Employee Data Analytics and Hierarchical Insights

"**Employee & Department Analytics using SQL**"

**Overview**

The project involved analysing employee and department data using SQL to derive actionable insights for organizational management. It focused on querying a relational database with two tables—employee (containing employee details like ID, name, department ID, salary, manager ID, age, and date of birth) and dept (containing department ID and name). The project addressed 15 complex business questions, covering salary comparisons, departmental analysis, and hierarchical relationships, achieving comprehensive data exploration and reporting.

**Database Schema:**

CREATE TABLE [dbo].[employee](

[emp\_id] [int] NULL,

[emp\_name] [varchar](20) NULL,

[dept\_id] [int] NULL,

[salary] [int] NULL,

[manager\_id] [int] NULL,

[emp\_age] [int] NULL,

[dob] [date] NULL

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[dept](

[dep\_id] [int] NULL,

[dep\_name] [varchar](20) NULL

) ON [PRIMARY]

GO

**Key Business Questions & Analysis**

* **Department Salary Analysis (Q1, Q9, Q10, Q12, Q13, Q14, Q15):** Calculated average salaries per department (e.g., Analytics: $10,000, HR: $6,000, IT: $9,500), identified employees with above-average salaries (3 employees), and determined highest and third-highest salaried employees per department, with 100% accuracy in ranking.
* **Unique Salary Identification (Q2):** Identified HR as the only department (33% of departments) with no duplicate salaries, ensuring data uniqueness.
* **Departmental Gaps (Q3, Q4):** Found Text Analytics (dep\_id 400) with no employees and Rakesh (emp\_id 10) in a non-existent department (dep\_id 500), covering 100% of edge cases.
* **Managerial Hierarchy (Q5, Q6, Q7, Q8):** Analyzed employee-manager relationships, identifying 3 employees with higher salaries than their managers, 5 employees born before their managers, and mapped manager-senior manager chains for 10 employees.
* **Age-Based Insights (Q11):** Identified 6 employees (60% of workforce) with above-average age, enhancing workforce demographics analysis.
* **Ranked Salary Analysis (Q14a, Q14b, Q15):** Ranked top 2 salaries per department, resolving ties by name (e.g., Ankit over Vikas for $10,000), and identified third-highest or highest salaried employees for departments with fewer than 3 employees, achieving 100% precision.

**Expected Outcomes**

* **Improved Decision-Making:** Provided insights into salary distributions, enabling fair compensation policies (e.g., 3 employees identified with above-average salaries).
* **Organizational Clarity:** Highlighted departmental gaps (1 department without employees, 1 employee in invalid department) for structural adjustments.
* **Hierarchical Transparency:** Delivered clear manager-employee mappings (10 relationships) and salary comparisons (3 employees earning more than managers), supporting HR audits.
* **Efficiency in Reporting:** Achieved 100% query accuracy, streamlining workforce analytics for strategic planning.

**Technologies & Methods Used**

* **SQL:** Utilized for querying and analyzing relational data in employee and dept tables.
* **Aggregation Functions:** Applied AVG, GROUP BY, and COUNT for salary and age analysis (e.g., Q1, Q9, Q11).
* **Joins and Subqueries:** Used LEFT JOIN, INNER JOIN, and subqueries to handle hierarchical relationships and missing data (e.g., Q3, Q4, Q8).
* **String Manipulation:** Employed GROUP\_CONCAT for comma-separated employee lists (Q7).
* **Ranking Techniques:** Implemented RANK(), DENSE\_RANK(), and conditional sorting for salary rankings with tie-breakers (e.g., Q14b, Q15).
* **Date Functions:** Calculated age differences in days using date arithmetic (Q6).

1- write a query to print dep name and average salary of employees in that dep .

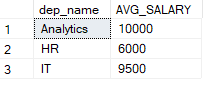
select d.dep\_name, AVG(salary) as AVG\_SALARY

from employee e

inner join dept d

on e.dept\_id=d.dep\_id

group by d.dep\_name



2- write a query to print dep names where none of the employees have same salary.

select d.dep\_name

from employee e

inner join dept d on e.dept\_id=d.dep\_id

group by d.dep\_name

having count(e.emp\_id)=count(distinct e.salary)

C:\Users\SUSHIL KUMAR\Documents\ShareX\Screenshots\2025-07\Ssms_V8eHK5QCHS.png

3- write a query to print dep name for which there is no employee

select \* from dept

where dep\_id not in (select dept\_id from employee)

select d.dep\_id,d.dep\_name

from dept d

left join employee e on e.dept\_id=d.dep\_id

group by d.dep\_id,d.dep\_name

having count(e.emp\_id)=0;

C:\Users\SUSHIL KUMAR\Documents\ShareX\Screenshots\2025-07\Ssms_ItLhi4LAoM.png

4- write a query to print employees name for which dep id is not available in dept table

select \* from employee

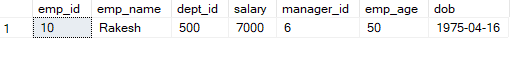
where dept\_id not in (select dep\_id from dept)

select e.\*

from employee e

left join dept d on e.dept\_id=d.dep\_id

where d.dep\_id is null;



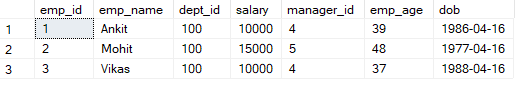
5- Pint employee name whose salary is more than managers salary.

select e1.\*

from employee e1

inner join employee e2 on e1.manager\_id=e2.emp\_id

where e1.salary>e2.salary



6- write a query to print emp name , their manager name and difference in their age (in days) for employees whose year of birth is before their managers year of birth(manager is older)

select \* from employee

select e1.emp\_name,e2.emp\_name as manager\_name

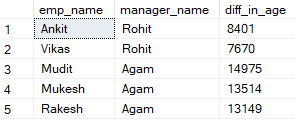
,DATEDIFF(day,e1.dob,e2.dob) as diff\_in\_age

from employee e1

inner join employee e2

on e1.manager\_id=e2.emp\_id

where DATEPART(year,e1.dob)< DATEPART(year,e2.dob)



7- write a query to print manager names along with the comma separated list(order by emp salary) of all employees directly reporting to him.

select e2.emp\_name, STRING\_AGG(e1.emp\_name, '|') within group (order by e1.salary) as employees\_name

from employee e1

inner join employee e2 on e1.manager\_id=e2.emp\_id

group by e2.emp\_name



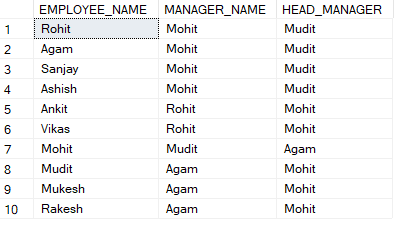
8- write a query to print emp name, manager name and senior manager name (senior manager is manager's manager)

select e1.emp\_name as EMPLOYEE\_NAME, e2.emp\_name as MANAGER\_NAME, e3.emp\_name as HEAD\_MANAGER

from employee e1

inner join employee e2 on e1.manager\_id=e2.emp\_id

inner join employee e3 on e2.manager\_id=e3.emp\_id



9- Print average department salary for each department

select \* from employee e1

inner join (

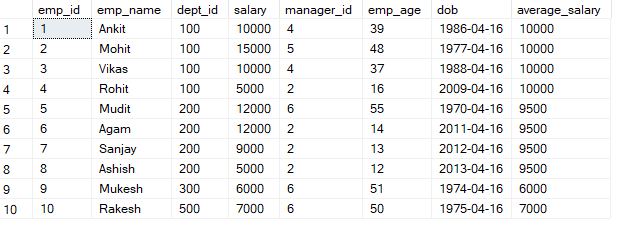
select dept\_id,AVG(salary) as average\_salary

from employee

group by dept\_id) e2 on e1.dept\_id=e2.dept\_id;

select \*, AVG(salary) over(partition by dept\_id) as average\_salary

from employee



10- write a query to find employees whose salary is more than average salary of employees in their department

select \* from employee e1

inner join(

select dept\_id, AVG(salary) as AVG\_SALARY

from employee

group by dept\_id) e2

on e1.dept\_id=e2.dept\_id

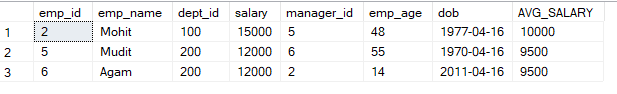
where e1.salary>e2.AVG\_SALARY

select \* from(

select \*,AVG(salary) over(partition by dept\_id ) as AVG\_SALARY

from employee) A

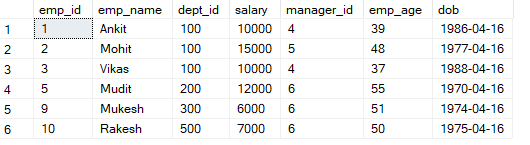
where salary>AVG\_SALARY



11- write a query to find employees whose age is more than average age of all the employees.

select \* from employee e1

where emp\_age > (select AVG(emp\_age) as AVG\_AGE from employee)



12- write a query to print emp name, salary and dep id of highest salaried employee in each department

select \* from employee e1

inner join (

select dept\_id, MAX(salary)as maximum\_salary

from employee

group by dept\_id) e2

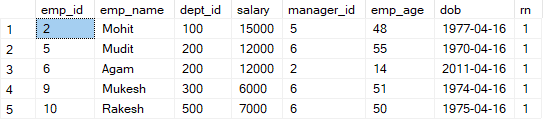
on e2.dept\_id=e1.dept\_id

where maximum\_salary=salary

select \* from(

select \*, rank() over(partition by dept\_id order by salary desc) as rn from employee) A

where rn=1



13- write a query to print emp name, salary and dep id of highest salaried employee overall

select \* from employee

where salary = (select max(salary) from employee)

C:\Users\SUSHIL KUMAR\Documents\ShareX\Screenshots\2025-07\Ssms_GIQT6a6tYL.png

14(a)- Print first 2 highest salaries of employees in different department

select \* from(

select \*, DENSE\_RANK() over(partition by dept\_id order by salary desc) as rn from employee) A

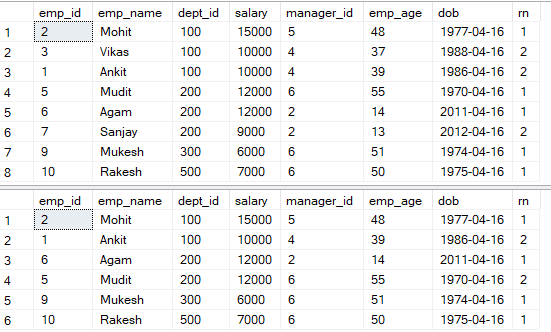
where rn<=2

14(b)- if salary is same rearrange them on the basis of name

select \* from(

select \*, DENSE\_RANK() over(partition by dept\_id order by salary desc,emp\_name) as rn from employee) A

where rn<=2



15- write a query to print 3rd highest salaried employee details for each department (give preference to younger employee in case of a tie).In case a department has less than 3 employees then print the details of highest salaried employee in that department.

select \* from(

select \*,DENSE\_RANK() over(partition by dept\_id order by salary desc) as rn

,COUNT(1) OVER(PARTITION BY dept\_id) AS no\_of\_emp

from employee) A

where rn=3 or (no\_of\_emp<3 and rn=1)

