7-1. 서브쿼리 복습

홍형경
chariehong@gmail.com
2021.06

1. 서브쿼리 (Subquery) - 개요

· 일반적인 쿼리(메인, 주 쿼리) 안에 있는 또 다른 쿼리 ➔ 보조, 하위 쿼리

· 메인 쿼리와 서브쿼리가 합쳐져 한 문장을 이룸

· 서브쿼리는 하나의 SELECT 문장으로, 괄호로 둘러싸인 형태

· 메인 쿼리 기준으로 여러 개의 서브 쿼리 사용 가능

1. 서브쿼리 (Subquery) - 종류

· 서브 쿼리 위치에 따라
  - 스칼라 서브쿼리 (Scalar Subquery)
  - 인라인 뷰 (Inline View)
  - 중첩 서브쿼리 (Nested Subquery)

2. 스칼라 서브쿼리 (Scalar Subquery)

· 메인쿼리의 **SELECT 절에 위치**한 서브쿼리

· SELECT 절에서 마치 하나의 **컬럼이나 표현식** 처럼 사용

· 스칼라(Scalar) : 크기만 가지는 값, 양을 의미 (수학, 물리)

· 서브쿼리 수행 결과가 하나의 값이 되므로 스칼라 서브쿼리라고 함(?)

2. 스칼라 서브쿼리 (Scalar Subquery)

· 서브쿼리가 최종 반환하는 로우 수는 **1**개

· 서브쿼리가 최종 반환하는 컬럼이나 표현식도 **1**개

· 서브쿼리에 별칭(Alias)을 주는 것이 일반적 ➔ 하나의 **컬럼 역할**을 하므로

· 서브쿼리 내에서 메인 쿼리와 조인 가능
  - 조인 하는 것이 일반적
  - 조인을 안하면 여러 건이 조회될 가능성이 많음
  - 조인을 한다는 것은 연관성 있는 서브쿼리란 뜻

3. 인라인 뷰 (Inline View)

· 메인쿼리의 **FROM 절**에 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

3. 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
  - 인라인 뷰의 쿼리가 여러 테이블을 조인해 읽어오는 경우가 많음
  - 복잡한 쿼리의 경우, 쿼리 작성을 좀 더 직관적으로 사용하기 위해

· **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

4. 중첩 서브쿼리 (Nested Subquery)

· 메인쿼리의 WHERE 절에 위치

· 서브쿼리가 조건절의 일부로 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

5. 서브쿼리 실습

(1) 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

사번	이름	급여	회사평균	부서평균	회사평균대비	부서평균대비

5. 서브쿼리 실습

(1) 사원의 급여를 회사 전체 평균 급여와 해당 사원이 속한 부서 평균 급여와 비교하라

· 필요 조건
  - 회사 전체 사원의 사번과 급여 조회
  - 회사 전체 평균 급여
  - 부서 평균 급여

5. 서브쿼리 실습

(1) 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

5. 서브쿼리 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

5. 서브쿼리 실습

(1) 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

5. 서브쿼리 실습

(1) 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

5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

· 가장 많은 급여 ➔ MAX(salary)

· 가장 적은 급여 ➔ MIN(salary)

5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

```
SELECT MIN(salary) min_sal,
       MAX(salary) max_sal
  FROM employees;
```

	MIN_SAL	MAX_SAL
1	2100	24000

5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.salary
  FROM employees a
 WHERE a.salary IN ( SELECT MIN(salary) min_sal,
                            MAX(salary) max_sal
                       FROM employees
                   );
```

```
ORA-00913: 값의 수가 너무 많습니다
00913, 00000 -  "too many values"
*Cause:
*Action:
14행, 22열에서 오류 발생
```


5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.salary
  FROM employees a
 WHERE a.salary IN ( SELECT MIN(salary) min_sal
                       FROM employees
                   );
```

	EMPLOYEE_ID	EMP_NAME	SALARY
1	132	TJ Olson	2100

5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.salary
  FROM employees a
 WHERE a.salary IN ( SELECT MIN(salary) min_sal
                       FROM employees
                   )
    OR a.salary IN ( SELECT MAX(salary) min_sal
                       FROM employees
                   );
```

	EMPLOYEE_ID	EMP_NAME	SALARY
1	100	Steven King	24000
2	132	TJ Olson	2100

5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.salary
  FROM employees a
  INNER JOIN  ( SELECT MIN(salary) min_sal,
                       MAX(salary) max_sal
                  FROM employees
              ) b
  ON a.salary = b.min_sal
  OR a.salary = b.max_sal;
```

	EMPLOYEE_ID	EMP_NAME	SALARY
1	100	Steven King	24000
2	132	TJ Olson	2100

5. 서브쿼리 실습

(2) 가장 급여가 많은 사원과 가장 적은 사원 이름과 급여 구하기

```
SELECT a.employee_id,
       a.first_name || ' ' || a.last_name emp_name,
       a.salary
  FROM employees a
  INNER JOIN  ( SELECT MIN(salary) min_sal,
                       MAX(salary) max_sal
                  FROM employees
              ) b
 ON a.salary IN ( b.min_sal, b.max_sal) ;
```

	EMPLOYEE_ID	EMP_NAME	SALARY
1	100	Steven King	24000
2	132	TJ Olson	2100

5. 서브쿼리 실습

(3) 사원에 할당되지 않은 부서 정보 조회

```
SELECT *
  FROM departments
 WHERE department_id NOT IN
                    ( SELECT a.department_id
                        FROM employees a
                    ) ;
```

- 178, Kimberly Grant의 department_id 값이 NULL 이기 때문

5. 서브쿼리 실습

(3) 사원에 할당되지 않은 부서 정보 조회

- department_id NOT IN (10, 20, 30, …, NULL)

- NOT (department_id = 10 OR
  department_id = 20 OR
  …
  department_id = NULL)

- department_id <> 10 AND
  department_id <> 20 AND
  …
  department_id <> NULL

- NULL 비교는 IS NULL, IS NOT NULL

5. 서브쿼리 실습

(3) 사원에 할당되지 않은 부서 정보 조회

```
SELECT *
  FROM departments a
 WHERE NOT EXISTS
                ( SELECT 1
                    FROM employees b
                   WHERE a.department_id = b.department_id
                 )
ORDER BY 1;
```

	DEPARTME...	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	120	Treasury	(null)	1700
2	130	Corporate Tax	(null)	1700
3	140	Control And Credit	(null)	1700
4	150	Shareholder Services	(null)	1700
5	160	Benefits	(null)	1700
6	170	Manufacturing	(null)	1700
7	180	Construction	(null)	1700
8	190	Contracting	(null)	1700
9	200	Operations	(null)	1700
10	210	IT Support	(null)	1700
11	220	NOC	(null)	1700
12	230	IT Helpdesk	(null)	1700
13	240	Government Sales	(null)	1700
14	250	Retail Sales	(null)	1700
15	260	Recruiting	(null)	1700
16	270	Payroll	(null)	1700

5. 서브쿼리 실습

(4) 입사 년도별 사원들의 급여 총액과 전년 대비 증가율을 구하라

입사연도	급여 총액	전년 급여	증감율
2001	17000	0	0
2002	68816	17000	304.8
2003	46500	68816	-32.43
2004	86000	46500	84.95
2005	197900	86000	130.12
2006	121100	197900	-38.81
2007	94900	121100	-21.64
2008	59200	94900	-32.62

5. 서브쿼리 실습

(4) 입사 년도별 사원들의 급여 총액과 전년 대비 증가율을 구하라

```
SELECT TO_CHAR(hire_date, 'YYYY') years, SUM(salary) sal
  FROM employees
  GROUP BY TO_CHAR(hire_date, 'YYYY')
 ORDER BY 1;
```

	YEARS	SAL
1	2001	17000
2	2002	68816
3	2003	46500
4	2004	86000
5	2005	197900
6	2006	121100
7	2007	94900
8	2008	59200

5. 서브쿼리 실습

(4) 입사 년도별 사원들의 급여 총액과 **전년 대비 증가율**을 구하라

➔ (금년 금액 – 전년 금액) / 전년 금액 * 100

➔ 금년도 ROW에서 전년 금액이 필요

입사연도	급여 총액	전년 급여	증감율
2001	17000	0	0
2002	68816	17000	304.8
2003	46500	68816	-32.43
2004	86000	46500	84.95
2005	197900	86000	130.12
2006	121100	197900	-38.81
2007	94900	121100	-21.64
2008	59200	94900	-32.62

5. 서브쿼리 실습

(4) 입사 년도별 사원들의 급여 총액과 전년 대비 증가율을 구하라

```
SELECT ty.years, ty.sal, ly.years, ly.sal
  FROM (
            SELECT TO_NUMBER(TO_CHAR(hire_date, 'YYYY')) years,
                        SUM(salary) sal
              FROM employees
             GROUP BY TO_CHAR(hire_date, 'YYYY')
         ) ty
 LEFT JOIN
        ( SELECT TO_NUMBER(TO_CHAR(hire_date, 'YYYY')) years,
                    SUM(salary) sal
             FROM employees
            GROUP BY TO_CHAR(hire_date, 'YYYY')
        ) ly
   ON ty.years - 1 = ly.years
ORDER BY 1;
```

	YEARS	SAL	YEARS_1	SAL_1
1	2001	17000	(null)	(null)
2	2002	68816	2001	17000
3	2003	46500	2002	68816
4	2004	86000	2003	46500
5	2005	197900	2004	86000
6	2006	121100	2005	197900
7	2007	94900	2006	121100
8	2008	59200	2007	94900

5. 서브쿼리 실습

(4) 입사 년도별 사원들의 급여 총액과 전년 대비 증가율을 구하라

```
SELECT ty.years, ty.sal, NVL(ly.sal,0) pre_sal,
        CASE WHEN NVL(ly.sal,0) = 0 THEN 0
                ELSE ROUND((ty.sal - ly.sal) / ly.sal * 100,2)
        END rates
 FROM ( SELECT TO_NUMBER(TO_CHAR(hire_date, 'YYYY')) years, SUM(salary) sal
          FROM employees
         GROUP BY TO_CHAR(hire_date, 'YYYY')
        ) ty
 LEFT JOIN
    ( SELECT TO_NUMBER(TO_CHAR(hire_date, 'YYYY')) years, SUM(salary) sal
       FROM employees
      GROUP BY TO_CHAR(hire_date, 'YYYY')
    ) ly
  ON ty.years - 1 = ly.years
ORDER BY 1;
```

	YEARS	SAL	PRE_SAL	RATES
1	2001	17000	0	0
2	2002	68816	17000	304.8
3	2003	46500	68816	-32.43
4	2004	86000	46500	84.95
5	2005	197900	86000	130.12
6	2006	121100	197900	-38.81
7	2007	94900	121100	-21.64
8	2008	59200	94900	-37.62

5. 서브쿼리 실습

(4) 입사 년도별 사원들의 급여 총액과 전년 대비 증가율을 구하라

```
WITH cte1 AS (
SELECT  TO_NUMBER(TO_CHAR(hire_date, 'YYYY')) years, SUM(salary) sal
  FROM employees
  GROUP BY TO_CHAR(hire_date, 'YYYY')
),
cte2 AS (
SELECT a.years, a.sal, b.years y2, NVL(b.sal,0) pre_sal
  FROM cte1 a
 LEFT JOIN cte1 b
   ON a.years - 1 = b.years
)
SELECT years, sal, pre_sal,
       CASE WHEN pre_sal = 0 THEN 0
            ELSE ROUND((sal - pre_sal) / pre_sal * 100,2)
       END rates
 FROM cte2
 ORDER BY 1;
```

	YEARS	SAL	PRE_SAL	RATES
1	2001	17000	0	0
2	2002	68816	17000	304.8
3	2003	46500	68816	-32.43
4	2004	86000	46500	84.95
5	2005	197900	86000	130.12
6	2006	121100	197900	-38.81
7	2007	94900	121100	-21.64
8	2008	59200	94900	-37.62

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.

