7-3. 서브쿼리 복습

홍형경
chariehong@gmail.com
2020.06

1. 인라인 뷰 (Inline View)

· 메인쿼리의 FROM 절에 위치

· 서브쿼리 자체가 마치 하나의 테이블 처럼 동작

· 서브쿼리가 최종 반환하는 로우와 컬럼, 표현식 수는 1개 이상 가능

· 서브쿼리에 대한 별칭(Alias)은 반드시 명시

· 메인쿼리와 조인조건은 메인 쿼리의 WHERE 절에서 처리가 일반적

1. 인라인 뷰 (Inline View)

· 인라인 뷰가 필요한 이유

- 기존 단일 테이블만 읽어서는 필요한 정보를 가져오기가 어려울 때

  예, 특정 조건으로 집계한 결과와 조인 필요 시

- 여러 개의 테이블을 조인하는 복잡한 쿼리 작성 시, 일단 인라인뷰 형태로 일부를 작성한 후, 나머지 조인 ➔ **복잡성 해소 측면**

· **LATERAL** 키워드 사용 시 서브쿼리 내에서 조인 가능 ➔ 스칼라 서브쿼리처럼 동작

- 과거 서브쿼리 내에서는 메인 쿼리 참조가 불가능 (조인 불가)
- 12c 부터 추가된 기능
- 서브쿼리 앞에 LATERAL 명시할 경우 메인 쿼리 컬럼 참조 가능

1. 인라인 뷰 - 실습

· 사원번호, 사원명, 부서번호, 부서명 조회 – 조인 사용

```
SELECT a.employee_id,
        a.first_name || a.last_name emp_name,
       a.department_id,
       b.department_name
FROM employees a,
     departments b
 WHERE a.department_id = b.department_id
 ORDER BY 1;
```


1. 인라인 뷰 - 실습

· 사원번호, 사원명, 부서번호, 부서명 조회 – 인라인 뷰 사용

```
SELECT a.employee_id,
        a.first_name || a.last_name emp_name,
       a.department_id,
       c.dept_name
FROM employees a,
       ( SELECT b.department_id,
                b.department_name  dept_name
          FROM departments b ) c
 WHERE a.department_id = c.department_id
ORDER BY 1;
```


1. 인라인 뷰 - 실습

```
-- 조인 사용
SELECT a.employee_id,
       a.first_name || a.last_name emp_name,
       a.department_id,
       b.department_name
FROM employees a,
     departments b
 WHERE a.department_id = b.department_id
 ORDER BY 1;
```

```
-- 인라인 뷰
SELECT a.employee_id,
       a.first_name || a.last_name emp_name,
       a.department_id,
       c.dept_name
FROM employees a,
     ( SELECT b.department_id,
              b.department_name  dept_name
       FROM departments b ) c
 WHERE a.department_id = c.department_id
 ORDER BY 1;
```


1. 인라인 뷰 실습

· 사번, 사원명, 직급명, 부서명, 부서거리명, 부서도시, 국가, 대륙명을 구하라

```
SELECT a.employee_id
      ,a.first_name || ' ' || a.last_name emp_names
      ,b.job_title ,c.department_name
      ,d.street_address, d.city  ,e.country_name ,f.region_name
  FROM employees a,
       jobs b,
       departments c,     locations d,
       countries e,    regions f
 WHERE a.job_id        = b.job_id
   AND a.department_id = c.department_id
   AND c.location_id   = d.location_id
   AND d.country_id    = e.country_id
   AND e.region_id     = f.region_id
 ORDER BY 1;
```

	EMPLOYEE_ID	EMP_NAMES	JOB_TITLE	DEPARTMENT_NAME	STREET_ADDRESS	CITY	COUNTRY_NAME	REGION_NAME
1	100	Steven King	President	Executive	2004 Charade Rd	Seattle	United States of America	Americas
2	101	Neena Kochhar	Administration Vice ...	Executive	2004 Charade Rd	Seattle	United States of America	Americas
3	102	Lex De Haan	Administration Vice ...	Executive	2004 Charade Rd	Seattle	United States of America	Americas
4	103	Alexander Hunold	Programmer	IT	2014 Jabberwocky Rd	Southlake	United States of America	Americas
5	104	Bruce Ernst	Programmer	IT	2014 Jabberwocky Rd	Southlake	United States of America	Americas
6	105	David Austin	Programmer	IT	2014 Jabberwocky Rd	Southlake	United States of America	Americas
7	106	Valli Pataballa	Programmer	IT	2014 Jabberwocky Rd	Southlake	United States of America	Americas
8	107	Diana Lorentz	Programmer	IT	2014 Jabberwocky Rd	Southlake	United States of America	Americas
9	108	Nancy Greenberg	Finance Manager	Finance	2004 Charade Rd	Seattle	United States of America	Americas
10	109	Daniel Faviet	Accountant	Finance	2004 Charade Rd	Seattle	United States of America	Americas
11	110	John Chen	Accountant	Finance	2004 Charade Rd	Seattle	United States of America	Americas
12	111	Ismael Sciarra	Accountant	Finance	2004 Charade Rd	Seattle	United States of America	Americas
13	112	Jose Manuel Urman	Accountant	Finance	2004 Charade Rd	Seattle	United States of America	Americas
14	113	Luis Popp	Accountant	Finance	2004 Charade Rd	Seattle	United States of America	Americas
15	114	Den Raphaely	Purchasing Manager	Purchasing	2004 Charade Rd	Seattle	United States of America	Americas
16	115	Alexander Khoo	Purchasing Clerk	Purchasing	2004 Charade Rd	Seattle	United States of America	Americas
17	116	Shelli Baida	Purchasing Clerk	Purchasing	2004 Charade Rd	Seattle	United States of America	Americas
18	117	Sigal Tobias	Purchasing Clerk	Purchasing	2004 Charade Rd	Seattle	United States of America	Americas
19	118	Guy Himuro	Purchasing Clerk	Purchasing	2004 Charade Rd	Seattle	United States of America	Americas
20	119	Karen Colmenares	Purchasing Clerk	Purchasing	2004 Charade Rd	Seattle	United States of America	Americas
21	120	Matthew Weiss	Stock Manager	Shipping	2011 Interiors Blvd	South San...	United States of America	Americas

1. 인라인 뷰 실습

· departments, locations, countries, regions 조회부분을 별로도 인라인 뷰로 작성

```
SELECT a.employee_id
     ,a.first_name || ' ' || a.last_name emp_names
     ,b.job_title ,k.department_name
     ,k.street_address, k.city
     ,k.country_name ,k.region_name
 FROM employees a,
          jobs b,
          ( SELECT c.department_id, c.department_name,
                   d.street_address, d.city,
                   e.country_name, f.region_name
              FROM departments c,
                   locations d,
                   countries e,
                   regions f
             WHERE c.location_id = d.location_id
               AND d.country_id  = e.country_id
               AND e.region_id   = f.region_id
          ) k
WHERE a.job_id       = b.job_id
  AND a.department_id = k.department_id
ORDER BY 1;
```


1. 인라인 뷰 실습

· 사원의 급여를 **회사 전체 평균 급여**와 해당 사원이 **속한 부서 평균 급여**와 비교하라

사번	이름	급여	회사평균	부서평균	회사평균대비	부서평균대비

1. 인라인 뷰 실습

· 사원의 급여를 **회사 전체 평균 급여**와 해당 사원이 **속한 부서 평균 급여**와 비교하라

· 필요 조건

- 회사 전체 사원의 사번과 급여 조회
- 회사 전체 평균 급여
- 부서 평균 급여

1. 인라인 뷰 실습

· 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

· 조인을 사용해 볼까?

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.salary,
       a.department_id, AVG(a.salary)
  FROM employees a,
       departments b
 WHERE a.department_id = b.department_id
 GROUP BY a.employee_id,
       a.first_name || ' ' || a.last_name ,
       a.salary,
       a.department_id
 ORDER BY 1  ;
```

	EMPLOYEE_ID	EMP_NAME	SALARY	DEPARTMENT_ID	DEPARTMENT_NAME	AVG(A.SALARY)
1	100	Steven King	24000	90	Executive	24000
2	101	Neena Kochhar	17000	90	Executive	17000
3	102	Lex De Haan	17000	90	Executive	17000
4	103	Alexander Hunold	9000	60	IT	9000
5	104	Bruce Ernst	6000	60	IT	6000
6	105	David Austin	4800	60	IT	4800
7	106	Valli Pataballa	4800	60	IT	4800
8	107	Diana Lorentz	4200	60	IT	4200
9	108	Nancy Greenberg	12008	100	Finance	12008
10	109	Daniel Faviet	9000	100	Finance	9000
11	110	John Chen	8200	100	Finance	8200
12	111	Ismael Sciarra	7700	100	Finance	7700
13	112	Jose Manuel Urman	7800	100	Finance	7800
14	113	Luis Popp	6900	100	Finance	6900
15	114	Den Raphaely	11000	30	Purchasing	11000
16	115	Alexander Khoo	3100	30	Purchasing	3100
17	116	Shelli Baida	2900	30	Purchasing	2900
18	117	Sigal Tobias	2800	30	Purchasing	2800

1. 인라인 뷰 실습

· 사원의 급여를 **회사 전체 평균 급여**와 해당 사원이 **속한 부서 평균 급여**와 비교하라

	EMPLOYEE_ID	EMP_NAME	SALARY	DEPARTMENT_ID	DEPARTMENT_NAME	AVG(A.SALARY)
1	100	Steven King	24000	90	Executive	24000
2	101	Neena Kochhar	17000	90	Executive	17000
3	102	Lex De Haan	17000	90	Executive	17000
4	103	Alexander Hunold	9000	60	IT	9000
5	104	Bruce Ernst	6000	60	IT	6000
6	105	David Austin	4800	60	IT	4800
7	106	Valli Pataballa	4800	60	IT	4800
8	107	Diana Lorentz	4200	60	IT	4200
9	108	Nancy Greenberg	12008	100	Finance	12008
10	109	Daniel Faviet	9000	100	Finance	9000
11	110	John Chen	8200	100	Finance	8200
12	111	Ismael Sciarra	7700	100	Finance	7700
13	112	Jose Manuel Urman	7800	100	Finance	7800
14	113	Luis Popp	6900	100	Finance	6900
15	114	Den Raphaely	11000	30	Purchasing	11000
16	115	Alexander Khoo	3100	30	Purchasing	3100
17	116	Shelli Baida	2900	30	Purchasing	2900
18	117	Sigal Tobias	2800	30	Purchasing	2800

· 결과가 이상함

➔ 부서별이 아닌 사원 급여가 그대로 조회

· 부서별 평균급여는 부서별로 집계되어야 하나, 사원별 집계가 됐음

· 사원 리스트와 부서별 급여 평균은 **집계 기준 자체가 다름**

· 조인만 사용해서는 원하는 결과를 얻을 수 없음

1. 인라인 뷰 실습

· 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

· 부서별 평균 급여

```
SELECT department_id, ROUND(AVG(salary),0) dept_avg_sal
  FROM employees
 GROUP BY department_id
 ORDER BY 1;
```

	DEPARTMENT_ID	DEPT_AVG_SAL
1	10	4400
2	20	9500
3	30	4150
4	40	6500
5	50	3476
6	60	5760
7	70	10000
8	80	8956
9	90	19333
10	100	8601
11	110	10154
12	(null)	7000

1. 인라인 뷰 실습

· 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

· 사원 테이블과 부서별 평균 급여 쿼리를 조인 – 서브쿼리 사용

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.department_id,
       a.salary, b.dept_avg_sal
  FROM employees a,
       ( SELECT department_id,
                ROUND(AVG(salary),0) dept_avg_sal
           FROM employees
          GROUP BY department_id
       ) b
 WHERE a.department_id = b.department_id
 ORDER BY 1;
```

	EMPLOYEE_ID	EMP_NAME	DEPARTMENT_ID	SALARY	DEPT_AVG_SAL
1	100	Steven King	90	24000	19333
2	101	Neena Kochhar	90	17000	19333
3	102	Lex De Haan	90	17000	19333
4	103	Alexander Hunold	60	9000	5760
5	104	Bruce Ernst	60	6000	5760
6	105	David Austin	60	4800	5760
7	106	Valli Pataballa	60	4800	5760
8	107	Diana Lorentz	60	4200	5760
9	108	Nancy Greenberg	100	12008	8601
10	109	Daniel Faviet	100	9000	8601
11	110	John Chen	100	8200	8601
12	111	Ismael Sciarra	100	7700	8601
13	112	Jose Manuel Urman	100	7800	8601
14	113	Luis Popp	100	6900	8601
15	114	Den Raphaely	30	11000	4150
16	115	Alexander Khoo	30	3100	4150
17	116	Shelli Baida	30	2900	4150
18	117	Sigal Tobias	30	2800	4150

1. 인라인 뷰 실습

· 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

· 회사 전체 급여 평균도 서브쿼리로 추가

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.department_id,
       a.salary, b.dept_avg_sal,    c.all_avg_sal
  FROM employees a,
       ( SELECT department_id,
                ROUND(AVG(salary),0) dept_avg_sal
           FROM employees
          GROUP BY department_id    ) b
      ,( SELECT ROUND(AVG(salary),0) all_avg_sal
           FROM employees    ) c
 WHERE a.department_id = b.department_id
 ORDER BY 1  ;
```

	EMPLOYEE_ID	EMP_NAME	DEPARTMENT_ID	SALARY	DEPT_AVG_SAL	ALL_AVG_SAL
1	100	Steven King	90	24000	19333	6462
2	101	Neena Kochhar	90	17000	19333	6462
3	102	Lex De Haan	90	17000	19333	6462
4	103	Alexander Hunold	60	9000	5760	6462
5	104	Bruce Ernst	60	6000	5760	6462
6	105	David Austin	60	4800	5760	6462
7	106	Valli Pataballa	60	4800	5760	6462
8	107	Diana Lorentz	60	4200	5760	6462
9	108	Nancy Greenberg	100	12008	8601	6462
10	109	Daniel Faviet	100	9000	8601	6462
11	110	John Chen	100	8200	8601	6462
12	111	Ismael Sciarra	100	7700	8601	6462
13	112	Jose Manuel Urman	100	7800	8601	6462
14	113	Luis Popp	100	6900	8601	6462
15	114	Den Raphaely	30	11000	4150	6462
16	115	Alexander Khoo	30	3100	4150	6462
17	116	Shelli Baida	30	2900	4150	6462

1. 인라인 뷰 실습

· 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

· 회사 전체 급여 평균 서브쿼리는 단일 값을 반환하므로 스칼라 서브쿼리 형태로 사용 가능

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.department_id,
       a.salary, b.dept_avg_sal,
      ( SELECT ROUND(AVG(salary),0)
          FROM employees    ) all_avg_sal
FROM employees a,
       ( SELECT department_id,
                ROUND(AVG(salary),0) dept_avg_sal
          FROM employees
         GROUP BY department_id    ) b
   WHERE a.department_id = b.department_id
 ORDER BY 1  ;
```

	EMPLOY...	EMP_NAME	DEPARTMENT_ID	SALARY	DEPT_AVG_SAL	ALL_AVG_SAL
1	100	Steven King	90	24000	19333	6462
2	101	Neena Kochhar	90	17000	19333	6462
3	102	Lex De Haan	90	17000	19333	6462
4	103	Alexander Hunold	60	9000	5760	6462
5	104	Bruce Ernst	60	6000	5760	6462
6	105	David Austin	60	4800	5760	6462
7	106	Valli Pataballa	60	4800	5760	6462
8	107	Diana Lorentz	60	4200	5760	6462
9	108	Nancy Greenberg	100	12008	8601	6462
10	109	Daniel Faviet	100	9000	8601	6462
11	110	John Chen	100	8200	8601	6462
12	111	Ismael Sciarra	100	7700	8601	6462
13	112	Jose Manuel Urman	100	7800	8601	6462
14	113	Luis Popp	100	6900	8601	6462
15	114	Den Raphaely	30	11000	4150	6462
16	115	Alexander Khoo	30	3100	4150	6462
17	116	Shelli Baida	30	2900	4150	6462

1. 인라인 뷰 실습

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       dept.department_name,
       loc.street_address, loc.city, loc.country_name
FROM employees a
   ,( SELECT *
       FROM departments b ) dept
   ,( SELECT l.location_id, l.street_address,
             l.city, c.country_name
        FROM locations l,
             countries c
       WHERE l.country_id = c.country_id
     ) loc
 WHERE a.department_id = dept.department_id
   AND dept.location_id = loc.location_id
 ORDER BY 1;
```

	EMPLOYEE_ID	EMP_NAME	DEPARTMENT_NAME	STREET_ADDRESS	CITY	COUNTRY_NAME
76	175	Alyssa Hutton	Sales	Magdalen Centre, The Oxford Science Park	Oxford	United Kingdom
77	176	Jonathon Taylor	Sales	Magdalen Centre, The Oxford Science Park	Oxford	United Kingdom
78	177	Jack Livingston	Sales	Magdalen Centre, The Oxford Science Park	Oxford	United Kingdom
79	179	Charles Johnson	Sales	Magdalen Centre, The Oxford Science Park	Oxford	United Kingdom
80	180	Winston Taylor	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
81	181	Jean Fleaur	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
82	182	Martha Sullivan	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
83	183	Girard Geoni	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
84	184	Nandita Sarchand	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
85	185	Alexis Bull	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
86	186	Julia Dellinger	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
87	187	Anthony Cabrio	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
88	188	Kelly Chung	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
89	189	Jennifer Dilly	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
90	190	Timothy Gates	Shipping	2011 Interiors Blvd	South San Francisco	United States of America
91	191	Randall Perkins	Shipping	2011 Interiors Blvd	South San Francisco	United States of America

2. 중첩 서브쿼리 (Nested Subquery)

· 메인쿼리의 **WHERE 절에 위치**

· 서브쿼리가 **조건절의 일부**로 사용됨

· 서브쿼리 최종 반환 값과 메인쿼리 테이블의 특정 컬럼 값을 비교 시 사용

· 서브쿼리가 최종 반환하는 로우와 컬럼, 표현식 수는 1개 이상 가능

· 조건절의 일부이므로 서브쿼리에 대한 별칭(Alias) 사용 불가

· 서브쿼리 내에서 메인쿼리와 조인 가능

2. 중첩 서브쿼리 (Nested Subquery)

· 사원이 속한 부서의 모든 데이터를 조회하라

- 부서의 모든 데이터 ➔ departments 테이블
- 사원이 속한 부서 번호 ➔ employees 테이블

· 방법 1 : 부서와 사원 테이블 조인

· 방법 2 : 부서 테이블에서 사원 테이블의 부서 번호가 있는 건만 체크

2. 중첩 서브쿼리 (Nested Subquery)

· 방법 1 : 부서와 사원 테이블 조인

```
SELECT a.department_id, a.department_name,
       a.manager_id, a.location_id
  FROM departments a,
       employees b
 WHERE a.department_id = b.department_id
 ORDER BY 1;
```

※ 데이터는 맞게 가져왔으나 사원 테이블과의 조인으로 인해 조회 결과가 많아졌고, 부서 데이터가 중복 조회됨

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	20	Marketing	201	1800
4	30	Purchasing	114	1700
5	30	Purchasing	114	1700
6	30	Purchasing	114	1700
7	30	Purchasing	114	1700
8	30	Purchasing	114	1700
9	30	Purchasing	114	1700
10	40	Human Resources	203	2400
11	50	Shipping	121	1500
12	50	Shipping	121	1500
13	50	Shipping	121	1500
14	50	Shipping	121	1500
15	50	Shipping	121	1500
16	50	Shipping	121	1500
17	50	Shipping	121	1500

2. 중첩 서브쿼리 (Nested Subquery)

· 방법 2 : 부서 테이블에서 사원 테이블의 부서 번호가 있는 건만 체크

```
SELECT *
  FROM departments
 WHERE department_id IN ( SELECT   department_id
                          FROM employees
                        ) ;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400
50	Shipping	121	1500
60	IT	103	1400
70	Public Relations	204	2700
80	Sales	145	2500
90	Executive	100	1700
100	Finance	108	1700
110	Accounting	205	1700

2. 중첩 서브쿼리 (Nested Subquery)

· IN 대신 EXISTS 연산자 사용

```
SELECT *
  FROM departments a
 WHERE EXISTS
        ( SELECT  1
            FROM employees b
           WHERE a.department_id = b.department_id
        ) ;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
30	Purchasing	114	1700
40	Human Resources	203	2400
50	Shipping	121	1500
60	IT	103	1400
70	Public Relations	204	2700
80	Sales	145	2500
90	Executive	100	1700
100	Finance	108	1700
110	Accounting	205	1700

2. 중첩 서브쿼리 (Nested Subquery)

· 급여가 10000 이상인 사원이 속한 부서 정보 조회

```
SELECT *
  FROM departments a
 WHERE EXISTS ( SELECT 'A'
                  FROM employees b
                 WHERE a.department_id = b.department_id
                   AND b.salary > 10000
              );
```

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	20	Marketing	201	1800
2	30	Purchasing	114	1700
3	80	Sales	145	2500
4	90	Executive	100	1700
5	100	Finance	108	1700
6	110	Accounting	205	1700

※ EXISTS 사용 시, 조건 체크는 서브쿼리의 WHERE 절에서 처리하므로,
서브쿼리 SELECT 절에는 어떤 값을 기술하든지 상관 없음

2. 중첩 서브쿼리 (Nested Subquery)

· 직급별 최소 급여를 받는 사원

```
SELECT employee_id,
       first_name || ' ' || last_name emp_name,
       job_id,
       salary
  FROM employees
 WHERE (job_id, salary ) IN ( SELECT job_id, min_salary
                              FROM jobs)
ORDER BY 1;
```

	EMPLOYEE_ID	EMP_NAME	JOB_ID	SALARY
1	119	Karen Colmenares	PU_CLERK	2500
2	182	Martha Sullivan	SH_CLERK	2500
3	191	Randall Perkins	SH_CLERK	2500

2. 중첩 서브쿼리 (Nested Subquery)

```
SELECT last_name, employee_id
            ,salary + NVL(commission_pct, 0)
            ,job_id, e.department_id
  FROM employees e
            ,departments d
WHERE e.department_id = d.department_id
  AND salary + NVL(commission_pct,0)
            > ( SELECT salary + NVL(commission_pct,0)
                  FROM employees
                 WHERE last_name = 'Pataballa')
ORDER BY last_name, employee_id;
```

	LAST_NAME	EMPLOYEE_ID	TOT_SALARY	JOB_ID	DEPARTMENT_ID
1	Abel	174	11000.3	SA_REP	80
2	Ande	166	6400.1	SA_REP	80
3	Baer	204	10000	PR_REP	70
4	Banda	167	6200.1	SA_REP	80
5	Bates	172	7300.15	SA_REP	80
6	Bernstein	151	9500.25	SA_REP	80
7	Bloom	169	10000.2	SA_REP	80
8	Cambrault	148	11000.3	SA_MAN	80
9	Cambrault	154	7500.2	SA_REP	80
10	Chen	110	8200	FI_ACCOUNT	100
11	De Haan	102	17000	AD_VP	90
12	Doran	160	7500.3	SA_REP	80
13	Ernst	104	6000	IT_PROG	60
14	Errazuriz	147	12000.3	SA_MAN	80
15	Faviet	109	9000	FI_ACCOUNT	100
16	Fay	202	6000	MK_REP	20
17	Fox	170	9600.2	SA_REP	80
18	Fripp	121	8200	ST_MAN	50
19	Gietz	206	8300	AC_ACCOUNT	110
20	Greenberg	108	12008	FI_MGR	100
21	Greene	163	9500.15	SA_REP	80
22	Hall	152	9000.25	SA_REP	80

학습정리

· 서브쿼리는 메인쿼리에 포함된 독립적인 SELECT 문장으로 괄호로 둘러싸인 쿼리를 말한다.

· 스칼라 서브쿼리는 메인 쿼리의 SELECT 절에 위치한 서브쿼리이다.

· 인라인 뷰는 메인 쿼리의 FROM 절에 위치한 서브쿼리이다.

· 중첩 서브쿼리는 메인 쿼리의 WHERE 절에 위치해 조건절의 일부로 사용된다.

