

Assignment 1

Create a Database name entri_assignment

```
CREATE DATABASE entri_assignment;
```

Create a Table with name departments

```
Department_id (pk) Department_name Location_id+
```

```
CREATE TABLE departments(
```

```
    department_id INT NOT NULL auto_increment,
```

```
    department_name VARCHAR(100) NOT NULL,
```

```
    location_id INT NOT NULL,
```

```
    PRIMARY KEY(department_id)
```

```
);
```

Create a Table with name employees

```
Employee_id (pk) ,first_name,last_name ,email,phone_number,hire_date,
```

```
job_id, salary, commission_pct, manager_id, department_id (fk  
reference
```

```
CREATE TABLE employees(
```

```
    Employee_id INT NOT NULL auto_increment,
```

```
    first_name VARCHAR(50) NOT NULL,
```

```
    last_name VARCHAR(50) NOT NULL,
```

```
    email VARCHAR(100) NOT NULL,
```

```
    phone_number VARCHAR(100) NOT NULL,
```

```
    hire_date DATE NOT NULL,
```

```
    job_id VARCHAR(20) NOT NULL,
```

```
    salary DECIMAL(10,2) NOT NULL,
```

```
    commision_pct DECIMAL(5,2),
```

```

manager_id INT,
department_id INT,
PRIMARY KEY(Employee_id),
FOREIGN KEY (department_id) REFERENCES departments(department_id)
);

```

```
## Insert into Departments table
```

```
INSERT INTO departments VALUES ( 170 , 'Payroll' , 1700);
```

```

INSERT INTO departments VALUES (20, 'Human Resources', 1001), (30,
'Marketing', 1002), (40, 'IT', 1003), (60, 'R&D', 1004), (100, 'Finance',
1005), (170, 'Accounting', 1006), (160, 'Legal', 1007), (150,
'Administration', 1008), (80, 'Purchasing', 1009), (70, 'Sales',
1010), (130, 'Warehouse', 1011), (50, 'Shipping', 1012), (90, 'Retail',
1013), (110, 'Manufacturing', 1014);

```

Query 1 departments x

Limit to 1000 rows

```
1 • SELECT * FROM entri_assignment.departments;
```

Result Grid

	department_id	department_name	location_id
▶	20	Human Resources	1001
	30	Marketing	1002
	40	IT	1003
	50	Shipping	1012
	60	R&D	1004
	70	Sales	1010
	80	Purchasing	1009
	90	Retail	1013
	100	Finance	1005
	110	Manufacturing	1014
	130	Warehouse	1011
	150	Administration	1008
	160	Legal	1007

departments 5 x

Output

employees table

```
; INSERT INTO employees V
```

```
## Insert into employees VALUES (101, 'Neena' , 'Kochhar' , 'NKOCHHAR' ,  
, '515.123.4568' , '1989-11-21' , 'AD_VP' , 17000 , NULL , 100 , 20);
```

```
INSERT INTO employees VALUES (102 , 'Lex' , 'De Haan' , 'LDEHAAN' ,  
'515.123.4569' , '1993-09-12' , 'AD_VP' , 17000 , NULL , 100 , 30);
```

```
INSERT INTO employees VALUES (104 , 'Bruce' , 'Ernst' , 'BERNST' ,  
'590.423.4568' , '1991-05-21', 'IT_PROG' , 6000 , NULL , 103 , 60);
```

```
INSERT INTO employees VALUES (105 , 'David' , 'Austin' , 'DAUSTIN' ,  
'590.423.4569' , '1997-06-25', 'IT_PROG' , 4800 , NULL , 103 , 60);
```

```
INSERT INTO employees VALUES (106 , 'Valli' , 'Pataballa' , 'VPATABAL' ,  
, '590.423.4560' , '1998-02-05', 'IT_PROG' , 4800 , NULL , 103 , 40);
```

```
INSERT INTO employees VALUES (107 , 'Diana' , 'Lorentz' , 'DLORENTZ' ,  
'590.423.5567' , '1999-02-09', 'IT_PROG' , 4200 , NULL , 103 , 40);
```

```
INSERT INTO employees VALUES (108 , 'Nancy' , 'Greenberg' , 'NGREENBE'  
, '515.124.4569' , '1994-08-17', 'FI_MGR' , 12000 , NULL , 101 ,  
100);
```

```
INSERT INTO employees VALUES (109 , 'Daniel' , 'Faviet' , 'DFAVIET' ,  
'515.124.4169' , '1994-08-12', 'FI_ACCOUNT' , 9000 , NULL , 108 ,  
170);
```

```
INSERT INTO employees VALUES (110 , 'John' , 'Chen' , 'JCHEN' ,  
'515.124.4269' , '1997-04-09', 'FI_ACCOUNT' , 8200 , NULL , 108 ,  
170);
```

```
INSERT INTO employees VALUES (111 , 'Ismael' , 'Sciarra' , 'ISCIARRA' ,  
'515.124.4369' , '1997-02-01', 'FI_ACCOUNT' , 7700 , NULL , 108 ,  
160);
```

```
INSERT INTO employees VALUES (112 , 'Jose Manuel' , 'Urman' ,  
'JMURMAN' , '515.124.4469' , '1998-06-03', 'FI_ACCOUNT' , 7800 , NULL  
8 , 150);
```

```
INSERT INTO employees VALUES (114 , 'Den' , 'Raphaely' , 'DRAPHEAL' ,  
'515.127.4561' , '1994-11-08', 'PU_MAN' , 11000 , NULL , 100 , 30);
```

```
INSERT INTO employees VALUES (115 , 'Alexander' , 'Khoo' , 'AKHOO' ,  
'515.127.4562' , '1995-05-12', 'PU_CLERK' , 3100 , NULL , 114 , 80);
```

```
INSERT INTO employees VALUES (116 , 'Shelli' , 'Baida' , 'SBAIDA' ,  
'515.127.4563' , '1997-12-13', 'PU_CLERK' , 2900 , NULL , 114 , 70);
```

```
INSERT INTO employees VALUES (117 , 'Sigal' , 'Tobias' , 'STOBIAS' ,  
'515.127.4564' , '1997-09-10', 'PU_CLERK' , 2800 , NULL , 114 , 30);
```

```
INSERT INTO employees VALUES (118 , 'Guy' , 'Himuro' , 'GHIMURO' ,  
'515.127.4565' , '1998-01-02', 'PU_CLERK' , 2600 , NULL , 114 , 60);
```

```
INSERT INTO employees VALUES (119 , 'Karen' , 'Colmenares' ,  
'KCOLMENA' , '515.127.4566' , '1999-04-08', 'PU_CLERK' , 2500 , NULL  
, 114 , 130);INSERT INTO employees VALUES (120 , 'Matthew' , 'Weiss' ,  
'MWEISS' , '650.123.1234' , '1996-07-18', 'ST_MAN' , 8000 , NULL , 100  
, 50);INSERT INTO employees VALUES (122 , 'Payam' , 'Kaufling' ,  
'PKAUFLIN' , '650.123.3234' , '1995-05-01', 'ST_MAN' , 7900 , NULL ,  
100 , 40);INSERT INTO employees VALUES (123 , 'Shanta' , 'Vollman' ,  
'SVOLLMAN' , '650.123.4234' , '1997-10-12', 'ST_MAN' , 6500 , NULL ,  
100 , 50);INSERT INTO employees VALUES (124 , 'Kevin' , 'Mourgos' ,  
'KMOURGOS' , '650.123.5234' , '1999-11-12', 'ST_MAN' , 5800 , NULL ,
```

```

100 , 80);INSERT INTO employees VALUES (125, 'Julia' , 'Nayer' ,
'JNAYER' , '650.124.1214' , '1997-07-02' , 'ST_CLERK' , 3200 , NULL ,
120 , 50);INSERT INTO employees VALUES (126, 'Irene' , 'Mikkilineni' ,
'IMIKKILI' , '650.124.1224' , '1998-11-12' , 'ST_CLERK' , 2700 , NULL ,
120 , 50);INSERT INTO employees VALUES (127, 'James' , 'Landry' ,
'JLANDRY' , '650.124.1334' , '1999-01-02' , 'ST_CLERK' , 2400 , NULL ,
120 , 90);INSERT INTO employees VALUES (128, 'Steven' , 'Markle' ,
'SMARKLE' , '650.124.1434' , '2000-03-04' , 'ST_CLERK' , 2200 , NULL ,
120 , 50);INSERT INTO employees VALUES (130, 'Mozhe' , 'Atkinson' ,
'MATKINSO' , '650.124.6234' , '1997-10-12' , 'ST_CLERK' , 2800 , NULL
, 121 , 110);

```

Query 1 departments employees

Limit to 1000 rows

1 • SELECT * FROM entri_assignment.employees;

Employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	commision_pct	manager_id	
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1989-11-21	AD_VP	17000.00	NULL	100	2
102	Lex	De Haan	LDEHAAN	515.123.4569	1993-09-12	AD_VP	17000.00	NULL	100	3
104	Bruce	Ernst	BERNST	590.423.4568	1991-05-21	IT_PROG	6000.00	NULL	103	6
105	David	Austin	DAUSTIN	590.423.4569	1997-06-25	IT_PROG	4800.00	NULL	103	6
106	Valli	Pataballa	VPATABAL	590.423.4560	1998-02-05	IT_PROG	4800.00	NULL	103	4
107	Diana	Lorentz	DLORENTZ	590.423.5567	1999-02-09	IT_PROG	4200.00	NULL	103	4
108	Nancy	Greenberg	NGREENBE	515.124.4569	1994-08-17	FI_MGR	12000.00	NULL	101	1
109	Daniel	Faviet	DFAVIET	515.124.4169	1994-08-12	FI_ACCOUNT	9000.00	NULL	108	1
110	John	Chen	JCHEN	515.124.4269	1997-04-09	FI_ACCOUNT	8200.00	NULL	108	1
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1997-02-01	FI_ACCOUNT	7700.00	NULL	108	1
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1998-06-03	FI_ACCOUNT	7800.00	NULL	8	1
114	Den	Raphaely	DRAPHEAL	515.127.4561	1994-11-08	PU_MAN	11000.00	NULL	100	3
115	Alexander	Khoo	AKHOO	515.127.4562	1995-05-12	PU_CLERK	3100.00	NULL	114	8
116	Shelli	Baida	SBAIDA	515.127.4563	1997-12-13	PU_CLERK	2900.00	NULL	114	7

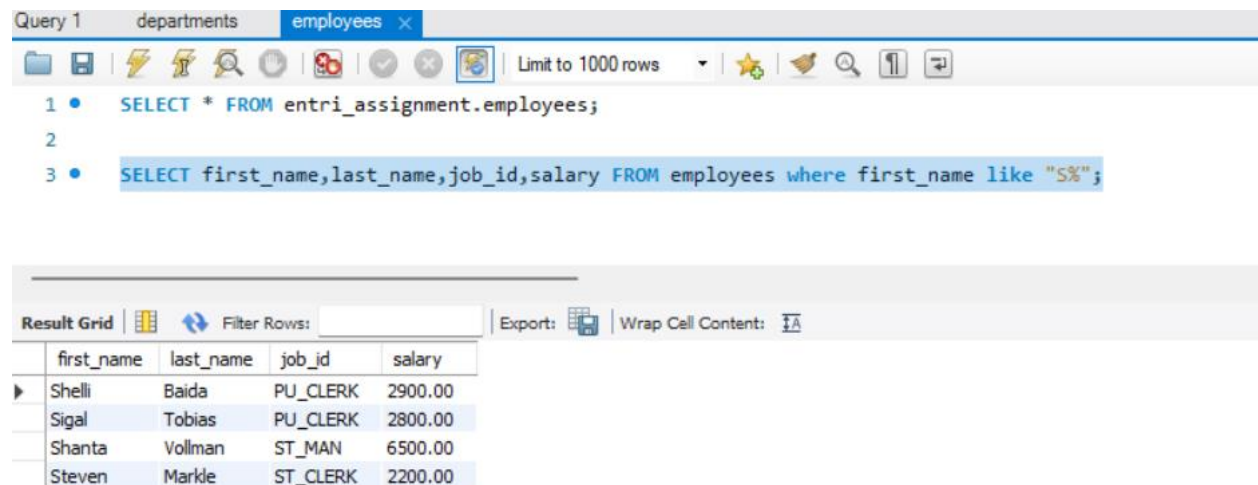
employees 5 x

Apply Revert

Solve SQL Exercises

1. Select employees first name, last name, job_id and salary whose first name starts with alphabet S

SELECT first_name,last_name,job_id,salary FROM employees where first_name like "S%";



The screenshot shows a SQL IDE interface. At the top, there are tabs for 'Query 1', 'departments', and 'employees'. Below the tabs is a toolbar with various icons. The main area displays a SQL query:

```
1 • SELECT * FROM entri_assignment.employees;
2
3 • SELECT first_name,last_name,job_id,salary FROM employees where first_name like "S%";
```

Below the query editor, there is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The result grid displays the following data:

	first_name	last_name	job_id	salary
▶	Shelli	Baida	PU_CLERK	2900.00
	Sigal	Tobias	PU_CLERK	2800.00
	Shanta	Vollman	ST_MAN	6500.00
	Steven	Markle	ST_CLERK	2200.00

2. Write a query to select employee with the highest salary (using an inner query)

SELECT first_name,last_name,job_id,salary FROM employees where SALARY = (SELECT MAX(salary) FROM employees);

The screenshot shows a SQL IDE interface. At the top, there's a tab labeled 'Query 1' with sub-tabs for 'departments' and 'employees'. Below the tabs is a toolbar with various icons, including a 'Limit to 1000 rows' dropdown. The main area contains a SQL query with four lines, where the third line is highlighted in blue:

```
1 • SELECT * FROM entri_assignment.employees;  
2 • SELECT first_name,last_name,job_id,salary FROM employees where first_name like "S%";  
3  
4 • SELECT first_name,last_name,job_id,salary FROM employees where SALARY = (SELECT MAX(salary) FROM employees);
```

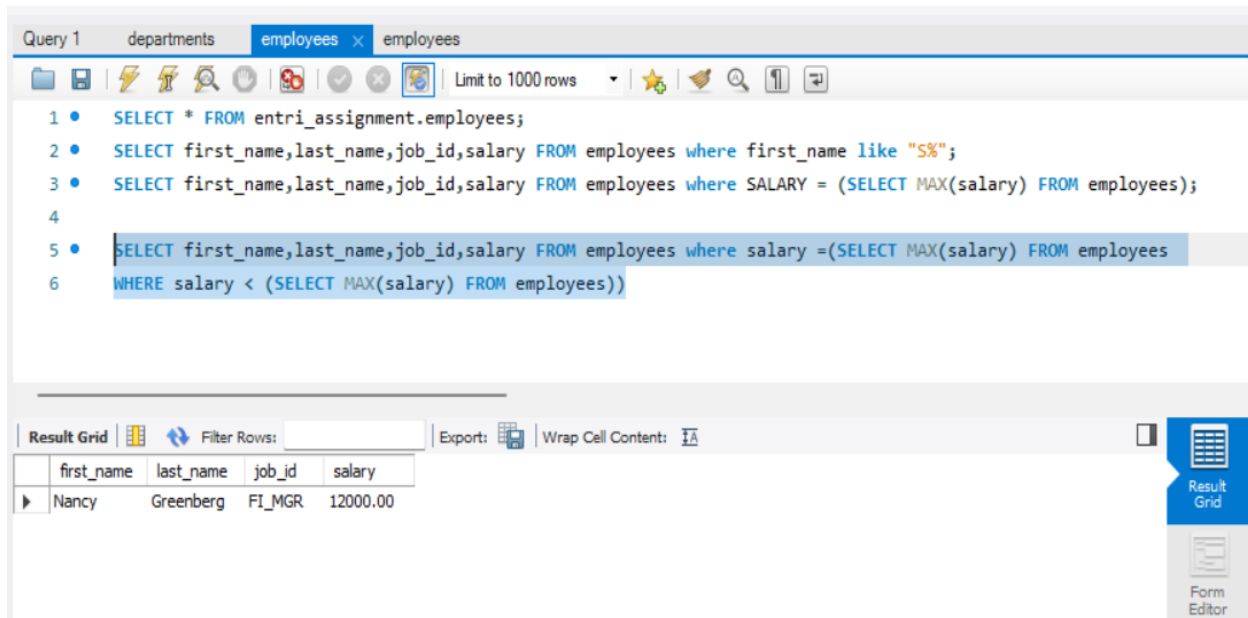
Below the query editor is a 'Result Grid' section. It has a toolbar with 'Filter Rows', 'Export', and 'Wrap Cell Content' options. The grid displays the following data:

	first_name	last_name	job_id	salary
▶	Neena	Kochhar	AD_VP	17000.00
	Lex	De Haan	AD_VP	17000.00

On the right side of the 'Result Grid' section, there is a vertical toolbar with a 'Result Grid' icon and a 'Full Screen' icon.

3. Select employee with the second highest salary

SELECT first_name,last_name,job_id,salary FROM employees where salary =(SELECT
MAX(salary) FROM employees WHERE salary < (SELECT MAX(salary) FROM employees))



Query 1 departments employees employees

Limit to 1000 rows

```
1 • SELECT * FROM entri_assignment.employees;
2 • SELECT first_name,last_name,job_id,salary FROM employees where first_name like "S%";
3 • SELECT first_name,last_name,job_id,salary FROM employees where SALARY = (SELECT MAX(salary) FROM employees);
4
5 • SELECT first_name,last_name,job_id,salary FROM employees where salary =(SELECT MAX(salary) FROM employees
6 WHERE salary < (SELECT MAX(salary) FROM employees))
```

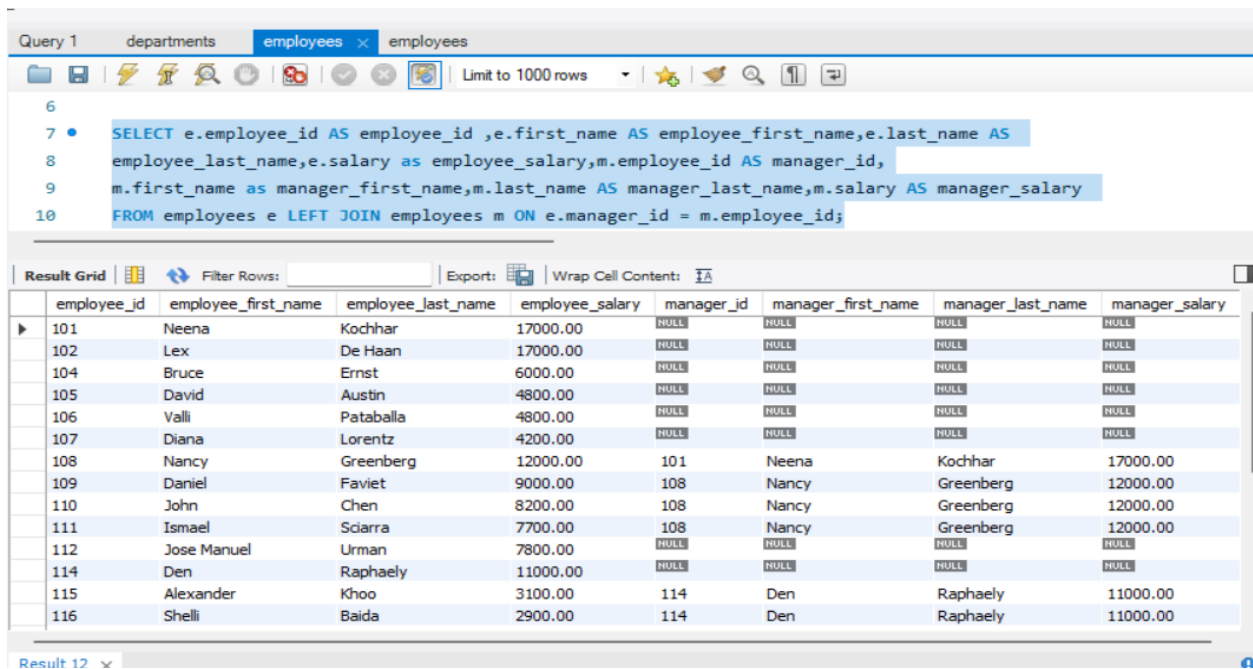
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	first_name	last_name	job_id	salary
▶	Nancy	Greenberg	FI_MGR	12000.00

Result Grid
Form Editor

4. Write a query to select employees and their corresponding managers and their salaries

```
SELECT e.employee_id AS employee_id ,e.first_name AS employee_first_name,e.last_name AS employee_last_name,e.salary as employee_salary,m.employee_id AS manager_id,m.first_name as manager_first_name,m.last_name AS manager_last_name,m.salary AS manager_salary FROM employees e LEFT JOIN employees m ON e.manager_id = m.employee_id;
```



The screenshot shows a database query editor with a query window and a result grid. The query window contains the following SQL query:

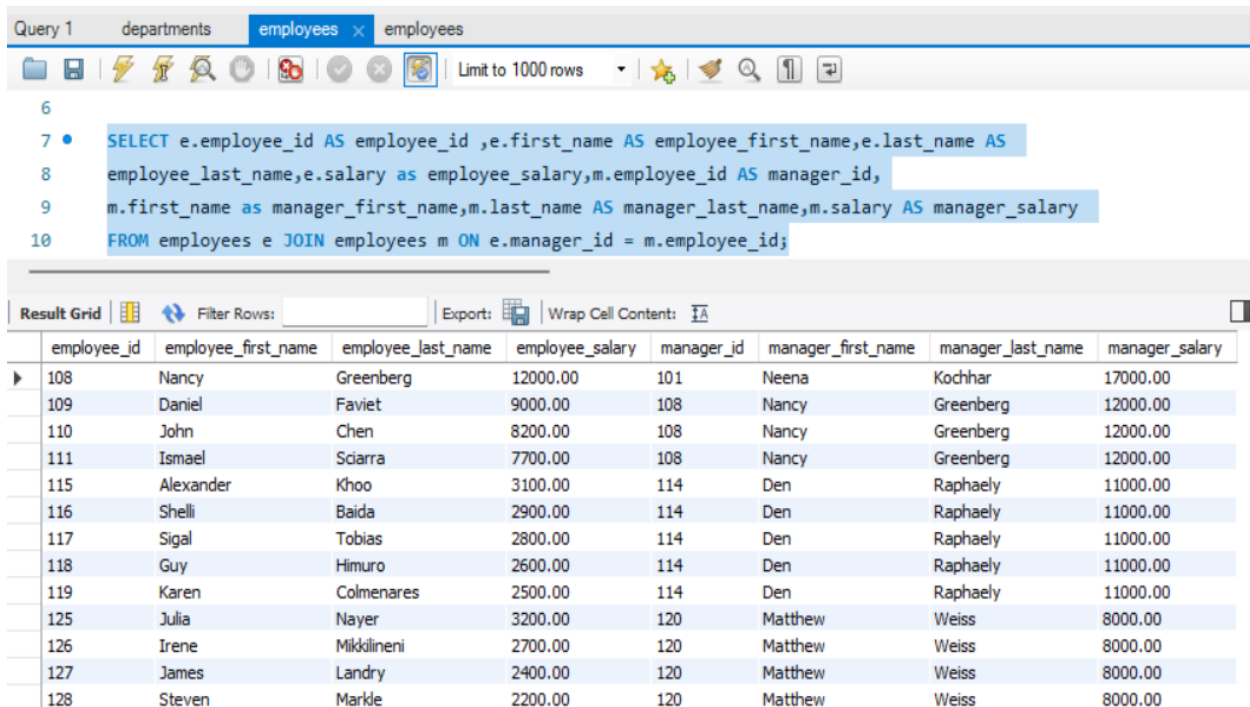
```
SELECT e.employee_id AS employee_id ,e.first_name AS employee_first_name,e.last_name AS employee_last_name,e.salary as employee_salary,m.employee_id AS manager_id,m.first_name as manager_first_name,m.last_name AS manager_last_name,m.salary AS manager_salary FROM employees e LEFT JOIN employees m ON e.manager_id = m.employee_id;
```

The result grid displays the following data:

employee_id	employee_first_name	employee_last_name	employee_salary	manager_id	manager_first_name	manager_last_name	manager_salary
101	Neena	Kochhar	17000.00	NULL	NULL	NULL	NULL
102	Lex	De Haan	17000.00	NULL	NULL	NULL	NULL
104	Bruce	Ernst	6000.00	NULL	NULL	NULL	NULL
105	David	Austin	4800.00	NULL	NULL	NULL	NULL
106	Valli	Pataballa	4800.00	NULL	NULL	NULL	NULL
107	Diana	Lorentz	4200.00	NULL	NULL	NULL	NULL
108	Nancy	Greenberg	12000.00	101	Neena	Kochhar	17000.00
109	Daniel	Faviet	9000.00	108	Nancy	Greenberg	12000.00
110	John	Chen	8200.00	108	Nancy	Greenberg	12000.00
111	Ismael	Sciarra	7700.00	108	Nancy	Greenberg	12000.00
112	Jose Manuel	Urman	7800.00	NULL	NULL	NULL	NULL
114	Den	Raphaely	11000.00	NULL	NULL	NULL	NULL
115	Alexander	Khoo	3100.00	114	Den	Raphaely	11000.00
116	Shelli	Baida	2900.00	114	Den	Raphaely	11000.00

5. Write a query to select employees and their corresponding managers and their salaries (SELF Join)

```
SELECT e.employee_id AS employee_id ,e.first_name AS employee_first_name,e.last_name AS employee_last_name,e.salary as employee_salary,m.employee_id AS manager_id,m.first_name as manager_first_name,m.last_name AS manager_last_name,m.salary AS manager_salary FROM employees e JOIN employees m ON e.manager_id = m.employee_id;
```



The screenshot shows a database query editor with a query window and a result grid. The query window displays a SQL query that performs a self-join on the employees table. The result grid shows the output of the query, listing employee and manager details.

Query 1 departments employees employees

Limit to 1000 rows

```
6
7 • SELECT e.employee_id AS employee_id ,e.first_name AS employee_first_name,e.last_name AS
8 employee_last_name,e.salary as employee_salary,m.employee_id AS manager_id,
9 m.first_name as manager_first_name,m.last_name AS manager_last_name,m.salary AS manager_salary
10 FROM employees e JOIN employees m ON e.manager_id = m.employee_id;
```

Result Grid


	employee_id	employee_first_name	employee_last_name	employee_salary	manager_id	manager_first_name	manager_last_name	manager_salary
▶	108	Nancy	Greenberg	12000.00	101	Neena	Kochhar	17000.00
	109	Daniel	Faviet	9000.00	108	Nancy	Greenberg	12000.00
	110	John	Chen	8200.00	108	Nancy	Greenberg	12000.00
	111	Ismael	Sciarra	7700.00	108	Nancy	Greenberg	12000.00
	115	Alexander	Khoo	3100.00	114	Den	Raphaely	11000.00
	116	Shelli	Baida	2900.00	114	Den	Raphaely	11000.00
	117	Sigal	Tobias	2800.00	114	Den	Raphaely	11000.00
	118	Guy	Himuro	2600.00	114	Den	Raphaely	11000.00
	119	Karen	Colmenares	2500.00	114	Den	Raphaely	11000.00
	125	Julia	Nayer	3200.00	120	Matthew	Weiss	8000.00
	126	Irene	Mikkilineni	2700.00	120	Matthew	Weiss	8000.00
	127	James	Landry	2400.00	120	Matthew	Weiss	8000.00
	128	Steven	Markle	2200.00	120	Matthew	Weiss	8000.00

6. Create a view for the above query




```
CREATE VIEW employee_manager_salaries AS SELECT e.employee_id AS  
employee_id,e.first_name AS employee_firts_name, e.last_name AS  
employee_last_name,e.salary AS employee_salary,m.employee_id AS  
manager_id,m.first_name AS manager_first_name,m.last_name AS  
manager_last_name,m.salary AS manager_salary FROM employees e JOIN employees m ON  
e.manager_id = m.employee_id;
```

```
SELECT * FROM employee_manager_salaries;
```

Query 1 departments employees employees

 Limit to 1000 rows

```
12 • CREATE VIEW employee_manager_salaries AS
13 SELECT e.employee_id AS employee_id,e.first_name AS employee_firts_name, e.last_name AS employee_last_name,
14 e.salary AS employee_salary,m.employee_id AS manager_id,m.first_name AS manager_first_name,m.last_name AS
15 manager_last_name,m.salary AS manager_salary FROM employees e JOIN employees m ON e.manager_id = m.employee_id;
16
17 • SELECT * FROM employee_manager_salaries;
18
```

Result Grid  Filter Rows: Export:  Wrap Cell Content: 

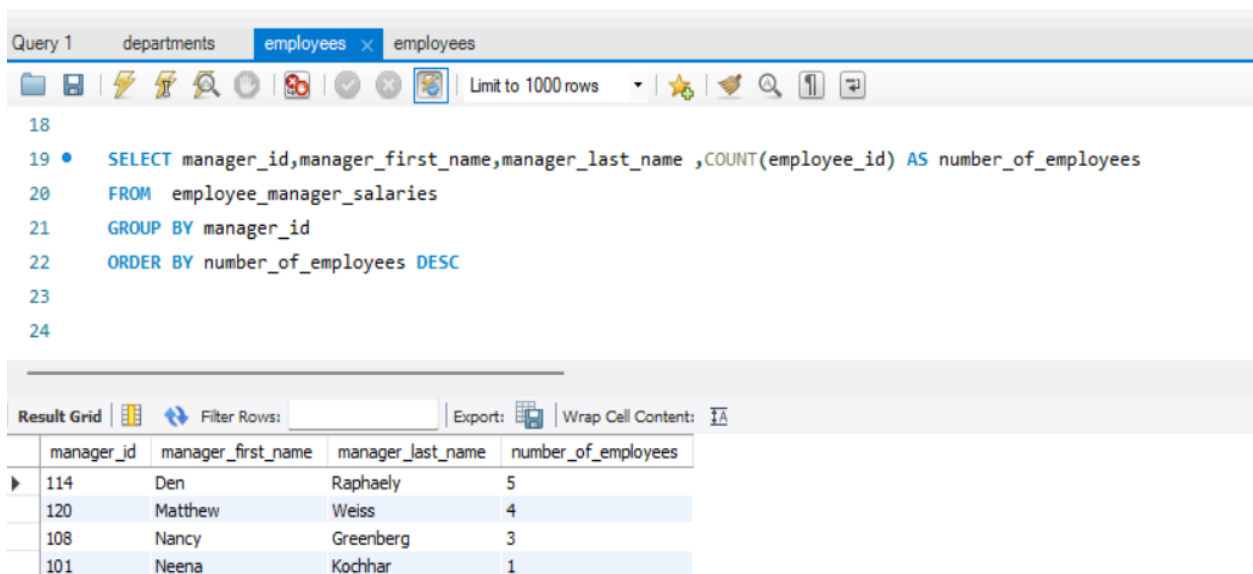
	employee_id	employee_firts_name	employee_last_name	employee_salary	manager_id	manager_first_name	manager_last_name	manager_salary
108	Nancy	Greenberg	12000.00	101	Neena	Kochhar	17000.00	
109	Daniel	Faviet	9000.00	108	Nancy	Greenberg	12000.00	
110	John	Chen	8200.00	108	Nancy	Greenberg	12000.00	
111	Ismael	Sciarra	7700.00	108	Nancy	Greenberg	12000.00	
115	Alexander	Khoo	3100.00	114	Den	Raphaely	11000.00	
116	Shelli	Baida	2900.00	114	Den	Raphaely	11000.00	
117	Sigal	Tobias	2800.00	114	Den	Raphaely	11000.00	
118	Guy	Himuro	2600.00	114	Den	Raphaely	11000.00	
119	Karen	Colmenares	2500.00	114	Den	Raphaely	11000.00	
125	Julia	Nayer	3200.00	120	Matthew	Weiss	8000.00	
126	Irene	Mikkilineni	2700.00	120	Matthew	Weiss	8000.00	
127	James	Landry	2400.00	120	Matthew	Weiss	8000.00	
128	Steven	Markle	2200.00	120	Matthew	Weiss	8000.00	

7. Write a query to show the count of employees under each manager in descending order (from view)

```
SELECT manager_id,manager_first_name,manager_last_name ,COUNT(employee_id) AS  
number_of_employees FROM employee_manager_salaries
```

```
GROUP BY manager_id
```

```
ORDER BY number_of_employees DESC
```



The screenshot shows a SQL IDE interface. At the top, there are tabs for 'Query 1', 'departments', 'employees', and 'employees'. Below the tabs is a toolbar with various icons. The main area displays a SQL query:

```
18  
19 • SELECT manager_id,manager_first_name,manager_last_name ,COUNT(employee_id) AS number_of_employees  
20 FROM employee_manager_salaries  
21 GROUP BY manager_id  
22 ORDER BY number_of_employees DESC  
23  
24
```

Below the query editor is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The result grid displays the following data:

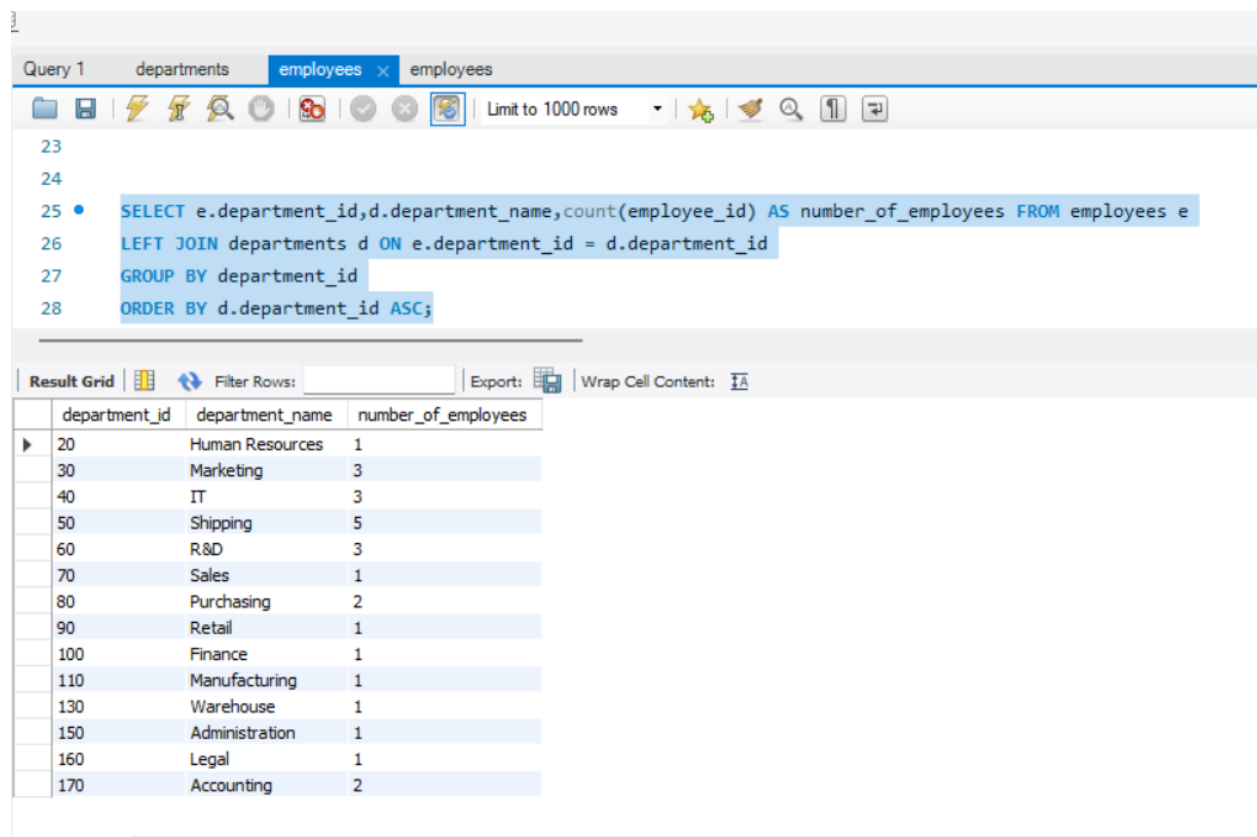
manager_id	manager_first_name	manager_last_name	number_of_employees
114	Den	Raphaely	5
120	Matthew	Weiss	4
108	Nancy	Greenberg	3
101	Neena	Kochhar	1

8. Find the count of employees in each department

```
SELECT e.department_id,d.department_name,count(employee_id) AS number_of_employees  
FROM employees e LEFT JOIN departments d ON e.department_id = d.department_id
```

```
GROUP BY department_id
```

```
ORDER BY d.department_id ASC;
```



The screenshot shows a database query editor with a query window and a results grid. The query window displays the following SQL query:

```
23  
24  
25 • SELECT e.department_id,d.department_name,count(employee_id) AS number_of_employees FROM employees e  
26 LEFT JOIN departments d ON e.department_id = d.department_id  
27 GROUP BY department_id  
28 ORDER BY d.department_id ASC;
```

The results grid shows the following data:

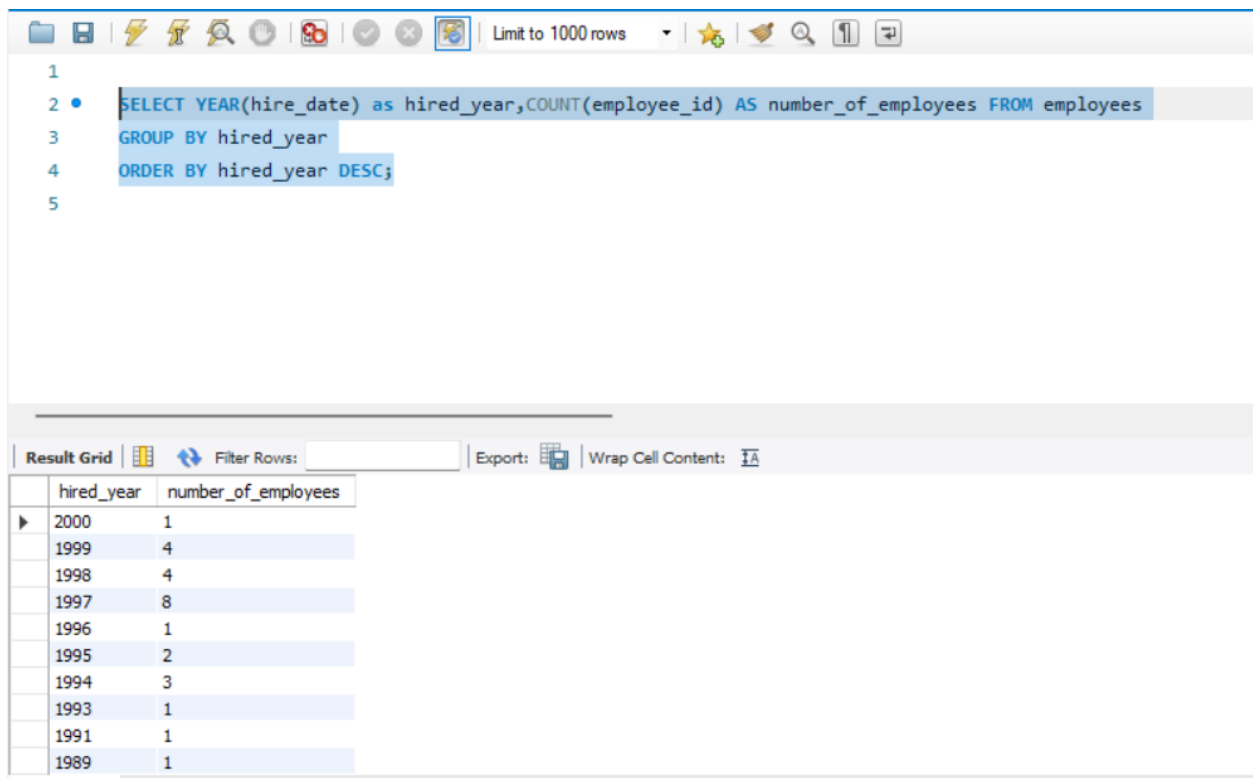
department_id	department_name	number_of_employees
20	Human Resources	1
30	Marketing	3
40	IT	3
50	Shipping	5
60	R&D	3
70	Sales	1
80	Purchasing	2
90	Retail	1
100	Finance	1
110	Manufacturing	1
130	Warehouse	1
150	Administration	1
160	Legal	1
170	Accounting	2

9. Get the count of employees hired year wise

```
SELECT YEAR(hire_date) as hired_year,COUNT(employee_id) AS number_of_employees  
FROM employees
```

```
GROUP BY hired_year
```

```
ORDER BY hired_year DESC;
```



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and settings, along with a 'Limit to 1000 rows' dropdown. The SQL editor contains the following query:

```
1  
2 • SELECT YEAR(hire_date) as hired_year,COUNT(employee_id) AS number_of_employees FROM employees  
3 GROUP BY hired_year  
4 ORDER BY hired_year DESC;  
5
```

Below the editor, the 'Result Grid' tab is active, displaying the query results in a table. The table has two columns: 'hired_year' and 'number_of_employees'. The results are ordered by year in descending order, from 2000 at the top to 1989 at the bottom.

hired_year	number_of_employees
2000	1
1999	4
1998	4
1997	8
1996	1
1995	2
1994	3
1993	1
1991	1
1989	1

10 . create a stored procedure to get the “ Get the count of employees hired in the input year”(IN year , OUT count)

```
DELIMITER //
```

```
CREATE PROCEDURE GetEmployeeCountByYear (
```

```
    IN input_year INT,
```

```
    OUT employee_count INT
```

```
)
```

```
BEGIN
```

```
    SELECT COUNT(employee_id) INTO employee_count
```

```
    FROM employees
```

```
    WHERE YEAR(hire_date) = input_year;
```

```
END //
```

```
DELIMITER ;
```

```
CALL GetEmployeeCountByYear(1997,@employee_count);
```

```
SELECT @employee_count AS number_of_employees_hired;
```

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

```
58 DELIMITER //  
59 CREATE PROCEDURE GetEmployeeCountByYear (  
60     IN input_year INT,  
61     OUT employee_count INT  
62 )  
63 BEGIN  
64     SELECT COUNT(employee_id) INTO employee_count  
65     FROM employees  
66     WHERE YEAR(hire_date) = input_year;  
67 END //  
68 DELIMITER ;  
69  
70 CALL GetEmployeeCountByYear(1997,@employee_count);  
71 SELECT @employee_count AS number_of_employees_hired;  
72
```

The results grid at the bottom shows the output of the query:

number_of_employees_hired
8

11. Select the employees whose first_name contains “an”

```
SELECT * FROM employees where first_name like '%an%';
```

[illegible]

12. Select employee first name and the corresponding phone number in the format (_ _ _)-(_ _ _)-(_ _ _)

```
SELECT first_name, CONCAT('(',SUBSTRING(phone_number,1,3),')-(' ,SUBSTRING(phone_number,5,3),')-(' ,SUBSTRING(phone_number,9,4),')') AS  
phone_number FROM employees;
```

The screenshot shows the Oracle SQL Developer interface. At the top, there are tabs for 'Query 1', 'departments', 'employees', and 'employees'. Below the tabs is a toolbar with various icons, including a 'Limit to 1000 rows' dropdown. The main area displays a SQL query:

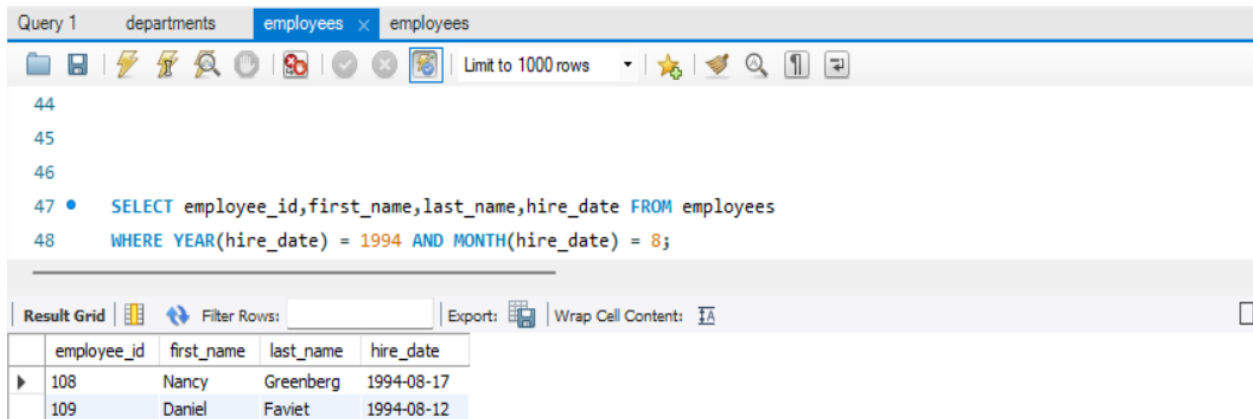
```
37  
38 • SELECT first_name,  
39      CONCAT('(',SUBSTRING(phone_number,1,3),')-(' ,SUBSTRING(phone_number,5,3),')-(' ,SUBSTRING(phone_number,9,4),')')  
40      AS phone_number  
41      FROM employees;  
42  
43
```

Below the query editor is the 'Result Grid' section. It includes a 'Filter Rows:' field, an 'Export:' button, and a 'Wrap Cell Content:' checkbox. The grid displays the following data:

first_name	phone_number
Neena	(515)-(123)-(4568)
Lex	(515)-(123)-(4569)
Bruce	(590)-(423)-(4568)
David	(590)-(423)-(4569)
Valli	(590)-(423)-(4560)
Diana	(590)-(423)-(5567)
Nancy	(515)-(124)-(4569)
Daniel	(515)-(124)-(4169)
John	(515)-(124)-(4269)
Ismael	(515)-(124)-(4369)
Jose Manuel	(515)-(124)-(4469)
Den	(515)-(127)-(4561)
Alexander	(515)-(127)-(4562)
Shelli	(515)-(127)-(4563)
Sigal	(515)-(127)-(4564)
Guy	(515)-(127)-(4565)
Karen	(515)-(127)-(4566)
Matthew	(650)-(123)-(1234)
Payam	(650)-(123)-(3234)
Chen	(650)-(123)-(4324)

13. Find the employees who joined in August, 1994.

```
SELECT employee_id,first_name,last_name,hire_date FROM employees  
  
WHERE YEAR(hire_date) = 1994 AND MONTH(hire_date) = 8;
```

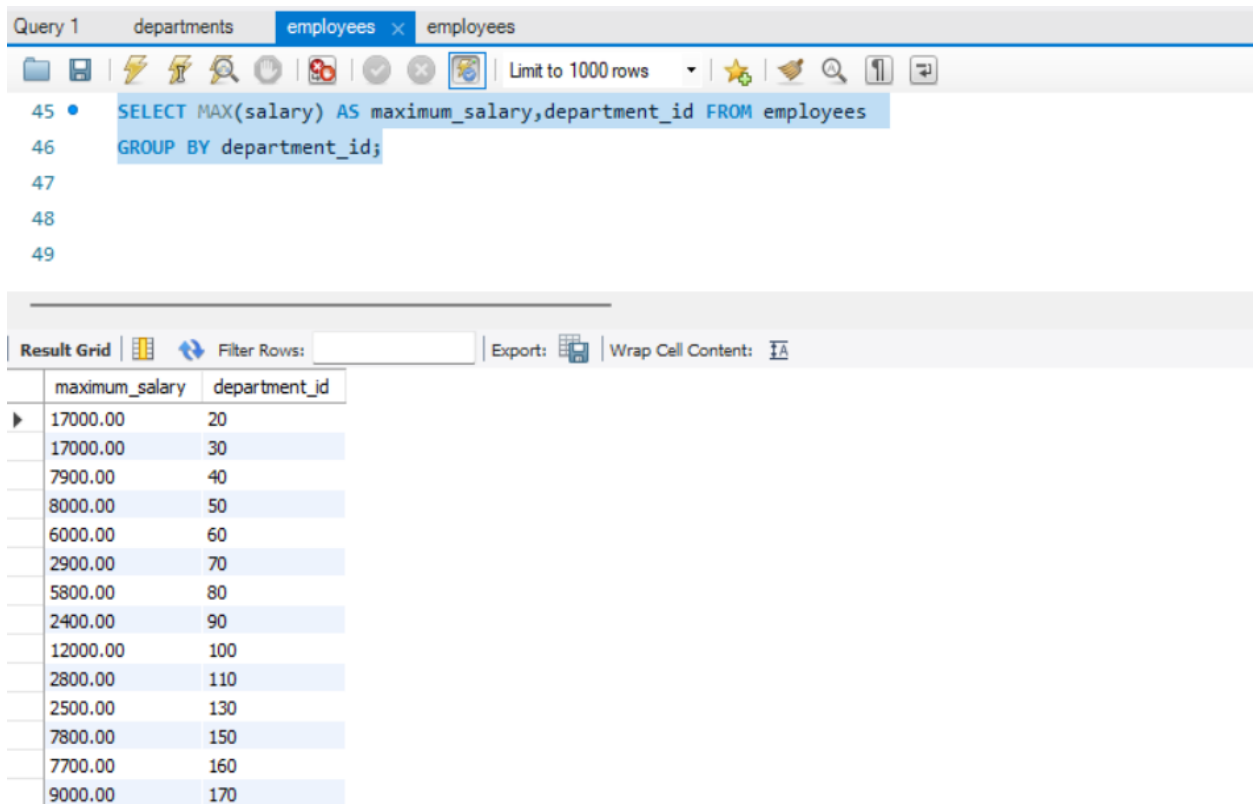


The screenshot shows a database query editor with a tab labeled 'employees'. The SQL query is entered in the editor, and the results are displayed in a table below. The table has four columns: employee_id, first_name, last_name, and hire_date. Two rows are shown, representing employees who joined in August 1994.

employee_id	first_name	last_name	hire_date
108	Nancy	Greenberg	1994-08-17
109	Daniel	Faviet	1994-08-12

14. Find the maximum salary from each department.

```
SELECT MAX(salary) AS maximum_salary, department_id FROM employees  
  
GROUP BY department_id;
```



The screenshot shows a database query editor with a tab labeled 'employees'. The SQL query is entered in the editor, and the results are displayed in a 'Result Grid' below. The query finds the maximum salary for each department. The results table has two columns: 'maximum_salary' and 'department_id'. The data is as follows:

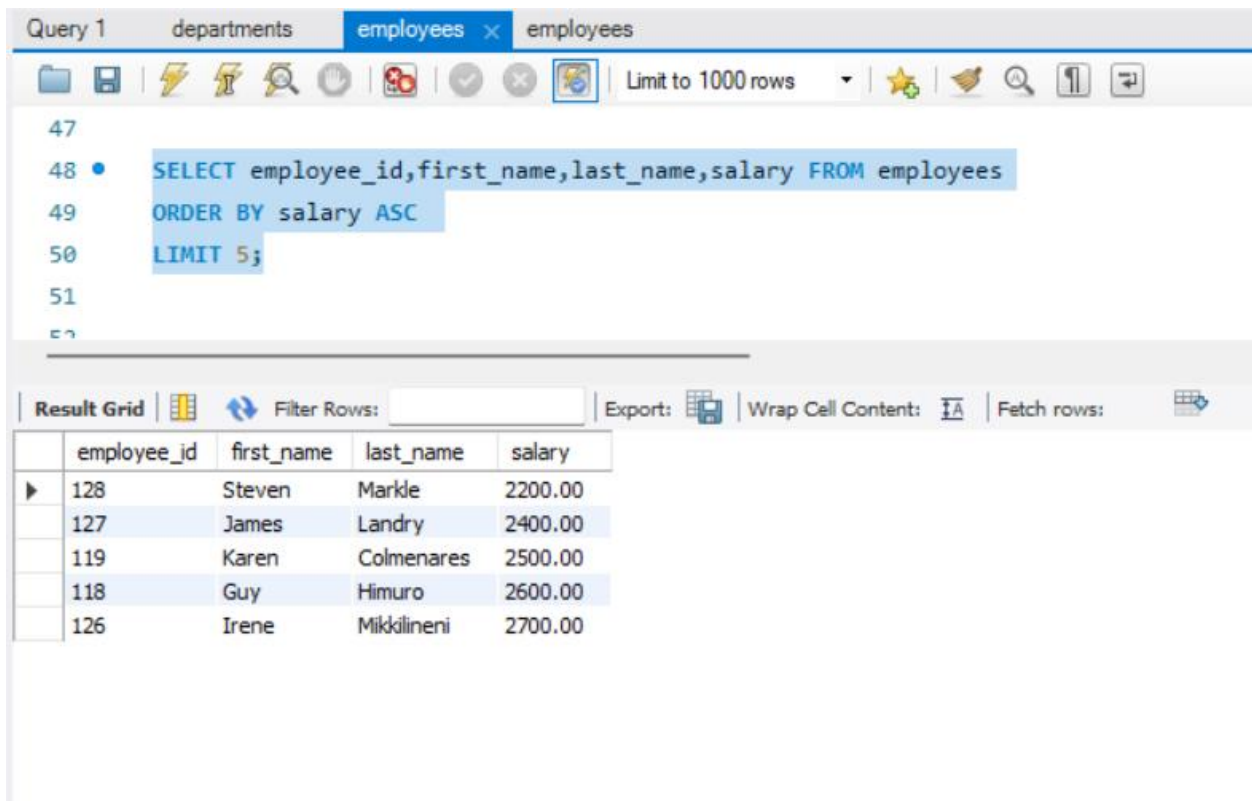
maximum_salary	department_id
17000.00	20
17000.00	30
7900.00	40
8000.00	50
6000.00	60
2900.00	70
5800.00	80
2400.00	90
12000.00	100
2800.00	110
2500.00	130
7800.00	150
7700.00	160
9000.00	170

15. Write a SQL query to display the 5 least earning employees

```
SELECT employee_id,first_name,last_name,salary FROM employees
```

```
ORDER BY salary ASC
```

```
LIMIT 5;
```



The screenshot shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL query:

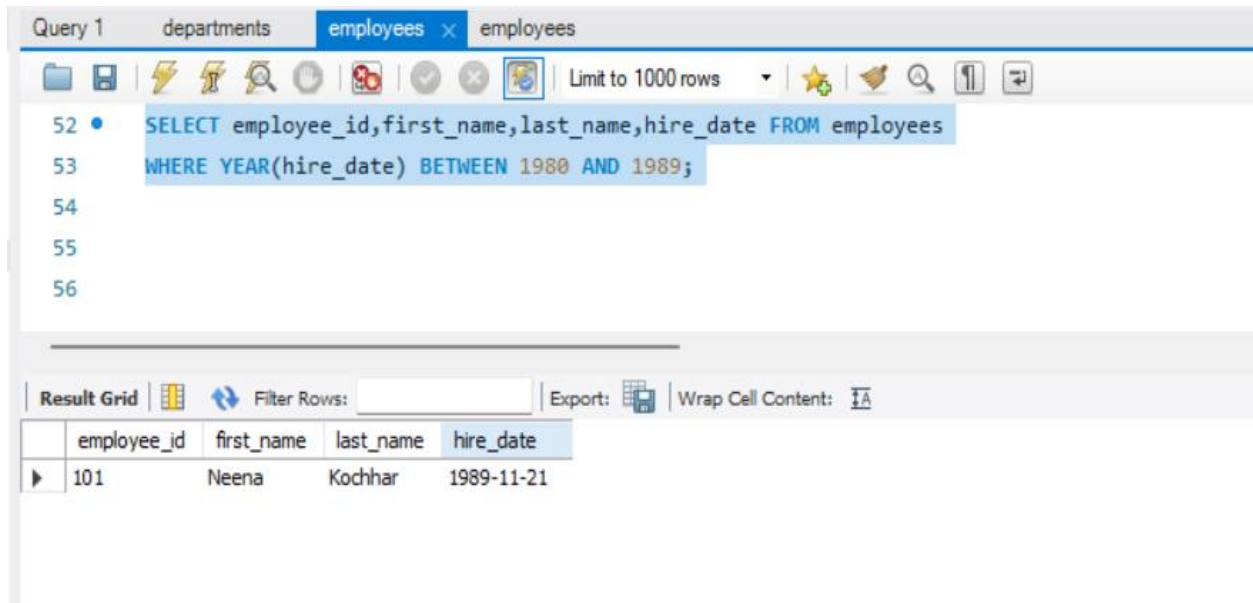
```
SELECT employee_id,first_name,last_name,salary FROM employees  
ORDER BY salary ASC  
LIMIT 5;
```

The result grid displays the following data:

	employee_id	first_name	last_name	salary
▶	128	Steven	Markle	2200.00
	127	James	Landry	2400.00
	119	Karen	Colmenares	2500.00
	118	Guy	Himuro	2600.00
	126	Irene	Mikkilineni	2700.00

16. Find the employees hired in the 80s

```
SELECT employee_id,first_name,last_name,hire_date FROM employees  
  
WHERE YEAR(hire_date) BETWEEN 1980 AND 1989;
```



The screenshot shows a SQL IDE interface. At the top, there are tabs for 'Query 1', 'departments', 'employees' (selected), and 'employees'. Below the tabs is a toolbar with various icons. The main area displays a SQL query:

```
52 • SELECT employee_id,first_name,last_name,hire_date FROM employees  
53 WHERE YEAR(hire_date) BETWEEN 1980 AND 1989;  
54  
55  
56
```

Below the query editor is a 'Result Grid' section. It includes a 'Filter Rows:' input field, an 'Export:' button, and a 'Wrap Cell Content:' checkbox. The result grid shows a single row of data:

	employee_id	first_name	last_name	hire_date
▶	101	Neena	Kochhar	1989-11-21

17. Find the employees who joined the company after 15th of the month

```
SELECT employee_id,first_name,last_name,hire_date FROM employees
```

```
WHERE DAY(hire_date) > 15;
```

The screenshot shows the Oracle SQL Developer interface. At the top, there are tabs for 'Query 1', 'departments', 'employees', and 'employees'. Below the tabs is a toolbar with various icons. The main area displays a SQL query:

```
54  
55 • SELECT employee_id,first_name,last_name,hire_date FROM employees  
56 WHERE DAY(hire_date) > 15;  
57  
58  
59
```

Below the query editor is the 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The result grid displays the following data:

	employee_id	first_name	last_name	hire_date
▶	101	Neena	Kochhar	1989-11-21
	104	Bruce	Ernst	1991-05-21
	105	David	Austin	1997-06-25
	108	Nancy	Greenberg	1994-08-17
	120	Matthew	Weiss	1996-07-18

