

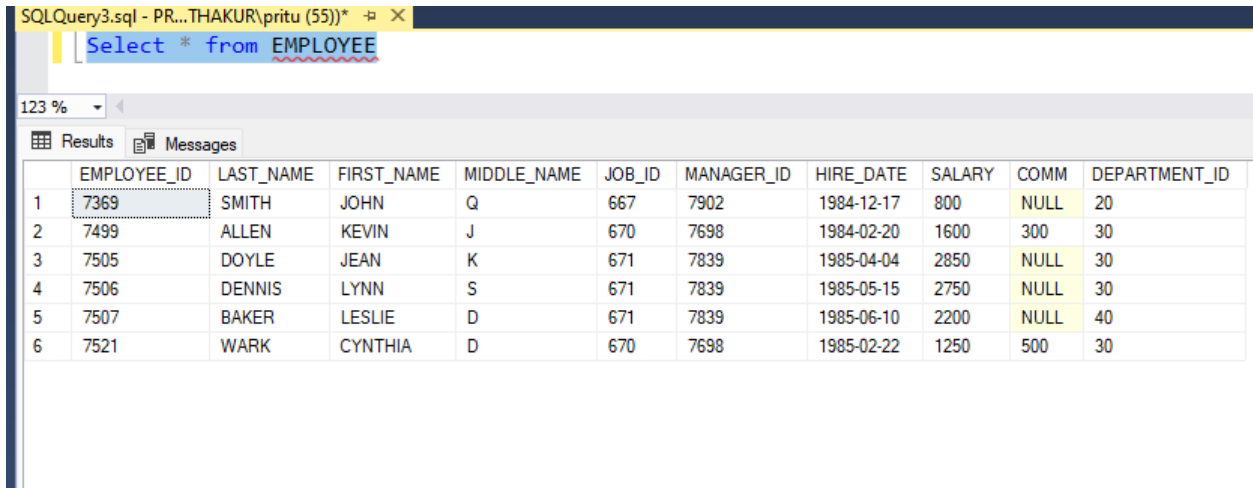
## SQL\_CaseStudy\_2

### Simple Queries

1. List all the employee details.

Query:

Select \* from EMPLOYEE



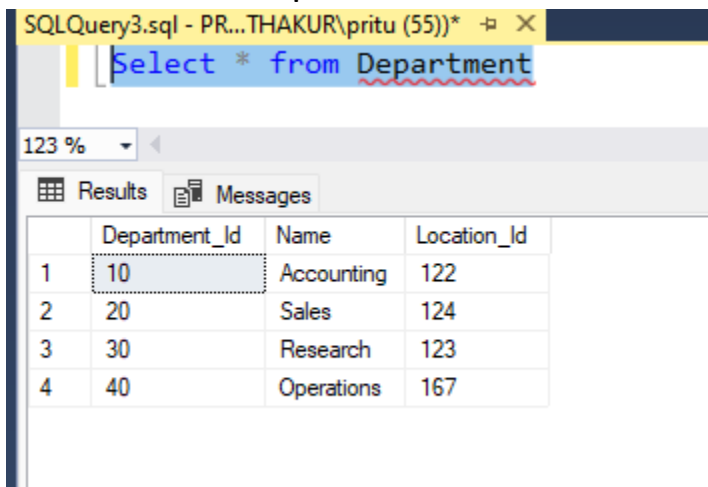
The screenshot shows a SQL query window with the text 'Select \* from EMPLOYEE'. Below the query, the 'Results' tab is active, displaying a table with 10 columns: EMPLOYEE\_ID, LAST\_NAME, FIRST\_NAME, MIDDLE\_NAME, JOB\_ID, MANAGER\_ID, HIRE\_DATE, SALARY, COMM, and DEPARTMENT\_ID. The table contains 6 rows of data.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20
2	7499	ALLEN	KEVIN	J	670	7698	1984-02-20	1600	300	30
3	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
4	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
5	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40
6	7521	WARK	CYNTHIA	D	670	7698	1985-02-22	1250	500	30

2. List all the department details.

Query:

Select \* from Department



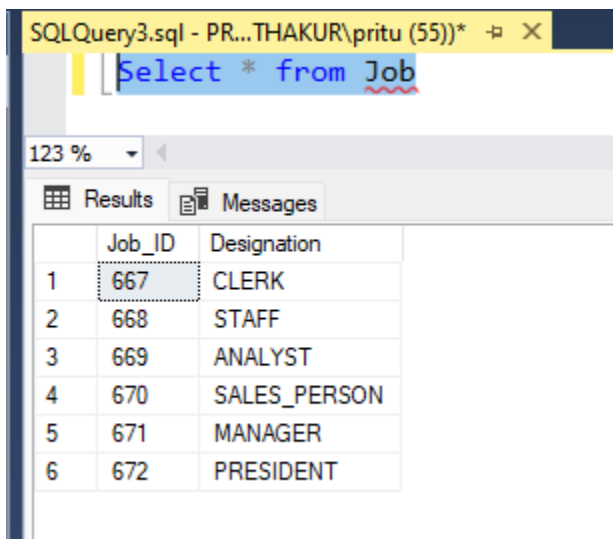
The screenshot shows a SQL query window with the text 'Select \* from Department'. Below the query, the 'Results' tab is active, displaying a table with 4 columns: Department\_Id, Name, and Location\_Id. The table contains 4 rows of data.

	Department_Id	Name	Location_Id
1	10	Accounting	122
2	20	Sales	124
3	30	Research	123
4	40	Operations	167

3. List all job details.

Query:

Select \* from Job



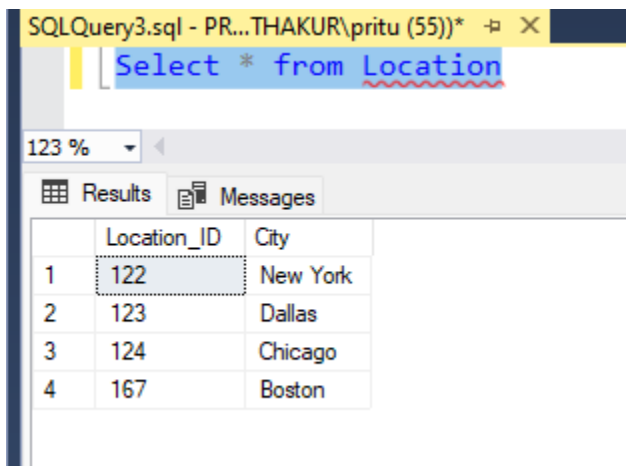
The screenshot shows a SQL Query Editor window with the title 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query 'Select \* from Job' is entered in the editor. Below the editor, the 'Results' tab is selected, displaying a table with 6 rows and 2 columns: 'Job\_ID' and 'Designation'. The zoom level is set to 123%.

	Job_ID	Designation
1	667	CLERK
2	668	STAFF
3	669	ANALYST
4	670	SALES_PERSON
5	671	MANAGER
6	672	PRESIDENT

4. List all the locations.

Query:

Select \* from Location



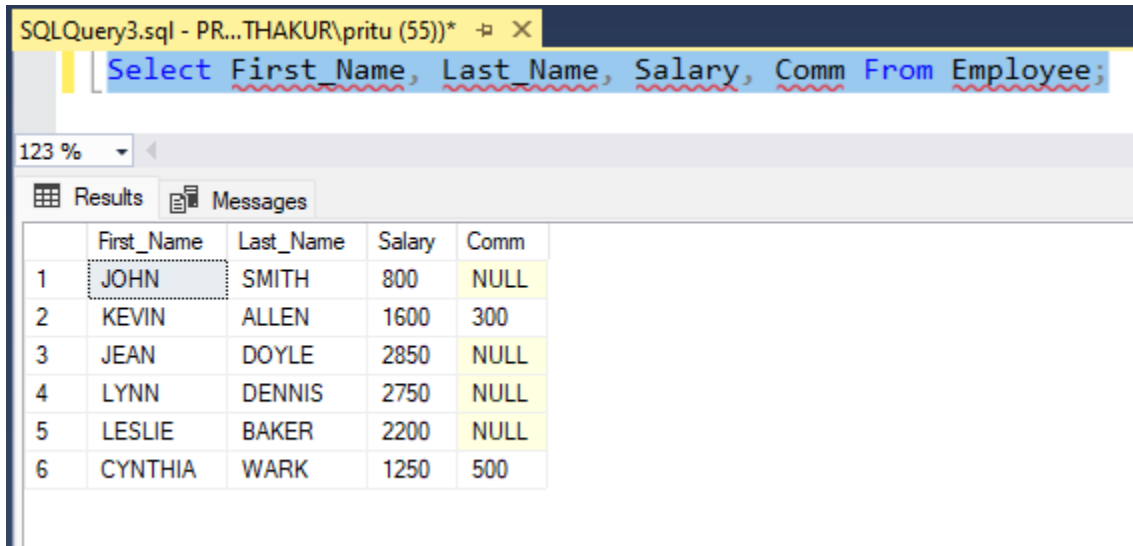
The screenshot shows a SQL Query Editor window with the title 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query 'Select \* from Location' is entered in the editor. Below the editor, the 'Results' tab is selected, displaying a table with 4 rows and 2 columns: 'Location\_ID' and 'City'. The zoom level is set to 123%.

	Location_ID	City
1	122	New York
2	123	Dallas
3	124	Chicago
4	167	Boston

5. List out the First Name, Last Name, Salary, Commission for all Employees.

Query:

Select First\_Name, Last\_Name, Salary, Comm From Employee;



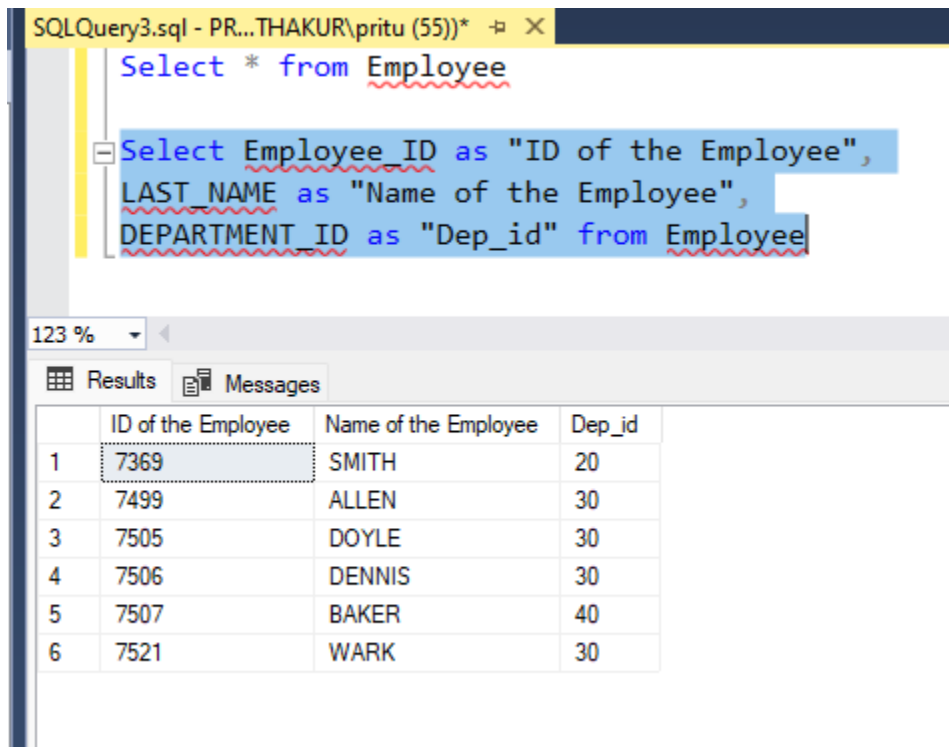
The screenshot shows a SQL Query Editor window titled 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query entered is 'Select First\_Name, Last\_Name, Salary, Comm From Employee;'. Below the query editor, the 'Results' tab is active, displaying a table with 6 rows and 5 columns: First\_Name, Last\_Name, Salary, and Comm. The first row is highlighted with a dotted border.

	First_Name	Last_Name	Salary	Comm
1	JOHN	SMITH	800	NULL
2	KEVIN	ALLEN	1600	300
3	JEAN	DOYLE	2850	NULL
4	LYNN	DENNIS	2750	NULL
5	LESLIE	BAKER	2200	NULL
6	CYNTHIA	WARK	1250	500

6. List out the Employee ID, Last Name, Department ID for all employees and alias Employee ID as "ID of the Employee", Last Name as "Name of the Employee", Department ID as "Dep\_id".

Query:

```
Select Employee_ID as "ID of the Employee",  
LAST_NAME as "Name of the Employee",  
DEPARTMENT_ID as "Dep_id" from Employee
```



The screenshot shows a SQL Developer window with a query editor and a results grid. The query editor contains the following SQL statement:

```
Select * from Employee
```

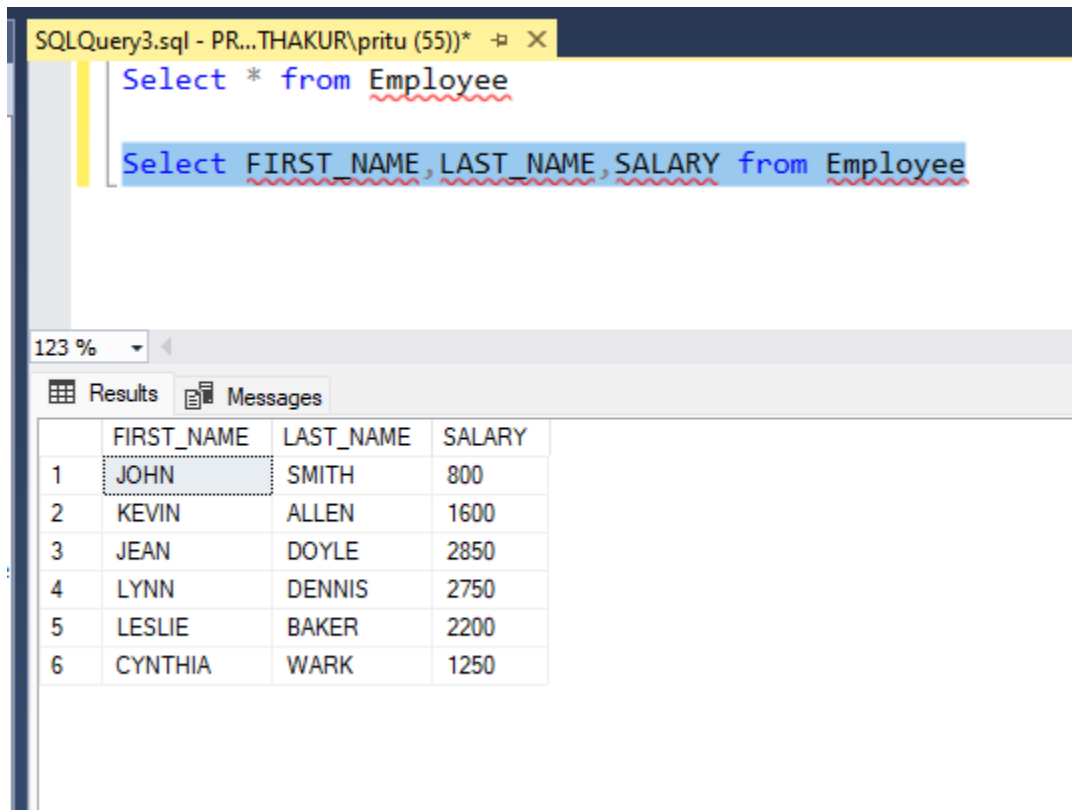
Below the query editor, the results grid is displayed, showing the output of the query. The grid has four columns: "ID of the Employee", "Name of the Employee", and "Dep\_id". The first row is highlighted, showing the employee with ID 7369, name SMITH, and department 20.

	ID of the Employee	Name of the Employee	Dep_id
1	7369	SMITH	20
2	7499	ALLEN	30
3	7505	DOYLE	30
4	7506	DENNIS	30
5	7507	BAKER	40
6	7521	WARK	30

7. List out the annual salary of the employees with their names only.

Query:

Select FIRST\_NAME, LAST\_NAME, SALARY from Employee



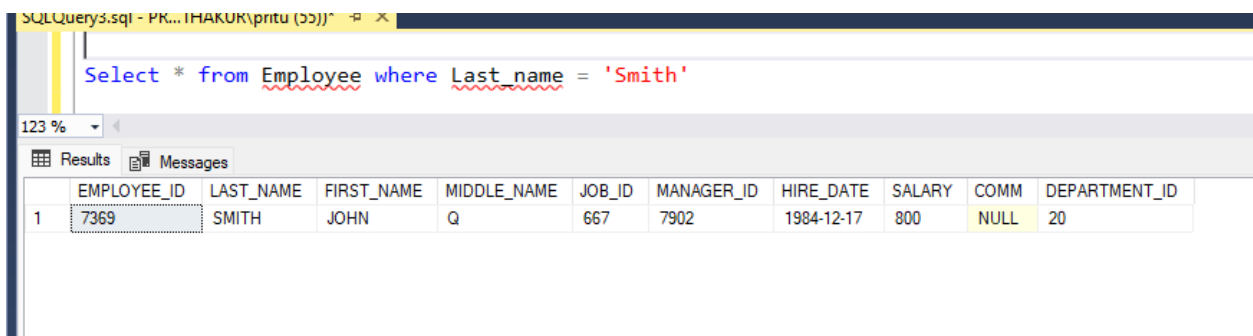
The screenshot shows a SQL Developer window with the title 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query editor contains two lines of SQL code: 'Select \* from Employee' and 'Select FIRST\_NAME, LAST\_NAME, SALARY from Employee'. The second line is highlighted. Below the editor, the 'Results' tab is active, displaying a table with 6 rows and 4 columns: FIRST\_NAME, LAST\_NAME, and SALARY. The first row is highlighted.

	FIRST_NAME	LAST_NAME	SALARY
1	JOHN	SMITH	800
2	KEVIN	ALLEN	1600
3	JEAN	DOYLE	2850
4	LYNN	DENNIS	2750
5	LESLIE	BAKER	2200
6	CYNTHIA	WARK	1250

## Where Condition

1. List the details about "Smith".

Query: Select \* from Employee where Last\_name = 'Smith'

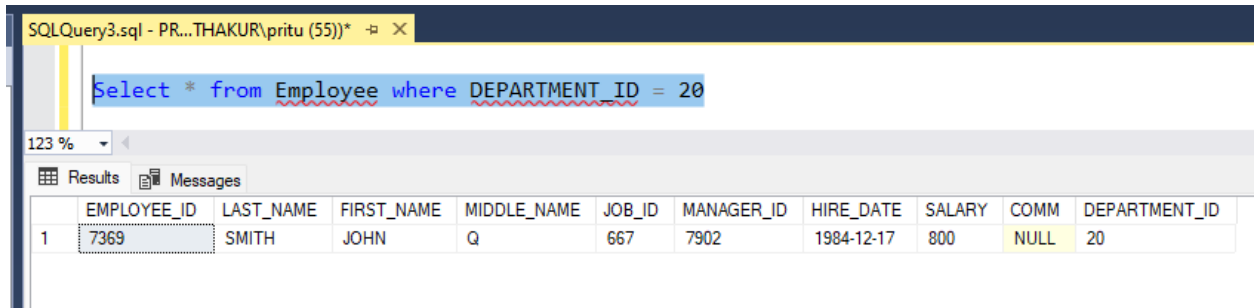


The screenshot shows a SQL Developer window with the title 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query editor contains the SQL code: 'Select \* from Employee where Last\_name = 'Smith''. Below the editor, the 'Results' tab is active, displaying a table with 11 columns: EMPLOYEE\_ID, LAST\_NAME, FIRST\_NAME, MIDDLE\_NAME, JOB\_ID, MANAGER\_ID, HIRE\_DATE, SALARY, COMM, and DEPARTMENT\_ID. The first row is highlighted.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20

2. List out the employees who are working in department 20.

Query: Select \* from Employee where DEPARTMENT\_ID = 20

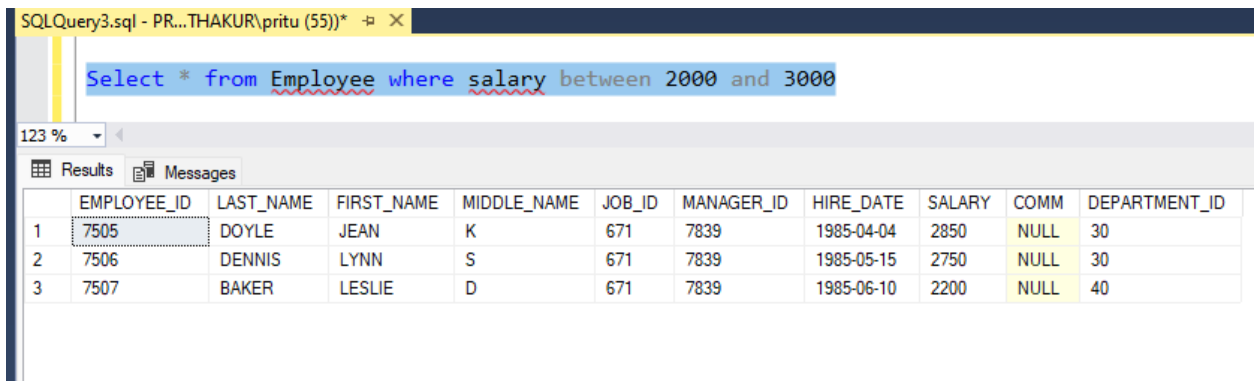


The screenshot shows a SQL query window titled 'SQLQuery3.sql - PR...THAKUR\pritu (55))' with the query: `Select * from Employee where DEPARTMENT_ID = 20`. The results pane shows a single row for employee SMITH (ID 7369) in department 20.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20

3. List out the employees who are earning salary between 2000 and 3000.

Query: Select \* from Employee where salary between 2000 and 3000

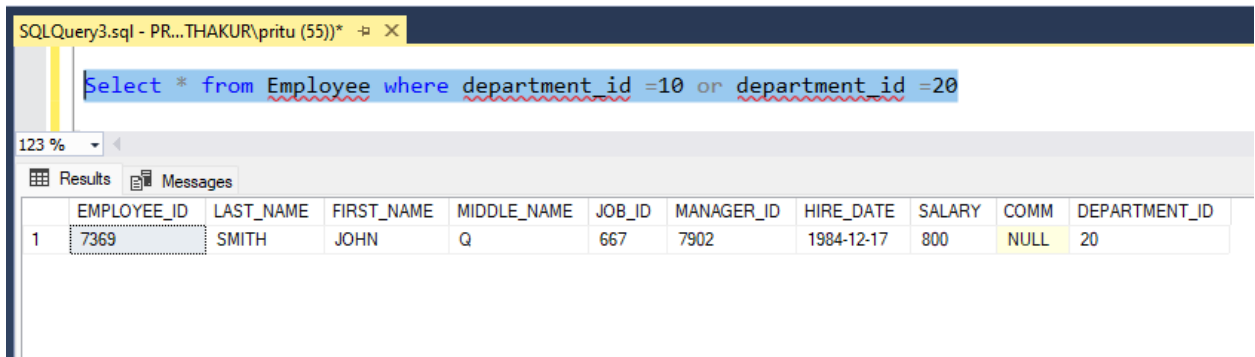


The screenshot shows a SQL query window titled 'SQLQuery3.sql - PR...THAKUR\pritu (55))' with the query: `Select * from Employee where salary between 2000 and 3000`. The results pane shows three rows of employees with salaries between 2000 and 3000.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
2	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
3	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40

4. List out the employees who are working in department 10 or 20.

Query: Select \* from Employee where department\_id =10 or department\_id =20

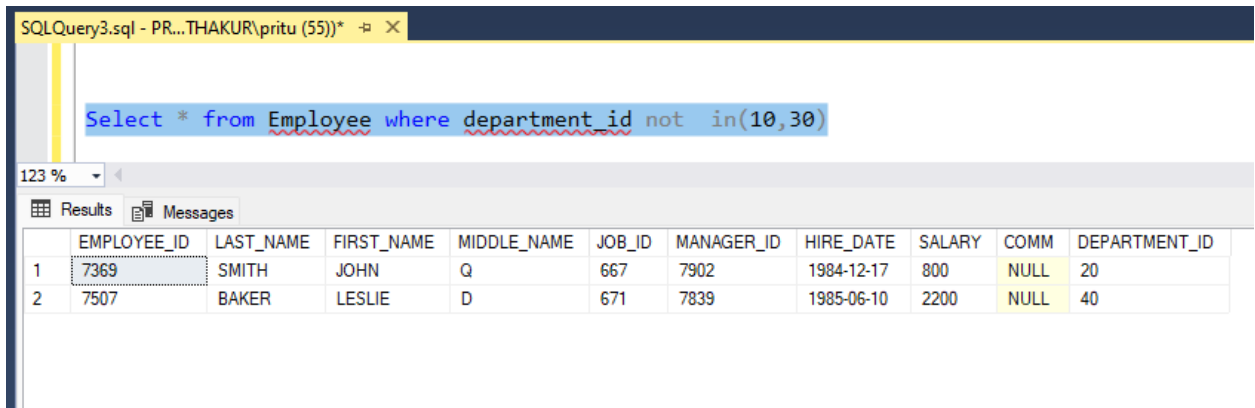


The screenshot shows a SQL Query Editor window titled "SQLQuery3.sql - PR...THAKUR\pritu (55)". The query entered is "Select \* from Employee where department\_id =10 or department\_id =20". The results pane shows a single row with the following data:

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20

5. Find out the employees who are not working in department 10 or 30.

Query: Select \* from Employee where department\_id not in(10,30)

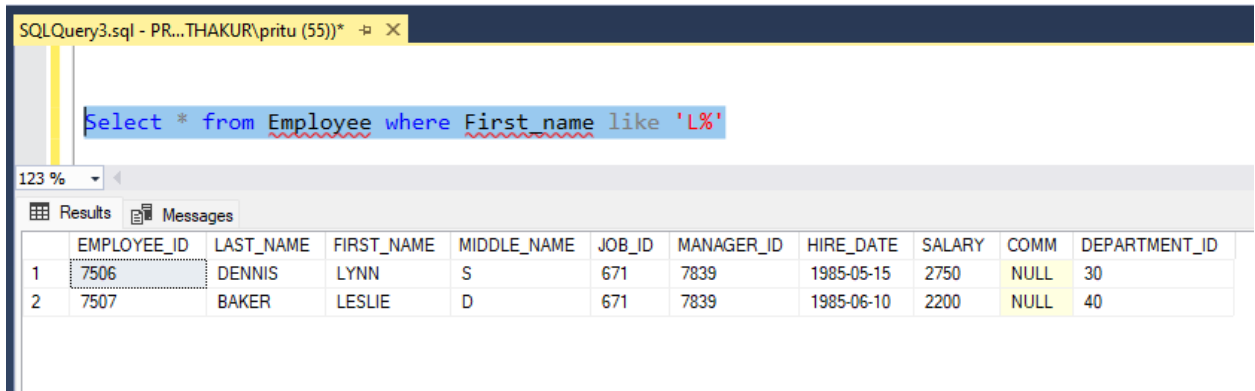


The screenshot shows a SQL Query Editor window titled "SQLQuery3.sql - PR...THAKUR\pritu (55)". The query entered is "Select \* from Employee where department\_id not in(10,30)". The results pane shows two rows with the following data:

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20
2	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40

6. List out the employees whose name starts with 'L'.

Query: Select \* from Employee where First\_name like 'L%'

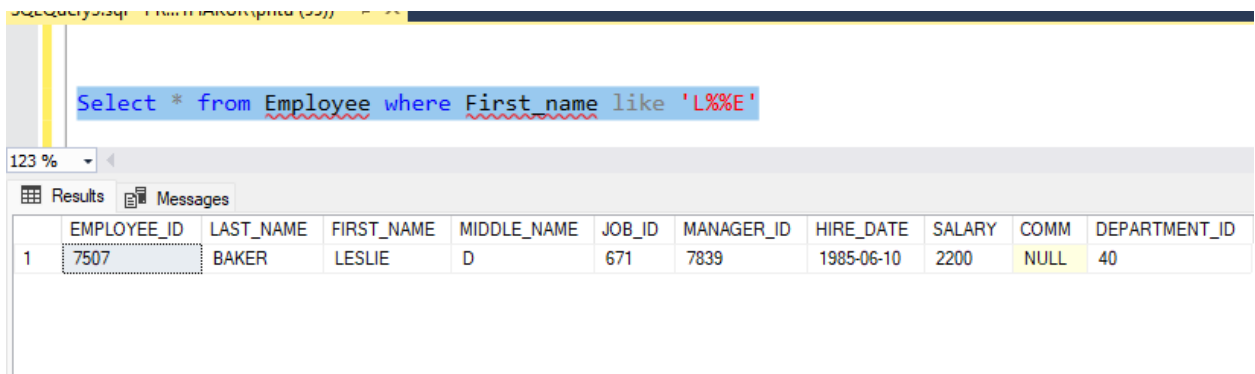


The screenshot shows a SQL Developer window with a query editor at the top containing the SQL statement: `Select * from Employee where First_name like 'L%'`. Below the editor, the 'Results' tab is active, displaying a table with 11 columns: EMPLOYEE\_ID, LAST\_NAME, FIRST\_NAME, MIDDLE\_NAME, JOB\_ID, MANAGER\_ID, HIRE\_DATE, SALARY, COMM, and DEPARTMENT\_ID. Two rows of data are shown, both with first names starting with 'L'.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
2	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40

7. List out the employees whose name starts with 'L' and ends with 'E'.

Query: Select \* from Employee where First\_name like 'L%%E'



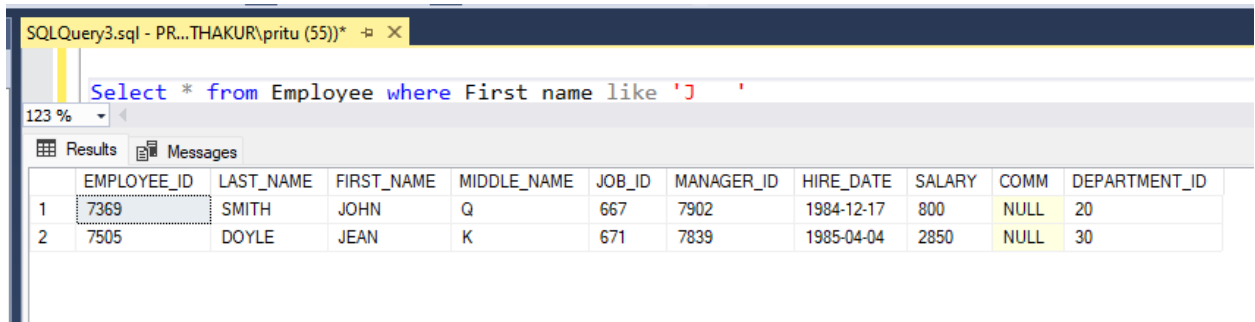
The screenshot shows a SQL Developer window with a query editor at the top containing the SQL statement: `Select * from Employee where First_name like 'L%%E'`. Below the editor, the 'Results' tab is active, displaying a table with 11 columns: EMPLOYEE\_ID, LAST\_NAME, FIRST\_NAME, MIDDLE\_NAME, JOB\_ID, MANAGER\_ID, HIRE\_DATE, SALARY, COMM, and DEPARTMENT\_ID. Only one row of data is shown, with a first name ending in 'E'.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40



8. List out the employees whose name length is 4 and start with 'J'.

Query: Select \* from Employee where First\_name like 'J\_\_\_\_'

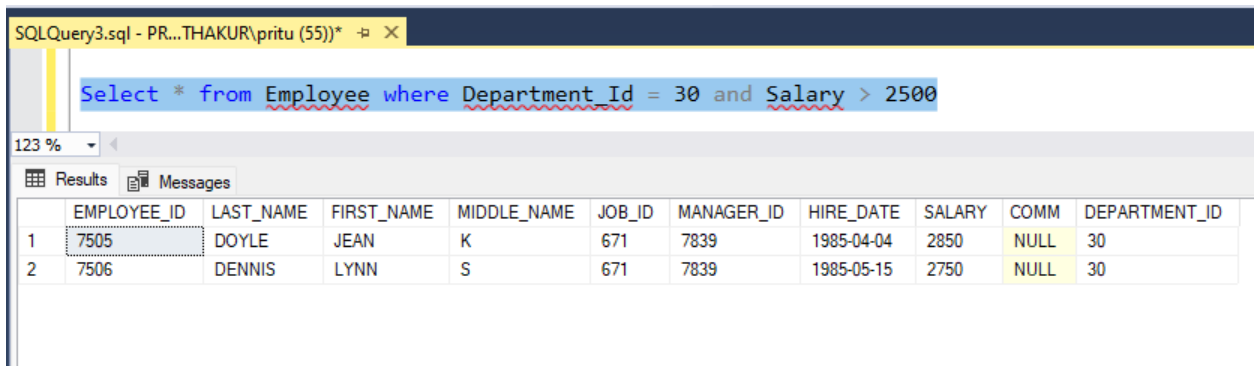


The screenshot shows a SQL query window with the query: `Select * from Employee where First_name like 'J____'`. The results pane displays two rows of data from the Employee table.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20
2	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30

9. List out the employees who are working in department 30 and draw the salaries more than 2500.

Query: Select \* from Employee where Department\_Id = 30 and Salary > 2500

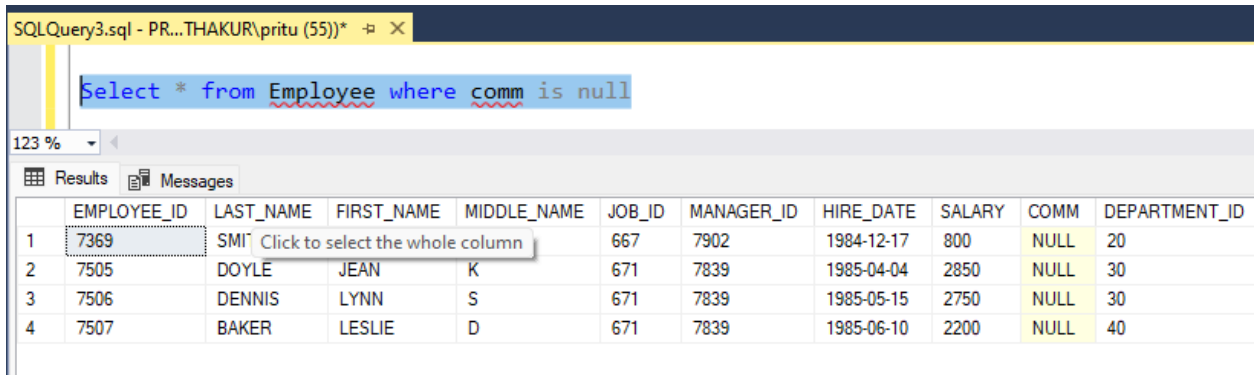


The screenshot shows a SQL query window with the query: `Select * from Employee where Department_Id = 30 and Salary > 2500`. The results pane displays two rows of data from the Employee table.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
2	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30

10. List out the employees who are not receiving commission.

Query: Select \* from Employee where comm is null



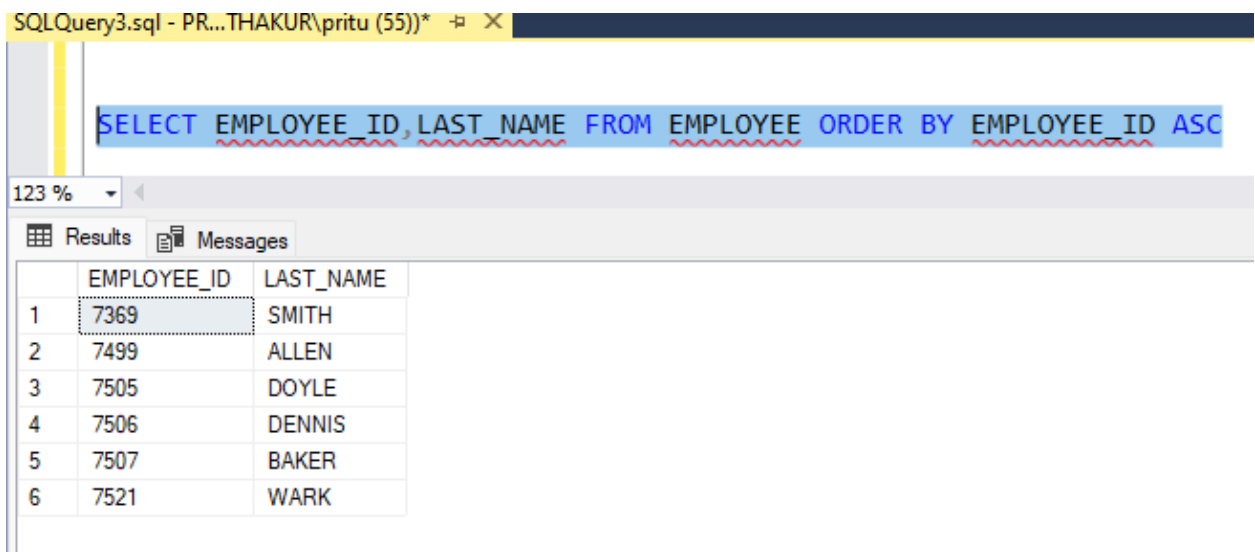
The screenshot shows a SQL query window with the text "Select \* from Employee where comm is null". Below the query, the "Results" tab is active, displaying a table with 10 columns: EMPLOYEE\_ID, LAST\_NAME, FIRST\_NAME, MIDDLE\_NAME, JOB\_ID, MANAGER\_ID, HIRE\_DATE, SALARY, COMM, and DEPARTMENT\_ID. The table contains 4 rows of data, all with NULL values in the COMM column.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH			667	7902	1984-12-17	800	NULL	20
2	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
3	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
4	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40

## ORDER BY Clause:

1. List out the Employee ID and Last Name in ascending order based on the Employee ID.

Query: SELECT EMPLOYEE\_ID, LAST\_NAME FROM EMPLOYEE ORDER BY EMPLOYEE\_ID ASC

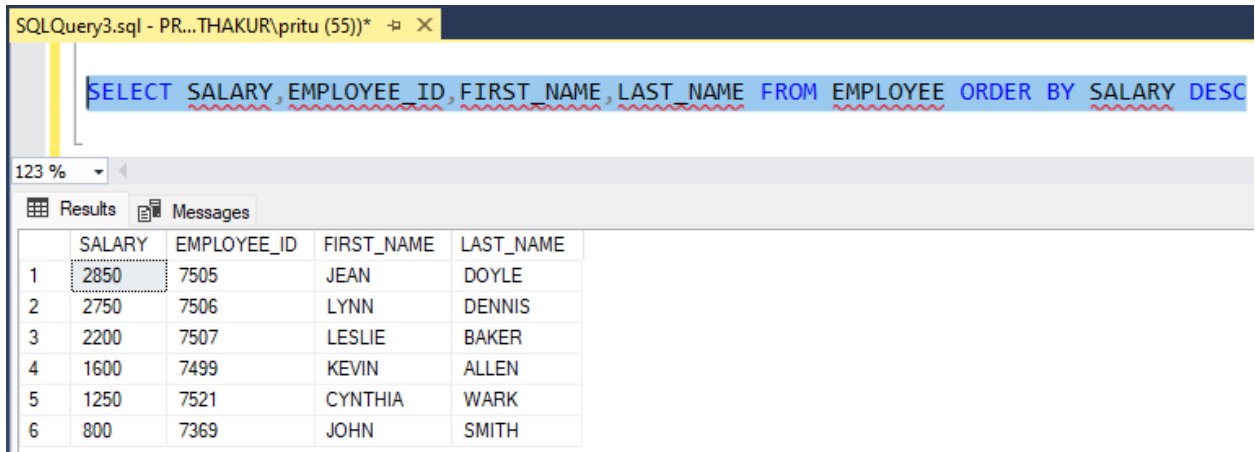


The screenshot shows a SQL query window with the text "SELECT EMPLOYEE\_ID, LAST\_NAME FROM EMPLOYEE ORDER BY EMPLOYEE\_ID ASC". Below the query, the "Results" tab is active, displaying a table with 2 columns: EMPLOYEE\_ID and LAST\_NAME. The table contains 6 rows of data, sorted in ascending order by EMPLOYEE\_ID.

	EMPLOYEE_ID	LAST_NAME
1	7369	SMITH
2	7499	ALLEN
3	7505	DOYLE
4	7506	DENNIS
5	7507	BAKER
6	7521	WARK

2. List out the Employee ID and Name in descending order based on salary.

Query: SELECT SALARY,EMPLOYEE\_ID,FIRST\_NAME,LAST\_NAME  
FROM EMPLOYEE ORDER BY SALARY DESC

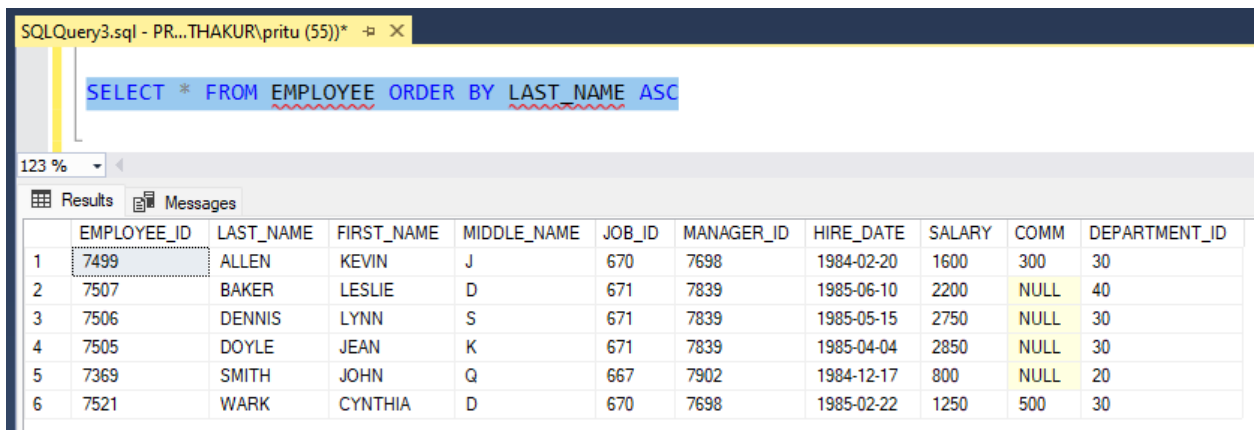


The screenshot shows a SQL Developer window with the title 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query editor contains the SQL statement: `SELECT SALARY,EMPLOYEE_ID,FIRST_NAME,LAST_NAME FROM EMPLOYEE ORDER BY SALARY DESC`. Below the editor, the 'Results' tab is active, displaying a table with 5 columns: SALARY, EMPLOYEE\_ID, FIRST\_NAME, and LAST\_NAME. The results are ordered by salary in descending order, with 6 rows shown.

	SALARY	EMPLOYEE_ID	FIRST_NAME	LAST_NAME
1	2850	7505	JEAN	DOYLE
2	2750	7506	LYNN	DENNIS
3	2200	7507	LESLIE	BAKER
4	1600	7499	KEVIN	ALLEN
5	1250	7521	CYNTHIA	WARK
6	800	7369	JOHN	SMITH

3. List out the employee details according to their Last Name in ascending-order.

Query: SELECT \* FROM EMPLOYEE ORDER BY LAST\_NAME ASC



The screenshot shows a SQL Developer window with the title 'SQLQuery3.sql - PR...THAKUR\pritu (55))'. The query editor contains the SQL statement: `SELECT * FROM EMPLOYEE ORDER BY LAST_NAME ASC`. Below the editor, the 'Results' tab is active, displaying a table with 10 columns: EMPLOYEE\_ID, LAST\_NAME, FIRST\_NAME, MIDDLE\_NAME, JOB\_ID, MANAGER\_ID, HIRE\_DATE, SALARY, COMM, and DEPARTMENT\_ID. The results are ordered by last name in ascending order, with 6 rows shown.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7499	ALLEN	KEVIN	J	670	7698	1984-02-20	1600	300	30
2	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40
3	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
4	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
5	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20
6	7521	WARK	CYNTHIA	D	670	7698	1985-02-22	1250	500	30

4. List out the employee details according to their Last Name in ascending order and then Department ID in descending order.

Query: SELECT \* FROM employee

ORDER BY last\_name ASC, department\_id DESC;

SELECT \* FROM employee

ORDER BY department\_id DESC, last\_name ASC;

SQLQuery3.sql - PR...THAKUR\pritu (55))\*

```
Select * from Employee
```

```
SELECT *  
FROM employee  
ORDER BY last_name ASC, department_id DESC;
```

123 %

Results Messages

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7499	ALLEN	KEVIN	J	670	7698	1984-02-20	1600	300	30
2	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40
3	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
4	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
5	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20
6	7521	WARK	CYNTHIA	D	670	7698	1985-02-22	1250	500	30

SQLQuery3.sql - PR...THAKUR\pritu (55))\*

```
SELECT * FROM employee  
ORDER BY department_id DESC, last_name ASC;
```

123 %

Results Messages

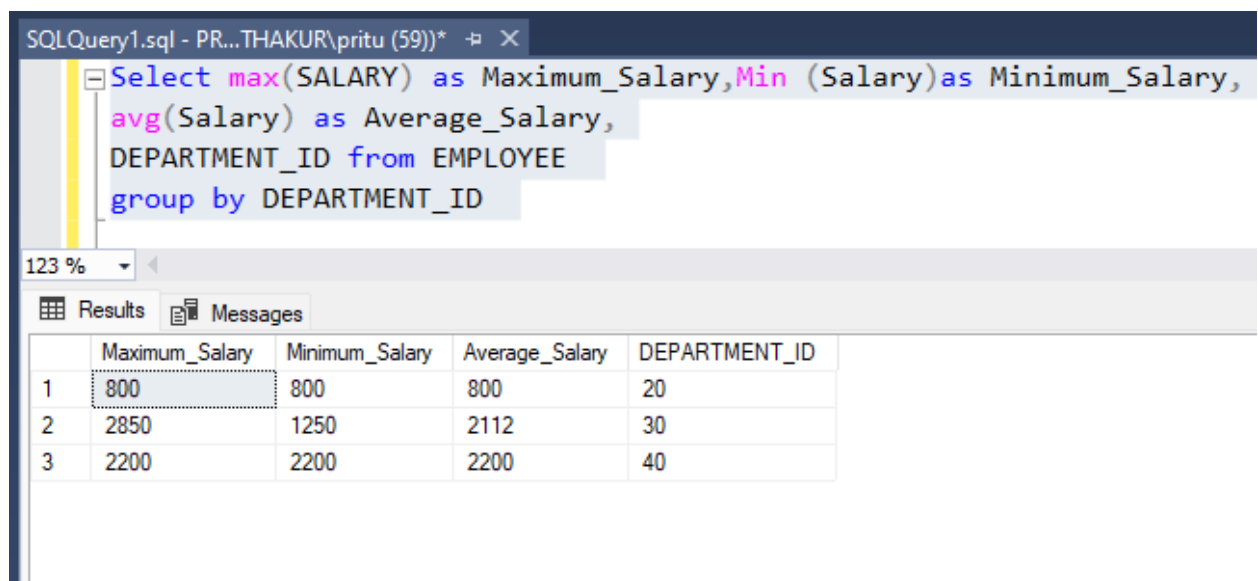
	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40
2	7499	ALLEN	KEVIN	J	670	7698	1984-02-20	1600	300	30
3	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
4	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
5	7521	WARK	CYNTHIA	D	670	7698	1985-02-22	1250	500	30
6	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20

## GROUP BY and HAVING Clause:

1. List out the department wise maximum salary, minimum salary, and average salary of the employees.

Query: Select max(SALARY) as Maximum\_Salary, Min (Salary) as Minimum\_Salary, avg(Salary) as Average\_Salary, DEPARTMENT\_ID  
from EMPLOYEE

group by DEPARTMENT\_ID



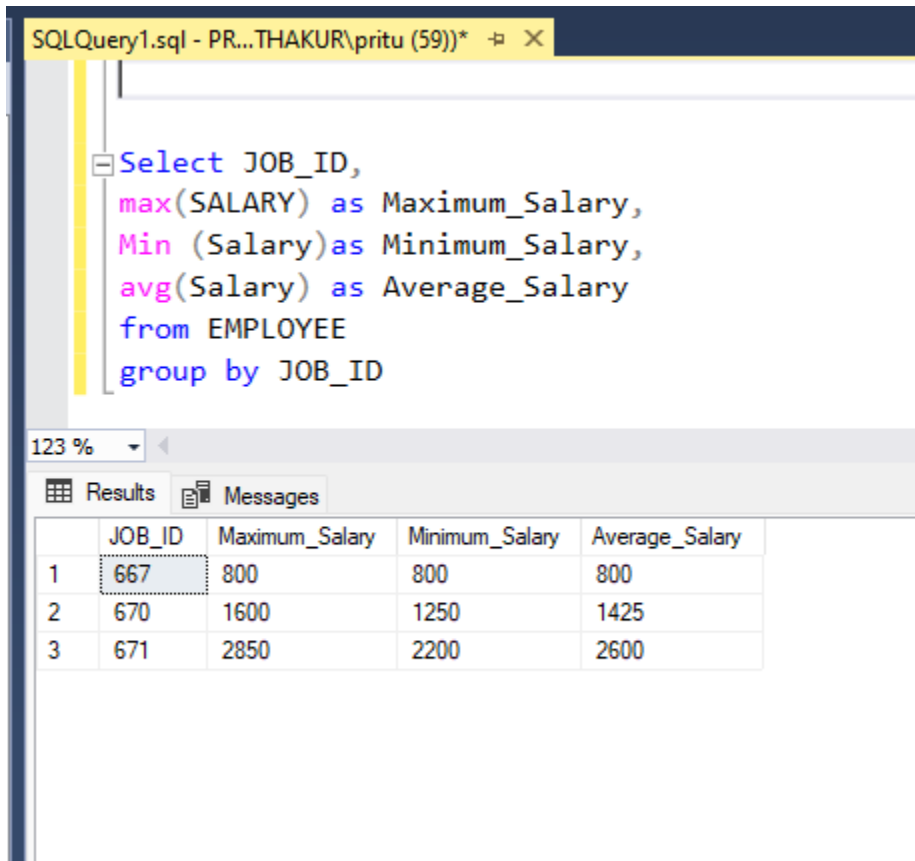
The screenshot shows a SQL Developer window with a query editor and a results grid. The query is: `Select max(SALARY) as Maximum_Salary, Min (Salary) as Minimum_Salary, avg(Salary) as Average_Salary, DEPARTMENT_ID from EMPLOYEE group by DEPARTMENT_ID`. The results grid displays three rows of data for departments 20, 30, and 40.

	Maximum_Salary	Minimum_Salary	Average_Salary	DEPARTMENT_ID
1	800	800	800	20
2	2850	1250	2112	30
3	2200	2200	2200	40

2. List out the job wise maximum salary, minimum salary and average salary of the employees.

Query: Select JOB\_ID,  
max(SALARY) as Maximum\_Salary,  
Min (Salary) as Minimum\_Salary,  
avg(Salary) as Average\_Salary  
from EMPLOYEE

group by JOB\_ID



The screenshot shows a SQL Developer window with a query editor and a results pane. The query editor contains the following SQL code:

```
Select JOB_ID,  
max(SALARY) as Maximum_Salary,  
Min (Salary)as Minimum_Salary,  
avg(Salary) as Average_Salary  
from EMPLOYEE  
group by JOB_ID
```

The results pane shows a table with 5 columns: JOB\_ID, Maximum\_Salary, Minimum\_Salary, and Average\_Salary. The table contains 3 rows of data.

	JOB_ID	Maximum_Salary	Minimum_Salary	Average_Salary
1	667	800	800	800
2	670	1600	1250	1425
3	671	2850	2200	2600

3. List out the number of employees who joined each month in ascending order.

Query:

SELECT

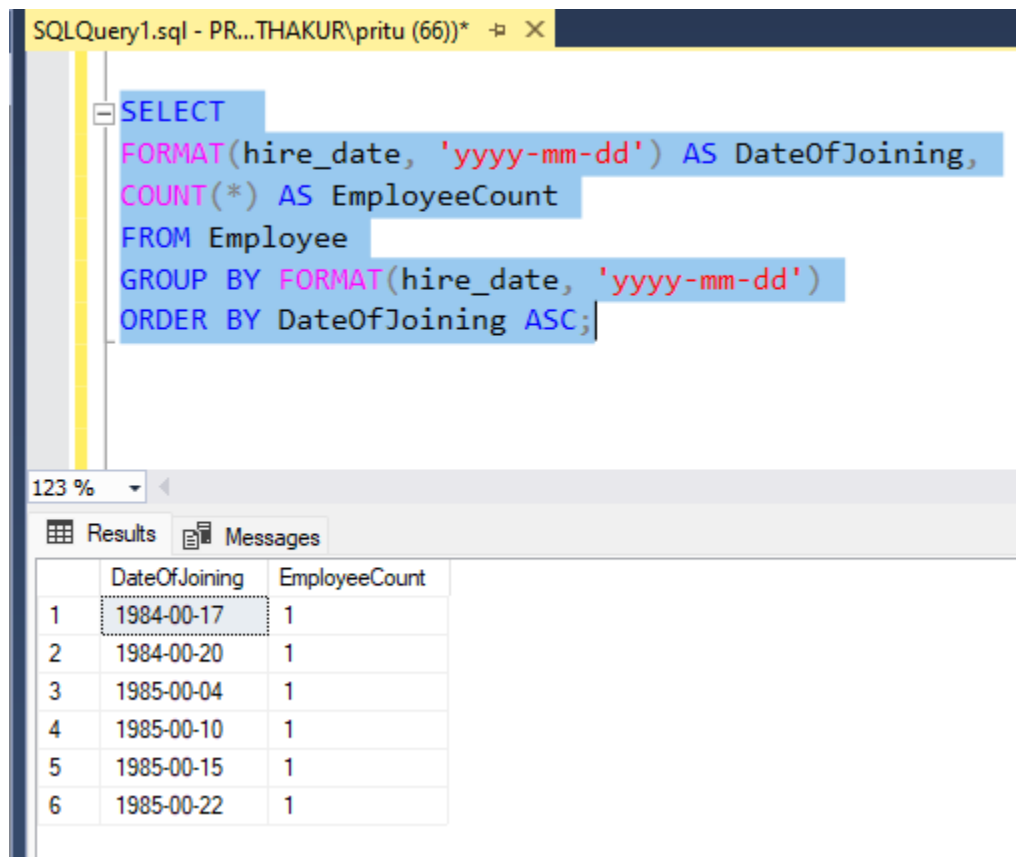
FORMAT(hire\_date, 'yyyy-mm-dd') AS DateOfJoining,

COUNT(\*) AS EmployeeCount

FROM Employee

GROUP BY FORMAT(hire\_date, 'yyyy-mm-dd')

ORDER BY DateOfJoining ASC;



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor contains the following SQL code:

```
SELECT  
    FORMAT(hire_date, 'yyyy-mm-dd') AS DateOfJoining,  
    COUNT(*) AS EmployeeCount  
FROM Employee  
GROUP BY FORMAT(hire_date, 'yyyy-mm-dd')  
ORDER BY DateOfJoining ASC;
```

The results pane shows a table with two columns: DateOfJoining and EmployeeCount. The table contains six rows of data, sorted by DateOfJoining in ascending order.

	DateOfJoining	EmployeeCount
1	1984-00-17	1
2	1984-00-20	1
3	1985-00-04	1
4	1985-00-10	1
5	1985-00-15	1
6	1985-00-22	1

4. List out the number of employees for each month and year in ascending order based on the year and month.

Query: SELECT

YEAR(HIRE\_DATE) AS Year,

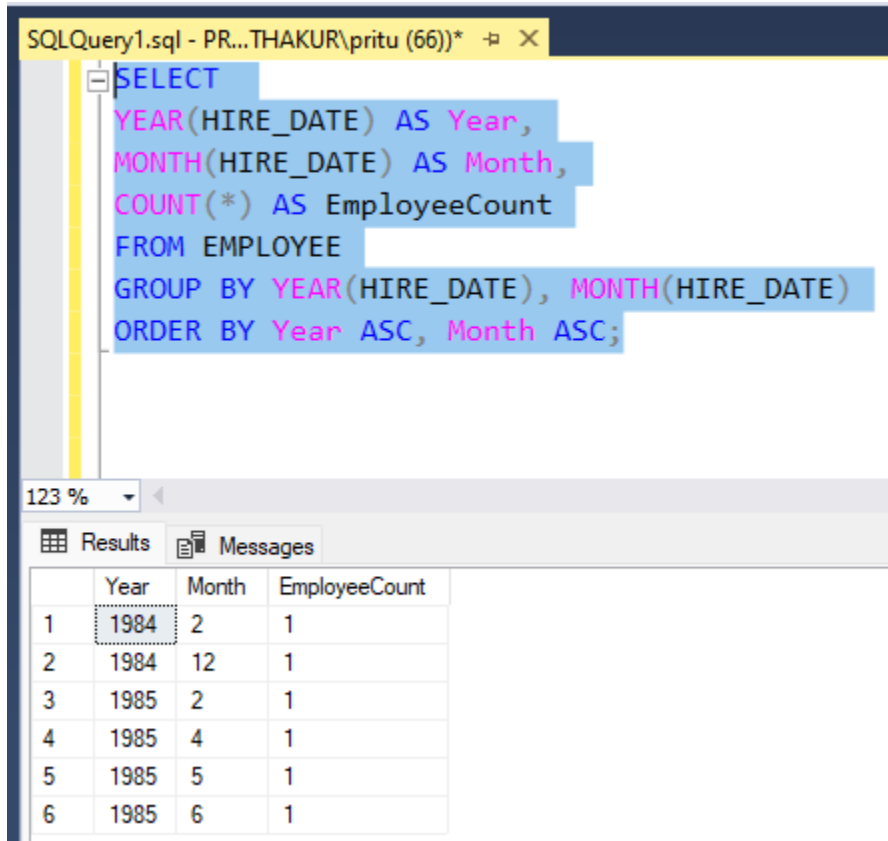
MONTH(HIRE\_DATE) AS Month,

COUNT(\*) AS EmployeeCount

FROM EMPLOYEE

GROUP BY YEAR(HIRE\_DATE), MONTH(HIRE\_DATE)

ORDER BY Year ASC, Month ASC;



The screenshot shows a SQL query window titled "SQLQuery1.sql - PR...THAKUR\pritu (66))\*". The query is as follows:

```
SELECT  
YEAR(HIRE_DATE) AS Year,  
MONTH(HIRE_DATE) AS Month,  
COUNT(*) AS EmployeeCount  
FROM EMPLOYEE  
GROUP BY YEAR(HIRE_DATE), MONTH(HIRE_DATE)  
ORDER BY Year ASC, Month ASC;
```

Below the query window, the "Results" tab is active, displaying a table with 3 columns: Year, Month, and EmployeeCount. The table contains 6 rows of data.

	Year	Month	EmployeeCount
1	1984	2	1
2	1984	12	1
3	1985	2	1
4	1985	4	1
5	1985	5	1
6	1985	6	1

6. List out the Department ID having at least four employees.

Query:

```
SELECT DEPARTMENT_ID,COUNT(*) AS EMPLOYEE_COUNT FROM  
EMPLOYEE
```

```
GROUP BY DEPARTMENT_ID HAVING COUNT(*) >= 4
```



SQLQuery1.sql - PR...THAKUR\pritu (66))\*

```

SELECT DEPARTMENT_ID,
COUNT(*) AS EMPLOYEE_COUNT FROM EMPLOYEE
GROUP BY DEPARTMENT_ID HAVING COUNT(*) >= 4

```

123 %

Results Messages

	DEPARTMENT_ID	EMPLOYEE_COUNT
1	30	4

6. How many employees joined in February month.

Query:

```

SELECT HIRE_DATE,EMPLOYEE_ID FROM EMPLOYEE GROUP BY
HIRE_DATE,EMPLOYEE_ID HAVING MONTH(HIRE_DATE)=2

```

SQLQuery1.sql - PR...THAKUR\pritu (66))\*

```

SELECT * FROM EMPLOYEE

```

```

SELECT HIRE_DATE,EMPLOYEE_ID FROM EMPLOYEE
GROUP BY HIRE_DATE,EMPLOYEE_ID
HAVING MONTH(HIRE_DATE)=2

```

123 %

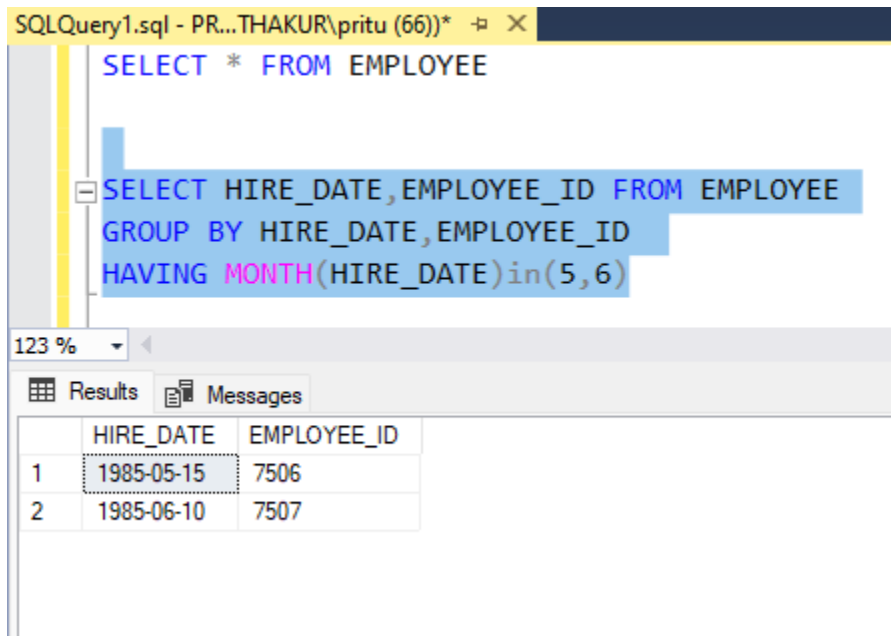
Results Messages

	HIRE_DATE	EMPLOYEE_ID
1	1984-02-20	7499
2	1985-02-22	7521

7. How many employees joined in May or June month.

Query:

```
SELECT HIRE_DATE,EMPLOYEE_ID FROM EMPLOYEE  
GROUP BY HIRE_DATE,EMPLOYEE_ID  
HAVING MONTH(HIRE_DATE)in(5,6)
```



The screenshot shows a SQL Developer window with a query editor and a results grid. The query editor contains the following SQL statement:

```
SELECT * FROM EMPLOYEE  
  
SELECT HIRE_DATE,EMPLOYEE_ID FROM EMPLOYEE  
GROUP BY HIRE_DATE,EMPLOYEE_ID  
HAVING MONTH(HIRE_DATE)in(5,6)
```

The results grid displays the output of the second query, showing two rows of employee data:

	HIRE_DATE	EMPLOYEE_ID
1	1985-05-15	7506
2	1985-06-10	7507

8. How many employees joined in 1985?

Query:

```
SELECT HIRE_DATE,EMPLOYEE_ID FROM EMPLOYEE  
GROUP BY HIRE_DATE,EMPLOYEE_ID  
HAVING YEAR(HIRE_DATE) = 1985
```

SQLQuery1.sql - PR...THAKUR\pritu (66))\* ✕

```
SELECT * FROM EMPLOYEE
```

```
SELECT HIRE_DATE,EMPLOYEE_ID FROM EMPLOYEE  
GROUP BY HIRE_DATE,EMPLOYEE_ID  
HAVING YEAR(HIRE_DATE) = 1985
```

123 %

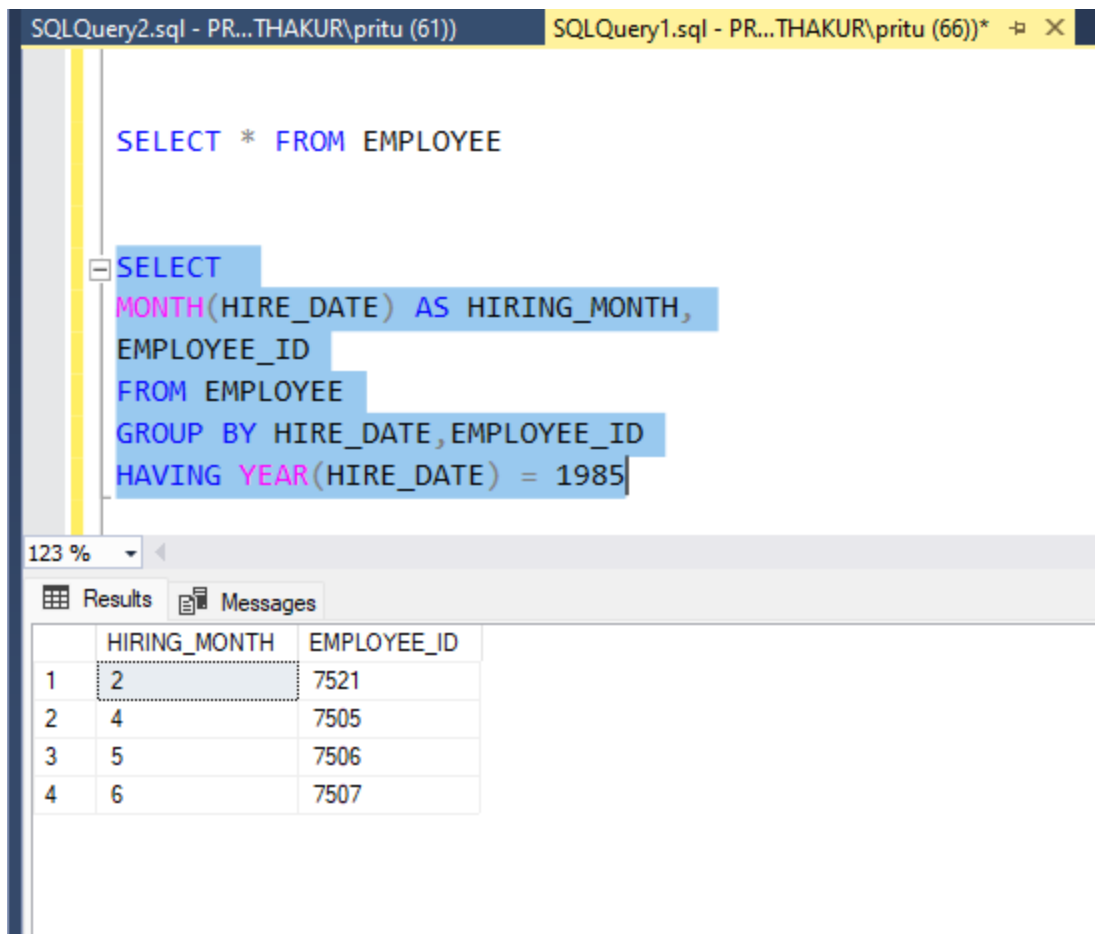
Results Messages

	HIRE_DATE	EMPLOYEE_ID
1	1985-02-22	7521
2	1985-04-04	7505
3	1985-05-15	7506
4	1985-06-10	7507

9. How many employees joined each month in 1985?

Query:

```
SELECT  
MONTH(HIRE_DATE) AS HIRING_MONTH,  
EMPLOYEE_ID  
FROM EMPLOYEE  
GROUP BY HIRE_DATE,EMPLOYEE_ID  
HAVING YEAR(HIRE_DATE) = 1985
```



```
SQLQuery2.sql - PR...THAKUR\pritu (61))  SQLQuery1.sql - PR...THAKUR\pritu (66))*  X
```

```
SELECT * FROM EMPLOYEE
```

```
SELECT  
MONTH(HIRE_DATE) AS HIRING_MONTH,  
EMPLOYEE_ID  
FROM EMPLOYEE  
GROUP BY HIRE_DATE, EMPLOYEE_ID  
HAVING YEAR(HIRE_DATE) = 1985
```

123 %

Results Messages

	HIRING_MONTH	EMPLOYEE_ID
1	2	7521
2	4	7505
3	5	7506
4	6	7507

10. How many employees were joined in April 1985?

Query:

```
SELECT COUNT(*) AS EmployeeHiredInApril85
```

```
FROM EMPLOYEE
```

```
GROUP BY YEAR(hire_date), MONTH(hire_date)
```

```
HAVING YEAR(hire_date) = 1985
```

```
AND MONTH(hire_date) = 4;
```

The screenshot shows a SQL Server Enterprise Manager window with two tabs: 'SQLQuery2.sql - PR...THAKUR\pritu (61)' and 'SQLQuery1.sql - PR...THAKUR\pritu (66)\*'. The active tab displays the following SQL query:

```
SELECT * FROM EMPLOYEE
SELECT COUNT(*) AS EmployeeHiredInApril85
FROM EMPLOYEE
GROUP BY YEAR(hire_date), MONTH(hire_date)
HAVING YEAR(hire_date) = 1985
AND MONTH(hire_date) = 4;
```

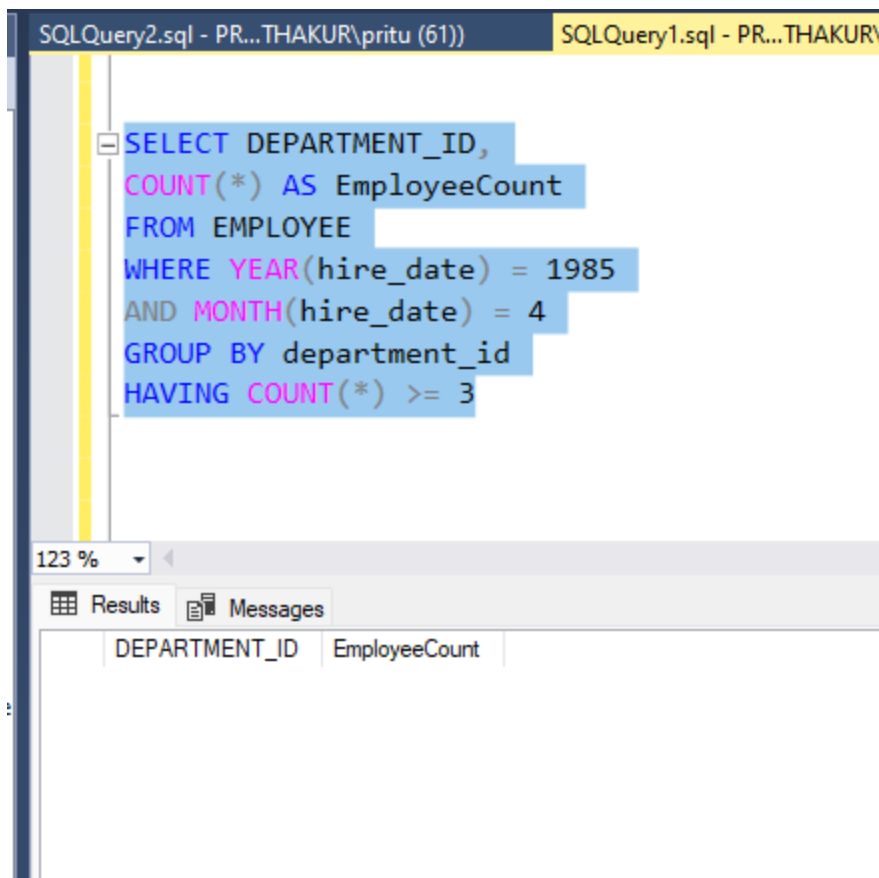
Below the query editor, the 'Results' tab is selected, showing a single row of data:

EmployeeHiredInApril85
1

11. Which is the Department ID having greater than or equal to 3 employees joining in April 1985?

Query:

```
SELECT DEPARTMENT_ID,
COUNT(*) AS EmployeeCount
FROM EMPLOYEE
WHERE YEAR(hire_date) = 1985
AND MONTH(hire_date) = 4
GROUP BY department_id
HAVING COUNT(*) >= 3
```



## Joins:

1. List out employees with their department names.

Query:

```
Select e.EMPLOYEE_ID,e.LAST_NAME,e.FIRST_NAME,e.JOB_ID,  
e.MANAGER_ID,e.SALARY,e.DEPARTMENT_ID ,d.Name from EMPLOYEE  
as e  
JOIN DEPARTMENT as d  
ON e.DEPARTMENT_ID=d.Department_Id
```

SQLQuery1.sql - PR...THAKUR\pritu (59))\* -> X

```

Select * from DEPARTMENT
Select * from EMPLOYEE

Select e.EMPLOYEE_ID,e.LAST_NAME,e.FIRST_NAME,e.JOB_ID,
e.MANAGER_ID,e.SALARY,e.DEPARTMENT_ID ,d.Name from EMPLOYEE as e
JOIN DEPARTMENT as d
ON e.DEPARTMENT_ID=d.Department_Id

```

123 %

Results Messages

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	JOB_ID	MANAGER_ID	SALARY	DEPARTMENT_ID	Name
1	7369	SMITH	JOHN	667	7902	800	20	Sales
2	7499	ALLEN	KEVIN	670	7698	1600	30	Research
3	7505	DOYLE	JEAN	671	7839	2850	30	Research
4	7506	DENNIS	LYNN	671	7839	2750	30	Research
5	7507	BAKER	LESLIE	671	7839	2200	40	Operations
6	7521	WARK	CYNTHIA	670	7698	1250	30	Research

2. Display employees with their designations.

Query:

```

Select e.EMPLOYEE_ID,e.LAST_NAME,e.FIRST_NAME,
e.JOB_ID,j.Designation from EMPLOYEE as e
JOIN JOB as j
ON e.JOB_ID=j.Job_ID

```

SQLQuery1.sql - PR...THAKUR\pritu (59))\*

```

Select * from DEPARTMENT
Select * from LOCATION

Select * from EMPLOYEE
Select * from JOB

Select e.EMPLOYEE_ID,e.LAST_NAME,e.FIRST_NAME,
e.JOB_ID,j.Designation from EMPLOYEE as e
JOIN JOB as j
ON e.JOB_ID=j.Job_ID

```

123 %

Results Messages

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	JOB_ID	Designation
1	7369	SMITH	JOHN	667	CLERK
2	7499	ALLEN	KEVIN	670	SALES_PERSON
3	7505	DOYLE	JEAN	671	MANAGER
4	7506	DENNIS	LYNN	671	MANAGER
5	7507	BAKER	LESLIE	671	MANAGER
6	7521	WARK	CYNTHIA	670	SALES_PERSON

3. Display the employees with their department names and city.

Query:

```

Select E.EMPLOYEE_ID,E.LAST_NAME,E.FIRST_NAME,
E.DEPARTMENT_ID,L.Location_ID,L.City,D.Name from EMPLOYEE as E
JOIN DEPARTMENT as D
ON E.DEPARTMENT_ID = D.Department_Id
Join LOCATION as L
on D.Location_Id =L.Location_ID

```



SQLQuery1.sql - PR...THAKUR\pritu (59))* - X							
<pre> Select * from JOB Select E.EMPLOYEE_ID,E.LAST_NAME,E.FIRST_NAME, E.DEPARTMENT_ID,L.Location_ID,L.City,D.Name from EMPLOYEE as E JOIN DEPARTMENT as D ON E.DEPARTMENT_ID = D.Department_Id Join LOCATION as L on D.Location_Id =L.Location_ID </pre>							
123 %							
Results Messages							
	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	DEPARTMENT_ID	Location_ID	City	Name
1	7369	SMITH	JOHN	20	124	Chicago	Sales
2	7499	ALLEN	KEVIN	30	123	Dallas	Research
3	7505	DOYLE	JEAN	30	123	Dallas	Research
4	7506	DENNIS	LYNN	30	123	Dallas	Research
5	7507	BAKER	LESLIE	40	167	Boston	Operations
6	7521	WARK	CYNTHIA	30	123	Dallas	Research

4. How many employees are working in different departments?  
 Display with department names.

Query:

Select Count(EMPLOYEE\_ID) as E\_Count,d.Name from EMPLOYEE as  
 e

join DEPARTMENT as d

on E.DEPARTMENT\_ID = d.Department\_Id

group by d.Name

SQLQuery1.sql - PR...THAKUR\pritu (59))\*

```

Select Count(EMPLOYEE_ID) as E_Count, d.Name from EMPLOYEE as e
join DEPARTMENT as d
on E.DEPARTMENT_ID = d.Department_Id
group by d.Name

```

123 %

Results Messages

	E_Count	Name
1	1	Operations
2	4	Research
3	1	Sales

5. How many employees are working in the sales department?

Query;

```

Select Count(e.EMPLOYEE_ID) as E_Count from EMPLOYEE as e
join DEPARTMENT as d
on E.DEPARTMENT_ID = d.Department_Id
where d.Name = 'Sales'

```

SQLQuery1.sql - PR...THAKUR\pritu (59))\*

```

Select Count(e.EMPLOYEE_ID) as E_Count from EMPLOYEE as e
join DEPARTMENT as d
on E.DEPARTMENT_ID = d.Department_Id
where d.Name = 'Sales'

```

123 %

Results Messages

	E_Count
1	1

6. Which is the department having greater than or equal to 3 employees and display the department names in ascending order.

Query:

```
Select Count(E.EMPLOYEE_ID) as Employee_Count,D.Name from  
Employee as E
```

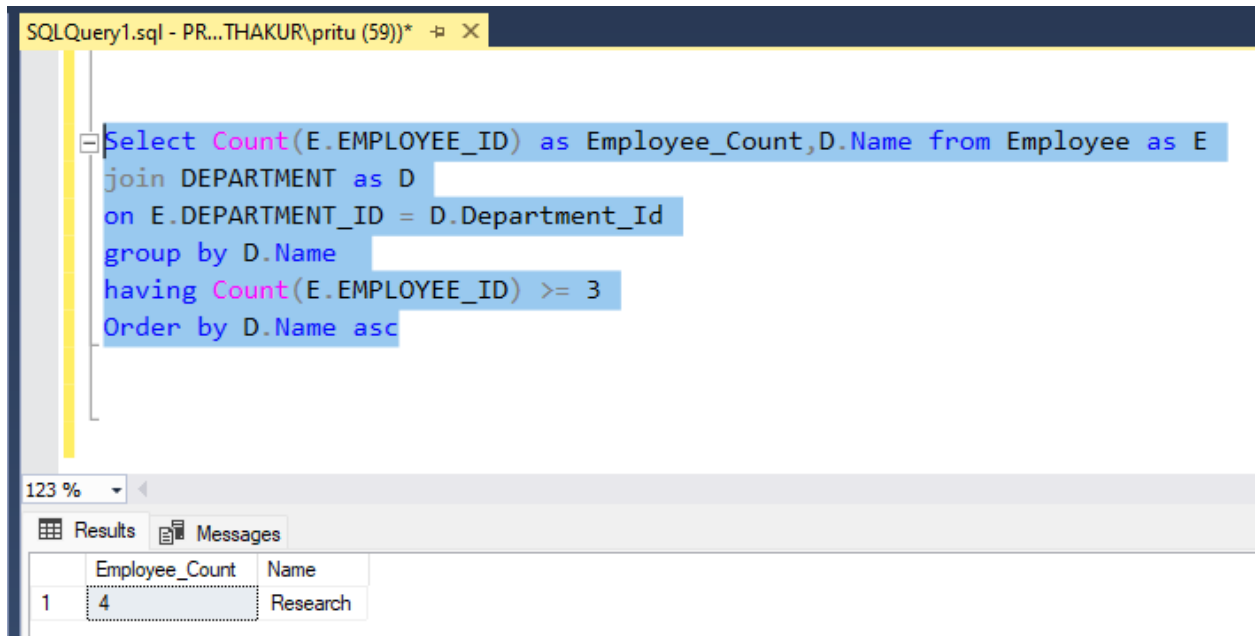
```
join DEPARTMENT as D
```

```
on E.DEPARTMENT_ID = D.Department_Id
```

```
group by D.Name
```

```
having Count(E.EMPLOYEE_ID) >= 3
```

```
Order by D.Name asc
```



The screenshot shows a SQL Developer window with a query editor and a results pane. The query editor contains the following SQL code:

```
Select Count(E.EMPLOYEE_ID) as Employee_Count,D.Name from Employee as E  
join DEPARTMENT as D  
on E.DEPARTMENT_ID = D.Department_Id  
group by D.Name  
having Count(E.EMPLOYEE_ID) >= 3  
Order by D.Name asc
```

The results pane shows a table with two columns: Employee\_Count and Name. The first row shows a count of 4 for the department named Research.

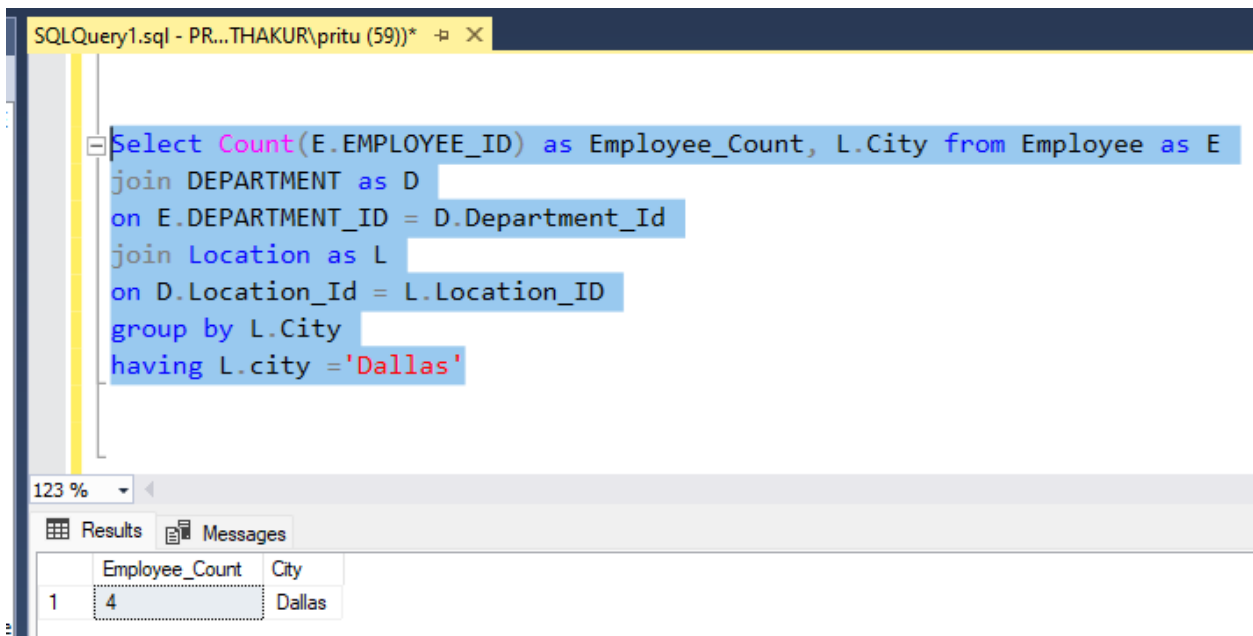
	Employee_Count	Name
1	4	Research

7. How many employees are working in 'Dallas'?

Query:

```
Select Count(E.EMPLOYEE_ID) as Employee_Count, L.City from  
Employee as E
```

```
join DEPARTMENT as D
on E.DEPARTMENT_ID = D.Department_Id
join Location as L
on D.Location_Id = L.Location_ID
group by L.City
having L.city ='Dallas'
```



The screenshot shows a SQL Developer window with a query titled 'SQLQuery1.sql - PR...THAKUR\pritu (59))'. The query is as follows:

```
Select Count(E.EMPLOYEE_ID) as Employee_Count, L.City from Employee as E
join DEPARTMENT as D
on E.DEPARTMENT_ID = D.Department_Id
join Location as L
on D.Location_Id = L.Location_ID
group by L.City
having L.city ='Dallas'
```

Below the query editor, the 'Results' tab is active, displaying a table with the following data:

	Employee_Count	City
1	4	Dallas

8. Display all employees in sales or operation departments.

Query:

```
Select E.EMPLOYEE_ID, E.FIRST_NAME,E.LAST_NAME,
E.JOB_ID,E.MANAGER_ID,E.SALARY, D.Name from Employee as E
join DEPARTMENT as D
on E.DEPARTMENT_ID = D.Department_Id
where D.Name in ('Sales','Operations')
```

SQLQuery1.sql - PR...THAKUR\pritu (59))\*

```

Select E.EMPLOYEE_ID, E.FIRST_NAME, E.LAST_NAME,
E.JOB_ID, E.MANAGER_ID, E.SALARY, D.Name from Employee as E
join DEPARTMENT as D
on E.DEPARTMENT_ID = D.Department_Id
where D.Name in ('Sales', 'Operations')

```

123 %

Results Messages

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	JOB_ID	MANAGER_ID	SALARY	Name
1	7369	JOHN	SMITH	667	7902	800	Sales
2	7507	LESLIE	BAKER	671	7839	2200	Operations

## CONDITIONAL STATEMENT

1. Display the employee details with salary grades. Use conditional statement to create a grade column.

Query:

Select EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, SALARY,

CASE

When Salary >= 2500 Then 'A'

When Salary >= 2000 Then 'B'

When Salary >= 1200 Then 'C'

When Salary >= 800 Then 'D'

Else 'E'

End as Salary\_Grade

from EMPLOYEE;

SQLQuery1.sql - PR...THAKUR\pritu (59))*					
<pre> Select EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY, CASE   When Salary &gt;= 2500 Then 'A'   When Salary &gt;= 2000 Then 'B'   When Salary &gt;= 1200 Then 'C'   When Salary &gt;= 800 Then 'D'   Else 'E' End as Salary_Grade from EMPLOYEE; </pre>					
123 %					
Results Messages					
	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	Salary_Grade
1	7369	JOHN	SMITH	800	D
2	7499	KEVIN	ALLEN	1600	C
3	7505	JEAN	DOYLE	2850	A
4	7506	LYNN	DENNIS	2750	A
5	7507	LESLIE	BAKER	2200	B
6	7521	CYNTHIA	WARK	1250	C

2. List out the number of employees grade wise. Use conditional statement to create a grade column.

Query: Select

CASE

When Salary >= 2500 Then 'A'

When Salary >= 2000 Then 'B'

When Salary >= 1200 Then 'C'

When Salary >= 800 Then 'D'

Else 'E'

End as Salary\_Grade,

```
COUNT(EMPLOYEE_ID) AS EMPLOYEE_COUNT
from EMPLOYEE
Group By
CASE
When Salary >= 2500 Then 'A'
When Salary >= 2000 Then 'B'
When Salary >= 1200 Then 'C'
When Salary >= 800 Then 'D'
Else 'E'
End
Order By Salary_Grade
```

SQLQuery2.sql - PR...THAKUR\pritu (62))\*    SQLQuery1.sql - PR...THA

```

Select
CASE
When Salary>= 2500 Then 'A'
When Salary>= 2000 Then 'B'
When Salary>= 1200 Then 'C'
When Salary>= 800 Then 'D'
Else 'E'
End as Salary_Grade,
COUNT(EMPLOYEE_ID) AS EMPLOYEE_COUNT
from EMPLOYEE
Group By
CASE
When Salary>= 2500 Then 'A'
When Salary>= 2000 Then 'B'
When Salary>= 1200 Then 'C'
When Salary>= 800 Then 'D'
Else 'E'
End
Order By Salary_Grade

```

123 %

Results    Messages

	Salary_Grade	EMPLOYEE_COUNT
1	A	2
2	B	1
3	C	2
4	D	1

3. Display the employee salary grades and the number of employees between 2000 to 5000 range of salary.

Query:

Select

CASE



When Salary >= 2500 Then 'A'

When Salary >= 2000 Then 'B'

When Salary >= 1200 Then 'C'

When Salary >= 800 Then 'D'

Else 'E'

End as Salary\_Grade,

COUNT(EMPLOYEE\_ID) AS EMPLOYEE\_COUNT

from EMPLOYEE WHERE SALARY BETWEEN 2000 AND 5000

Group By

CASE

When Salary >= 2500 Then 'A'

When Salary >= 2000 Then 'B'

When Salary >= 1200 Then 'C'

When Salary >= 800 Then 'D'

Else 'E'

End

Order By Salary\_Grade

SQLQuery2.sql - PR...THAKUR\pritu (62))\*    SQLQuery1.sql - PR...THAKUR\pritu (59))\*

```
Select
CASE
When Salary >= 2500 Then 'A'
When Salary >= 2000 Then 'B'
When Salary >= 1200 Then 'C'
When Salary >= 800 Then 'D'
Else 'E'
End as Salary_Grade,
COUNT(EMPLOYEE_ID) AS EMPLOYEE_COUNT
from EMPLOYEE WHERE SALARY BETWEEN 2000 AND 5000
Group By
CASE
When Salary >= 2500 Then 'A'
When Salary >= 2000 Then 'B'
When Salary >= 1200 Then 'C'
When Salary >= 800 Then 'D'
Else 'E'
End
Order By Salary_Grade
```

123 %

Results    Messages

	Salary_Grade	EMPLOYEE_COUNT
1	A	2
2	B	1

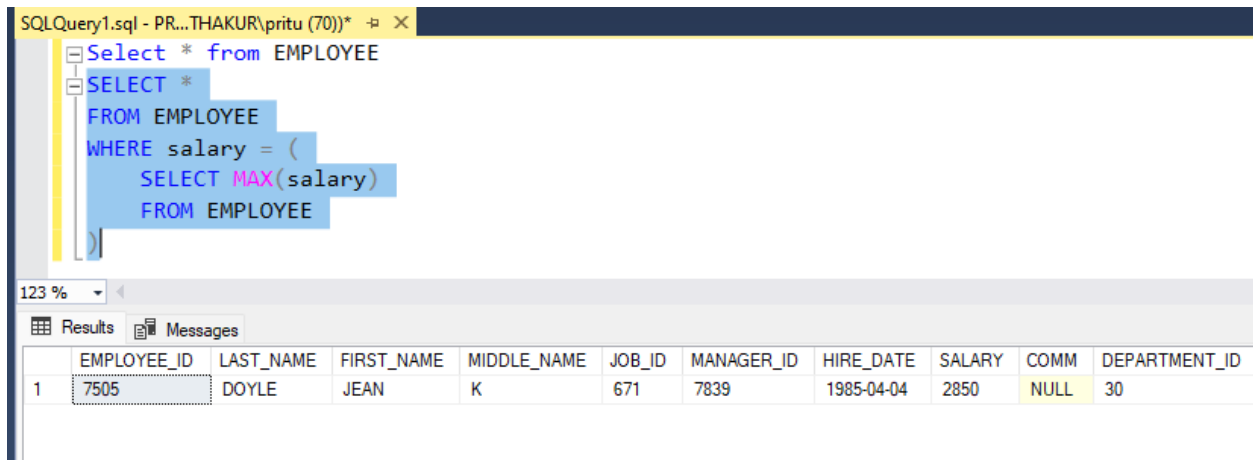
## Subqueries:

1. Display the employees list who got the maximum salary.

Query: SELECT \*

FROM EMPLOYEE

```
WHERE salary = (  
    SELECT MAX(salary)  
    FROM EMPLOYEE  
)
```



The screenshot shows a SQL Developer window with a query editor and a results grid. The query in the editor is:

```
Select * from EMPLOYEE  
SELECT *  
FROM EMPLOYEE  
WHERE salary = (  
    SELECT MAX(salary)  
    FROM EMPLOYEE  
)
```

The results grid shows one row of data:

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30

2. Display the employees who are working in the sales department.

```
Query: SELECT *  
FROM EMPLOYEE  
WHERE DEPARTMENT_ID = (  
    SELECT DEPARTMENT_ID  
    FROM DEPARTMENT  
    WHERE Name = 'sales'  
)
```

SQLQuery1.sql - PR...THAKUR\pritu (70))\*

```

Select * from EMPLOYEE
SELECT *
FROM EMPLOYEE
WHERE DEPARTMENT_ID = (
  SELECT DEPARTMENT_ID
  FROM DEPARTMENT
  WHERE Name = 'sales'
)

```

123 %

Results Messages

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20

3. Display the employees who are working as 'Clerk'.

Query: SELECT \*

FROM EMPLOYEE

WHERE JOB\_ID = (

SELECT JOB\_ID

FROM JOB

WHERE Designation = 'Clerk')

SQLQuery1.sql - PR...THAKUR\pritu (70))\*

```

Select * from EMPLOYEE
Select * from JOB
SELECT *
FROM EMPLOYEE
WHERE JOB_ID = (
  SELECT JOB_ID
  FROM JOB
  WHERE Designation = 'Clerk')

```

123 %

Results Messages

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	800	NULL	20

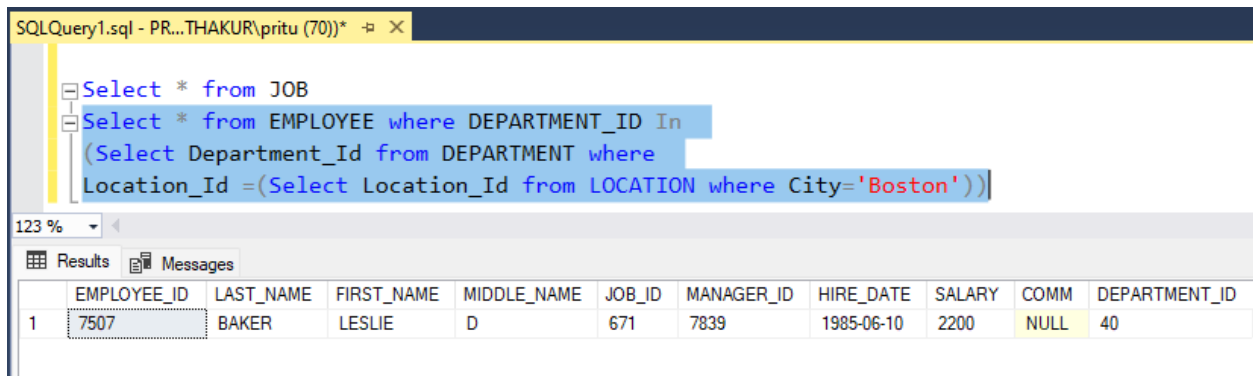
4. Display the list of employees who are living in 'Boston'.

Query:

Select \* from EMPLOYEE where DEPARTMENT\_ID In

(Select Department\_Id from DEPARTMENT where

Location\_Id =(Select Location\_Id from LOCATION where City='Boston'))



The screenshot shows a SQL Developer window with a query editor and a results grid. The query in the editor is:

```
Select * from JOB  
Select * from EMPLOYEE where DEPARTMENT_ID In  
(Select Department_Id from DEPARTMENT where  
Location_Id =(Select Location_Id from LOCATION where City='Boston'))
```

The results grid shows one row of data:

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40

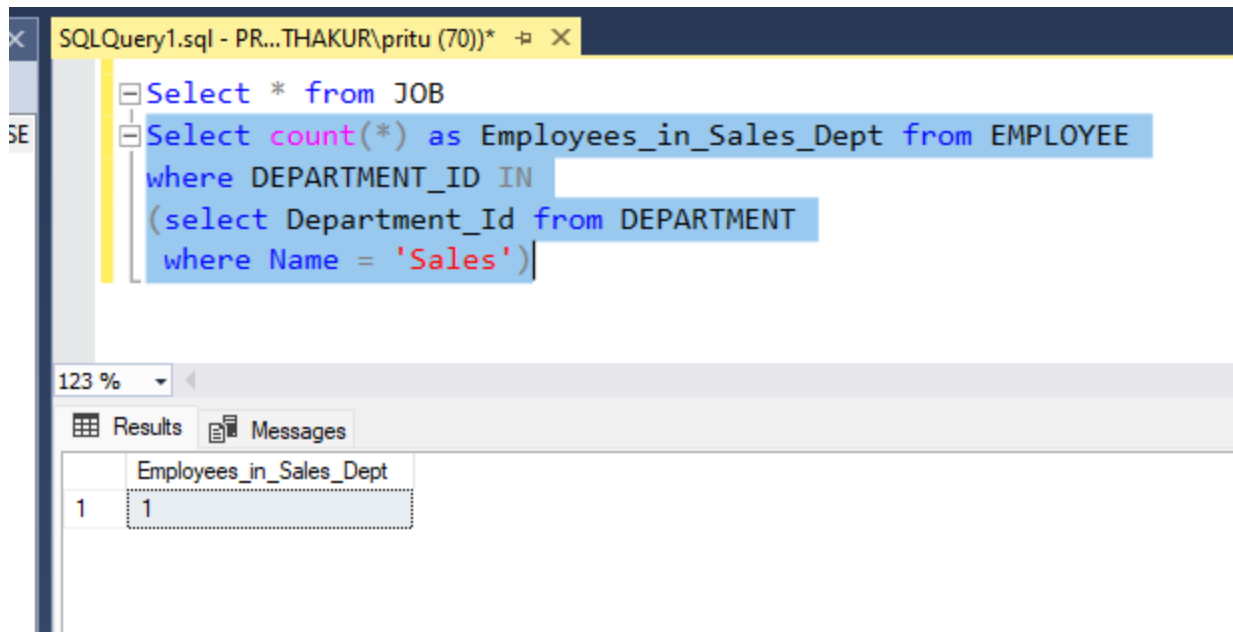
5. Find out the number of employees working in the sales department.

Query: Select count(\*) as Employees\_in\_Sales\_Dept from EMPLOYEE

where DEPARTMENT\_ID IN

(select Department\_Id from DEPARTMENT

where Name = 'Sales')



6. Update the salaries of employees who are working as clerks on the basis of 10%.

Query: UPDATE EMPLOYEE

SET Salary = Salary \* 1.10

Where Job\_Id IN

(Select J.Job\_ID from JOB J where Designation='Clerk')

Select \* from JOB

Select \* From EMPLOYEE

```
SQLQuery1.sql - PR...THAKUR\pritu (57))* ✕
UPDATE EMPLOYEE
SET Salary = Salary * 1.10
Where Job_Id IN
(Select J.Job_ID from JOB J where Designation='Clerk')
```

123 %

Messages

(1 row affected)

Completion time: 2025-01-06T17:38:31.8970013-06:00

SQLQuery1.sql - PR...THAKUR\pritu (57))\* ✕

```
UPDATE EMPLOYEE
SET Salary = Salary *1.10
Where Job_Id IN
(Select J.Job_ID from JOB J where Designation='Clerk')

Select * from JOB
Select * From EMPLOYEE
```

123 %

Results Messages

	Job_ID	Designation
1	667	CLERK
2	668	STAFF
3	669	ANALYST
4	670	SALES_PERSON
5	671	MANAGER
6	672	PRESIDENT

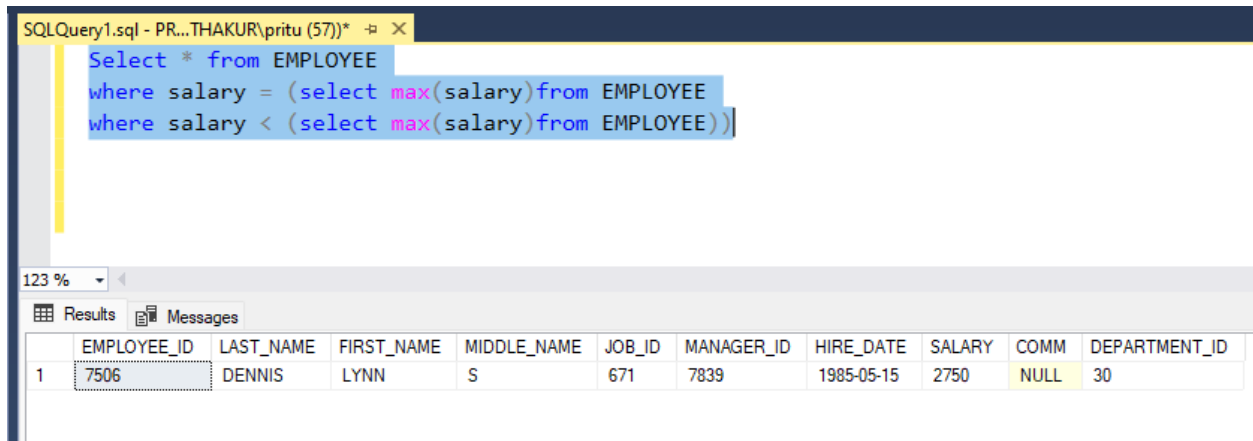
	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7369	SMITH	JOHN	Q	667	7902	1984-12-17	880	NULL	20
2	7499	ALLEN	KEVIN	J	670	7698	1984-02-20	1600	300	30
3	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
4	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
5	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40
6	7521	WARK	CYNTHIA	D	670	7698	1985-02-22	1250	500	30

7. Display the second highest salary drawing employee details.

Query: Select \* from EMPLOYEE

where salary = (select max(salary)from EMPLOYEE

where salary < (select max(salary)from EMPLOYEE))



The screenshot shows a SQL Developer window with a query editor at the top and a results grid below. The query editor contains the following SQL code:

```
Select * from EMPLOYEE  
where salary = (select max(salary)from EMPLOYEE  
where salary < (select max(salary)from EMPLOYEE))
```

The results grid shows a single row of data for employee ID 7506, Dennis Lynn Smith, with a salary of 2750.

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30

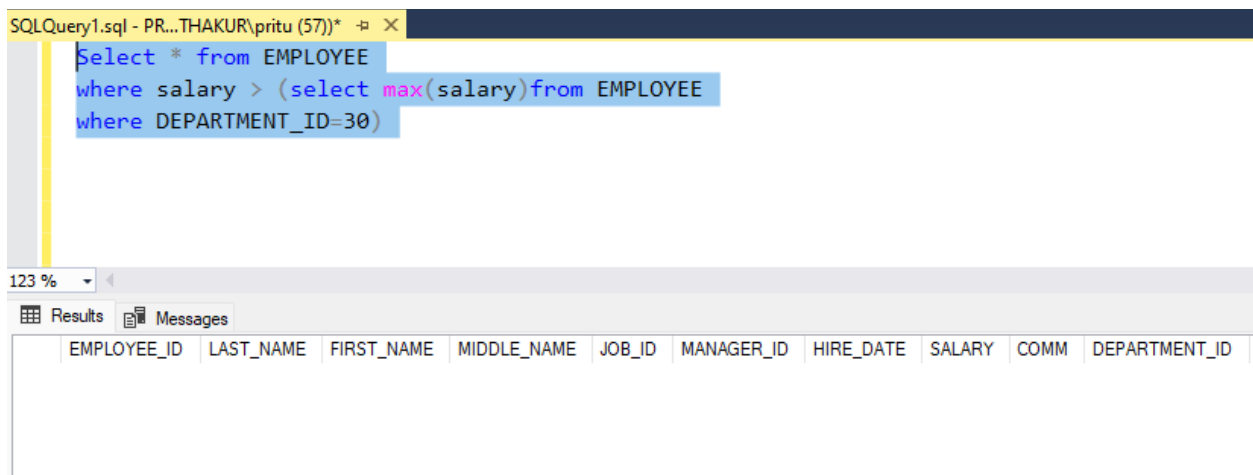
8. List out the employees who earn more than every employee in department 30.

Query:

Select \* from EMPLOYEE

where salary > (select max(salary)from EMPLOYEE

where DEPARTMENT\_ID=30)



The screenshot shows a SQL Developer window with a query editor at the top and a results grid below. The query editor contains the following SQL code:

```
Select * from EMPLOYEE  
where salary > (select max(salary)from EMPLOYEE  
where DEPARTMENT_ID=30)
```

The results grid is currently empty, showing only the column headers.

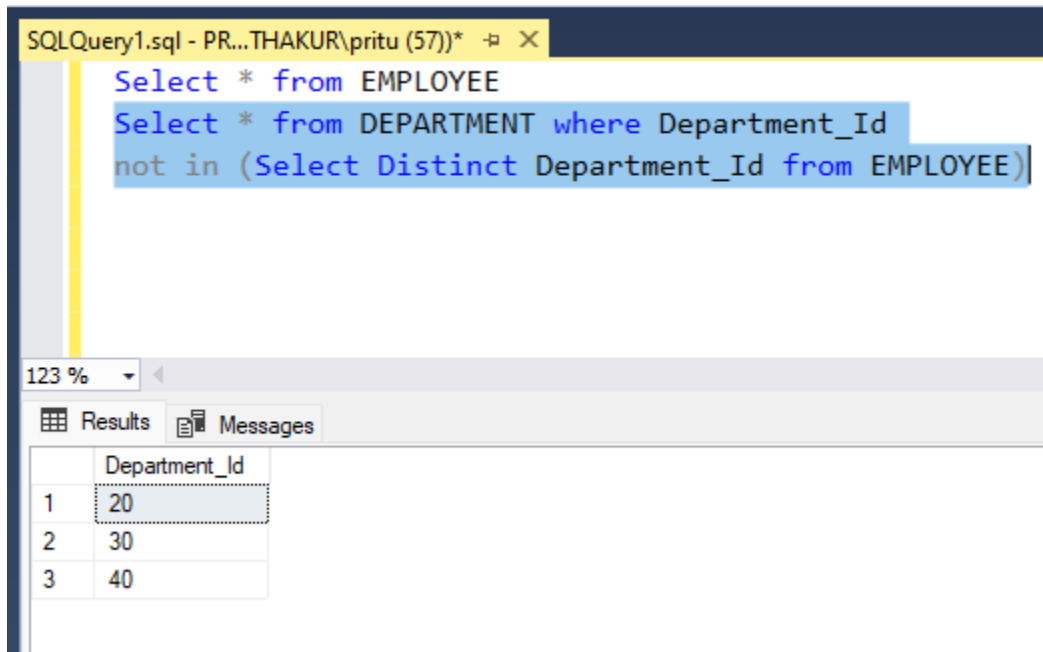
	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
--	-------------	-----------	------------	-------------	--------	------------	-----------	--------	------	---------------



9. Find out which department has no employees.

Query:

Select \* from DEPARTMENT where Department\_Id  
not in (Select Distinct Department\_Id from EMPLOYEE)



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor contains the following SQL code:

```
SQLQuery1.sql - PR...THAKUR\pritu (57))* ✕  
Select * from EMPLOYEE  
Select * from DEPARTMENT where Department_Id  
not in (Select Distinct Department_Id from EMPLOYEE)
```

The results pane shows a table with the following data:

	Department_Id
1	20
2	30
3	40

10. Find out the employees who earn greater than the average salary for their department.

Query:

Select \* from EMPLOYEE E where E.SALARY >  
(Select avg(e.SALARY) from EMPLOYEE e  
where e.DEPARTMENT\_ID = E.DEPARTMENT\_ID)

SQLQuery1.sql - PR...THAKUR\pritu (57))\* - ✕

```
Select * from DEPARTMENT
```

```
Select * from LOCATION
```

```
Select * from JOB
```

```
Select * from EMPLOYEE E where E.SALARY >  
(Select avg(e.SALARY) from EMPLOYEE e  
where e.DEPARTMENT_ID = E.DEPARTMENT_ID)
```

123 %

Results Messages

	EMPLOYEE_ID	LAST_NAME	FIRST_NAME	MIDDLE_NAME	JOB_ID	MANAGER_ID	HIRE_DATE	SALARY	COMM	DEPARTMENT_ID
1	7505	DOYLE	JEAN	K	671	7839	1985-04-04	2850	NULL	30
2	7506	DENNIS	LYNN	S	671	7839	1985-05-15	2750	NULL	30
3	7507	BAKER	LESLIE	D	671	7839	1985-06-10	2200	NULL	40