SQL Intermediate

03 Subquery & Aggregation

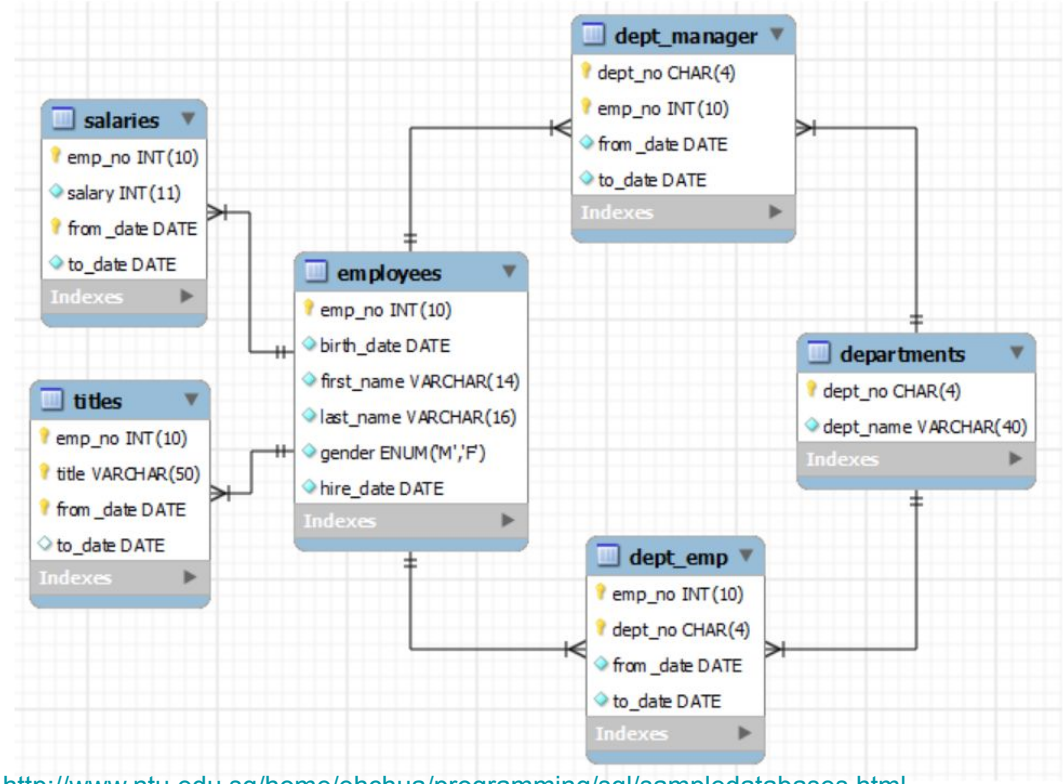
SQL Intermediate

1	1. 연산자
	  1. 비교 연산자
	  2. 산술
	  3. 논리연산자
	  4. 기타연산자 (Like : 특정 패턴을 검색하기 위한 연산자)
	  와일드카드 : 문자열 내에서 임의의 문자나 문자열을 대체하기 위해서 사용하는 기호
	2. 타입변환
	3. 기타함수

1	1. Alias
	2. Joins
	  a. INNER JOIN
	  b. OUTER JOIN
	  c. LEFT JOIN
	  d. RIGHT JOINT

1	**1. SubQuery** **2. Aggregate Functions**
.5	1. Optimization 소개 a. EXPLAIN
	과제 연습

오늘 활용할 테이블



http://www.ntu.edu.sg/home/ehchua/programming/sql/sampledatabases.html

- employees [직원]
- salaries [연봉]
- titles [직급]
- departments [부서]
- dept_emp [부서-직원]
- dept_manager [부서-매니저]

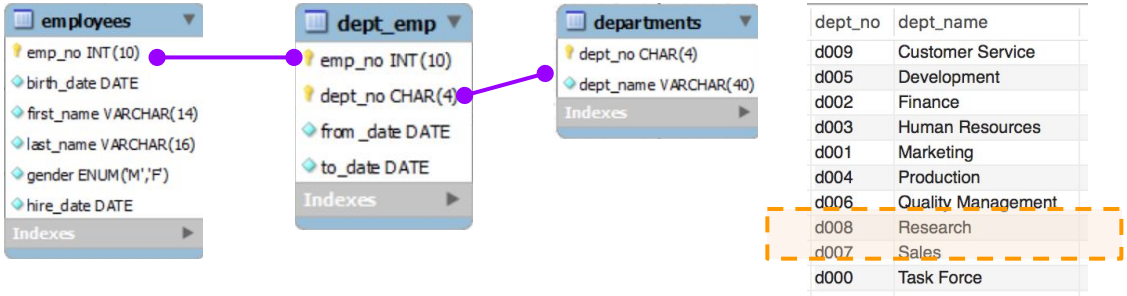
1. Subquery

다른 쿼리 내부에 포함되어있는 SELECT 문을 지칭함.
서브 쿼리는 반드시 ()로 감싸져 있어야만 합니다.

- **Outer query** : subquery를 포함하고 있는 외부 쿼리.
  - SELECT, INSERT, UPDATE, DELETE, SET, DO 문 등이 subquery를 포함 할 수 있음.
- **Sub(Inner) query** : 내부 쿼리

- 장점
  - 쿼리를 구조화 하여 명확히 구분할 수 있게 해준다. **Indentation이 중요!!**
  - 복잡한 JOIN, UNION 동작을 수행할 수 있는 다른 방법을 제공함.
- 단점
  - Subquery 갯수의 제한은 없지만, 너무 많이 사용되면 성능이 떨어지게 됨.

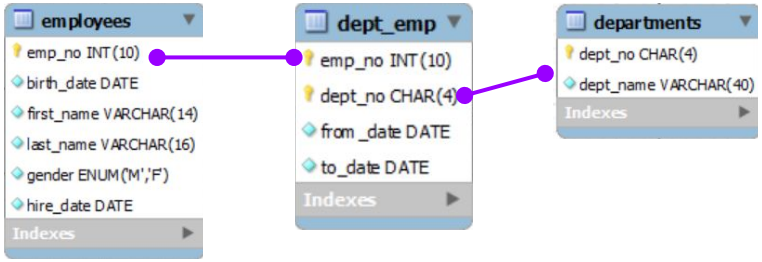
Subquery



1. Sales와 Research 부서 소속 직원 정보만 보고 싶다.

Sales	직원 번호	직원 이름	....
	10002	Mary Kim...	
	…..	...	
Research	…..	…..	…..

Subquery



dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

1. **Sales와 Research 부서 소속 직원 정보만 보고 싶다.**
   a. **Sales 부서번호 찾기 : SELECT dept_no FROM departments WHERE dept_name IN ('Sales', 'Research');**

```
5 •  SELECT dept_no FROM departments WHERE dept_name IN ('Sales', 'Research');
```

dept_no
d008
d007

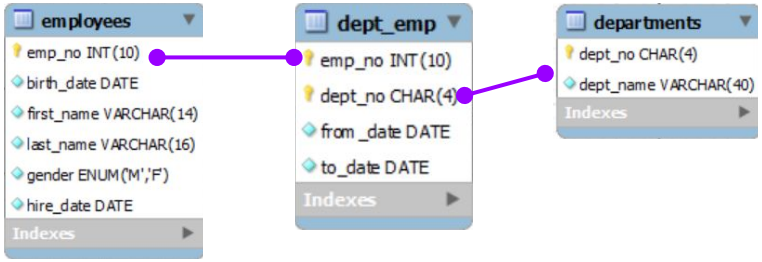
   b. **직원과 부서 정보를 JOIN으로 찾기 :**

```
7 •  SELECT e.emp_no, first_name, last_name, dept_no
8    FROM employees AS e
9    INNER JOIN dept_emp AS d
10   ON e.emp_no = d.emp_no;
```

emp_no	first_name	last_name	dept_no
10001	Georgi	Facello	d005
10002	Bezalel	Simmel	d007
10003	Parto	Bamford	d004
10004	Chirstian	Koblick	d004

```
SELECT *
FROM employees AS e
INNER JOIN dept_emp AS d
ON e.emp_no = d.emp_no;
```

Subquery



dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

1. **Sales와 Research 부서 소속 직원 정보만 보고 싶다.**

   c. **Subquery로 묶기 :**

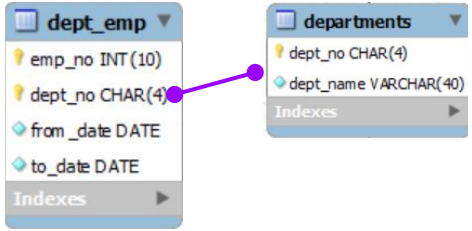
```
12 •  SELECT e.emp_no, first_name, last_name, dept_no
13    FROM employees AS e
14    INNER JOIN dept_emp AS d
15    ON e.emp_no = d.emp_no
16    WHERE d.dept_no IN (SELECT dept_no
17                        FROM departments
18                        WHERE dept_name
19                        IN ('Sales', 'Research'));
```

emp_no	first_name	last_name	dept_no
10007	Tzvetan	Zielinski	d008
10015	Guoxiang	Nooteboom	d008
10019	Lillian	Haddadi	d008
10040	Weiyi	Meriste	d008
10046	Lucien	Rosenbaum	d008
10052	Heping	Nitsch	d008

```
SELECT e.emp_no, first_name, last_name, dept_no   a
FROM employees AS e
INNER JOIN dept_emp AS d
ON e.emp_no = d.emp_no
WHERE d.dept_no IN (SELECT dept_no                 b
                    FROM departments
                    WHERE dept_name
                    IN ('Sales', 'Research'));
```

Subquery가 WHERE 절 내부에

Subquery & Aggregation

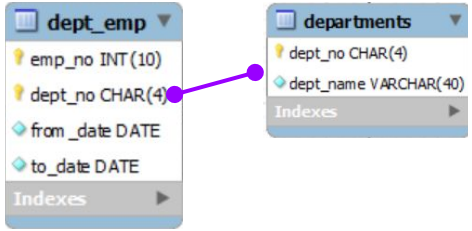


dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

2. 현시점의 각 부서별 직원 수를 확인하고 싶다.

부서	직원수
Sales	20
…..	….

Subquery & Aggregation



dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

2. 현시점의 각 부서별 직원 수를 확인하고 싶다.

a. 하나의 부서에 대해 직원-부서 기록 숫자 세보기 :

SELECT dept_no, COUNT(emp_no) FROM dept_emp WHERE dept_no = 'd009';

COUNT() : 선택된 필드에서 특정 조건을 만족하는 레코드의 **총 개수**를 반환.

```
24 •  select dept_no, count(emp_no)
25    from dept_emp
26    where dept_no='d009';
```

dept_no	count(emp_no)
d009	105

b. 전체 부서에 대해 직원-부서 기록 숫자 세보기 :

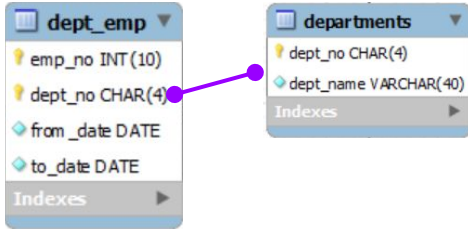
SELECT dept_no, COUNT(emp_no) FROM dept_emp GROUP BY dept_no;

GROUP BY : 선택된 레코드의 집합을 필드의 값이나 표현식에 의해 **그룹화한 결과 집합**을 반환

```
29 •  select dept_no, count(emp_no)
30    from dept_emp
31    group by dept_no;
```

dept_no	count(emp_no)
d001	102
d002	104
d003	102

Subquery & Aggregation



dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

2. 현시점의 각 부서별 직원 수를 확인하고 싶다.

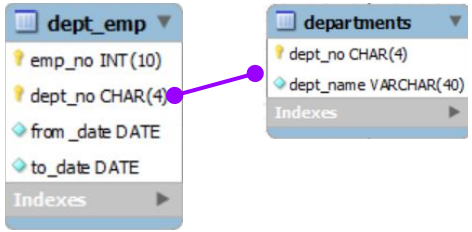
c. 직원-부서 기록 중, 소속 기간이 현재까지 진행 중인 데이터만 찾기 :

```
SELECT *
FROM dept_emp
WHERE to_date >= CURRENT_DATE();
```

```
25 •  select * from dept_emp where to_date >= current_date();
```

emp_no	dept_no	from_date	to_date
10001	d005	1986-06-26	9999-01-01
10002	d007	1996-08-03	9999-01-01
10003	d004	1995-12-03	9999-01-01
10004	d004	1986-12-01	9999-01-01
10005	d003	1989-09-12	9999-01-01

Subquery & Aggregation



dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

2. 현시점의 각 부서별 직원 수를 확인하고 싶다.

d. Subquery로 합치기 : **b**

```
SELECT dept_no, COUNT(emp_no)
FROM (SELECT *
      FROM dept_emp
      WHERE to_date >= CURRENT_DATE()) AS cur_det_emp
GROUP BY dept_no;
```

c

Subquery가 FROM 절 내부에 하나의 테이블로 가정되기 때문에 alias 필요

```
38  select dept_no, count(emp_no)
39  from (
40      select *
41      from dept_emp
42      where to_date >= current_date()
43      ) as cur_det_emp
44  group by dept_no;
```

dept_no	count(emp_no)
d001	75
d002	68
d003	83
d004	84
d005	81
d006	86
d007	83
d008	66
d009	81

departments

dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

추출한 결과

dept_no	count(emp_no)
d001	75
d002	68
d003	83
d004	84
d005	81
d006	86
d007	83
d008	66
d009	81

2. 현시점의 각 부서별 직원 수를 확인하고 싶다.

e. 전체 부서를 표기하고 싶다 :

```
SELECT *
FROM departments AS d
LEFT JOIN (SELECT dept_no, COUNT(emp_no)                                d
           FROM (SELECT *
                 FROM dept_emp
                 WHERE to_date >= CURRENT_DATE()) AS cur_det_emp
           GROUP BY dept_no) AS cnt_det_emp
ON  d.dept_no = cnt_det_emp.dept_no;
```

departments

dept_no	dept_name
d009	Customer Service
d005	Development
d002	Finance
d003	Human Resources
d001	Marketing
d004	Production
d006	Quality Management
d008	Research
d007	Sales
d000	Task Force

추출한 결과

dept_no	count(emp_no)
d001	75
d002	68
d003	83
d004	84
d005	81
d006	86
d007	83
d008	66
d009	81

```
47 • SELECT *
48   FROM departments AS d
49   LEFT JOIN (SELECT dept_no, COUNT(emp_no)
50             FROM (SELECT *
51                   FROM dept_emp
52                   WHERE to_date >= CURRENT_DATE()) AS cur_det_emp
53             GROUP BY dept_no) AS cnt_det_emp
54   ON  d.dept_no = cnt_det_emp.dept_no;
```

dept_no	dept_name	dept_no	COUNT(emp_no)
d009	Customer Service	d009	81
d005	Development	d005	81
d002	Finance	d002	68
d003	Human Resources	d003	83
d001	Marketing	d001	75
d004	Production	d004	84
d006	Quality Management	d006	86
d008	Research	d008	66
d007	Sales	d007	83
d000	Task Force	NULL	NULL

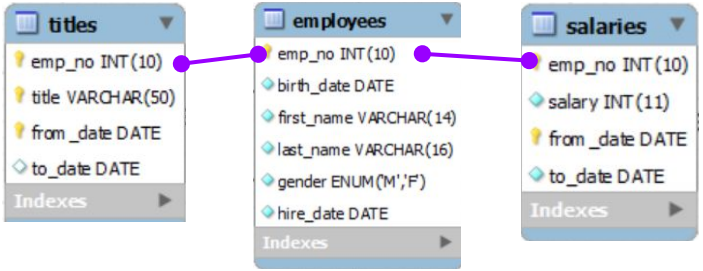
2. 현시점의 각 부서별 직원 수를 확인하고 싶다.

e. 전체 부서를 표기하고 싶다 :

```
SELECT *
FROM departments AS d
LEFT JOIN (SELECT dept_no, COUNT(emp_no)
           FROM (SELECT *
                 FROM dept_emp
                 WHERE to_date >= CURRENT_DATE()) AS cur_det_emp
           GROUP BY dept_no) AS cnt_det_emp
ON  d.dept_no = cnt_det_emp.dept_no;
```

d

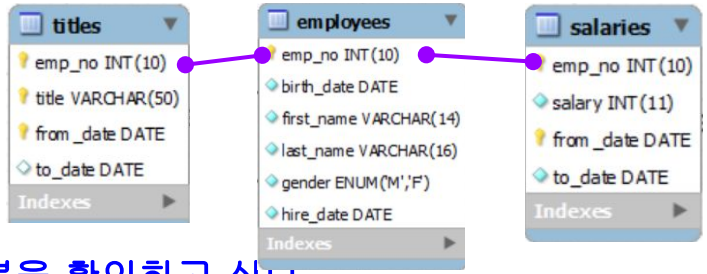
Subquery & Aggregation



3. 현시점의 직원 정보, 직함, 연봉을 확인하고 싶다.

직원명	직함	연봉	.....
Mary Kim	Senior Engineer	8000	...
.....	....	....	....

Subquery & Aggregation



3. 현시점의 직원 정보, 직함, 연봉을 확인하고 싶다.

a. 현시점의 직원과 직함 정보 찾기 : **SELECT ***

FROM employees AS e

INNER JOIN (SELECT *

FROM titles

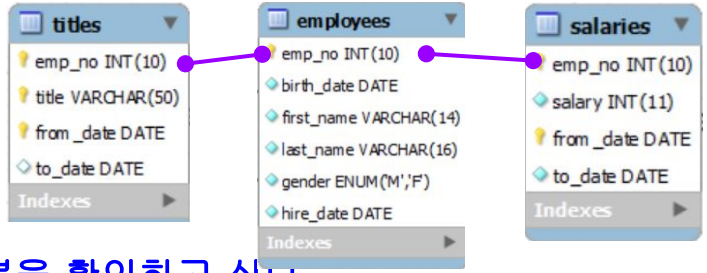
WHERE to_date >= CURRENT_DATE()) AS cur_t

ON e.emp_no = cur_t.emp_no;

```
61 • SELECT *
62   FROM employees AS e
63   INNER JOIN (SELECT *
64               FROM titles
65               WHERE to_date >= CURRENT_DATE()) AS cur_t
66   ON e.emp_no = cur_t.emp_no;
```

emp_no	birth_date	first_name	last_name	gender	hire_date	emp_no	title	from_date	to_date
10001	1953-09-02	Georgi	Facello	M	1986-06-26	10001	Senior Engineer	1986-06-26	9999-01-01
10002	1964-06-02	Bezalel	Simmel	F	1985-11-21	10002	Staff	1996-08-03	9999-01-01
10003	1959-12-03	Parto	Bamford	M	1986-08-28	10003	Senior Engineer	1995-12-03	9999-01-01
10004	1954-05-01	Chirstian	Koblick	M	1986-12-01	10004	Senior Engineer	1995-12-01	9999-01-01
10005	1955-01-21	Kyoichi	Maliniak	M	1989-09-12	10005	Senior Staff	1996-09-12	9999-01-01
10006	1953-04-20	Anneke	Preusig	F	1989-06-02	10006	Senior Engineer	1990-08-05	9999-01-01

Subquery & Aggregation



3. 현시점의 직원 정보, 직함, 연봉을 확인하고 싶다.

b. 현시점의 직원과 연봉 정보 찾기 : **SELECT ***

FROM employees AS e

INNER JOIN (SELECT *

FROM titles

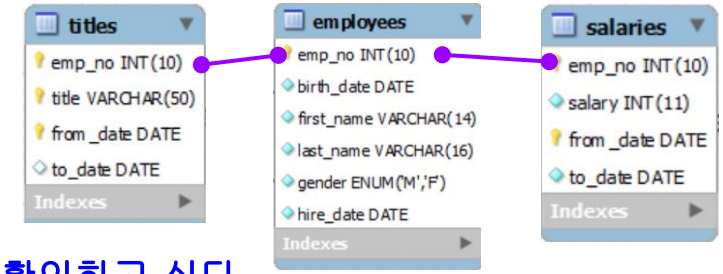
WHERE to_date >= CURRENT_DATE()) AS cur_t

ON e.emp_no = cur_t.emp_no;

```
61 • SELECT *
62   FROM employees AS e
63   INNER JOIN (SELECT *
64               FROM titles
65               WHERE to_date >= CURRENT_DATE()) AS cur_t
66   ON e.emp_no = cur_t.emp_no;
```

emp_no	birth_date	first_name	last_name	gender	hire_date	emp_no	title	from_date	to_date
10001	1953-09-02	Georgi	Facello	M	1986-06-26	10001	Senior Engineer	1986-06-26	9999-01-01
10002	1964-06-02	Bezalel	Simmel	F	1985-11-21	10002	Staff	1996-08-03	9999-01-01
10003	1959-12-03	Parto	Bamford	M	1986-08-28	10003	Senior Engineer	1995-12-03	9999-01-01
10004	1954-05-01	Chirstian	Koblick	M	1986-12-01	10004	Senior Engineer	1995-12-01	9999-01-01
10005	1955-01-21	Kyoichi	Maliniak	M	1989-09-12	10005	Senior Staff	1996-09-12	9999-01-01
10006	1953-04-20	Anneke	Preusig	F	1989-06-02	10006	Senior Engineer	1990-08-05	9999-01-01

Subquery & Aggregation



3. 현시점의 직원 정보, 직함, 연봉을 확인하고 싶다.

c. 앞선 두개의 결과 합치기 :

```
78  SELECT e.emp_no, first_name, last_name, title, salary
79  FROM employees AS e
80  INNER JOIN (SELECT *
81              FROM titles
82              WHERE to_date >= CURRENT_DATE()) AS cur_t
83  ON e.emp_no = cur_t.emp_no
84  INNER JOIN (SELECT *
85              FROM salaries
86              WHERE to_date >= CURRENT_DATE()) as cur_s
87  ON e.emp_no = cur_s.emp_no;
```

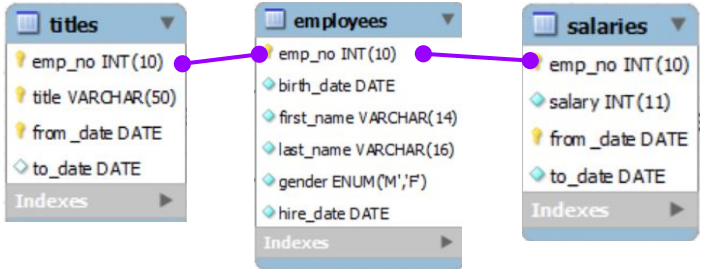
100% 28:87

Result Grid Filter Rows: Search Export:

emp_no	first_name	last_name	title	salary
10001	Georgi	Facello	Senior Engineer	88958
10002	Bezalel	Simmel	Staff	72527
10003	Parto	Bamford	Senior Engineer	43311
10004	Chirstian	Koblick	Senior Engineer	74057

```
SELECT e.emp_no, first_name, last_name, title, salary
FROM employees AS e
INNER JOIN (SELECT *
            FROM titles
            WHERE  to_date >= CURRENT_DATE()) AS cur_t
ON e.emp_no = cur_t.emp_no
INNER JOIN (SELECT *
            FROM salaries
            WHERE  to_date >= CURRENT_DATE()) AS cur_s
ON e.emp_no = cur_s.emp_no;
```

Subquery & Aggregation



- AVG() : 선택된 필드의 평균값 반환

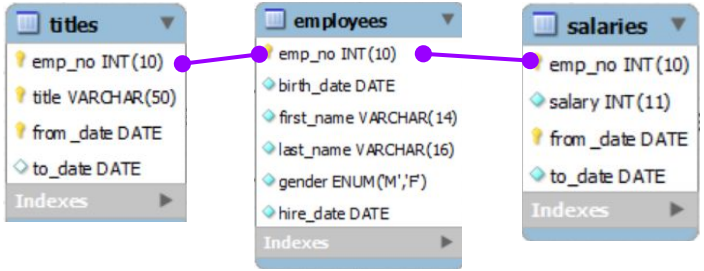
4. 앞선 직함과 연봉 정보를 기반으로 직함별 평균 연봉을 얻고 싶다.

```
SELECT title, AVG(salary)
FROM (기존 추출 쿼리) AS t_avg
GROUP BY title;
```

```
SELECT title, AVG(salary)
FROM (SELECT e.emp_no, first_name, last_name, title, salary
      FROM employees AS e
      INNER JOIN (SELECT *
                  FROM titles
                  WHERE  to_date >= CURRENT_DATE()) AS cur_t
      ON e.emp_no = cur_t.emp_no
      INNER JOIN (SELECT *
                  FROM salaries
                  WHERE  to_date >= CURRENT_DATE()) AS cur_s
      ON e.emp_no = cur_s.emp_no) AS t_avg
GROUP BY title;
```

emp_no	first_name	last_name	title	salary
10001	Georgi	Facello	Senior Engineer	88958
10002	Bezalel	Simmel	Staff	72527
10003	Parto	Bamford	Senior Engineer	43311
10004	Chirstian	Koblick	Senior Engineer	74057
10005	Kyoichi	Maliniak	Senior Staff	94692
10006	Anneke	Preusig	Senior Engineer	59755
10007	Tzvetan	Zielinski	Senior Staff	88070
10009	Sumant	Peac	Senior Engineer	94409
10010	Duangkaew	Piveteau	Engineer	80324
10012	Patricio	Bridgland	Senior Engineer	54423
10013	Eberhardt	Terkki	Senior Staff	68901
10014	Berni	Genin	Engineer	60598
10016	Kazuhito	Cappelletti	Staff	77935
10017	Cristinel	Bouloucos	Senior Staff	99651
10018	Kazuhide	Peha	Senior Engineer	84672
10019	Lillian	Haddadi	Staff	50032
10020	Mayuko	Warwick	Engineer	47017
10022	Shahaf	Famili	Engineer	41348

Subquery & Aggregation



- AVG() : 선택된 필드의 평균값 반환

4. 앞선 직함과 연봉 정보를 기반으로 직함별 평균 연봉을 얻고 싶다.

SELECT title, AVG(salary)
FROM (기존 추출 쿼리) AS t_avg
GROUP BY title;

```
SELECT title, AVG(salary)
FROM (SELECT e.emp_no, first_name, last_name, title, salary
      FROM employees AS e
      INNER JOIN (SELECT *
                  FROM titles
                  WHERE  to_date >= CURRENT_DATE()) AS cur_t
      ON e.emp_no = cur_t.emp_no
      INNER JOIN (SELECT *
                  FROM salaries
                  WHERE  to_date >= CURRENT_DATE()) AS cur_s
      ON e.emp_no = cur_s.emp_no) AS t_avg
GROUP BY title;
```

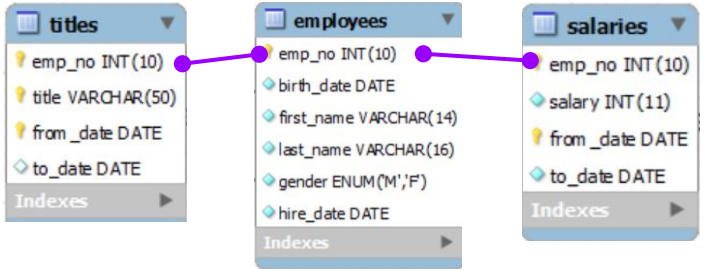
```
92  SELECT title, AVG(salary)
93  FROM (SELECT e.emp_no, first_name, last_name, title, salary
94        FROM employees as e
95        INNER JOIN (SELECT *
96                    FROM titles
97                    WHERE to_date >= CURRENT_DATE()) as cur_t
98        ON e.emp_no = cur_t.emp_no
99        INNER JOIN (SELECT *
100                   FROM salaries
101                   WHERE to_date >= CURRENT_DATE()) as cur_s
102       ON e.emp_no = cur_s.emp_no) AS t_avg
103 GROUP BY title;
```

100% 3:89

Result Grid Filter Rows: Search Export:

title	AVG(salary)
Senior Engineer	70320.7826
Staff	64483.3130
Senior Staff	76936.6313
Engineer	61096.0175
Assistant Engineer	63713.6250
Technique Leader	70421.3784
Manager	77723.6667

Subquery & Aggregation



- ORDER BY : 정렬 기준 값 지정.
  ASC : 오름차순 | DESC : 내림차순

5. 직함별 평균 연봉을 높은 순으로 보고 싶다.
(기존 추출 쿼리) ORDER BY AVG(salary);

```
SELECT title, AVG(salary)
FROM (SELECT e.emp_no, first_name, last_name, title, salary
        FROM employees AS e
        INNER JOIN (SELECT *
                    FROM titles
                    WHERE  to_date >= CURRENT_DATE()) AS cur_t
        ON e.emp_no = cur_t.emp_no
        INNER JOIN (SELECT *
                    FROM salaries
                    WHERE  to_date >= CURRENT_DATE()) AS cur_s
        ON e.emp_no = cur_s.emp_no) AS t_avg
GROUP BY title
ORDER BY AVG(salary);
```

```
SELECT title, AVG(salary)
FROM (SELECT e.emp_no, first_name, last_name, title, salary
        FROM employees as e
        INNER JOIN (SELECT *
                    FROM titles
                    WHERE to_date >= CURRENT_DATE()) as cur_t
        ON e.emp_no = cur_t.emp_no
        INNER JOIN (SELECT *
                    FROM salaries
                    WHERE to_date >= CURRENT_DATE()) as cur_s
        ON e.emp_no = cur_s.emp_no) AS t_avg
GROUP BY title
ORDER BY AVG(salary) DESC;
```

100% 21:106

Result Grid Filter Rows: Search Export:

title	AVG(salary)
Manager	77723.6667
Senior Staff	76936.6313
Technique Leader	70421.3784
Senior Engineer	70320.7826
Staff	64483.3130
Assistant Engineer	63713.6250
Engineer	61096.0175