

4. The following instruction basically similarly those in the lecture slide. Run and observe the output

a) Aggregate Function (AVG, COUNT, SUM, MAX, MIN)

i) SELECT AVG (salary) FROM employees;

	AVG (salary)
▶	8060.000000

ii) SELECT AVG (salary) FROM employees WHERE job_id = 6;

	AVG (salary)
▶	7920.000000

iii)

```
SELECT
    COUNT(*) as employee_count
FROM
    employees
WHERE
    job_id = 9;
```

	employee_count
▶	5




iii)

```
SELECT SUM(salary)
FROM employees;
```



	SUM(salary)
▶	322400.00

Repeat practise (i) but list the maximum and minimum salary using MAX, MIN function

```
1 • use hr;
2
3 • SELECT MIN(salary) FROM employees;
```

Result Grid			Filter Rows: <input type="text"/>	Exports: 
	MIN(salary)			
▶	2500.00			

```
1 • use hr;
2
3 • SELECT MAX(salary) FROM employees;
```

Result Grid			Filter Rows: <input type="text"/>	Export
	MAX(salary)			
▶	24000.00			

Repeat practise (ii) but list the average salary of employee in department_id=5

b) Subqueries (Nested query)

i) SELECT employee_id, salary FROM employees WHERE department_id = (SELECT department_id FROM departments WHERE department_name = "Sales")

	employee_id	salary
▶	145	14000.00
	146	13500.00
	176	8600.00
	177	8400.00
	178	7000.00
	179	6200.00
*	NULL	NULL

SELECT employee_id, salary

FROM employees

WHERE department_id = (SELECT department_id FROM departments WHERE department_name = "Human Resource")

	employee_id	salary
▶	203	6500.00
*	NULL	NULL

ii) SELECT employee_id, first_name, last_name FROM employees WHERE department_id IN (SELECT department_id FROM departments WHERE location_id = 1700) ORDER BY first_name , last_name;

	employee_id	first_name	last_name
▶	115	Alexander	Khoo
	109	Daniel	Faviet
	114	Den	Raphaely
	118	Guy	Himuro
	111	Ismael	Sciarra
	200	Jennifer	Whalen
	110	John	Chen
	112	Jose Manuel	Urman
	119	Karen	Colmenares
	102	Lex	De Haan
	113	Luis	Popp
	108	Nancy	Greenberg
	101	Neena	Kochhar

iii)

SELECT employee_id, first_name, last_name, salary FROM employees WHERE salary = (SELECT MAX(salary) FROM employees) ORDER BY first_name , last_name;

	employee_id	first_name	last_name	salary
▶	100	Steven	King	24000.00
*	NULL	NULL	NULL	NULL

Repeat practise (iii) but assign salary with minimum salary and then with average salary Use function sum to find the total salary paid according to department.

SELECT employee_id, first_name, last_name, salary FROM employees

WHERE salary = (SELECT MIN(salary) FROM employees) ORDER BY first_name , last_name;

	employee_id	first_name	last_name	salary
▶	119	Karen	Colmenares	2500.00
*	NULL	NULL	NULL	NULL

SELECT employee_id, first_name, last_name, salary FROM employees
WHERE salary = (SELECT AVG(salary) FROM employees) ORDER BY first_name , last_name;

	employee_id	first_name	last_name	salary
	NULL	NULL	NULL	NULL

205 18:19:42 SELECT employee_id, first_name, last_name, salary FROM employees WHERE salary = (SELECT AVG(salary... 0 row(s) returned

0.000 sec / 0.000 sec

c) Join table

SELECT employee_id, first_name, last_name, department_name
FROM employees NATURAL JOIN departments;

	employee_id	first_name	last_name	department_name
▶	200	Jennifer	Whalen	Administration
	201	Michael	Hartstein	Marketing
	202	Pat	Fay	Marketing
	114	Den	Raphaely	Purchasing
	115	Alexander	Khoo	Purchasing
	116	Shelli	Baida	Purchasing
	117	Sigal	Tobias	Purchasing
	118	Guy	Himuro	Purchasing
	119	Karen	Colmenares	Purchasing
	203	Susan	Mavris	Human Resources
	120	Matthew	Weiss	Shipping
	121	Adam	Fripp	Shipping

SELECT employee_id, first_name, last_name, department_name, city
FROM employees NATURAL JOIN departments
NATURAL JOIN locations;

	employee_id	first_name	last_name	department_name	city
▶	103	Alexander	Hunold	IT	Southlake
	104	Bruce	Ernst	IT	Southlake
	105	David	Austin	IT	Southlake
	106	Valli	Pataballa	IT	Southlake
	107	Diana	Lorentz	IT	Southlake
	120	Matthew	Weiss	Shipping	South San Francisco
	121	Adam	Fripp	Shipping	South San Francisco
	122	Payam	Kaufling	Shipping	South San Francisco
	123	Shanta	Vollman	Shipping	South San Francisco
	126	Irene	Mikkilineni	Shipping	South San Francisco
	192	Sarah	Bell	Shipping	South San Francisco
	193	Britney	Everett	Shipping	South San Francisco

SELECT employee_id, first_name, last_name, department_id, department_name
FROM employees JOIN departments USING (department_id);

	employee_id	first_name	last_name	department_id	department_name
▶	200	Jennifer	Whalen	1	Administration
	201	Michael	Hartstein	2	Marketing
	202	Pat	Fay	2	Marketing
	114	Den	Raphaely	3	Purchasing
	115	Alexander	Khoo	3	Purchasing
	116	Shelli	Baida	3	Purchasing
	117	Sigal	Tobias	3	Purchasing
	118	Guy	Himuro	3	Purchasing
	119	Karen	Colmenares	3	Purchasing
	203	Susan	Mavris	4	Human Resources
	120	Matthew	Weiss	5	Shipping
	121	Adam	Fripp	5	Shipping

```

SELECT employees.employee_id, employees.first_name, employees.last_name,
departments.department_id, departments.department_name
FROM employees JOIN departments
ON employees.department_id = departments.department_id;

```

	employee_id	first_name	last_name	department_id	department_name
▶	200	Jennifer	Whalen	1	Administration
	201	Michael	Hartstein	2	Marketing
	202	Pat	Fay	2	Marketing
	114	Den	Raphaely	3	Purchasing
	115	Alexander	Khoo	3	Purchasing
	116	Shelli	Baida	3	Purchasing
	117	Sigal	Tobias	3	Purchasing
	118	Guy	Himuro	3	Purchasing
	119	Karen	Colmenares	3	Purchasing
	203	Susan	Mavris	4	Human Resources
	120	Matthew	Weiss	5	Shipping
	121	Adam	Fripp	5	Shipping

```

SELECT DISTINCT e.manager_id, m.first_name, m.last_name
FROM employees e INNER JOIN employees m
ON e.manager_id = m.employee_id;

```

	manager_id	first_name	last_name
▶	100	Steven	King
	101	Neena	Kochhar
	102	Lex	De Haan
	103	Alexander	Hunold
	108	Nancy	Greenberg
	114	Den	Raphaely
	120	Matthew	Weiss
	123	Shanta	Vollman
	201	Michael	Hartstein
	205	Shelley	Higgins

```

SELECT department_name, first_name
FROM departments LEFT OUTER JOIN employees
ON (departments.department_id = employees.department_id);

```

	department_name	first_name
▶	Administration	Jennifer
	Marketing	Michael
	Marketing	Pat
	Purchasing	Den
	Purchasing	Alexander
	Purchasing	Shelli
	Purchasing	Sigal
	Purchasing	Guy
	Purchasing	Karen
	Human Resources	Susan
	Shipping	Matthew
	Shipping	Adam
	Shipping	Payam

```

SELECT department_name, first_name
FROM departments
RIGHT OUTER JOIN employees
ON (departments.department_id = employees.department_id);

```

	department_name	first_name
▶	Executive	Steven
	Executive	Neena
	Executive	Lex
	IT	Alexander
	IT	Bruce
	IT	David
	IT	Valli
	IT	Diana
	Finance	Nancy
	Finance	Daniel
	Finance	John
	Finance	Ismael
	Finance	Jose Manuel

```

SELECT
first_name, last_name, employees.department_id, departments.department_id,
department_name
FROM employees
INNER JOIN departments ON departments.department_id = employees.department_id
WHERE employees.department_id IN (1 , 2, 3);

```

	first_name	last_name	department_id	department_id	department_name
▶	Jennifer	Whalen	1	1	Administration
	Michael	Hartstein	2	2	Marketing
	Pat	Fay	2	2	Marketing
	Den	Raphaely	3	3	Purchasing
	Alexander	Khoo	3	3	Purchasing
	Shelli	Baida	3	3	Purchasing
	Sigal	Tobias	3	3	Purchasing
	Guy	Himuro	3	3	Purchasing
	Karen	Colmenares	3	3	Purchasing

d) GROUP BY statement

i)

```
SELECT
department_name,
COUNT(employee_id) headcount
FROM
employees e
INNER JOIN departments d ON d.department_id = e.department_id
GROUP BY
department_name;
```

	department_name	headcount
▶	Administration	1
	Marketing	2
	Purchasing	6
	Human Resources	1
	Shipping	7
	IT	5
	Public Relations	1
	Sales	6
	Executive	3
	Finance	6
	Accounting	2

ii)

```
SELECT
department_name,
COUNT(employee_id) headcount
FROM
employees e
INNER JOIN
departments d ON d.department_id = e.department_id
GROUP BY department_name
ORDER BY headcount DESC;
```

	department_name	headcount
▶	Shipping	7
	Purchasing	6
	Sales	6
	Finance	6
	IT	5
	Executive	3
	Marketing	2
	Accounting	2
	Administration	1
	Human Resources	1
	Public Relations	1

e) HAVING statement (observe the different between GROUP BY and HAVING statement)

i)

```
SELECT
  manager_id,
  first_name,
  last_name,
  COUNT(employee_id) direct_reports
FROM
  employees
WHERE
  manager_id IS NOT NULL
GROUP BY manager_id
HAVING direct_reports >= 5;
```

	manager_id	first_name	last_name	direct_reports
▶	100	Neena	Kochhar	14
	101	Nancy	Greenberg	5
	108	Daniel	Faviet	5
	114	Alexander	Khoo	5

ii)

```
SELECT
  department_id, SUM(salary)
FROM
  employees
GROUP BY department_id
HAVING SUM(salary) BETWEEN 20000 AND 30000
ORDER BY SUM(salary);
```

	department_id	SUM(salary)
▶	11	20300.00
	3	24900.00
	6	28800.00

iii)

```
SELECT
  e.department_id,
  department_name,
  ROUND(AVG(salary), 2)
FROM
  employees e
INNER JOIN departments d ON d.department_id = e.department_id
GROUP BY
  e.department_id
HAVING
  AVG(salary) BETWEEN 5000
  AND 7000
ORDER BY
  AVG(salary);
```

	department_id	department_name	ROUND(AVG(salary), 2)
▶	6	IT	5760.00
	5	Shipping	5885.71
	4	Human Resources	6500.00

f) RANK statement

```
SELECT first_name, last_name, salary,  
RANK() OVER (ORDER BY salary) salary_rank  
FROM employees;
```

	first_name	last_name	salary	salary_rank
▶	Karen	Colmenares	2500.00	1
	Guy	Himuro	2600.00	2
	Irene	Mikkilineni	2700.00	3
	Sigal	Tobias	2800.00	4
	Shelli	Baida	2900.00	5
	Alexander	Khoo	3100.00	6
	Britney	Everett	3900.00	7
	Sarah	Bell	4000.00	8
	Diana	Lorentz	4200.00	9
	Jennifer	Whalen	4400.00	10

g) ROLLUP

```
SELECT  
department_id, SUM(salary) as 'Total_salaryRollup'  
FROM employees  
GROUP BY department_id WITH ROLLUP ;
```

	department_id	Total_salaryRollup
▶	1	4400.00
	2	19000.00
	3	24900.00
	4	6500.00
	5	41200.00
	6	28800.00
	7	10000.00
	8	57700.00
	9	58000.00
	10	51600.00