KH정보교육원

서브쿼리(SUB QUERY)

Oracle - Sub Query

KH정보교육원

SubQuery란?

하나의 SELECT 문장의 절 안에 포함된 또 하나의 SELECT 문장

[전 직원의 평균 급여보다 많은 급여를 받고 있는 직원의 사번, 이름, 직급코드, 급여를 조회하세요.]

```
SELECT EMP_ID,
       EMP_NAME,
       JOB_CODE,
       SALARY
FROM EMPLOYEE
WHERE SARLRY >= (SELECT AVG(SALARY)
                 FROM EMPLOYEE);
```

	EMP_ID	EMP_NAME	JOB_CODE	SALARY
1	200	선동일	J1	8000000
2	201	송종기	J2	6000000
3	202	노옹철	J2	3700000
4	204	유재식	J3	3400000
5	205	정중하	J3	3900000
6	209	심봉선	J3	3500000
7	215	대북혼	J5	3760000
8	217	전지연	J6	3660000

- 서브쿼리는 메인쿼리가 실행되기 이전에 한번만 실행되며, 비교연산자의 오른쪽에 기술해야 하며, 반드시 괄호로 묶어야 한다.
- 서브쿼리와 비교할 항목은 반드시 서브쿼리의 SELECT한 항목의 개수와 자료형을 일치시켜야 한다.

Oracle - Sub Query

KH정보교육원

Sub Query의 유형

1. 단일행 서브쿼리

- 서브쿼리의 조회 결과 값의 개수가 1개 일 때

2. 다중행 서브쿼리

- 서브쿼리의 조회 결과 값의 행이 여러 개 일 때

3. 다중열 서브쿼리

- 서브쿼리의 조회 결과 컬럼의 개수가 여러 개 일 때

4. 다중행 다중열 서브쿼리

- 서브쿼리의 조회 결과 컬럼의 개수와 행의 개수가 여러 개 일 때

5. 상(호연)관 서브쿼리

- 서브쿼리가 만든 결과값을 메인 쿼리가 비교 연산 할 때, 메인 쿼리 테이블의 값이 변경되면 서브쿼리의 결과값도 바뀐다.

6. 스칼라 서브쿼리

- 상관쿼리 이면서 결과값이 한 개인 서브쿼리

Oracle - Sub Query

KH정보교육원

단일행(Single Row) 서브쿼리

[전 직원의 급여 평균보다 급여를 많이 받는 직원의 이름, 직급, 부서, 급여 조회]

```
SELECT EMP_NAME,
       JOB_CODE,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE SALARY >= (SELECT AVG(SALARY)
                 FROM EMPLOYEE)
ORDER BY 2;
```

	EMP_NAME	JOB_CODE	DEPT_CODE	SALARY
1	선동일	J1	D9	8000000
2	송종기	J2	D9	6000000
3	노옹철	J2	D9	3700000
4	유재식	J3	D6	3400000
5	정중하	J3	D6	3900000
6	심봉선	J3	D5	3500000
7	대북혼	J5	D5	3760000
8	전지연	J6	D1	3660000

	AVG(SALARY)
1	3047662.60869565217391304347826086956522

Oracle - Sub Query

KH정보교육원

다중행(Multiple Row) 서브쿼리

[부서별 최고 급여를 받는 직원의 이름, 직급, 부서, 급여 조회]

```
SELECT EMP_NAME,
       JOB_CODE,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE SALARY IN (SELECT MAX(SALARY)
                 FROM EMPLOYEE
                 GROUP BY DEPT_CODE)
ORDER BY 3;
```

** 다중행 서브쿼리 앞에는
일반 비교연산자 사용 불가

** 사용 가능 연산자
IN / NOT IN,
>ANY / <ANY,
>ALL / <ALL,
EXIST / NOT EXIST 등

	EMP_NAME	JOB_CODE	DEPT_CODE	SALARY
1	전지연	J6	D1	3660000
2	이중석	J4	D2	2490000
3	대북혼	J5	D5	3760000
4	정중하	J3	D6	3900000
5	장쯔위	J6	D8	2550000
6	선동일	J1	D9	8000000
7	이오리	J7	(null)	2890000

	MAX(SALARY)
1	2890000
2	3660000
3	8000000
4	3760000
5	3900000
6	2490000
7	2550000

Oracle - Sub Query

KH정보교육원

FROM절에서의 서브쿼리(인라인뷰)

FROM절에 서브쿼리를 사용한 것을 인라인뷰(INLINE-VIEW)라고 함.

```
SELECT ROWNUM,
       EMP_NAME,
       SALARY
FROM EMPLOYEE
WHERE ROWNUM <= 5;
```

	ROWNUM	EMP_NAME	SALARY
1	1	선동일	8000000
2	2	송종기	6000000
3	3	노옹철	3700000
4	5	유재식	3400000
5	4	송은희	2800000

** ROWNUM은 FROM절을 수행 하면서 붙여지기 때문에 top-N분석시 SELECT절에 사용한 ROWNUM이 의미 없게 됨

```
SELECT ROWNUM,
       EMP_NAME,
       SALARY
FROM (SELECT *
      FROM EMPLOYEE
      ORDER BY SALARY DESC)
WHERE ROWNUM <= 5;
```

	ROWNUM	EMP_NAME	SALARY
1	1	선동일	8000000
2	2	송종기	6000000
3	3	정중하	3900000
4	4	대북혼	3760000
5	5	노옹철	3700000

** FROM절에 이미 정렬이 수행된 서브쿼리(인라인뷰)를 적용시, ROWNUM이 top-N분석에 사용될 수 있음.

Oracle - Sub Query

WITH 구문

서브쿼리에 이름을 붙여 주고, 인라인뷰로 사용시 서브쿼리의 이름으로 FROM절에 기술할 수 있다.

같은 서브쿼리가 여러 번 사용될 경우에 중복 작성을 피할 수 있고 실행 속도도 빨라진다는 장점이 있다.

```
WITH TOPN_SAL AS (SELECT EMP_ID,
                         EMP_NAME,
                         SALARY
                  FROM EMPLOYEE
                  ORDER BY SALARY DESC)
SELECT ROWNUM,
       EMP_NAME,
       SALARY
FROM TOPN_SAL;
```

	ROWNUM	EMP_NAME	SALARY
1	1	선동일	8000000
2	2	송종기	6000000
3	3	정중하	3900000
4	4	대북혼	3760000
5	5	노옹철	3700000
6	6	전지연	3660000
7	7	심봉선	3500000
8	8	유재식	3400000
9	9	이오리	2890000
10	10	송은희	2800000
11	11	차태연	2780000
12	12	장쯔위	2550000
13	13	김해술	2500000
14	14	이중석	2490000
15	15	유하진	2480000
16	16	이태림	2436240
17	17	하동운	2320000
18	18	하이유	2200000
19	19	전형돈	2000000
20	20	윤은해	2000000
21	21	박나라	1800000
22	22	임시환	1550000
23	23	방명수	1380000

Oracle - Sub Query

KH정보교육원

- RANK() OVER 구문

중복 순위 다음은 해당 개수만큼 건너뛰고 반환

즉, 1,2,3,4,5,6,6,6,9,10

```
SELECT 순위, EMP_NAME, SALARY
FROM (SELECT EMP_NAME,
             SALARY,
             RANK() OVER(ORDER BY SALARY DESC) AS 순위
      FROM EMPLOYEE
      ORDER BY SALARY DESC);
```

	순위	EMP_NAME	SALARY
1	1	선동일	8000000
2	2	송종기	6000000
3	3	정중하	3900000
4	4	대북혼	3760000
5	5	노옹철	3700000
6	6	전지연	3660000
7	7	심봉선	3500000
8	8	유재식	3400000
9	9	이오리	2890000
10	10	송은희	2800000
11	11	차태연	2780000
12	12	장쯔위	2550000
13	13	김해술	2500000
14	14	이중석	2490000
15	15	유하진	2480000
16	16	이태림	2436240
17	17	하동운	2320000
18	18	하이유	2200000
19	19	전형돈	2000000
20	19	윤은해	2000000
21	21	박나라	1800000
22	22	임시환	1550000
23	23	방명수	1380000

Oracle - Sub Query

KH정보교육원

- DENSE_RANK() OVER 구문

중복 순위 상관없이 순차적으로 반환

즉, 1,2,3,4,5,6,6,6,7,8,9,10

```
SELECT 순위, EMP_NAME, SALARY
FROM (SELECT EMP_NAME,
             SALARY,
             DENSE_RANK() OVER(ORDER BY SALARY DESC)
                                                AS 순위
      FROM EMPLOYEE
      ORDER BY SALARY DESC);
```

	순위	EMP_NAME	SALARY
1	1	선동일	8000000
2	2	송종기	6000000
3	3	정중하	3900000
4	4	대북혼	3760000
5	5	노옹철	3700000
6	6	전지연	3660000
7	7	심봉선	3500000
8	8	유재식	3400000
9	9	이오리	2890000
10	10	송은희	2800000
11	11	차태연	2780000
12	12	장쯔위	2550000
13	13	김해술	2500000
14	14	이중석	2490000
15	15	유하진	2480000
16	16	이태림	2436240
17	17	하동운	2320000
18	18	하이유	2200000
19	19	전형돈	2000000
20	19	윤은해	2000000
21	20	박나라	1800000
22	21	임시환	1550000
23	22	방명수	1380000

DDL(Data Definition Language)

KH정보교육원

SUBQUERY를 이용한 CREATE TABLE

서브쿼리를 이용해서 SELECT의 조회 결과로 테이블을 생성하는 방법이다. 컬럼명과 데이터타입, 값이 복사되고, 제약조건은 NOT NULL만 복사된다.

```
CREATE TABLE EMPLOYEE_COPY
AS SELECT EMP_ID, EMP_NAME, SALARY, DEPT_TITLE, JOB_NAME
    FROM EMPLOYEE
    LEFT JOIN DEPARTMENT ON(DEPT_CODE = DEPT_ID)
    LEFT JOIN JOB USING(JOB_CODE);
```

SQL | 인출된 모든 행: 23(0.016초)

	EMP_ID	EMP_NAME	SALARY	DEPT_TITLE	JOB_NAME
1	214	방명수	1380000	인사관리부	사원
2	216	차태연	2780000	인사관리부	대리
3	217	전지연	3660000	인사관리부	대리
4	219	임시환	1550000	회계관리부	차장
5	220	이중석	2490000	회계관리부	차장
6	221	유하진	2480000	회계관리부	차장
7	206	박나라	1800000	해외영업1부	사원

.
.
.

21	200	선동일	8000000	총무부	대표
22	218	이오리	2890000	(null)	사원
23	213	하동운	2320000	(null)	대리