

DBMS Lab Assignment 5

Team 4

1. Illustrate logical ANY, ALL and LIKE operators. One query explaining the difference between ANY and ALL.

For ANY Operator:

Query:

USE University;

SELECT Department_Name

FROM T4_Department

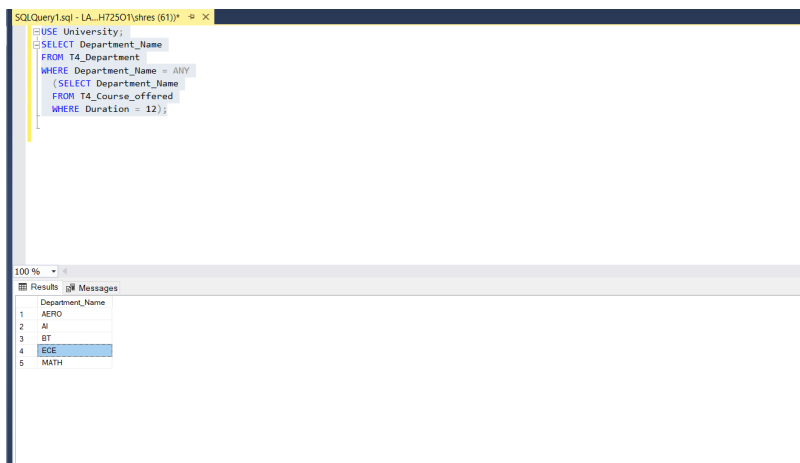
WHERE Department_Name = ANY

(SELECT Department_Name

FROM T4_Course_offered

WHERE Duration = 12);

OUTPUT:



The screenshot shows the SQL Developer interface. The top pane displays the following SQL query:

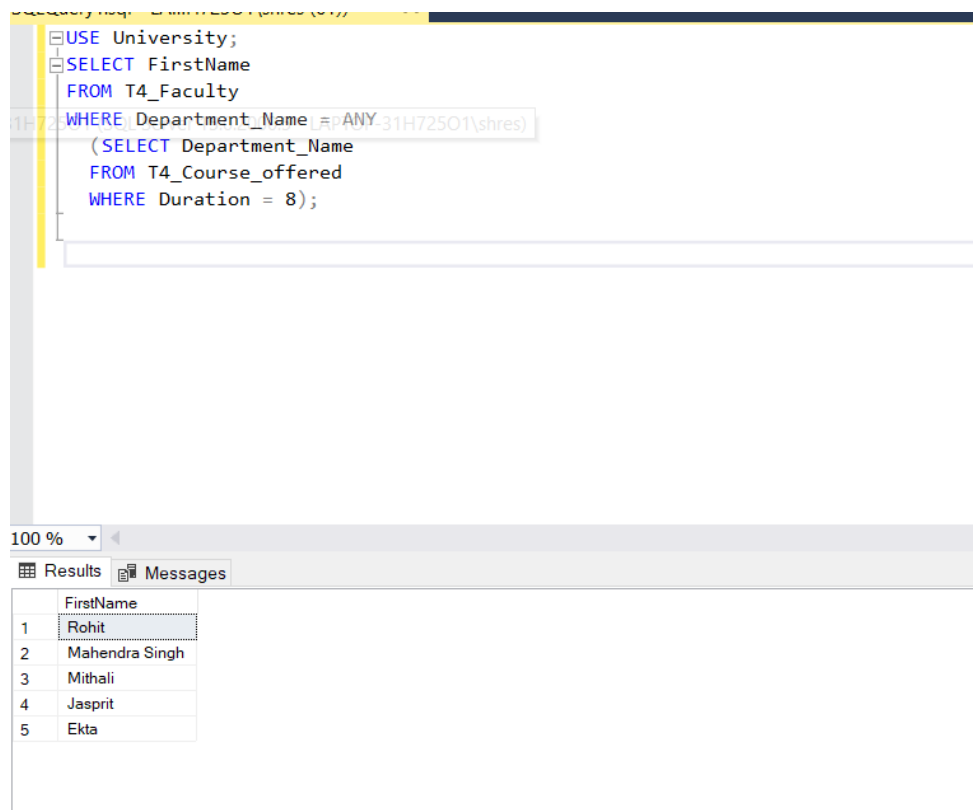
```
USE University;
SELECT Department_Name
FROM T4_Department
WHERE Department_Name = ANY
  (SELECT Department_Name
   FROM T4_Course_offered
   WHERE Duration = 12);
```

The bottom pane shows the results of the query in a table with the following data:

Department_Name
AERO
AI
BT
EDGE
MATH

Query:

```
USE University;  
  
SELECT FirstName  
  
FROM T4_Faculty  
  
WHERE Department_Name = ANY  
  
    (SELECT Department_Name  
  
    FROM T4_Course_offered  
  
    WHERE Duration = 8);
```

Output:

The screenshot shows a SQL Server Enterprise Manager interface. The top pane displays a query script with the following text:

```
USE University;  
  
SELECT FirstName  
  
FROM T4_Faculty  
  
WHERE Department_Name = ANY  
  
    (SELECT Department_Name  
  
    FROM T4_Course_offered  
  
    WHERE Duration = 8);
```

The bottom pane shows the results of the query. It includes a 'Results' tab and a 'Messages' tab. The 'Results' tab is active, displaying a table with the following data:

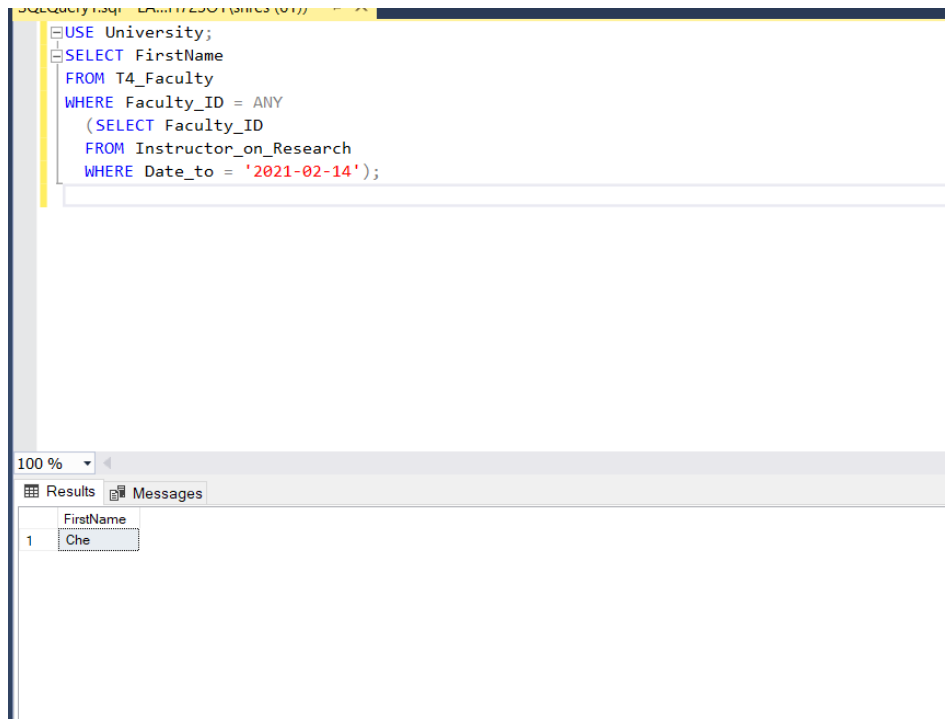
	FirstName
1	Rohit
2	Mahendra Singh
3	Mithali
4	Jasprit
5	Ekta

Query:

```
USE University;  
  
SELECT FirstName  
  
FROM T4_Faculty
```

```
WHERE Faculty_ID = ANY  
  
(SELECT Faculty_ID  
  
FROM Instructor_on_Research  
  
WHERE Date_to = '2021-02-14');
```

Output:



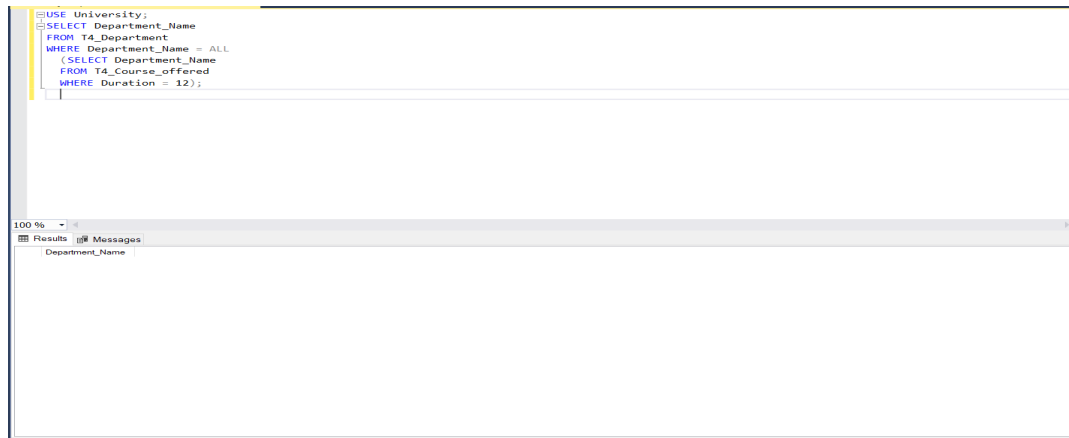
```
USE University;  
SELECT FirstName  
FROM T4_Faculty  
WHERE Faculty_ID = ANY  
      (SELECT Faculty_ID  
       FROM Instructor_on_Research  
       WHERE Date_to = '2021-02-14');
```

Results	Messages
1	Che

For ALL operator

```
USE University;  
  
SELECT Department_Name  
  
FROM T4_Department  
  
WHERE Department_Name = ALL  
  
(SELECT Department_Name  
  
FROM T4_Course_offered  
  
WHERE Duration = 12);
```

Output:

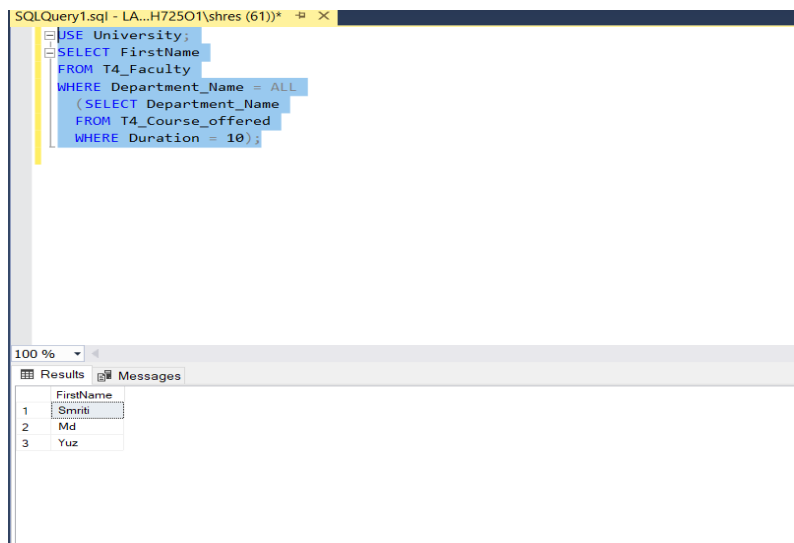


Query :

```
USE University;

SELECT FirstName
FROM T4_Faculty
WHERE Department_Name = ALL
    (SELECT Department_Name
     FROM T4_Course_offered
     WHERE Duration = 10);
```

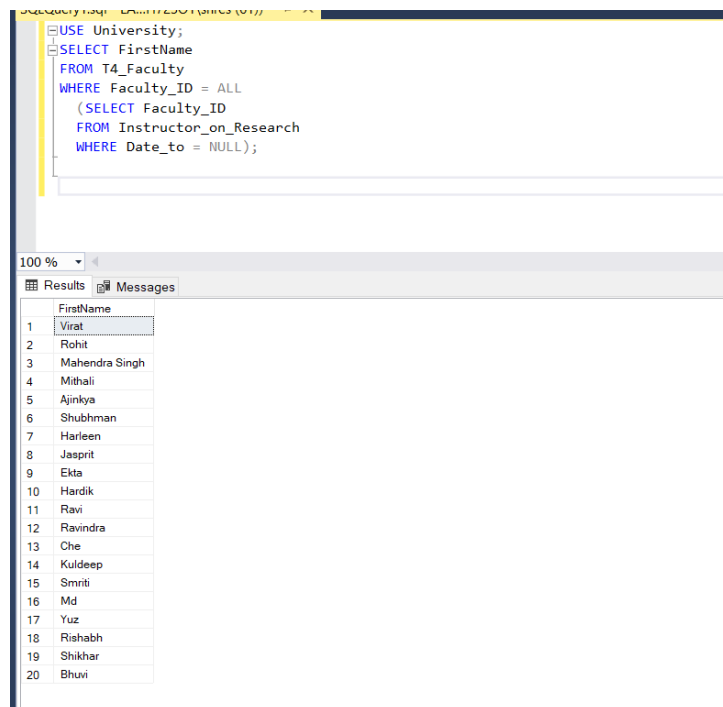
Output:



Query:

```
USE University;  
  
SELECT FirstName  
  
FROM T4_Faculty  
  
WHERE Faculty_ID = ALL  
  
    (SELECT Faculty_ID  
  
    FROM Instructor_on_Research  
  
    WHERE Date_to = NULL);
```

Output:



100 %

Results Messages

	FirstName
1	Virat
2	Rohit
3	Mahendra Singh
4	Mithali
5	Ajinkya
6	Shubhman
7	Harleen
8	Jasprit
9	Ekta
10	Hardik
11	Ravi
12	Ravindra
13	Che
14	Kuldeep
15	Smriti
16	Md
17	Yuz
18	Rishabh
19	Shikhar
20	Bhuvn

For LIKE Operator :

Query:

```
USE University;  
  
SELECT FirstName  
  
FROM T4_Faculty  
  
WHERE FirstName LIKE 'm%';
```

Output:

```
USE University;
SELECT FirstName
FROM T4_Faculty
WHERE FirstName LIKE 'm%';
```

100 %

	FirstName
1	Mahendra Singh
2	Mithali
3	Md

Query:

USE University;

SELECT FirstName

FROM T4_Faculty

WHERE FirstName LIKE '%t';

Output:

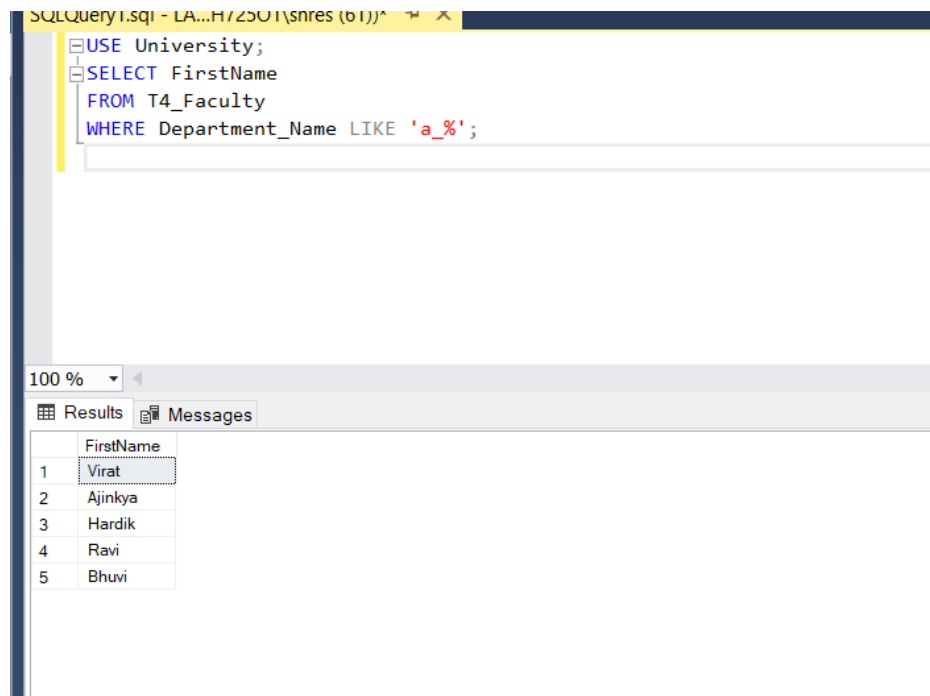
```
USE University;
SELECT FirstName
FROM T4_Faculty
WHERE FirstName LIKE '%t';
```

100 %

	FirstName
1	Virat
2	Rohit
3	Jasprit

Query :

```
USE University;  
  
SELECT FirstName  
  
FROM T4_Faculty  
  
WHERE Department_Name LIKE 'a_%';
```

Output:

The screenshot shows a SQL query window with the following text:

```
USE University;  
SELECT FirstName  
FROM T4_Faculty  
WHERE Department_Name LIKE 'a_%';
```

Below the query window, the 'Results' tab is active, displaying a table with 5 rows and 1 column (FirstName). The data is as follows:

	FirstName
1	Virat
2	Ajinkya
3	Hardik
4	Ravi
5	Bhuv

Difference between ANY and ALL operator:

From the above queries of ANY and ALL it is clear that :

- ALL returns TRUE if ALL of the subquery values meet the condition.
- ANY returns TRUE if ANY of the subquery values meet the condition.

2. Query for each Aggregate function.**Query:**

```
/* creating new table with Salary values for Faculty for aggregate functions computations */
```

```
CREATE TABLE T4_Faculty_Salary
(
    Faculty_ID INT PRIMARY KEY FOREIGN KEY REFERENCES T4_Faculty(Faculty_ID) NOT
    NULL ,
    Salary INT
);
```

```
INSERT INTO T4_Faculty_Salary
```

```
VALUES
```

```
(100, 120000),
```

```
(101, 100000),
```

```
(102, 125000),
```

```
(103, 100000),
```

```
(104, 90000),
```

```
(105, 100000),
```

```
(106, 80000),
```

```
(107, 90000),
```

```
(108, 70000),
```

```
(109, 75000),
```

```
(110, 85000),
```

```
(111, 90000),
```

```
(112, 100000),
```

```
(113, 75000),
```

```
(114, 95000),
```

```
(115, 75000),
```

```
(116, NULL),
```

```
(117, 85000),
```



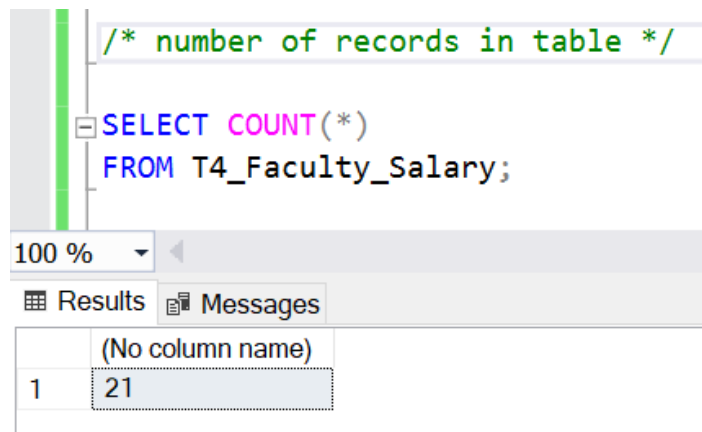
```
(118, 90000),  
(119, 90000),  
(120, 95000)  
;
```

i) COUNT commands

/* number of records in table */

```
SELECT COUNT(*)  
FROM T4_Faculty_Salary;
```

Output:



```
/* number of records in table */  
  
SELECT COUNT(*)  
FROM T4_Faculty_Salary;
```

100 %

Results Messages

	(No column name)
1	21

/* number of values in Salary column */

```
SELECT COUNT(Salary)  
FROM T4_Faculty_Salary;
```

Output:

```

/* number of values in Salary column */
SELECT COUNT(Salary)
FROM T4_Faculty_Salary;

```

100 %	
Results	Messages
	(No column name)
1	20

/* number of distinct Salary values */

```
SELECT COUNT(DISTINCT Salary)
```

```
FROM T4_Faculty_Salary;
```

Output:

```

/* number of distinct Salary values */
SELECT COUNT(DISTINCT Salary)
FROM T4_Faculty_Salary;

```

00 %	
Results	Messages
	(No column name)
1	9

ii) SUM commands

/* Sum of all salaries*/

```
SELECT SUM(Salary)
```

```
FROM T4_Faculty_Salary;
```

Output:

<pre> /* Sum of all salaries*/ SELECT SUM(Salary) FROM T4_Faculty_Salary; </pre>	
.00 %	
Results	Messages
	(No column name)
1	1830000

```

/* Sum of distinct salaries*/
SELECT SUM(DISTINCT Salary)
FROM T4_Faculty_Salary;

```

Output:

<pre> /* Sum of distinct salaries*/ SELECT SUM(DISTINCT Salary) FROM T4_Faculty_Salary; </pre>	
00 %	
Results	Messages
	(No column name)
1	840000

iii) AVG commands

```

SELECT AVG(Salary)
FROM T4_Faculty_Salary;

```

Output:

/* average of all salaries*/	
SELECT AVG(Salary)	
FROM T4_Faculty_Salary;	
100 %	
Results	Messages
	(No column name)
1	91500

/* average of specified salary */

```
SELECT AVG(Salary)
FROM T4_Faculty_Salary
WHERE Salary>90000;
```

Output:

/* average of specified salary */	
SELECT AVG(Salary)	
FROM T4_Faculty_Salary	
WHERE Salary>90000;	
00 %	
Results	Messages
	(No column name)
1	104375

iv) MAX command

```
SELECT MAX(Salary)
FROM T4_Faculty_Salary;
```

Output:

The screenshot shows a SQL query editor with the following text:

```
SELECT MAX(Salary)
FROM T4_Faculty_Salary;
```

Below the query, there is a 'Results' tab and a 'Messages' tab. The 'Results' tab is active, showing a table with one row and one column. The column header is '(No column name)' and the value is 125000.

	(No column name)
1	125000

v) MIN command

```
SELECT MIN(Salary)
FROM T4_Faculty_Salary;
```

Output:

The screenshot shows a SQL query editor with the following text:

```
SELECT MIN(Salary)
FROM T4_Faculty_Salary;
```

Below the query, there is a 'Results' tab and a 'Messages' tab. The 'Results' tab is active, showing a table with one row and one column. The column header is '(No column name)' and the value is 70000.

	(No column name)
1	70000

3. Illustrate the usage of order by, group by and having clause .

Solution:

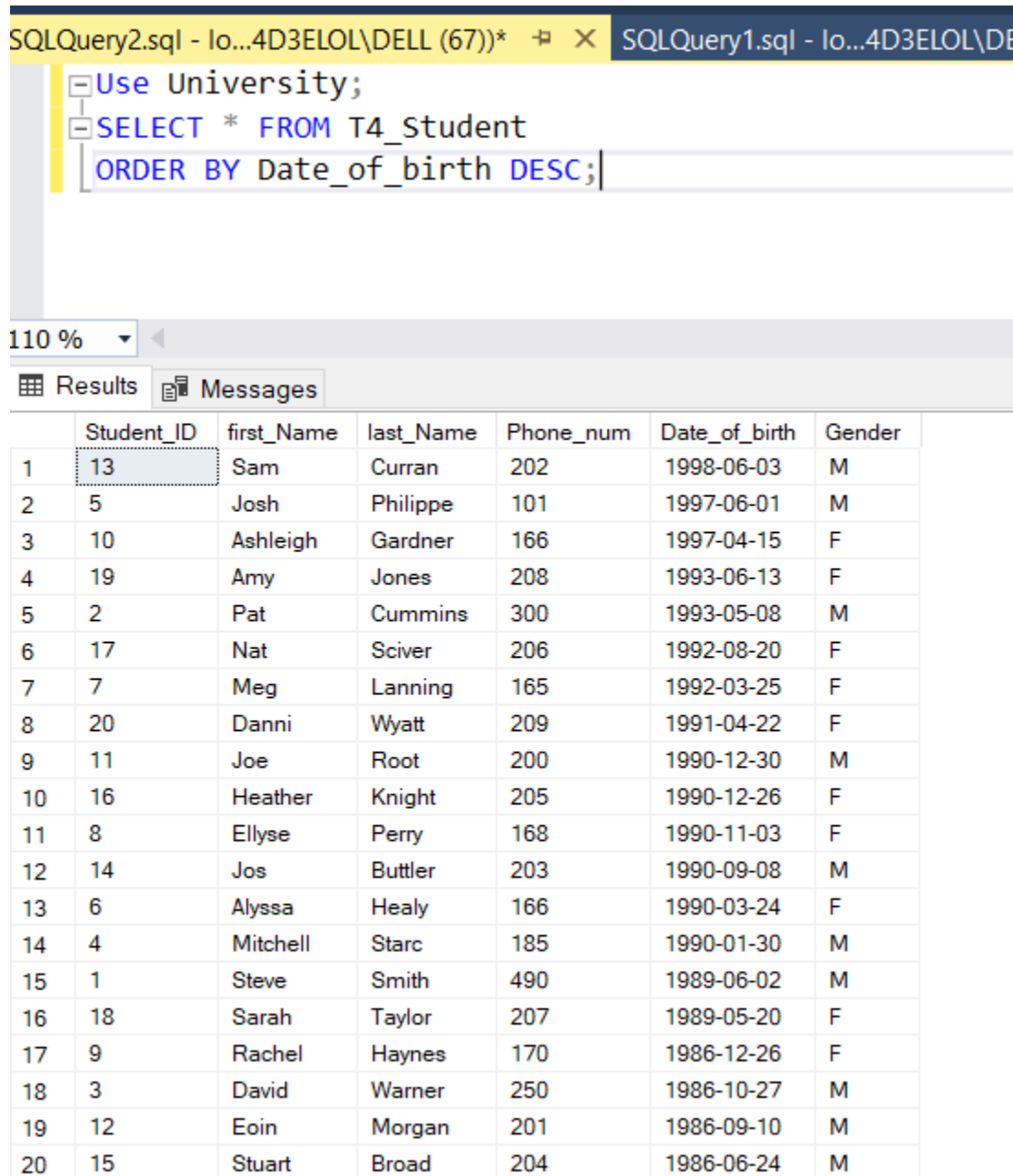
a) Usage of ORDER BY:

Query:

Use University;

```
SELECT * FROM T4_Student
ORDER BY Date_of_birth DESC;
```

Output:



The screenshot shows a SQL Server Enterprise Manager window with two tabs: 'SQLQuery2.sql' and 'SQLQuery1.sql'. The 'SQLQuery2.sql' tab is active, displaying the following SQL query:

```
Use University;  
SELECT * FROM T4_Student  
ORDER BY Date_of_birth DESC;
```

Below the query editor, the 'Results' tab is selected, showing a table with 20 rows of student data. The table has the following columns: Student_ID, first_Name, last_Name, Phone_num, Date_of_birth, and Gender. The data is sorted by Date_of_birth in descending order.

	Student_ID	first_Name	last_Name	Phone_num	Date_of_birth	Gender
1	13	Sam	Curran	202	1998-06-03	M
2	5	Josh	Philippe	101	1997-06-01	M
3	10	Ashleigh	Gardner	166	1997-04-15	F
4	19	Amy	Jones	208	1993-06-13	F
5	2	Pat	Cummins	300	1993-05-08	M
6	17	Nat	Sciver	206	1992-08-20	F
7	7	Meg	Lanning	165	1992-03-25	F
8	20	Danni	Wyatt	209	1991-04-22	F
9	11	Joe	Root	200	1990-12-30	M
10	16	Heather	Knight	205	1990-12-26	F
11	8	Ellyse	Perry	168	1990-11-03	F
12	14	Jos	Buttler	203	1990-09-08	M
13	6	Alyssa	Healy	166	1990-03-24	F
14	4	Mitchell	Starc	185	1990-01-30	M
15	1	Steve	Smith	490	1989-06-02	M
16	18	Sarah	Taylor	207	1989-05-20	F
17	9	Rachel	Haynes	170	1986-12-26	F
18	3	David	Warner	250	1986-10-27	M
19	12	Eoin	Morgan	201	1986-09-10	M
20	15	Stuart	Broad	204	1986-06-24	M

Query:

Use University;

SELECT * FROM T4_Student

ORDER BY first_Name DESC,Student_ID ASC;

Output:

SQLQuery2.sql - lo...4D3EOL\DELL (67))* X SQLQuery1.sql - lo...4D3EOL\DEL

[-] Use University;

[-] SELECT * FROM T4_Student

ORDER BY first_Name DESC, Student_ID ASC;

110 %

Results

Messages

	Student_ID	first_Name	last_Name	Phone_num	Date_of_birth	Gender
1	15	Stuart	Broad	204	1986-06-24	M
2	1	Steve	Smith	490	1989-06-02	M
3	18	Sarah	Taylor	207	1989-05-20	F
4	13	Sam	Curran	202	1998-06-03	M
5	9	Rachel	Haynes	170	1986-12-26	F
6	2	Pat	Cummins	300	1993-05-08	M
7	17	Nat	Sciver	206	1992-08-20	F
8	4	Mitchell	Starc	185	1990-01-30	M
9	7	Meg	Lanning	165	1992-03-25	F
10	5	Josh	Philippe	101	1997-06-01	M
11	14	Jos	Buttler	203	1990-09-08	M
12	11	Joe	Root	200	1990-12-30	M
13	16	Heather	Knight	205	1990-12-26	F
14	12	Eoin	Morgan	201	1986-09-10	M
15	8	Ellyse	Perry	168	1990-11-03	F
16	3	David	Warner	250	1986-10-27	M
17	20	Danni	Wyatt	209	1991-04-22	F
18	10	Ashleigh	Gardner	166	1997-04-15	F
19	19	Amy	Jones	208	1993-06-13	F
20	6	Alyssa	Healy	166	1990-03-24	F

b)Usage of GROUP BY and having clause:

Query:

Use University;

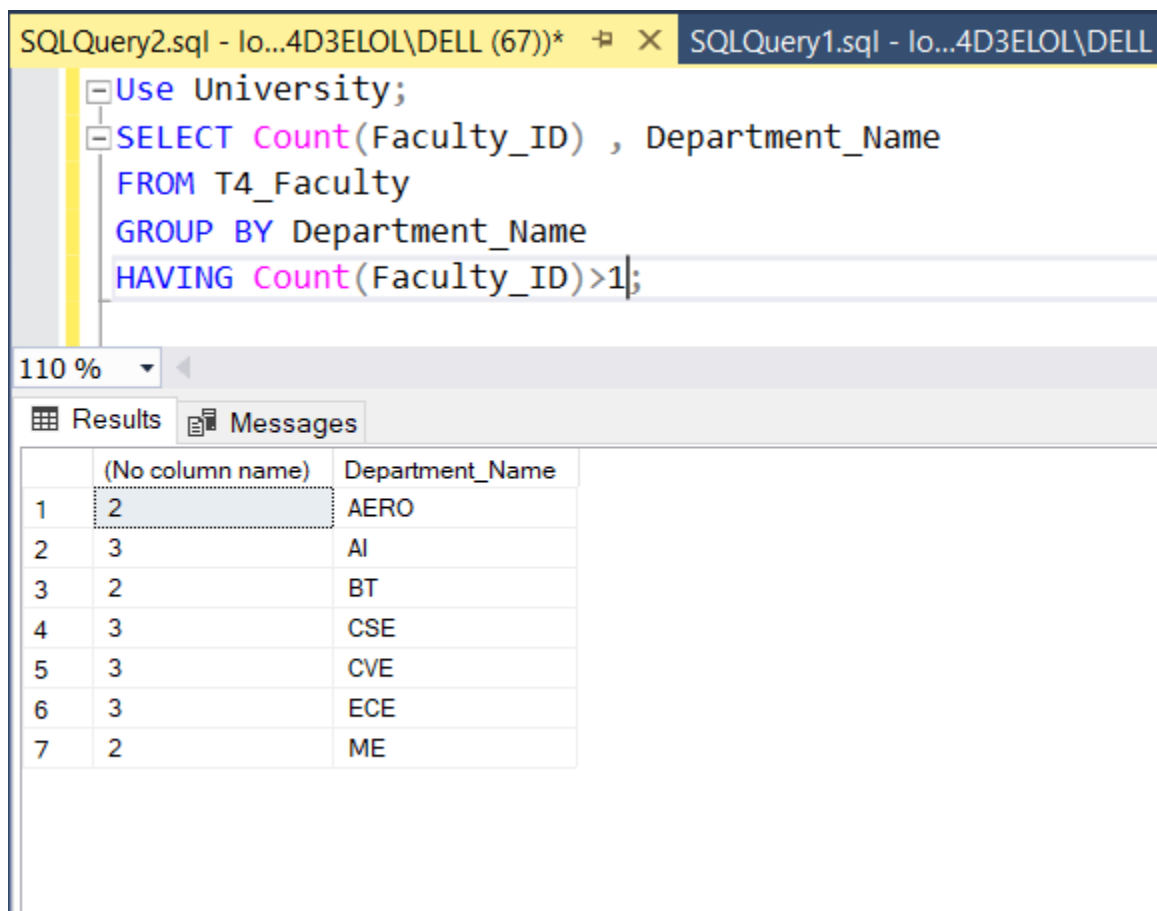
SELECT Count(Faculty_ID) , Department_Name

FROM T4_Faculty

GROUP BY Department_Name

HAVING Count(Faculty_ID)>1;

Output:



The screenshot shows a SQL Server Enterprise Manager window with two tabs: 'SQLQuery2.sql' and 'SQLQuery1.sql'. The 'SQLQuery2.sql' tab is active, displaying the following SQL query:

```
Use University;  
SELECT Count(Faculty_ID) , Department_Name  
FROM T4_Faculty  
GROUP BY Department_Name  
HAVING Count(Faculty_ID)>1;
```

Below the query editor, the 'Results' tab is selected, showing the output of the query. The results are displayed in a table with two columns: '(No column name)' and 'Department_Name'. The table contains 7 rows of data.

	(No column name)	Department_Name
1	2	AERO
2	3	AI
3	2	BT
4	3	CSE
5	3	CVE
6	3	ECE
7	2	ME

Query:

Use University;

```
SELECT Count(Course_name) As Number_of_Courses , Department_Name
```

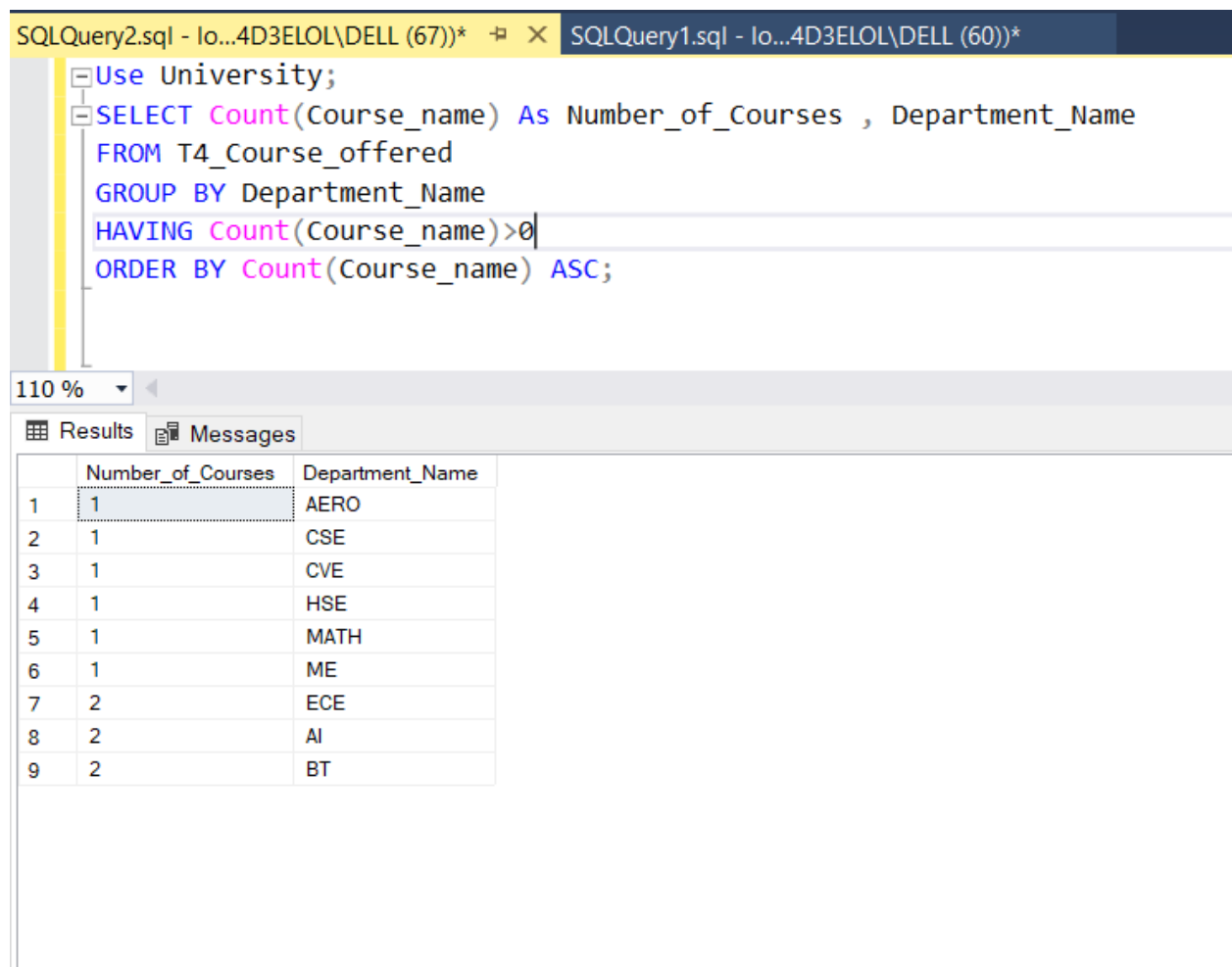
```
FROM T4_Course_offered
```

```
GROUP BY Department_Name
```

```
HAVING Count(Course_name)>0
```

```
ORDER BY Count(Course_name) ASC;
```

Output:



The screenshot shows a SQL Server Enterprise Manager window with two tabs: 'SQLQuery2.sql' and 'SQLQuery1.sql'. The 'SQLQuery1.sql' tab is active, displaying the following SQL query:

```
Use University;
SELECT Count(Course_name) As Number_of_Courses , Department_Name
FROM T4_Course_offered
GROUP BY Department_Name
HAVING Count(Course_name)>0
ORDER BY Count(Course_name) ASC;
```

Below the query editor, the 'Results' tab is selected, showing the output of the query. The results are displayed in a table with two columns: 'Number_of_Courses' and 'Department_Name'. The table contains 9 rows of data, sorted by 'Number_of_Courses' in ascending order.

	Number_of_Courses	Department_Name
1	1	AERO
2	1	CSE
3	1	CVE
4	1	HSE
5	1	MATH
6	1	ME
7	2	ECE
8	2	AI
9	2	BT

4. Use Aggregate function with group by and having

a)

Query:

```
SELECT Faculty_ID, AVG(Salary)
FROM T4_Faculty_Salary
GROUP BY Faculty_ID
HAVING AVG(Salary)>80000
```

Output:

Results Messages		
	Faculty_ID	(No column name)
1	100	120000
2	101	100000
3	102	125000
4	103	100000
5	104	90000
6	105	100000
7	107	90000
8	110	85000
9	111	90000
10	112	100000
11	114	95000
12	117	85000
13	118	90000
14	119	90000

b)

Query:

```
SELECT Faculty_ID,SUM(Salary)
FROM T4_Faculty_Salary
GROUP BY Faculty_ID
```

HAVING SUM(Salary)>100000

Output:

Results Messages		
	Faculty_ID	(No column name)
1	100	120000
2	102	125000

c)

Query:

```
SELECT Faculty_ID,MAX(Salary)
FROM T4_Faculty_Salary
GROUP BY Faculty_ID
HAVING Faculty_ID>110
```

Output:

Results Messages		
	Faculty_ID	(No column name)
1	111	90000
2	112	100000
3	113	75000
4	114	95000
5	115	75000
6	116	NULL
7	117	85000
8	118	90000
9	119	90000

d)

Query:

```
SELECT Faculty_ID,MIN(Salary)
FROM T4_Faculty_Salary
GROUP BY Faculty_ID,Salary
```

HAVING Salary<100000

Output:

Results Messages		
	Faculty_ID	(No column name)
1	104	90000
2	106	80000
3	107	90000
4	108	70000
5	109	75000
6	110	85000
7	111	90000
8	113	75000
9	114	95000
10	115	75000
11	117	85000
12	118	90000
13	119	90000

e)

Query:

```
SELECT Faculty_ID,COUNT(Salary)
FROM T4_Faculty_Salary
GROUP BY Faculty_ID,Salary
HAVING COUNT(Salary)<1
```

Output:

Results Messages		
	Faculty_ID	(No column name)
1	116	0

5. Write at least 3 nested queries using order by, group by and having clause.

QUERY:

```
SELECT Faculty_ID, FirstName, Department_Name FROM T4_Faculty WHERE Faculty_ID = ANY(
SELECT Faculty_ID FROM T4_Faculty WHERE Department_Name ="CSE") ORDER BY FirstName;
```

```
SELECT COUNT(Faculty_ID), Department_Name FROM T4_Faculty WHERE Faculty_ID=ANY (
SELECT Faculty_ID FROM T4_Faculty WHERE Department_Name ="CSE") GROUP BY
Department_Name;
```

```
SELECT COUNT(Faculty_ID), Department_Name FROM T4_Faculty WHERE Faculty_ID=ANY(
SELECT Faculty_ID FROM T4_Faculty WHERE Department_Name ="CSE") GROUP BY
Department_Name HAVING COUNT(Faculty_ID) > 3;
```

OUTPUT:



```
SELECT Faculty_ID, FirstName, Department_Name FROM T4_Faculty WHERE Faculty_ID = ANY( SELECT Faculty_ID FROM T4_Faculty WHERE Department_Name ='CSE') ORDER BY FirstName;
SELECT COUNT(Faculty_ID), Department_Name FROM T4_Faculty WHERE Faculty_ID=ANY ( SELECT Faculty_ID FROM T4_Faculty WHERE Department_Name ='CSE') GROUP BY Department_Name;
SELECT COUNT(Faculty_ID), Department_Name FROM T4_Faculty WHERE Faculty_ID=ANY( SELECT Faculty_ID FROM T4_Faculty WHERE Department_Name ='AI') GROUP BY Department_Name HAVING COUNT(Faculty_ID) > 3;
```

Faculty_ID	FirstName	Department_Name
102	Mahendra Singh	CSE
103	Mithali	CSE
101	Rohit	CSE

(No column name)	Department_Name
3	CSE

6. Illustrate the Usage of Except, Exists, Not Exists, Union, Intersect

a) EXCEPT QUERY

```
SELECT
```

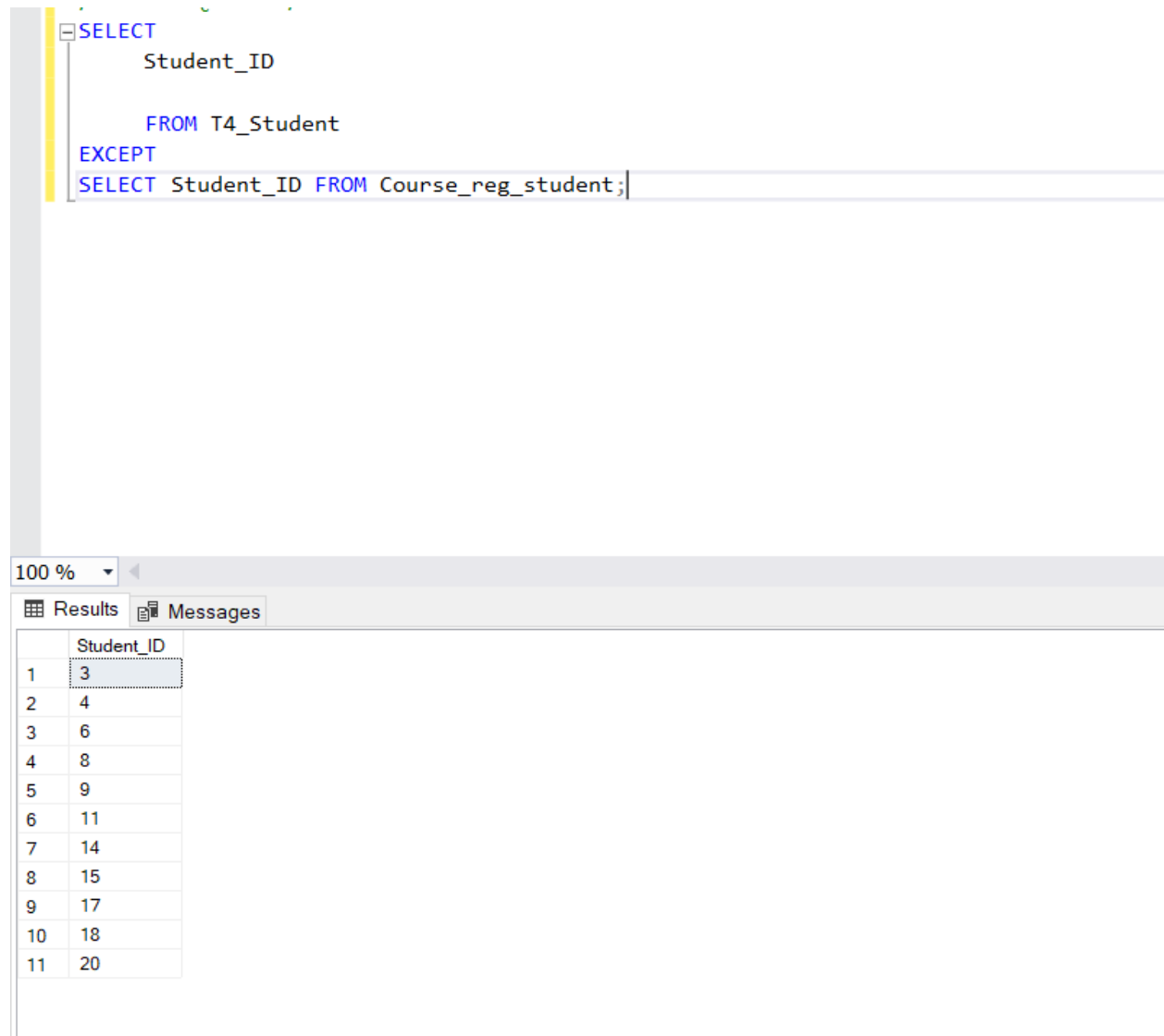
```
Studeny_ID
```

```
FROM T4_Student
```

EXCEPT

SELECT Student_ID FROM Course_reg_student;

OUTPUT



The screenshot shows a SQL IDE interface. The top pane contains a query with a syntax error. The query is:

```
SELECT
    Student_ID
FROM T4_Student
EXCEPT
SELECT Student_ID FROM Course_reg_student;
```

The bottom pane shows the 'Results' tab with a table of 11 rows. The first row is highlighted. The table has two columns: an index and 'Student_ID'.

	Student_ID
1	3
2	4
3	6
4	8
5	9
6	11
7	14
8	15
9	17
10	18
11	20

b) EXIST QUERY

```
SELECT
    Faculty_ID,
    Department_Name,
    FirstName,
    LastName,
    Phone
FROM
    T4_Faculty
WHERE
    EXISTS( SELECT Department_Name FROM T4_Course_offered
    WHERE
        T4_Faculty.Faculty_ID = T4_Course_offered.Faculty_ID);
```

OUTPUT

```

SELECT
    Faculty_ID,
    Department_Name,
    FirstName,
    LastName,
    Phone
FROM
    T4_Faculty
WHERE
    EXISTS(
        SELECT Department_Name FROM T4_Course_offered
        WHERE
            T4_Faculty.Faculty_ID = T4_Course_offered.Faculty_ID);

```

100 %

Results Messages

	Faculty_ID	Department_Name	FirstName	LastName	Phone
1	100	AI	Virat	Kohli	51188
2	102	CSE	Mahendra Singh	Dhoni	70781
3	104	AI	Ajinkya	Rahane	60688
4	105	ECE	Shubhman	Gill	70895
5	106	ECE	Harleen	Deol	66666
6	107	BT	Jasprit	Bumrah	99999
7	108	BT	Ekta	Bisht	54545
8	110	AERO	Ravi	Ashwin	11116
9	111	ME	Ravindra	Jadeja	33890
10	112	MATH	Che	Pujara	80806
11	114	CVE	Smriti	Mandhana	11112
12	118	HSE	Shikhar	Dhawan	11114

c) NOT EXISTS QUERY

SELECT

Student1.Student_id,

Student1.first_Name,

Student1.last_Name

FROM T4_Student AS Student1

WHERE NOT EXISTS(SELECT * FROM T4_Student AS Student2

WHERE Student1.Student_ID = Student2.Stduent_ID

AND GENDER IN ('M'));

OUTPUT


```

SELECT
    Student1.Student_ID,
    Student1.first_Name,
    Student1.last_Name
FROM T4_Student AS Student1

WHERE NOT EXISTS(SELECT * FROM T4_Student AS Student2
    WHERE Student1.Student_ID = Student2.Student_ID
    AND Gender in ('M'))

```

100 %

Results Messages

	Student_ID	first_Name	last_Name
1	6	Alyssa	Healy
2	7	Meg	Lanning
3	8	Ellyse	Perry
4	9	Rachel	Haynes
5	10	Ashleigh	Gardner
6	16	Heather	Knight
7	17	Nat	Sciver
8	18	Sarah	Taylor
9	19	Amy	Jones
10	20	Danni	Wyatt

d) UNION QUERY

SELECT

Student_ID,

first_Name,

last_Name

FROM T4_Student

UNION

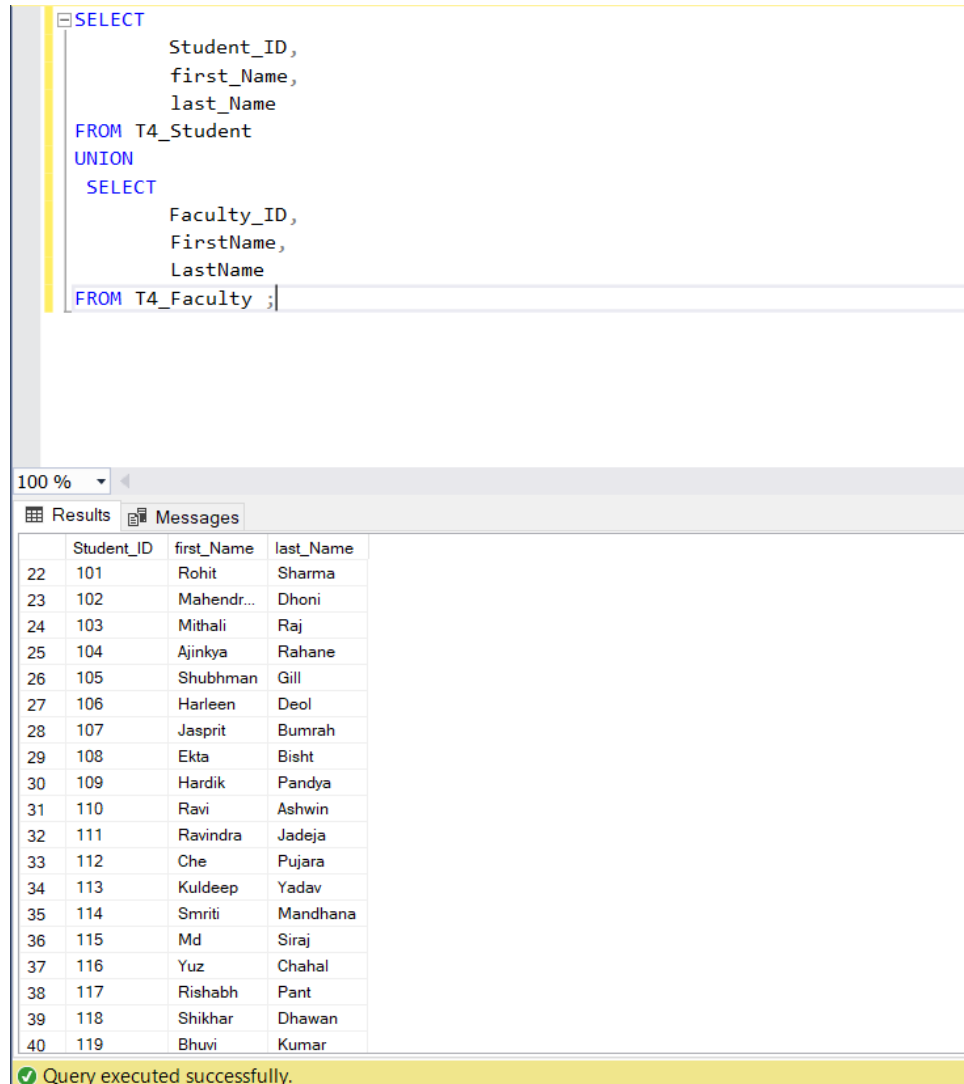
SELECT Faculty_ID,

FirstName,

LastName

FROM T4_Faculty ;

OUTPUT



```
SELECT
    Student_ID,
    first_Name,
    last_Name
FROM T4_Student
UNION
SELECT
    Faculty_ID,
    FirstName,
    LastName
FROM T4_Faculty ;
```

100 %

Results Messages

	Student_ID	first_Name	last_Name
22	101	Rohit	Sharma
23	102	Mahendr...	Dhoni
24	103	Mithali	Raj
25	104	Ajinkya	Rahane
26	105	Shubhman	Gill
27	106	Harleen	Deol
28	107	Jasprit	Bumrah
29	108	Ekta	Bisht
30	109	Hardik	Pandya
31	110	Ravi	Ashwin
32	111	Ravindra	Jadeja
33	112	Che	Pujara
34	113	Kuldeep	Yadav
35	114	Smriti	Mandhana
36	115	Md	Siraj
37	116	Yuz	Chahal
38	117	Rishabh	Pant
39	118	Shikhar	Dhawan
40	119	Bhuvi	Kumar

✓ Query executed successfully.

e) INTERSECT QUERY

SELECT Faculty_ID,

Department_Name,

FROM T4_Faculty

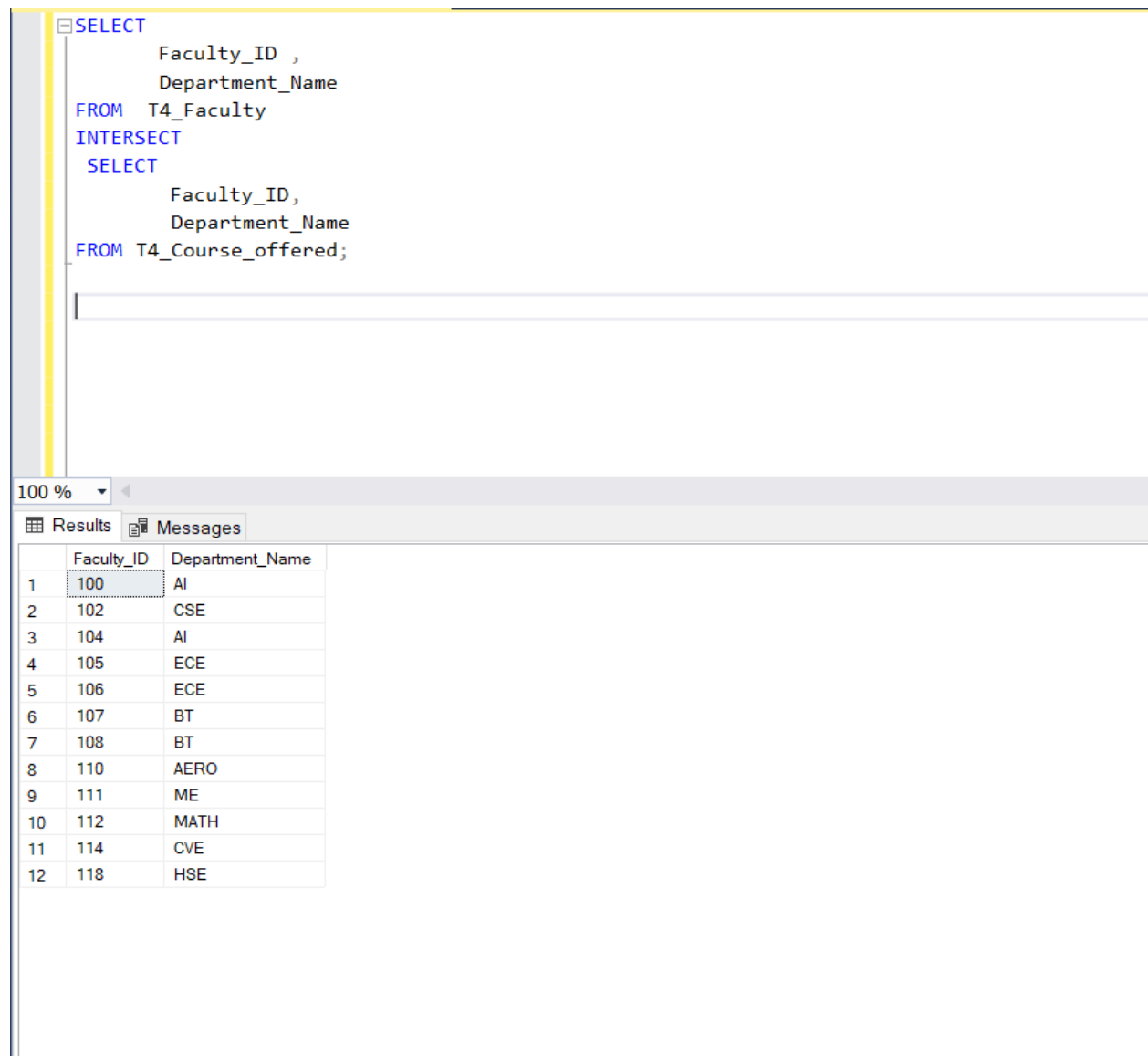
INTERSECT

SELECT Faculty_ID,

Department_Name

FROM T4_Course_offered ;

OUTPUT



```
SELECT
    Faculty_ID ,
    Department_Name
FROM T4_Faculty
INTERSECT
SELECT
    Faculty_ID,
    Department_Name
FROM T4_Course_offered;
```

100 %

Results Messages

	Faculty_ID	Department_Name
1	100	AI
2	102	CSE
3	104	AI
4	105	ECE
5	106	ECE
6	107	BT
7	108	BT
8	110	AERO
9	111	ME
10	112	MATH
11	114	CVE
12	118	HSE

7. INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN- 3 queries for each instance

and

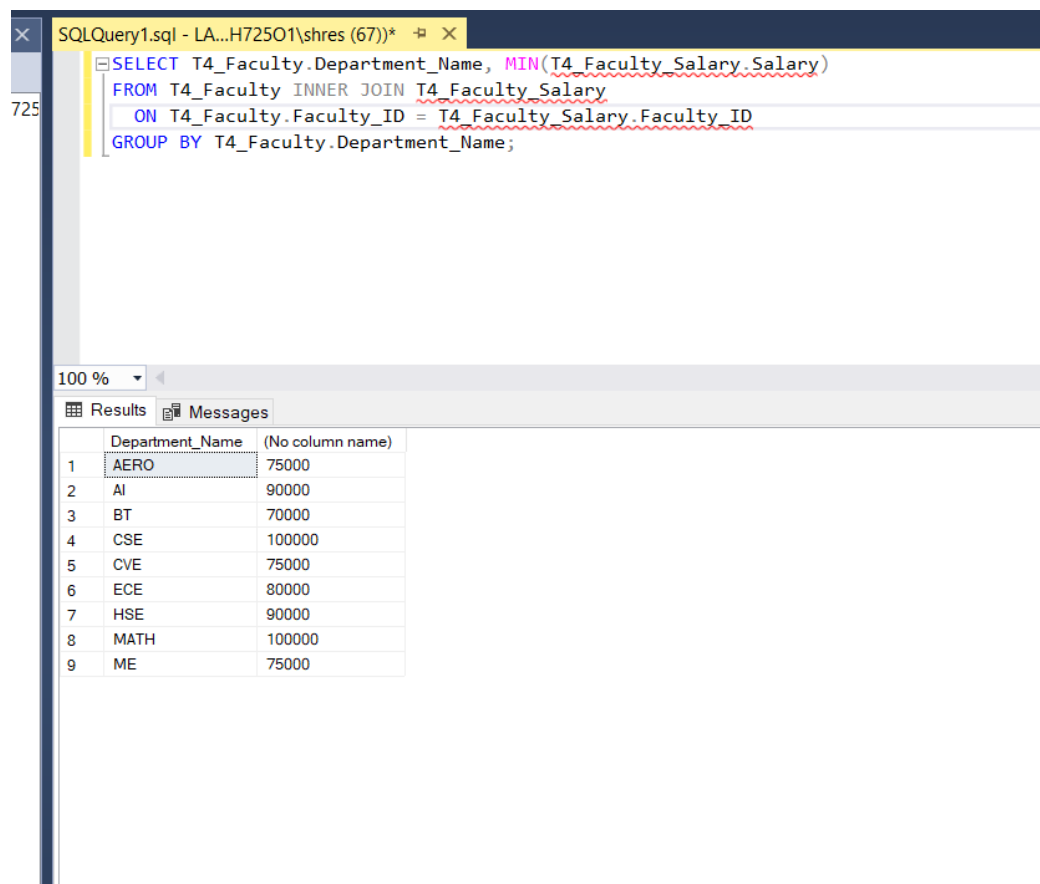
8. Use all the above conditions in JOIN as well.

INNER JOIN

Query:

```
SELECT T4_Faculty.Department_Name, MIN(T4_Faculty_Salary.Salary)
FROM T4_Faculty INNER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;
```

Output:



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 T4_Faculty.Department_Name, MIN(T4_Faculty_Salary.Salary)
FROM T4_Faculty INNER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;
```

The results pane shows the output of the query, which is a table with two columns: Department_Name and (No column name). The table contains 9 rows of data.

	Department_Name	(No column name)
1	AERO	75000
2	AI	90000
3	BT	70000
4	CSE	100000
5	CVE	75000
6	ECE	80000
7	HSE	90000
8	MATH	100000
9	ME	75000

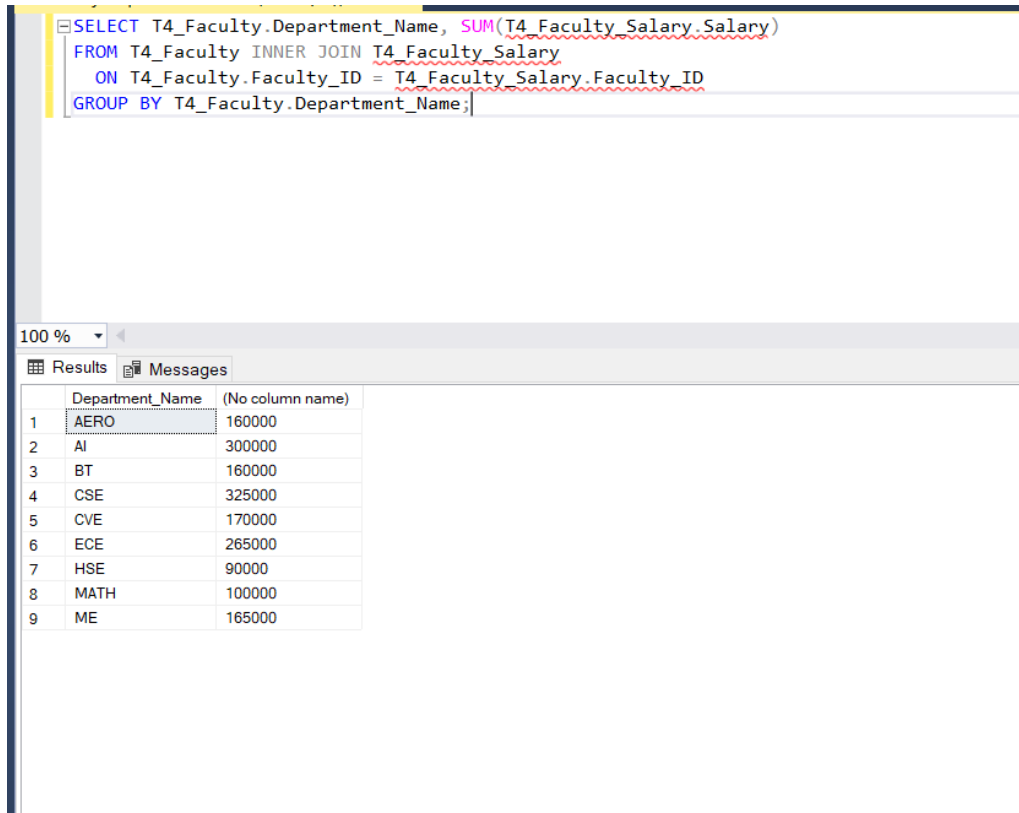
Query:

```
SELECT T4_Faculty.Department_Name, SUM(T4_Faculty_Salary.Salary)
FROM T4_Faculty INNER JOIN T4_Faculty_Salary
```

ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID

GROUP BY T4_Faculty.Department_Name;

Output:



```
SELECT T4_Faculty.Department_Name, SUM(T4_Faculty_Salary.Salary)
FROM T4_Faculty INNER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;
```

100 %

Results Messages

	Department_Name	(No column name)
1	AERO	160000
2	AI	300000
3	BT	160000
4	CSE	325000
5	CVE	170000
6	ECE	265000
7	HSE	90000
8	MATH	100000
9	ME	165000

Query:

SELECT T4_Faculty.Department_Name

FROM T4_Faculty INNER JOIN T4_Faculty_Salary

on T4_Faculty.Faculty_ID = ANY

(SELECT Faculty_ID from T4_Faculty where T4_Faculty.Faculty_ID =
T4_Faculty_Salary.Faculty_ID);

OUTPUT:

25

```
SELECT T4_Faculty.Department_Name
FROM T4_Faculty INNER JOIN T4_Faculty_Salary
on T4_Faculty.Faculty_ID = ANY
(SELECT Faculty_ID from T4_Faculty where T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID);
```

100 %

Results Messages

	Department_Name
1	AI
2	CSE
3	CSE
4	CSE
5	AI
6	ECE
7	ECE
8	BT
9	BT
10	AERO
11	AERO
12	ME
13	MATH
14	ME
15	CVE
16	CVE
17	CVE
18	ECE
19	HSE
20	AI

Left Outer Join

Query:

```
SELECT T4_Faculty.Department_Name, SUM(T4_Faculty_Salary.Salary)
FROM T4_Faculty LEFT OUTER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;
```

Output:

```

SELECT T4_Faculty.Department_Name, SUM(T4_Faculty_Salary_Salary)
FROM T4_Faculty LEFT OUTER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;

```

	Department_Name	(No column name)
1	AERO	160000
2	AI	300000
3	BT	160000
4	CSE	325000
5	CVE	170000
6	ECE	265000
7	HSE	90000
8	MATH	100000
9	ME	165000

Query:

```

SELECT T4_Faculty.Department_Name
FROM T4_Faculty LEFT OUTER JOIN T4_Faculty_Salary
on T4_Faculty.Faculty_ID = ALL
(SELECT Faculty_ID from T4_Faculty where T4_Faculty.Faculty_ID =
T4_Faculty_Salary.Faculty_ID);

```

Output:

```

SELECT T4_Faculty.Department_Name
FROM T4_Faculty LEFT OUTER JOIN T4_Faculty_Salary
on T4_Faculty.Faculty_ID = ALL
(SELECT Faculty_ID from T4_Faculty where T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID);

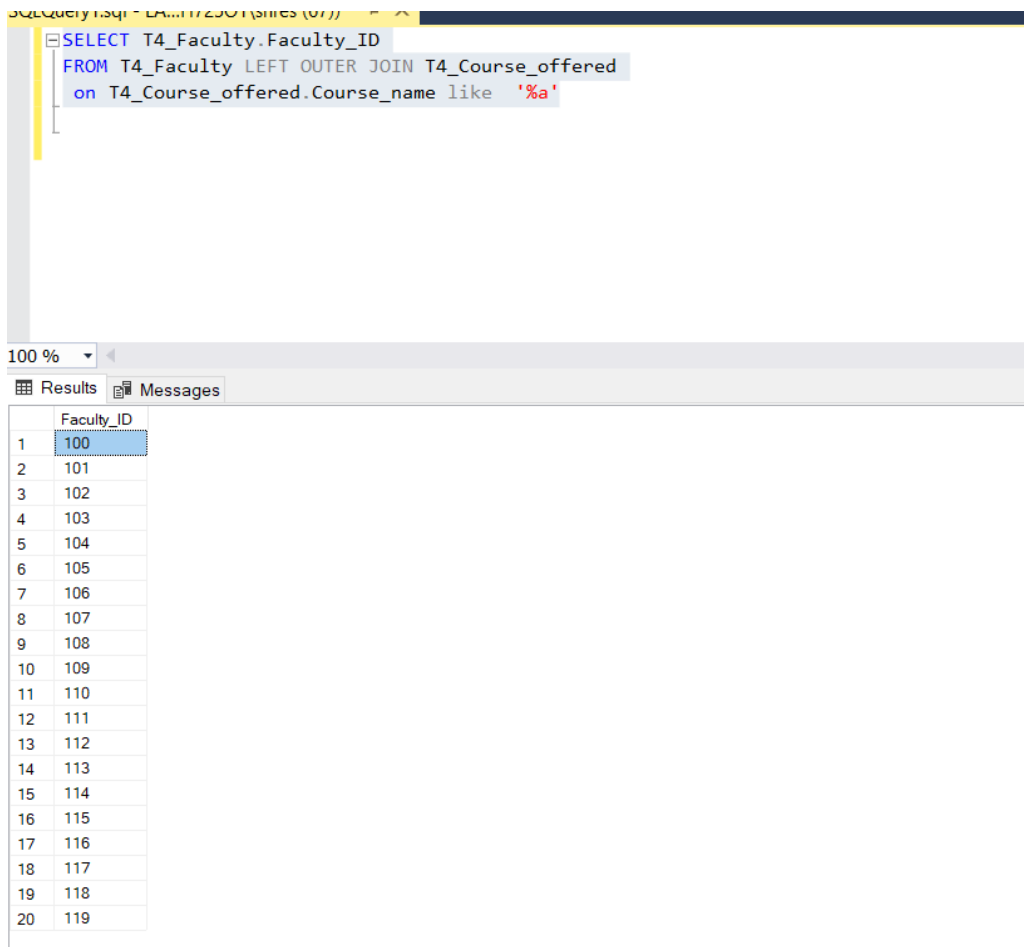
```

	Department_Name
1	AI
2	CSE
3	CSE
4	CSE
5	AI
6	ECE
7	ECE
8	BT
9	BT
10	AERO
11	AERO
12	ME
13	MATH
14	ME
15	CVE
16	CVE
17	CVE
18	ECE
19	HSE
20	AI

Query:

```
SELECT T4_Faculty.Faculty_ID
FROM T4_Faculty LEFT OUTER JOIN T4_Course_offered
on T4_Course_offered.Course_name like '%a'
```

Output



The screenshot shows a SQL Server Enterprise Manager interface. The top pane displays a query: `SELECT T4_Faculty.Faculty_ID FROM T4_Faculty LEFT OUTER JOIN T4_Course_offered on T4_Course_offered.Course_name like '%a'`. The bottom pane shows the results of this query, which is a table with two columns: `Faculty_ID` and an unlabeled column. The results are as follows:

	Faculty_ID	
1	100	
2	101	
3	102	
4	103	
5	104	
6	105	
7	106	
8	107	
9	108	
10	109	
11	110	
12	111	
13	112	
14	113	
15	114	
16	115	
17	116	
18	117	
19	118	
20	119	

Right Outer Join

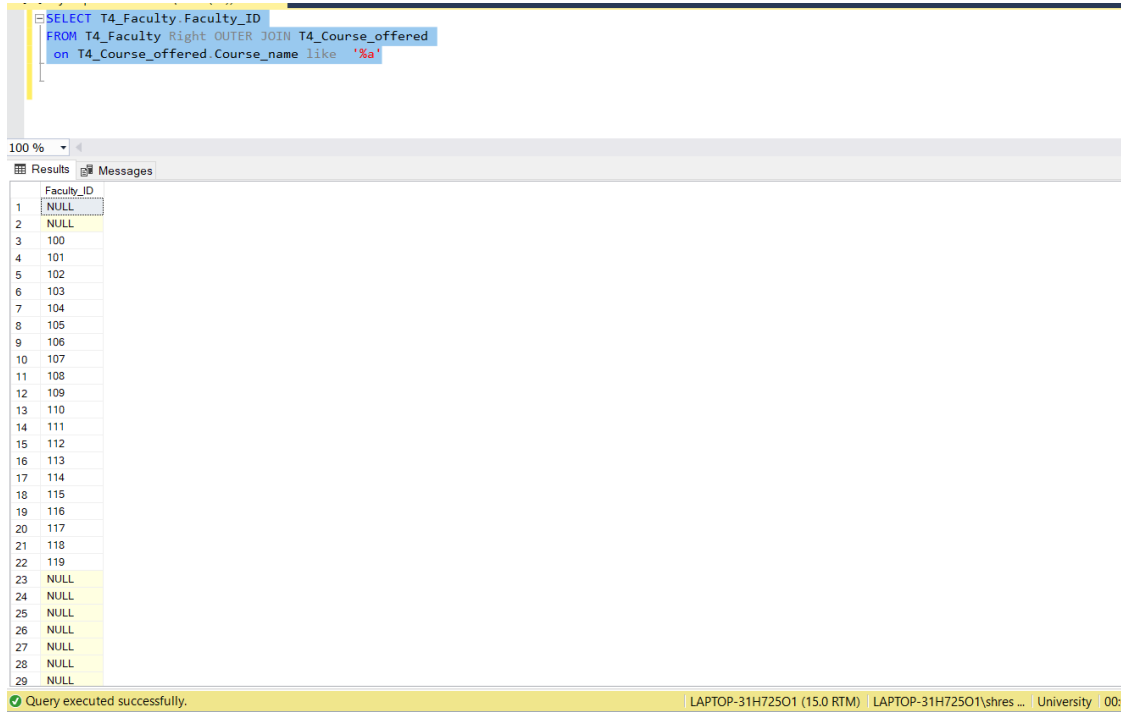
Query:

```
SELECT T4_Faculty.Faculty_ID
```



```
FROM T4_Faculty Right OUTER JOIN T4_Course_offered
on T4_Course_offered.Course_name like '%a'
```

OUTPUT:



The screenshot shows a SQL query window with the following text:

```
SELECT T4_Faculty.Faculty_ID
FROM T4_Faculty Right OUTER JOIN T4_Course_offered
on T4_Course_offered.Course_name like '%a'
```

Below the query window, the 'Results' tab is active, displaying a table with one column, 'Faculty_ID'. The table contains 29 rows. Rows 1 and 2 are highlighted in yellow and contain 'NULL'. Rows 3 through 22 contain numeric values from 100 to 119. Rows 23 through 29 are highlighted in yellow and contain 'NULL'.

Faculty_ID
NULL
NULL
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
NULL
NULL
NULL
NULL
NULL
NULL
NULL

At the bottom of the screenshot, a status bar indicates 'Query executed successfully.' and shows the file path 'LAPTOP-31H725O1 (15.0 RTM) | LAPTOP-31H725O1\shres ... | University | 004'.

Query:

```
SELECT T4_Faculty.Department_Name
FROM T4_Faculty Right OUTER JOIN T4_Faculty_Salary
on T4_Faculty.Faculty_ID = ALL
(SELECT Faculty_ID from T4_Faculty where T4_Faculty.Faculty_ID =
T4_Faculty_Salary.Faculty_ID);
```

OUTPUT :

SQLQuery1.sql - LA...H725O1\shres (67))*

```

SELECT T4_Faculty.Department_Name
FROM T4_Faculty Right OUTER JOIN T4_Faculty_Salary
on T4_Faculty.Faculty_ID = ALL
(SELECT Faculty_ID from T4_Faculty where T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID);

```

100 %

Results Messages

	Department_Name
1	AI
2	CSE
3	CSE
4	CSE
5	AI
6	ECE
7	ECE
8	BT
9	BT
10	AERO
11	AERO
12	ME
13	MATH
14	ME
15	CVE
16	CVE
17	CVE
18	ECE
19	HSE
20	AI

Query :

```

SELECT T4_Faculty.Department_Name, MIN(T4_Faculty_Salary.Salary)
FROM T4_Faculty Right OUTER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;

```

OUTPUT:

SQLQuery1.sql - LA...H725O1\shres (67))*

```

SELECT T4_Faculty.Department_Name, MIN(T4_Faculty_Salary.Salary)
FROM T4_Faculty Right OUTER JOIN T4_Faculty_Salary
ON T4_Faculty.Faculty_ID = T4_Faculty_Salary.Faculty_ID
GROUP BY T4_Faculty.Department_Name;

```

100 %

Results Messages

	Department_Name	(No column name)
1	AERO	75000
2	AI	90000
3	BT	70000
4	CSE	100000
5	CVE	75000
6	ECE	80000
7	HSE	90000
8	MATH	100000
9	ME	75000

