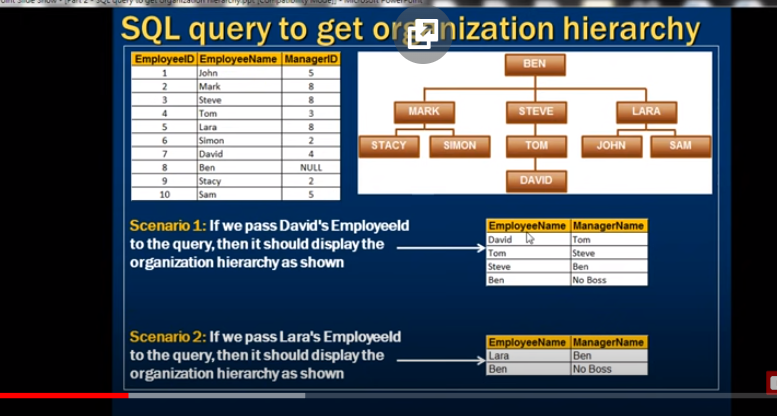
**Sql part2**



create table tblEmployee

(

id int identity ,

name varchar(20),

gender varchar(20),

salary int,

city varchar(20)

)

insert into tblEmployee values ('tom','male',4000,'london')

insert into tblEmployee values ('pam','female',3000,'newyork')

insert into tblEmployee values ('john','male',3500,'london')

insert into tblEmployee values ('sam','male',4500,'london')

insert into tblEmployee values ('tod','male',2800,'sydney')

insert into tblEmployee values ('ben','male',7000,'newyork')

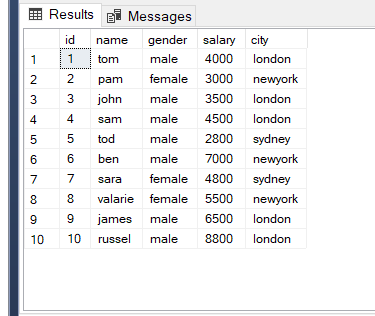
insert into tblEmployee values ('sara','female',4800,'sydney')

insert into tblEmployee values ('valarie','female',5500,'newyork')

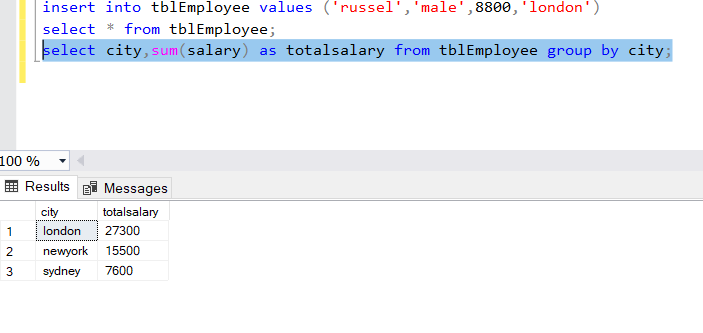
insert into tblEmployee values ('james','male',6500,'london')

insert into tblEmployee values ('russel','male',8800,'london')

select \* from tblEmployee;

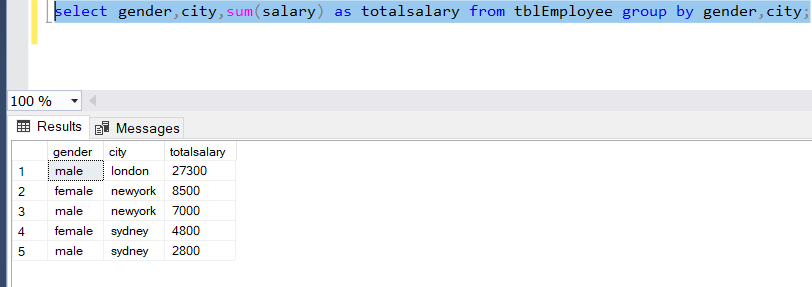


**Case1:I want an sql query, which gives total salaries paid by City**

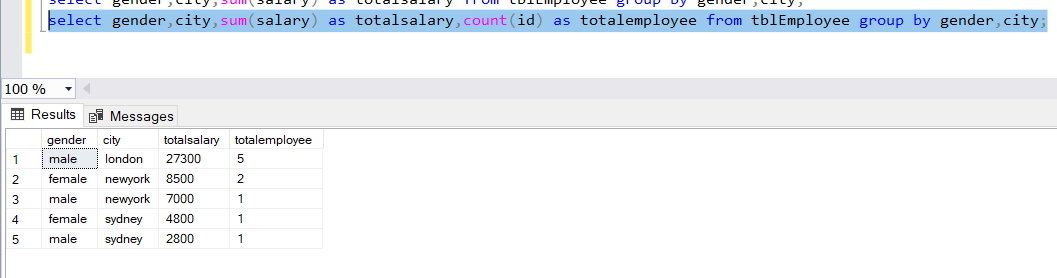


**Case2:**

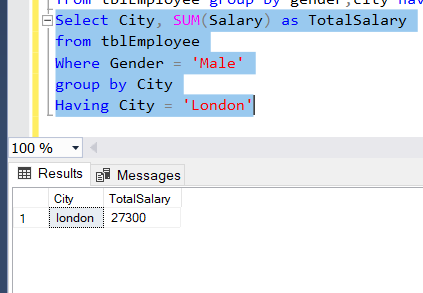
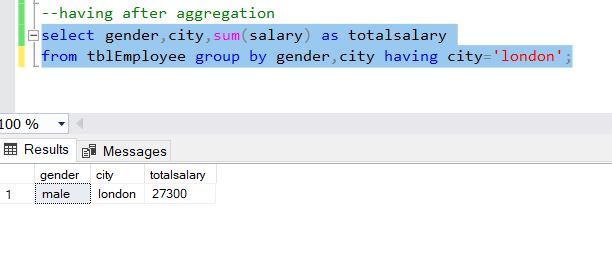
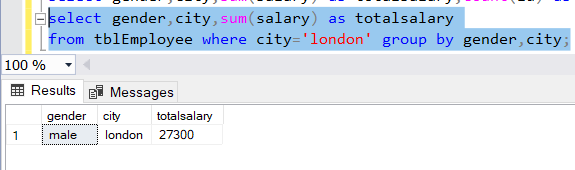
**Now, I want an sql query, which gives total salaries by City, by gender.**



**Case 3:Now, I want an sql query, which gives total salaries and total number of employees by City, and by gender.**



**Filtering Groups:**  
WHERE clause is used to filter rows before aggregation, where as HAVING clause is used to filter groups after aggregations. The following 2 queries produce the same result.



**Difference between WHERE and HAVING clause:**

1. WHERE clause can be used with - Select, Insert, and Update statements, where as HAVING clause can only be used with the Select statement.

2. WHERE filters rows before aggregation (GROUPING), where as, HAVING filters groups, after the aggregations are performed.

Aggregate functions cannot be used in the WHERE clause, unless it is in a sub query contained in a HAVING clause, whereas, aggregate functions can be used in Having clause.

Types of Join

**General Formula for Joins**  
SELECT      ColumnList  
FROM           LeftTableName  
JOIN\_TYPE  RightTableName  
ON                 JoinCondition

**CROSS JOIN**CROSS JOIN, produces the cartesian product of the 2 tables involved in the join. For example, in the Employees table we have 10 rows and in the Departments table we have 4 rows. So, a cross join between these 2 tables produces 40 rows. Cross Join shouldn't have ON clause.

Create table tblDepartment

(

ID int primary key,

DepartmentName nvarchar(50),

Location nvarchar(50),

DepartmentHead nvarchar(50)

)

Go

Insert into tblDepartment values (1, 'IT', 'London', 'Rick')

Insert into tblDepartment values (2, 'Payroll', 'Delhi', 'Ron')

insert into tblDepartment values (3, 'HR', 'New York', 'Christie')

Insert into tblDepartment values (4, 'Other Department', 'Sydney', 'Cindrella')

Go

Create table tblEmploye

(

ID int primary key,

Name nvarchar(50),

Gender nvarchar(50),

Salary int,

DepartmentId int foreign key references tblDepartment(Id)

)

Go

Insert into tblEmploye values (1, 'Tom', 'Male', 4000, 1)

Insert into tblEmploye values (2, 'Pam', 'Female', 3000, 3)

Insert into tblEmploye values (3, 'John', 'Male', 3500, 1)

Insert into tblEmploye values (4, 'Sam', 'Male', 4500, 2)

Insert into tblEmploye values (5, 'Todd', 'Male', 2800, 2)

Insert into tblEmploye values (6, 'Ben', 'Male', 7000, 1)

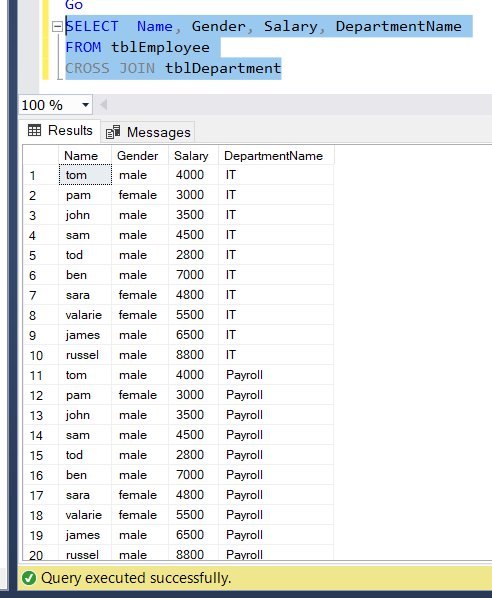
Insert into tblEmploye values (7, 'Sara', 'Female', 4800, 3)

Insert into tblEmploye values (8, 'Valarie', 'Female', 5500, 1)

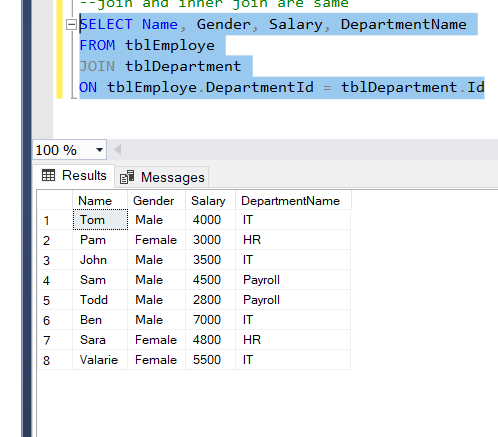
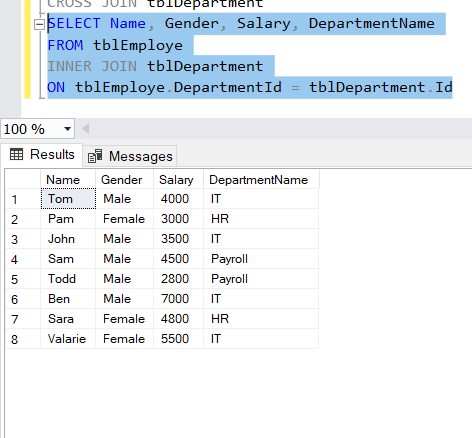
Insert into tblEmploye values (9, 'James', 'Male', 6500, NULL)

Insert into tblEmploye values (10, 'Russell', 'Male', 8800, NULL)

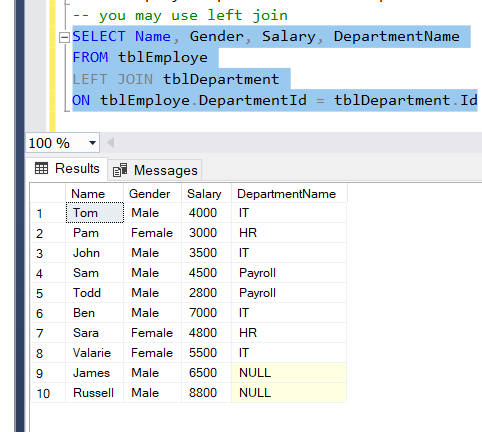
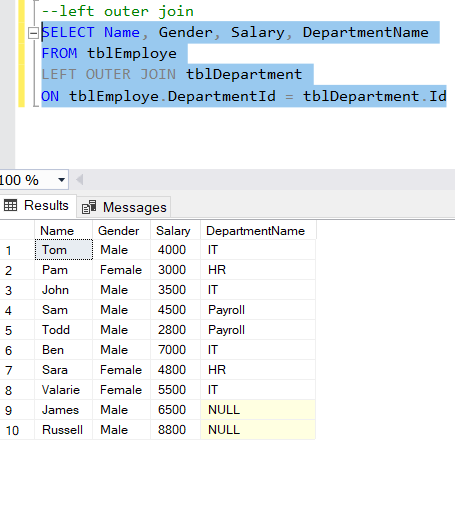
Go



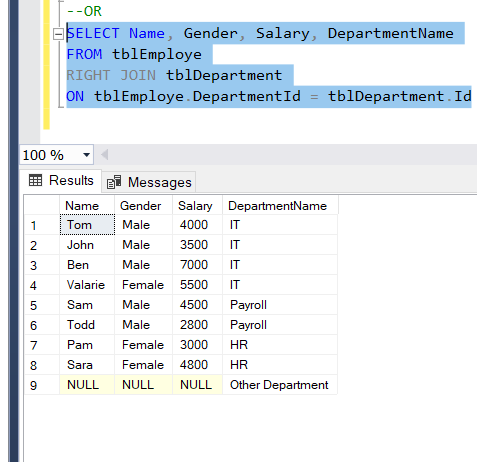
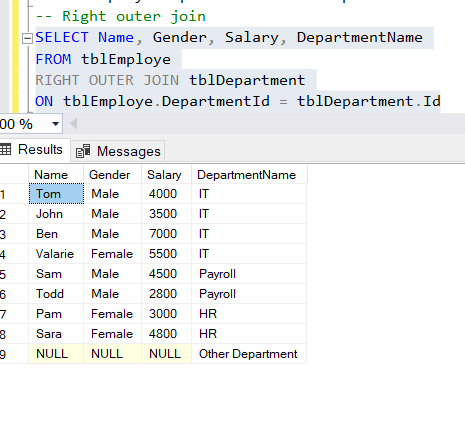
**JOIN or INNER JOIN**Write a query, to retrieve Name, Gender, Salary and DepartmentName from Employees and Departments table.



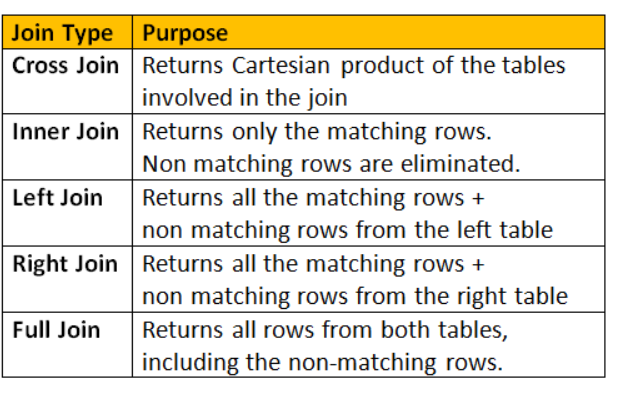
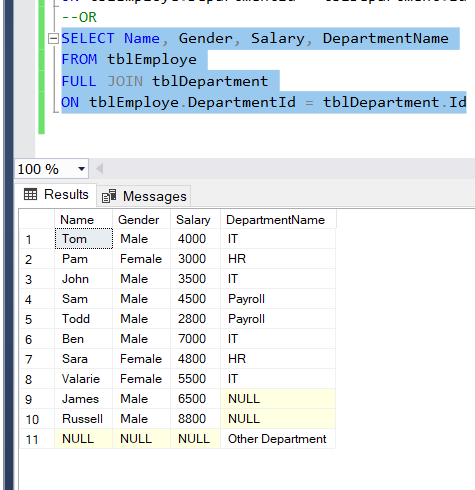
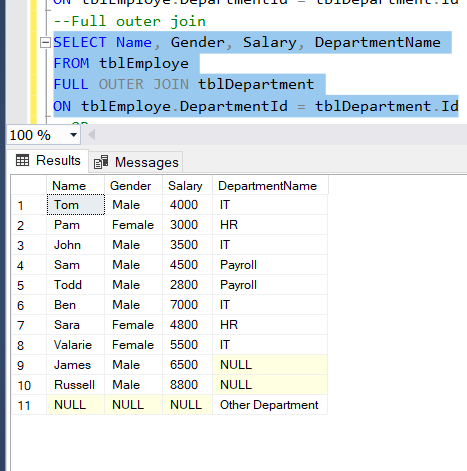
**LEFT JOIN or LEFT OUTER JOIN**  
Now, let's say, I want all the rows from the Employees table, including JAMES and RUSSELL records. I want the output, as shown below.



**RIGHT JOIN or RIGHT OUTER JOIN**  
I want, all the rows from the right table.



**FULL JOIN or FULL OUTER JOIN**I want all the rows from both the tables involved in the join.



Create Table TableEmployer

(

ID int identity(1,1)Primary Key NOT NULL,

Name nchar(50) Not NULL,

Gender nchar(10) NOT NULL,

Salary int Not NULL,

DepartamentID int

)

Create Table TableDepartament

(

ID int identity(1,1)Primary Key,

DepartamentName nchar(50) Not NULL,

Location nchar(59) Not Null,

DepartamentHead nchar(50) Not Null,

)

Alter Table TableEmployer

Add Constraint CK\_TableEmployer\_Salary

CHECK(Salary > 0)

Alter Table TableEmployer

ADD CONSTRAINT CK\_TableEmployer\_Gender

CHECK(Gender = 'Female' OR Gender = 'Male' OR Gender ='Unknown')

ALTER TABLE TableEmployer

ADD CONSTRAINT CK\_TableEmployer\_DepartamentID

CHECK(DepartamentID>=1 AND DepartamentID<=4)

INSERT INTO TableEmployer values ('Tom','Male',4000,1)

Insert into TableEmployer values('Ron','Male',4000,1)

Insert into TableEmployer values('Jessie','Female',4800,3)

Insert into TableEmployer values('Pam','Female',3500,1)

Insert into TableEmployer values('John','Male',3000,2)

Insert into TableEmployer values('Sam','Male',4500,2)

Insert into TableEmployer values('Linlna','Female',7300,3)

Insert into TableEmployer values('Artesa','Female',4000,2)

Insert into TableEmployer values('Dom','Male',5500,1)

Insert into TableEmployer values('Mayer','Male',9000,3)

Insert into TableEmployer values('Aryueta','Female',3000,2)

Insert into TableEmployer values('Bangklor','Male',4560,1)

--Insert into TableEmployer values('Bangklor','Male',4560,5)

Select Name ,Gender ,Salary ,DepartamentID

From TableEmployer

INNER JOIN TableDepartament

ON TableEmployer.DepartamentID = TableDepartament.ID

Select Name ,Gender ,Salary ,DepartamentID

From TableEmployer

LEFT JOIN TableDepartament

ON TableEmployer.DepartamentID = TableDepartament.ID

Select Name ,Gender ,Salary ,DepartamentID

From TableEmployer

RIGHT OUTER JOIN TableDepartament

ON TableEmployer.DepartamentID = TableDepartament.ID

Select Name ,Gender ,Salary ,DepartamentID

From TableEmployer

FULL OUTER JOIN TableDepartament

ON TableEmployer.DepartamentID = TableDepartament.ID

Select Name ,Gender ,Salary ,DepartamentID

From TableEmployer

CROSS JOIN TableDepartament

-- doesn't need a clause ON TableEmployer.DepartamentID = TableDepartament.ID

-- CROSS PRODUCES CONTATION PRODUCTS LEFTTABLE X RIGHTTABLE

-- IT TAKES EACH RECORD FROM LEFT TABLE AND ASSOCIATES WITH RIGHT TABLE

/\*Select ColumnList

From LeftTable

JoinType RighTable

ON JoinCondition \*/ -- GENERIC IDEA ON HOW TO CREATE A JOIN

Select Name,Gender,Salary ,DepartamentID

From TableEmployer

/\* any join \*/ INNER JOIN TableDepartament

ON TableEmployer.DepartamentID = TableDepartament.ID

