SUBQUERY

SUBQUERY

SUBQUERY

SUBQUERY

- 하나의 SELECT 문장 안에 포함된 또 하나의 SELECT 문장
- 서브쿼리는 메인쿼리 실행 전 한번만 실행

서브쿼리 조건

1. 서브쿼리는 반드시 소괄호로 묶어야 함 (SELECT …..)
2. 서브쿼리와 비교할 항목은 서브쿼리의 SELECT한 항목의 개수와 자료형이 일치해야함

SUBQUERY

SUBQUERY 예제

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

서브쿼리를 사용하지 않는 경우

1. 전직원 평균급여를 구한다.
   ```
   SELECT AVG(SALARY) FROM EMPLOYEE; -- 3047662원
   ```
2. 구한 평균급여를 이용하여 쿼리문을 작성한다.
   ```
   SELECT EMP_ID,EMP_NAME,DEPT_CODE,SALARY
   FROM EMPLOYEE
   WHERE SALARY >= 3047662;
   ```

SUBQUERY

SUBQUERY 유형

1. **단일행 서브쿼리**
   - 서브쿼리의 조회 결과 값의 개수가 1개인 경우(1행,1열)
2. **다중행 서브쿼리**
   - 서브쿼리의 조회 결과 값의 행이 여러 개 인 경우(N행,1열)
3. **다중열 서브쿼리**
   - 서브쿼리의 조회 결과 컬럼 개수가 여러 개 인 경우(1행,N열)
4. **다중행 다중열 서브쿼리**
   - 서브쿼리의 조회 결과 컬럼 개수와 행의 개수가 여러 개인 경우(M행,N열)
5. **상관 서브쿼리(상호연관 서브쿼리)**
   - 서브쿼리가 만든 결과 값을 메인쿼리가 비교 연산할 때, 메인 쿼리 테이블의 값이 변경되면 서브쿼리의 결과값도 바뀌는 경우
6. **스칼라 서브쿼리**
   - 상관쿼리이면서 결과값이 1개인 경우

SUBQUERY

1. 단일행 SUBQUERY

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

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

	EMP_ID	EMP_NAME	DEPT_CODE	SALARY
1	200	선동일	D9	8000000
2	201	송송기	D9	6000000
3	202	노옹철	D9	3700000
4	204	유재식	D6	3400000
5	205	정중하	D6	3900000
6	209	심봉선	D5	3500000
7	215	대북혼	D5	3760000
8	217	전지연	D1	3660000

SUBQUERY

2. 다중행 SUBQUERY

[부서별 최고 급여를 받는 직원의 이름,직급,부서,급여 조회]

※ 다중행 서브쿼리는 결과값이 여러 ROW이므로 일반 비교연산자 사용 불가
※ 사용가능 연산자 : IN/NOT IN, ANY, ALL, EXIST

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

	EMP_ID	EMP_NAME	DEPT_CODE	SALARY
1	200	선동일	D9	8000000
2	205	정중하	D6	3900000
3	212	장쯔위	D8	2550000
4	215	대북혼	D5	3760000
5	217	전지연	D1	3660000
6	218	이오리	(null)	2890000
7	220	이중석	D2	2490000

SUBQUERY

2. 다중행 SUBQUERY – 추가 연산자 ANY

ANY : 서브 쿼리의 결과 중에서 하나라도 참이면 참
- > ANY : 최소값 보다 크면 / => ANY : 최소값 보다 크거나 같으면
- < ANY : 최대값 보다 크면 / <= ANY : 최대값 보다 작거나 같으면
- = ANY : IN 과 같은 효과 / != ANY : NOT IN과 같은 효과

```
SELECT EMP_ID, EMP_NAME, DEPT_CODE, SALARY
FROM EMPLOYEE WHERE SALARY > ANY(2000000, 5000000);
```

```
SELECT EMP_ID, EMP_NAME, DEPT_CODE, SALARY
FROM EMPLOYEE WHERE SALARY > 2000000 OR SALARY > 5000000;
```

SUBQUERY

2. 다중행 SUBQUERY – 추가 연산자 ALL

ALL : 서브 쿼리의 결과가 모두 참이면 참
- > ALL : 최대값 보다 크면 / => ALL : 최대값 보다 크거나 같으면
- < ALL : 최소값 보다 크면 / <= ALL : 최소값 보다 작거나 같으면

```
SELECT EMP_ID, EMP_NAME, DEPT_CODE, SALARY
FROM EMPLOYEE WHERE SALARY > ALL(2000000, 5000000);
```

```
SELECT EMP_ID, EMP_NAME, DEPT_CODE, SALARY
FROM EMPLOYEE WHERE SALARY > 2000000 AND SALARY > 5000000;
```

SUBQUERY

2. 다중행 SUBQUERY – 추가 연산자 EXISTS

서브쿼리 결과 중에서 만족하는 값이 하나라도 존재하면 참
- 메인쿼리와 상관이 있는지 여부에 따라 결과가 다름

서브쿼리 수행결과가 3개
메인쿼리와 상관없으므로 전체출력

```
SELECT EMP_NAME,
       MANAGER_ID,
       BONUS
FROM EMPLOYEE E
WHERE EXISTS
 (SELECT EMP_NAME
FROM EMPLOYEE M
WHERE NVL(M.BONUS,0) >= 0.3);
```

서브쿼리 수행결과가 3개
메인쿼리에 일치하는 결과 3개만 출력

```
SELECT EMP_NAME,
       MANAGER_ID,
       BONUS
FROM EMPLOYEE E
WHERE EXISTS
 (SELECT EMP_NAME
FROM EMPLOYEE M
WHERE NVL(E.BONUS,0) >= 0.3);
```

SUBQUERY

3. 다중열 SUBQUERY

[퇴사한 여직원과 같은 부서, 같은 직급에 해당하는 사원의 이름, 직급 부서, 입사일을 조회]

```
SELECT EMP_NAME,
       JOB_CODE,
       DEPT_CODE,
       HIRE_DATE
FROM EMPLOYEE
WHERE
(DEPT_CODE,JOB_CODE)  IN
(SELECT DEPT_CODE,
        JOB_CODE
FROM EMPLOYEE WHERE SUBSTR(EMP_NO,8,1)=2 AND ENT_YN = 'Y');
```

	EMP_NAME	JOB_CODE	DEPT_CODE	HIRE_DATE
1	이태림	J6	D8	97/09/12
2	전형돈	J6	D8	12/12/12
3	장쯔위	J6	D8	15/06/17

SUBQUERY

4. 다중행 다중열 SUBQUERY

[직급별 최소 급여를 받는 직원의 사번, 이름, 직급, 급여 조회]

```
SELECT EMP_ID,
            EMP_NAME,
            JOB_CODE,
            SALARY
FROM EMPLOYEE
WHERE
(JOB_CODE,SALARY)  IN
(SELECT JOB_CODE,  MIN(SALARY)
FROM EMPLOYEE GROUP BY JOB_CODE);
```

	EMP_ID	EMP_NAME	JOB_CODE	SALARY
1	200	선동일	J1	8000000
2	202	노옹철	J2	3700000
3	204	유재식	J3	3400000
4	219	임시환	J4	1550000
5	207	하이유	J5	2200000
6	211	전형돈	J6	2000000
7	214	방명수	J7	1380000

SUBQUERY

5. 상관 SUBQUERY

[관리자가 있는 사원들 중 관리자의 사번이 EMPLOYEE 테이블에 존재하는 직원의 사번인 직원의 사번, 이름, 소속부서, 관리자 사번을 조회]

```
SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       MANAGER_ID
FROM EMPLOYEE E
WHERE
EXISTS
(SELECT EMP_ID
FROM EMPLOYEE M
WHERE E.MANAGER_ID = M.EMP_ID);
```

	EMP_ID	EMP_NAME	DEPT_CODE	MANAGER_ID
1	201	송종기	D9	200
2	202	노옹철	D9	201
3	203	송은희	D6	204
4	204	유재식	D6	200
5	205	정중하	D6	204
6	206	박나라	D5	207
7	207	하이유	D5	200
8	208	김해술	D5	207
9	209	심봉선	D5	207
10	210	윤은해	D5	207
11	211	전형돈	D8	200
12	212	장쯔위	D8	211
13	214	방명수	D1	200
14	216	차태연	D1	214
15	217	전지연	D1	214

SUBQUERY

6. 스칼라 SUBQUERY – SELECT 절

[모든 사원의 사번, 이름, 관리자사번, 관리자명을 조회]

```
SELECT E.EMP_ID,
       E.EMP_NAME,
       E.MANAGER_ID,
NVL(
(SELECT M.EMP_NAME
FROM EMPLOYEE M
WHERE E.MANAGER_ID = M.EMP_ID)
, '없음') AS 관리자명
FROM EMPLOYEE E;
```

	EMP_ID	EMP_NAME	MANAGER_ID	관리자명
1	200	선동일	(null)	없음
2	201	송송기	200	선동일
3	202	노옹철	201	송송기
4	203	송은희	204	유재식
5	204	유재식	200	선동일
6	205	정중하	204	유재식
7	206	박나라	207	하이유
8	207	하이유	200	선동일
9	208	김해술	207	하이유
10	209	심봉선	207	하이유
11	210	윤은해	207	하이유
12	211	전형돈	200	선동일
13	212	장쯔위	211	전형돈
14	213	하동운	(null)	없음
15	214	방명수	200	선동일
16	215	대북혼	(null)	없음
17	216	차태연	214	방명수
18	217	전지연	214	방명수

SUBQUERY

6. 스칼라 SUBQUERY – WHERE 절

[자신이 속한 직급의 평균 급여보다 많이 받는 직원의 이름, 직급, 급여 조회]

```
SELECT EMP_NAME,
       JOB_CODE,
       SALARY
FROM EMPLOYEE E
WHERE SALARY>=
(SELECT AVG(SALARY)
FROM EMPLOYEE M
WHERE E.JOB_CODE = M.JOB_CODE)
;
```

	EMP_NAME	JOB_CODE	SALARY
1	선동일	J1	8000000
2	송송기	J2	6000000
3	송은희	J4	2800000
4	정중하	J3	3900000
5	대북혼	J5	3760000
6	차태연	J6	2780000
7	전지연	J6	3660000
8	이오리	J7	2890000
9	이중석	J4	2490000
10	유하진	J4	2480000

SUBQUERY

6. 스칼라 SUBQUERY – ORDER BY 절

[직원의, 사번, 이름, 부서코드를 부서명 내림차순으로 정렬]

```
SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE
FROM EMPLOYEE
ORDER BY
(SELECT DEPT_TITLE
FROM DEPARTMENT
WHERE
DEPT_CODE = DEPT_ID)
DESC NULLS LAST;
```

	EMP_ID	EMP_NAME	DEPT_CODE
1	219	임시환	D2
2	220	이중석	D2
3	221	유하진	D2
4	204	유재식	D6
5	203	송은희	D6
6	205	정중하	D6
7	206	박나라	D5
8	207	하이유	D5
9	208	김해술	D5
10	209	심봉선	D5
11	210	윤은해	D5
12	215	대북혼	D5
13	202	노옹철	D9
14	200	선동일	D9
15	201	송종기	D9
16	216	차태연	D1
17	217	전지연	D1
18	214	방명수	D1
19	212	장쯔위	D8
20	211	전형돈	D8
21	222	이태림	D8
22	213	하동운	(null)
23	218	이오리	(null)

SUBQUERY

SUBQUERY – FROM 절(인라인 뷰)

[직원들 중 급여를 가장 많이 받는 5명의 순위,이름,급여를 급여순으로 출력]
※ ROWNUM : SELECT되는 행마다 자동으로 순차적을 번호를 붙여 줌

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

	ROWNUM	EMP_N...	SALARY
1	1	선동일	8000000
2	2	송송기	6000000
3	3	정중하	3900000
4	4	대북혼	3760000
5	5	노옹철	3700000

※ columns 에서 가장 큰 N개의 값 또는 가장 작은 N개의 값을 요청할 때 사용되는 기법을 TOP-N 분석기법이라고 함

SUBQUERY

SUBQUERY – FROM 절(인라인 뷰) – WITH

서브쿼리에 이름을 붙여주고, 인라인뷰로 사용 시 서브쿼리 이름으로 FROM절에 기술하는 기법으로 같은 서브쿼리가 여러 번 사용될 경우 중복을 피할 수 있고 실행속도도 빨라진다.

```
WITH TOPN_SAL AS
(SELECT *
FROM EMPLOYEE
ORDER BY SALARY DESC)

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

	ROWNUM	EMP_N...	SALARY
1	1	선동일	8000000
2	2	송송기	6000000
3	3	정중하	3900000
4	4	대북혼	3760000
5	5	노옹철	3700000

SUBQUERY

SUBQUERY – FROM 절(인라인 뷰) – RANK방식

1) RANK() OVER

중복 순위 발생 시 다음은 해당 개수 만큼 건너뛰고 반환
[회사 연봉순위 출력]

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

	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

SUBQUERY

SUBQUERY – FROM 절(인라인 뷰) – RANK방식
2) DENSE_RANK() OVER

중복 순위 발생 시 상관없이 순차적으로 반환
[회사 연봉순위 출력]

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

	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

SUBQUERY

SUBQUERY – FROM 절(인라인 뷰) – RANK방식
3) ROW_NUMBER() OVER

중복 순위 발생 시 상관없이 순차적으로 반환
[회사 연봉순위 출력]

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

	순위	EMP_NAME	SALARY
1	1	선동일	8000000
2	2	송송기	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	하농운	2320000
18	18	하이유	2200000
19	19	선형돈	2000000
20	20	윤은해	2000000
21	21	박나라	1800000
22	22	임시환	1550000
23	23	방명수	1380000