	Rahul Tiwari
24	Swati Mishra	Engineering	Sneha Iyer
25	Nikhil Bhatt	Engineering	Sneha Iyer
5	Vikram Nair	Human Re...	Arjun Mehta
26	Preethi Nair	Human Re...	Vikram Nair
27	Gaurav Sharma	Human Re...	Vikram Nair
28	Shalini Gupta	Human Re...	Vikram Nair
29	Rohit Soni	Human Re...	Vikram Nair
30	Tanya Khanna	Human Re...	Vikram Nair

Query 7: Departments with more than 5 reviews

Shows departments actively participating in appraisals.

```
SELECT e.dept_id, COUNT(*) AS review_count
FROM employees e
JOIN performance_reviews p ON e.emp_id = p.emp_id
GROUP BY e.dept_id
HAVING COUNT(*) > 5;
```

	dept_id	review_count
▶	1	40
	2	16
	3	12
	4	8
	6	16
	5	8
	9	8
	10	16

Query 8: Managers handling more than one employee

Identifies supervisors with teams.

```
SELECT supervisor_id, COUNT(*) AS team_size FROM employees
WHERE supervisor_id IS NOT NULL
GROUP BY supervisor_id
HAVING COUNT(*) > 1;
```

supervisor_id	team_size
1	3
2	5
3	5
4	6
5	5
6	5
7	4
8	3
11	4
12	7
13	4
14	3
15	3

Query 9: Employees with deductions above average

Used to find employees with high deductions.

```
SELECT DISTINCT e.emp_id, e.name
FROM employees e
JOIN payroll p ON e.emp_id = p.emp_id
WHERE p.deductions > (
   SELECT AVG(deductions) FROM payroll
);
```

emp_id	name
1	Arjun Mehta
2	Priya Sharma
3	Ravi Kumar
4	Sneha Iyer
5	Vikram Nair
6	Divya Pillai
7	Aditya Rao
8	Meera Das
9	Suresh Menon
10	Kavya Reddy
11	Nitin Joshi
12	Pooja Bhat
13	Rahul Tiwari
14	Ananya Singh
15	Karthik Balaji
17	Ritu Ghosh
21	Harsha Reddy
25	Nikhil Bhatt

Query 10: Employees with zero training enrollments

Finds employees who have not enrolled in any training.

```
SELECT e.emp_id, e.name
FROM employees e
LEFT JOIN training_enrollments te ON e.emp_id = te.emp_id
WHERE te.emp_id IS NULL;
```

emp_id	name
13	Rahul Tiwari
14	Ananya Singh
15	Karthik Balaji
35	Manish Goel
39	Kiran Rao
44	Ishaan Malhotra
49	Nandini Rao

Query 11: Top 10 performers

Ranks employees based on rating.

```
SELECT e.name, pr.rating
FROM performance_reviews pr
JOIN employees e ON pr.emp_id = e.emp_id
ORDER BY pr.rating DESC
LIMIT 10;
```

	name	rating
▶	Harsha Reddy	5
	Puja Sen	5
	Akash Tomar	5
	Ritu Ghosh	5
	Mohit Saxena	5
	Harsha Reddy	5
	Tarun Bajaj	5
	Nikhil Bhatt	5
	Puja Sen	5
	Ankit Desai	5

Query 12: Department-wise payroll expenditure

Calculates total salary spent per department.

```
SELECT d.name, SUM(p.net_salary) AS expenditure
FROM payroll p
JOIN employees e ON p.emp_id = e.emp_id
JOIN departments d ON e.dept_id = d.dept_id
GROUP BY d.name;
```

	name	expenditure
▶	Engineering	27624300
	Human Resources	3582600
	Finance	3907250
	Marketing	3657400
	Operations	7133850
	Sales	7272400
	Legal	2353000
	Product	2820000
	Customer Support	3087550
	Research & Dev	5188750

Query 13: Employee with highest attendance

Finds the most regular employee.

```
SELECT emp_id
FROM attendance
WHERE status != 'leave'
GROUP BY emp_id
ORDER BY COUNT(*) DESC
LIMIT 1;
```

emp_id
1

Query 14: Quarterly performance trend

Shows performance trend per quarter.

```
SELECT QUARTER(period),
AVG(rating), MAX(rating), MIN(rating)
FROM performance_reviews
GROUP BY QUARTER(period);
```

QUARTER(period)	AVG(rating)	MAX(rating)	MIN(rating)
2	3.6452	5	1
4	4.0484	5	1

Query 15: Training modules with no enrollments

Used for skill gap analysis.

```
SELECT tm.module_id
FROM training_modules tm
LEFT JOIN training_enrollments te
ON tm.module_id = te.module_id
WHERE te.module_id IS NULL;
```

module_id
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Query 16: Monthly salary report

Shows employee count and salary payout per month.

```
SELECT MONTH(month), COUNT(emp_id), SUM(net_salary)
FROM payroll
GROUP BY MONTH(month);
```

MONTH(month)	COUNT(emp_id)	SUM(net_salary)
1	60	7654700
2	25	4674200
3	25	4674200
4	25	4674200
5	25	4890700
6	60	7758900
7	25	4674200
8	25	4674200
9	25	4890700
10	25	4674200
11	25	4674200
12	60	8712700

Query 17: Employees earning above department average

Used to compare salaries within departments.

```
SELECT e.emp_id, e.name
FROM employees e
```

```
JOIN payroll p ON e.emp_id = p.emp_id
WHERE p.net_salary >
(
   SELECT AVG(p2.net_salary)
   FROM payroll p2
   JOIN employees e2 ON p2.emp_id = e2.emp_id
   WHERE e2.dept_id = e.dept_id
);
```

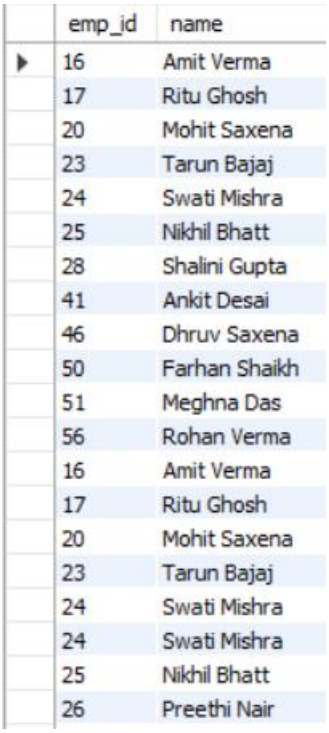
emp_id	name
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
1	Arjun Mehta
2	Priya Sharma
2	Priya Sharma
2	Priya Sharma
2	Priya Sharma
2	Priya Sharma
2	Priya Sharma
2	Priya Sharma
2	Priya Sharma

Query 18: Employees whose performance improved

Detects employees with better ratings over time.

```
SELECT curr.emp_id, e.name
FROM performance_reviews curr
JOIN performance_reviews prev
ON curr.emp_id = prev.emp_id
AND curr.period > prev.period
```

JOIN employees e ON e.emp_id = curr.emp_id
WHERE curr.rating > prev.rating;

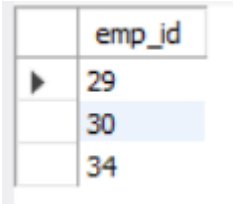


emp_id	name
16	Amit Verma
17	Ritu Ghosh
20	Mohit Saxena
23	Tarun Bajaj
24	Swati Mishra
25	Nikhil Bhatt
28	Shalini Gupta
41	Ankit Desai
46	Dhruv Saxena
50	Farhan Shaikh
51	Meghna Das
56	Rohan Verma
16	Amit Verma
17	Ritu Ghosh
20	Mohit Saxena
23	Tarun Bajaj
24	Swati Mishra
24	Swati Mishra
25	Nikhil Bhatt
26	Preethi Nair

Query 19: Employees linked only to optional training

Ensures no mandatory training is assigned.

SELECT emp_id
FROM training_enrollments
GROUP BY emp_id
HAVING SUM(status = 'mandatory') = 0;



emp_id
29
30
34

Query 20: Employee Dashboard View

Creates a summarized employee report.

CREATE OR REPLACE VIEW emp_dashboard AS
SELECT e.emp_id, e.name, d.name AS department, r.title,

```
ROUND(att.present_days * 100 / att.total_days, 2) AS attendance_pct,
ROUND(pr.avg_rating, 2) AS avg_rating,
p.net_salary
FROM employees e
JOIN departments d ON e.dept_id = d.dept_id
JOIN roles r ON e.role_id = r.role_id
LEFT JOIN (
   SELECT emp_id, COUNT(*) total_days,
   SUM(status IN ('present','wfh')) present_days
   FROM attendance GROUP BY emp_id
) att ON e.emp_id = att.emp_id
LEFT JOIN (
   SELECT emp_id, AVG(rating) avg_rating
   FROM performance_reviews GROUP BY emp_id
) pr ON e.emp_id = pr.emp_id
LEFT JOIN payroll p ON e.emp_id = p.emp_id;
```

```
CREATE OR REPLACE VIEW emp_dashboard AS
SELECT e.emp_id, e.name, d.name AS department, r.title,
ROUND(att.present_days * 100 / att.total_days, 2) AS attendance_pct,
ROUND(pr.avg_rating, 2) AS avg_rating,
p.net_salary
FROM employees e
JOIN departments d ON e.dept_id = d.dept_id
JOIN roles r ON e.role_id = r.role_id
LEFT JOIN (
   SELECT emp_id, COUNT(*) total_days,
   SUM(status IN ('present','wfh')) present_days
   FROM attendance GROUP BY emp_id
) att ON e.emp_id = att.emp_id
LEFT JOIN (
   SELECT emp_id, AVG(rating) avg_rating
   FROM performance_reviews GROUP BY emp_id
) pr ON e.emp_id = pr.emp_id
LEFT JOIN payroll p ON e.emp_id = p.emp_id;
```

emp_id	name	department	title	attendance_pct	avg_rating	net_salary
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	395000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	395000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	370000
1	Arjun Mehta	Engineering	CEO	86.46	NULL	420000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000
2	Priya Sharma	Engineering	CTO	75.00	NULL	354000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000
2	Priya Sharma	Engineering	CTO	75.00	NULL	339000

---

8. Conclusion

The WorkForcePro system successfully demonstrates:

- ER modeling
- Normalization
- SQL schema design
- Advanced queries
- Views and reporting

This project shows how DBMS concepts can be applied to a real-world HR system.