**Mini Project**

**Name:** Priya Kesarwani

**Email Id:** priyapk1808@gmail.com

**Batch** **Date:** 06 May, 2022

# SQL

**Data used:** <https://dev.mysql.com/doc/employee/en/employees-validation.html>

**Data used:** https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms

**Question 1.** Find the average salary of the male and female employees in each department.

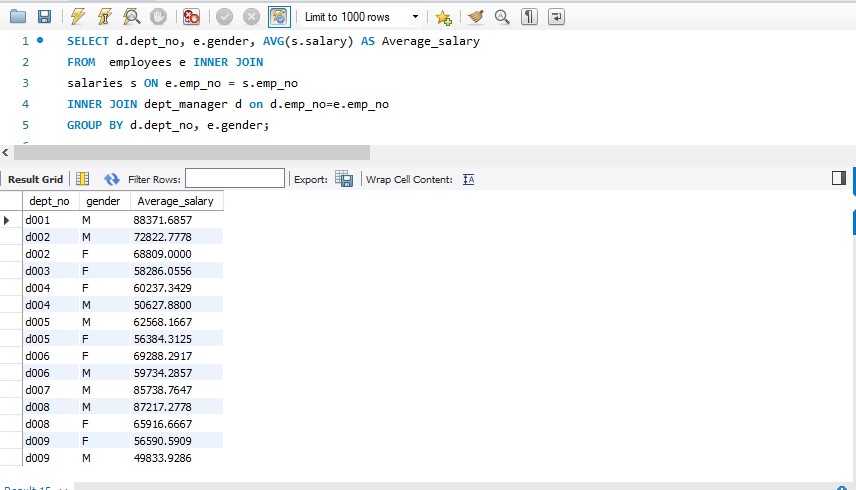
Solution:

* Table used (employees, dept\_manager, salaries)
* select d.dept\_no, e.gender, AVG(s.salary) AS Average\_salary from employees e INNER JOIN salaries s on e.emp\_no = s.emp\_no

INNER JOIN dept\_manager d on d.emp\_no=e.emp\_no

GROUP BY d.dept\_no, e.gender;

Output:

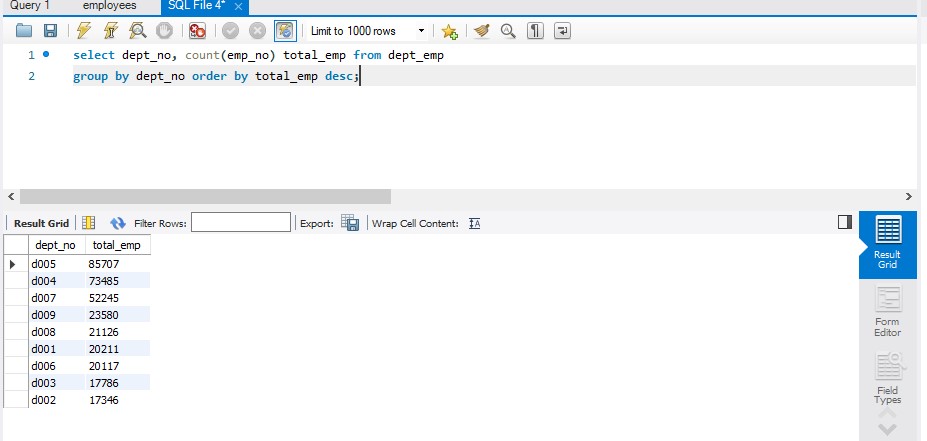


**Question 2.** Find the total employees encountered in the 'dept\_emp' table.

Solution:

* Table used (dept\_emp )
* select dept\_no, count(emp\_no) total\_emp from dept\_emp group by dept\_no order by total\_emp desc;

Output:

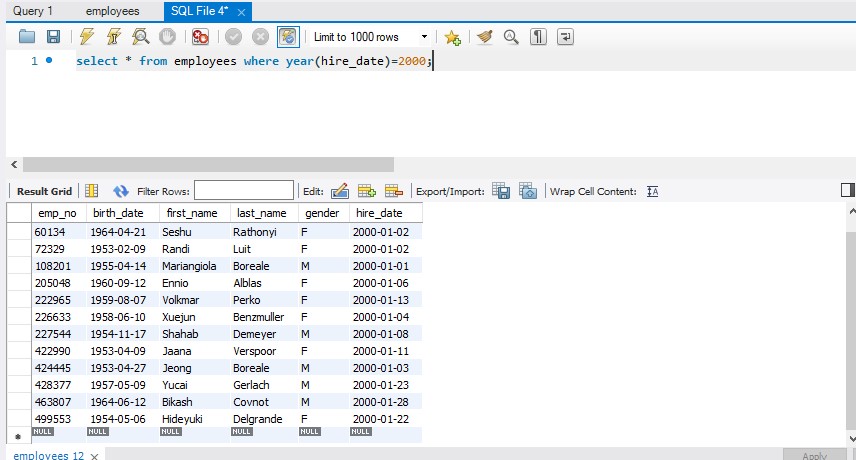


**Question 3.** Retrieve a list of all employees that have been hired in 2000.

Solution:

* Table used (employees )
* select \* from employees where year(hire\_date)=2000;

Output:

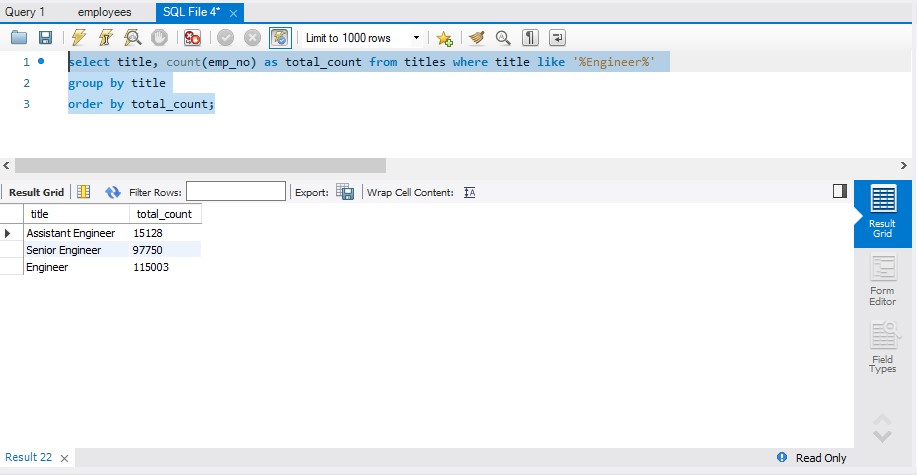


**Question 4.** Retrieve a distinct count of all employees from the ‘titles’ table who are engineers.

Solution:

* Table used (titles)
* Select title, count(emp\_no) as total\_count from titles where title like '%Engineer%' group by title order by total\_count;

Output:



**Question 5.** How many contracts have been registered in the 'salaries' table with duration of more than one year and of value higher than or equal to $100,000?

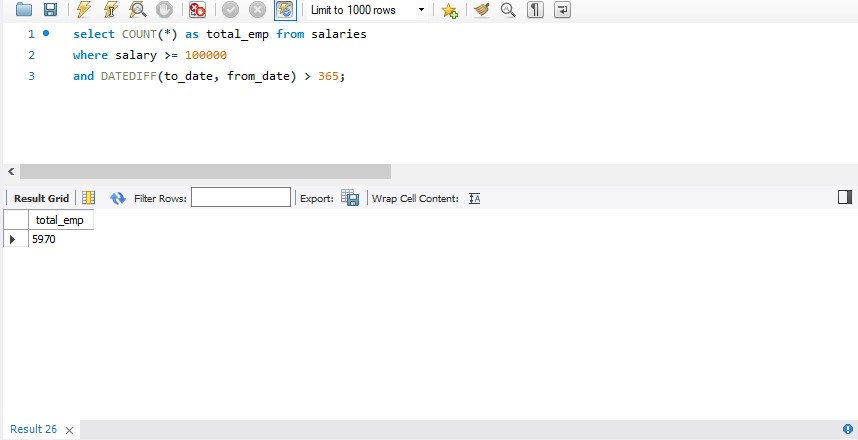
Solution:

* Table used (salaries)
* select count(\*) as total\_emp from salaries

where salary >= 100000 and DATEDIFF(to\_date, from\_date) > 365;

Output

:



**Question 6.** Obtain a table containing the following three fields for all individuals whose employee number is not greater than 10040

* Employee number
* Lowest department number among the departments where the employee has worked
* Assign '110022' as 'manager' to all individuals whose employee number is lower than or equal to 10020, and '110039' to those whose number is between 10021 and 10040 inclusive.

Solution:

* Table used (dept\_emp) and newly created table named emp\_manager
* create table emp\_manager

select emp\_no,

(select min(dept\_no) from dept\_emp d where e.emp\_no = d.emp\_no) dept\_no,

CASE

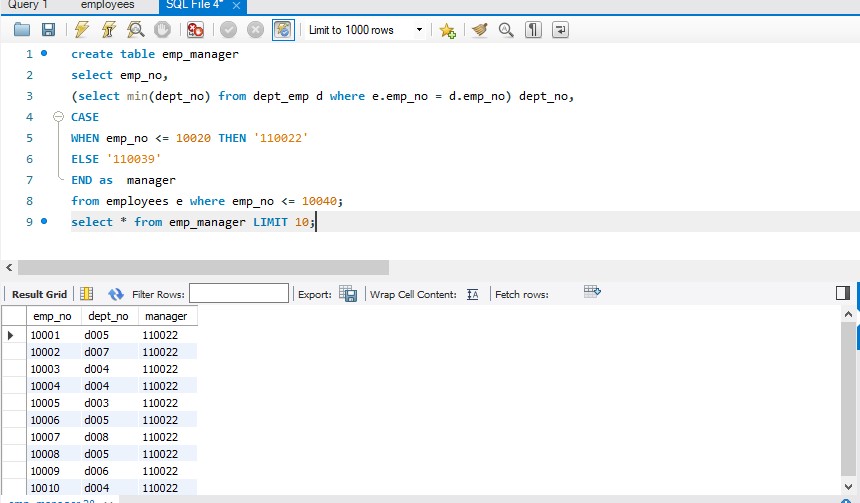
WHEN emp\_no <= 10020 THEN '110022'

ELSE '110039' END as manager from employees e where emp\_no <= 10040;

• select \* from emp\_manager LIMIT 10;

Output

:



**Question 7.** Write an SQL query to list the customers who will be getting 10% discount. Customers eligible for 10% discount are those who have done shopping for three consecutive months and transaction amount of successive months must be greater than the previous month.

Solution:

with test1 as (

select custname,txnmonth,txnamount,MONTH('1' + [txnmonth] + '00' )as rn

from customer)

select a.custnam from test1 a join test1 b

on a.custname = b.custname and a.rn+1 = b.rn and b.txnamount>a.txnamount

join test1 c

on c.custname = a.custname and a.rn+2 = c.rn and c.txnamount>b.txnamount;

**Question 8.** What should be the output of below query:

select distinct Deptname from Dept where upper (Deptname) = 'hr'  
a) HR b) Hr c) hr d) No Data Found

Solution: No data found but SQL server is case sensitive so it will return HR in output

**Question 9.** How many items with ListPrice more than $1000 have been sold?

Solution: SELECT COUNT(salesorderid) Total FROM Sales.SalesOrderDetail s JOIN Production.Product p ON s.productid = p.productid WHERE listprice > 1000;

**Question 10.** Give the CompanyName of those customers with orders over $100000.

Solution: SELECT sh.SalesOrderID

FROM Sales.Customer c JOIN Sales.SalesOrderHeader sh ON c.customerid = sh.customerid

WHERE subtotal + taxamt + freight > 100000;

**Question 11.** Show the SalesOrderID and the UnitPrice for every Single Item Order.

Solution: WITH cte AS (

SELECT salesorderid, SUM(orderqty) as items

FROM Sales.SalesOrderDetail

GROUP BY salesorderid

HAVING SUM(orderqty)=1

)

SELECT salesorderid, unitprice

FROM Sales.SalesOrderDetail

WHERE salesorderid IN (SELECT salesorderid FROM cte);

**Question 12.** List the product name and the CompanyName for all Customers who ordered ProductModel 'Racing Socks'.

Solution: SELECT p.Name FROM Sales.Customer c JOIN Sales.SalesOrderHeader sh on c.CustomerID=sh.CustomerID JOIN Sales.SalesOrderDetail sd

ON sh.salesorderid = sd.salesorderid JOIN Production.Product p ON sd.ProductID=p.ProductID JOIN Production.ProductModel pm

ON p.ProductModelID=pm.ProductModelID

WHERE pm.Name='Racing Socks';

**Question 13.** Show the product description for culture 'fr' for product with ProductID 736.

Solution: SELECT description FROM Production.Product p JOIN Production.ProductModel pm

ON p.productmodelid = pm.productmodelid

JOIN Production.ProductModelProductDescriptionCulture pmpdc ON pm.productmodelid = pmpdc.productmodelid JOIN Production.ProductDescription pd ON pmpdc.productdescriptionid = pd.productdescriptionid WHERE (productid = 736) AND (CultureID = 'fr');

**Question 14.** How many products in ProductCategory ‘Accessories’ have been sold to an address in 'London'?

Solution: SELECT SUM(orderqty) total

FROM Person.Address a JOIN Sales.SalesOrderHeader sh ON a.addressid = sh.billtoaddressid

JOIN Sales.SalesOrderDetail sd ON sh.salesorderid = sd.salesorderid

JOIN Production.Product p ON sd.productid = p.productid

JOIN Production.ProductCategory pc ON pc.ProductCategoryID = pc.productcategoryid

WHERE (city = 'London') AND (pc.name = 'Accessories');

**Question 15.** For each order show the SalesOrderID and SubTotal calculated three ways:

A) From the SalesOrderHeade B) Sum of OrderQty\*UnitPrice

C) Sum of OrderQty\*ListPrice

Solution: WITH tempA AS (

SELECT salesorderid, subtotal A\_total

FROM Sales.SalesOrderHeader

), tempB AS (

SELECT salesorderid, SUM(orderqty \* unitprice) B\_total

FROM Sales.SalesOrderDetail

GROUP BY salesorderid

), tempC AS (

SELECT salesorderid, SUM(orderqty \* listprice) C\_total

FROM Sales.SalesOrderDetail sd JOIN Production.Product p ON sd.productid = p.productid

GROUP BY salesorderid

)

SELECT tempA.salesorderid, A\_total, B\_total, C\_total

FROM tempA JOIN tempB ON tempA.salesorderid = tempB.salesorderid

JOIN tempC ON tempB.salesorderid = tempC.salesorderid;

**Question 16.** Show the bestselling item by value.

Solution: SELECT Top 1 name, SUM(orderqty \* unitprice) total\_value

FROM SALES.SalesOrderDetail sd JOIN Production.Product p ON sd.productid = p.productid

GROUP BY name

ORDER BY total\_value DESC;

**Question 17.** Show the total order value for each CountryRegion. List by value with the highest first.

Solution: SELECT countyregion, SUM(subtotal) as total

FROM Person.Address a JOIN Sales.SalesOrderHeader sh ON a.addressid = sh.shiptoaddressid

GROUP BY countyregion order by total desc;

**Question 18.** Show OrdeQty, the Name and the ListPrice of the order made by CustomerID 16518

Solution: SELECT OrderQty,Name,ListPrice

FROM Sales.SalesOrderHeader JOIN Sales.SalesOrderDetail

ON SalesOrderDetail.SalesOrderID = SalesOrderHeader.SalesOrderID

JOIN Production.Product

ON SalesOrderDetail.ProductID=Product.ProductID

WHERE CustomerID=16518;

**Question 19.** Find the best customer in each region.

Solution: WITH temp1 AS (

SELECT countyregion, companyname, SUM(subtotal) total,

RANK() OVER (PARTITION BY countyregion ORDER BY total DESC) rnk

FROM Person.Address a JOIN Sales.SalesOrderHeader sh ON a.addressid = sh.shiptoaddressid

JOIN Sales.Customer c ON sh.customerid = c.customerid

GROUP BY countyregion, companyname

)

SELECT countyregion, companyname, total

FROM temp1

WHERE rnk = 1;

**Question 20.** List the SalesOrderNumber for the customers 'Good Toys' and 'Bike World'.

Solution: SELECT companyname, salesorderid

FROM Sales.Customer c LEFT JOIN Sales.SalesOrderHeader sh ON c.customerid = sh.customerid

WHERE companyname LIKE '%Good Toys%' OR companyname LIKE '%Bike World%';

**Question 21.** Pivot the Occupation column in OCCUPATIONS so that each Name is sorted alphabetically and displayed underneath its corresponding Occupation. The output column headers should be Doctor, Professor, Singer, and Actor, respectively.

Note: Print NULL when there are no more names corresponding to an occupation.

Solution: Select

max(case when temp.Occupation = "Doctor" then temp.Name end) as NAME,

max(case when temp.Occupation = "Professor" then temp.Name end)as NAME,

max(case when temp.Occupation = "Singer" then temp.Name end) as NAME,

max(case when temp.Occupation = "Actor" then temp.Name end) as NAME

FROM

(select \*, row\_number() over (partition by Occupation order by Name) row\_num

from OCCUPATIONS) temp group by row\_num;

**Question 22.** Delete duplicate data from cars table. Duplicate record is identified based on the model and brand name.

Solution1: Using SELF join

delete from cars where id in (

select c2.id from cars c1

join cars c2 on c1.model = c2.model and c1.brand = c2.brand

where c1.id < c2.id);

Solution2: Using Window function

delete from cars where id in (

Select id from (select id, brand, model, row\_number() over(partition by model, brand order by id) as rn from cars) a where a.rn > 1);

**Question 23.** Delete duplicate data from cars table. Duplicate record is identified based on the all of the columns.

Solution1: Creating a backup table without dropping the original table.

create table cars\_bkp as

select distinct \* from cars;

truncate table cars;

insert into cars

select distinct \* from cars\_bkp;

drop table cars\_bkp;

**Question 24.** Write query for those students who have scored more than average marks in each subject.

Solution: with cte as (

select subject, AVG(marks) as avg\_marks from students group by subject)

select s.studentid, s.studentname,s.marks from students s JOIN cte c on s.subject=c.subject where s.marks>c.avg\_marks;

**Question 25.** Write query for second highest marks and second lowest marks in each subject.

Solution: select subject,

sum(case when rnk2=2 then marks else NULL end) as second\_highest\_marks,

sum(case when rnk1=2 then marks else NULL end) as second\_lowest\_marks

from (

select subject, marks,

rank() over(partition by subject order by marks asc) as rnk1,

rank() over(partition by subject order by marks desc) as rnk2

from students) A

group by subject;

**Question 26.** Why rank skips the sequence in SQL?

Solution: When there are duplicate values same ranking is assigned, and a gap appears in the sequence for each duplicate ranking in database.

**Question 27.** Why column name alias can't be used in WHERE CLAUSE but can be used in ORDER BY CLAUSE?

Solution: WHERE CLAUSE is a filtered condition which is applied on database columns which filters only to column with actual name.

Where as ORDER BY CLAUSE will sort the data based on the specifications of the SELECT STATEMENT.

**Question 28.** In which scenario IN operator fails?

Solution: The IN operator fails because there is a limited number of inputs and it won't handle null values. To overcome this issue, we can use exists operator in SQL.

**Question 29.** Can we use aggregate functions without Group by clause?

Solution: Yes, we can use aggregate functions without Group by clause.

Ex: Select count(emp\_no), max(salary), min(salary), avg(salary) from salaries;

**Question 30.** Can we use Group by clause without aggregate functions?

Solution: Yes, We can use Group by clause without aggregate functions

Ex: Select emp\_no, salary from salaries group by emp\_no, salary LIMIT 50;