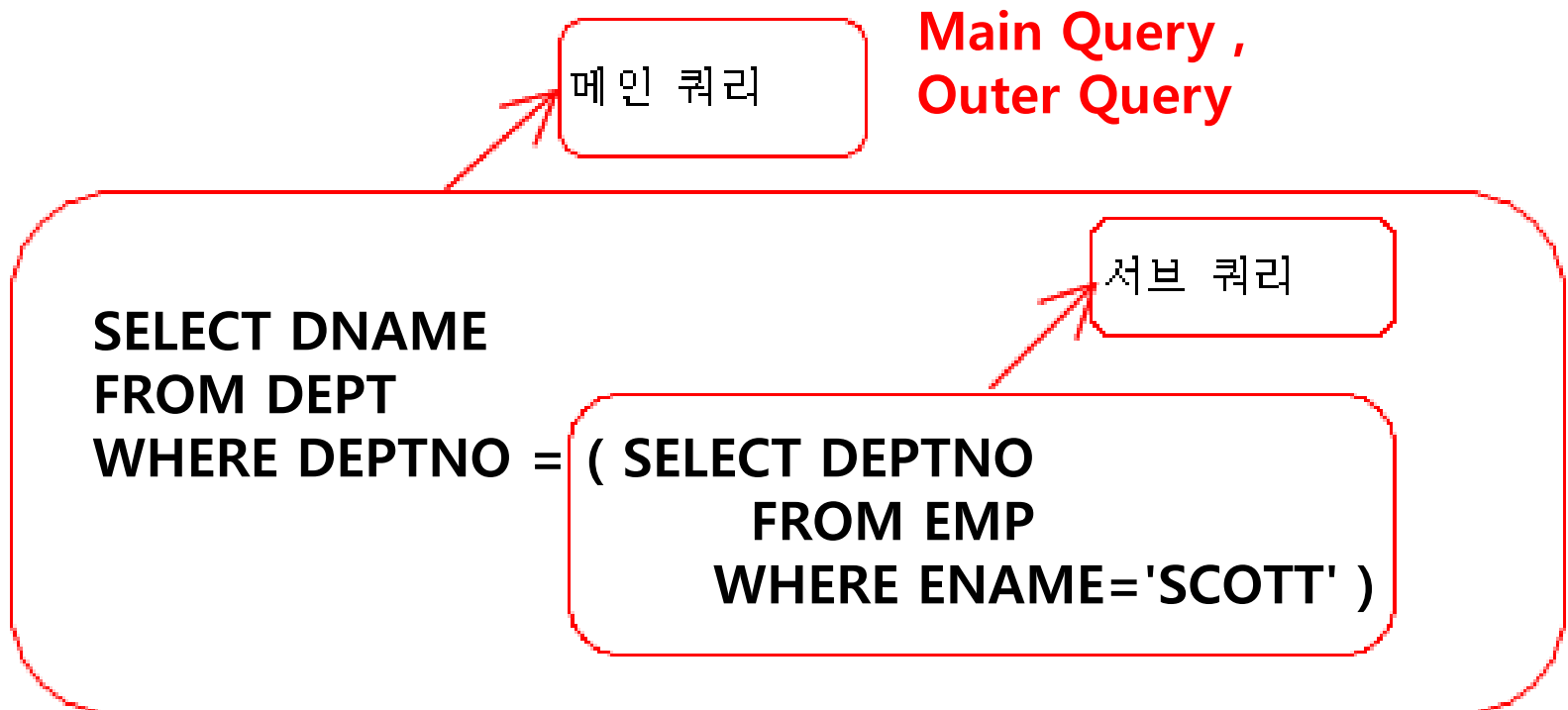


## 07. 서브 쿼리

# 01. 서브 쿼리의 기본 개념

- ❖ SCOTT의 부서명을 알아내기 위한 서브 쿼리문 부터 살펴봅시다.



# 01. 서브 쿼리의 기본 개념

- ❖ 서브 쿼리는 하나의 SELECT 문장의 절 안에 포함된 또 하나의 SELECT 문장입니다.
- ❖ 그렇기에 서브 쿼리를 포함하고 있는 쿼리문을 메인 쿼리, 포함된 또 하나의 쿼리를 서브 쿼리라 합니다.
- ❖ 서브 쿼리는 비교 연산자의 오른쪽에 기술해야 하고 반드시 괄호로 둘러싸아야 합니다.
- ❖ 서브 쿼리는 메인 쿼리가 실행되기 이전에 한번만 실행이 됩니다.

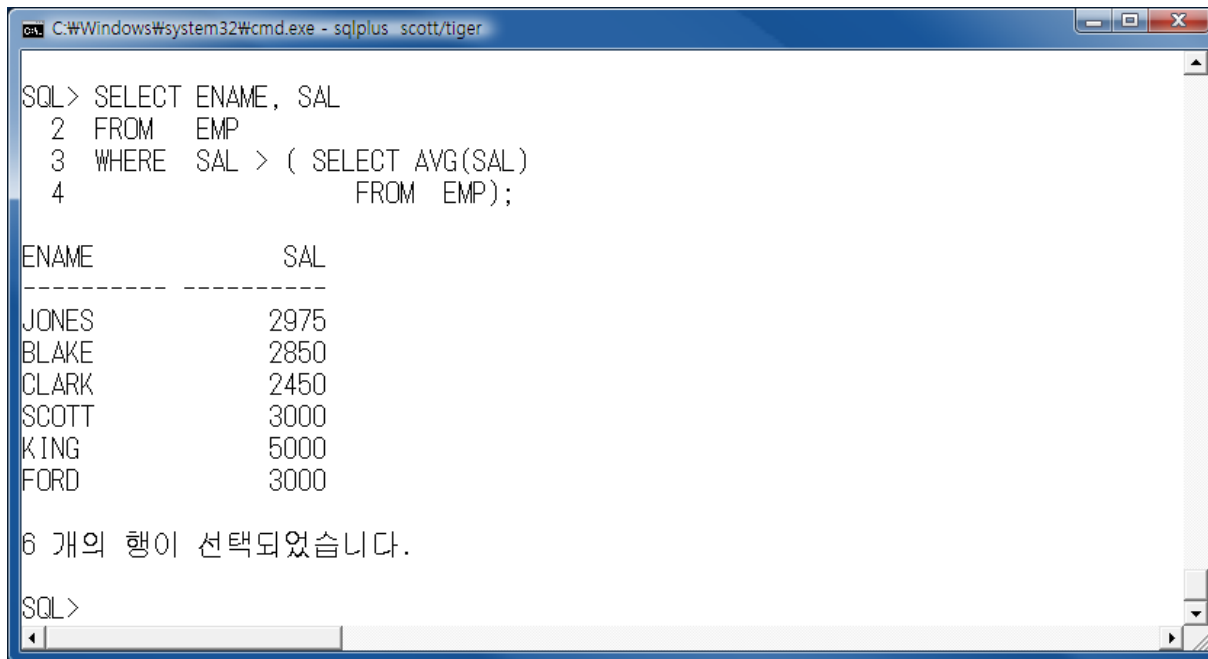
## 02. 단일 행 서브 쿼리

- ❖ 단일 행(Single Row) 서브 쿼리는 수행 결과가 오직 하나의 로우(행, row)만을 반환하는 서브 쿼리를 갖는 것을 말합니다.
- ❖ 단일 행 서브 쿼리문에서는 이렇게 오직 하나의 로우(행, row)로 반환되는 서브 쿼리의 결과는 메인 쿼리에 보내게 되는데 메인 쿼리의 WHERE 절에서는 단일 행 비교 연산자인  $=$ ,  $>$ ,  $>=$ ,  $<$ ,  $<=$ ,  $<>$ 를 사용해야 합니다.

## 03. 서브 쿼리에서 그룹 함수의 사용

- ❖ 평균 급여를 구하는 쿼리문을 서브 쿼리로 사용하여 평균 급여보다 더 많은 급여를 받는 사원을 검색하는 문장은 다음과 같습니다.

```
SELECT ENAME, SAL
FROM EMP
WHERE SAL > ( SELECT AVG(SAL)
               FROM EMP);
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". Inside the window, the following SQL query is entered and executed:

```
SQL> SELECT ENAME, SAL
2  FROM EMP
3  WHERE SAL > ( SELECT AVG(SAL)
4                FROM EMP);
```

The output of the query is displayed as a table:

ENAME	SAL
JONES	2975
BLAKE	2850
CLARK	2450
SCOTT	3000
KING	5000
FORD	3000

Below the table, the message "6 개의 행이 선택되었습니다." (6 rows selected) is displayed. The prompt "SQL>" is visible at the bottom of the window.

# 04. 다중 행 서브 쿼리

❖ 다중 행 서브 쿼리는 서브 쿼리에서 반환되는 결과가 하나 이상의 행일 때 사용하는 서브 쿼리입니다. 다중 행 서브 쿼리는 반드시 다중 행 연산자(Multiple Row Operator)와 함께 사용해야 합니다.

종류	의미
IN	메인 쿼리의 비교 조건('=' 연산자로 비교할 경우)이 서브 쿼리의 결과 중에서 하나라도 일치하면 참입니다.
ANY, SOME	메인 쿼리의 비교 조건이 서브 쿼리의 검색 결과와 하나 이상이 일치하면 참입니다.
ALL	메인 쿼리의 비교 조건이 서브 쿼리의 검색 결과와 모든 값이 일치하면 참입니다.
EXIST	메인 쿼리의 비교 조건이 서브 쿼리의 결과 중에서 만족하는 값이 하나라도 존재하면 참입니다.

## 04. 다중 행 서브 쿼리

- ❖ 결과가 2개 이상 구해지는 쿼리문을 서브 쿼리로 기술할 경우에는 다중 행 연산자와 함께 사용해야 합니다.
- ❖ 주어진 문제가 3000 이상 받는 사원이 소속된 부서(10번, 20번)와 동일한 부서에서 근무하는 사원이기에 서브 쿼리의 결과 중에서 하나라도 일치하면 참인 결과를 구하는 IN 연산자와 함께 사용되어야 합니다.

```
SELECT ENAME, SAL, DEPTNO  
FROM EMP  
WHERE DEPTNO IN ( SELECT DISTINCT DEPTNO  
FROM EMP  
WHERE SAL>=3000);
```

## 4.2 ALL 연산자

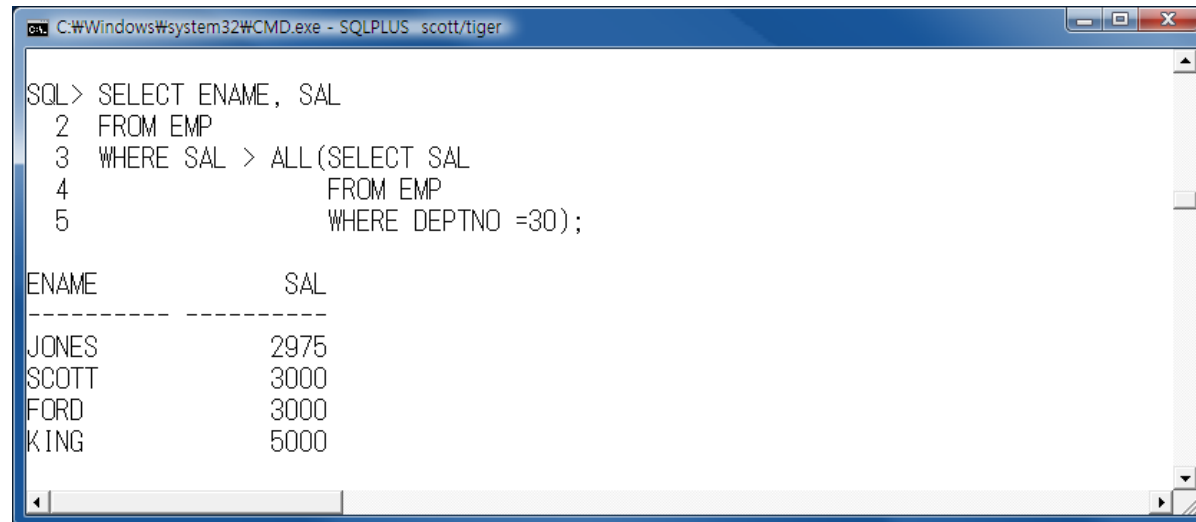
- ❖ ALL 조건은 메인 쿼리의 비교 조건이 서브 쿼리의 검색 결과와 모든 값이 일치하면 참입니다.
- ❖ 찾아진 값에 대해서 AND 연산을 해서 모두 참이면 참이 되는 셈이 됩니다. > ALL 은 “모든 비교값 보다 크냐”고 묻는 것이 되므로 최대값보다 더 크면 참이 됩니다.



## 4.2 ALL 연산자

- ❖ 30번 소속 직원들 중에서 급여를 가장 많이 받는 직원보다 더 많은 급여를 받는 사람의 이름, 급여를 출력하는 쿼리문을 작성해 봅시다.

```
SELECT ENAME, SAL
FROM EMP
WHERE SAL > ALL(SELECT SAL
                  FROM EMP
                  WHERE DEPTNO =30);
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\CMD.exe - SQLPLUS scott/tiger". The user has entered the following SQL query:

```
SQL> SELECT ENAME, SAL
2 FROM EMP
3 WHERE SAL > ALL(SELECT SAL
4                 FROM EMP
5                 WHERE DEPTNO =30);
```

The output of the query is displayed as a table with two columns: ENAME and SAL. The results are as follows:

ENAME	SAL
JONES	2975
SCOTT	3000
FORD	3000
KING	5000

## 4.3 ANY 연산자

- ❖ ANY 조건은 메인 쿼리의 비교 조건이 서브 쿼리의 검색 결과와 하나 이상만 일치하면 참입니다.
- ❖ > ANY는 찾아진 값에 대해서 하나라도 크면 참이 되는 셈이 됩니다. 그러므로 찾아진 값 중에서 가장 작은 값 즉, 최소값 보다 크면 참이 됩니다.

## 4.3 ANY 연산자

- ❖ 다음은 부서번호가 30번인 직원들의 급여 중 가장 작은 값(950)보다 많은 급여를 받는 직원의 이름, 급여를 출력하는 예제를 작성해 봅시다.

```
SELECT ENAME, SAL
FROM EMP
WHERE SAL > ANY ( SELECT SAL
                   FROM EMP
                   WHERE DEPTNO = 30 );
```

# 부속질의

- ❖ 스칼라 부속질의 – SELECT 부속질의
- ❖ 인라인 뷰 – FROM 부속질의
- ❖ 중첩질의 – WHERE 부속질의

# 부속질의

## 부속질의의 종류

명칭	위치	영문 및 동의어	설명
스칼라 부속질의	SELECT 절	scalar subquery	SELECT 절에서 사용되며 단일 값을 반환하기 때문에 스칼라 부속질의라고 함.
인라인 뷰	FROM 절	inline view, table subquery	FROM 절에서 결과를 뷰(view) 형태로 반환하기 때문에 인라인 뷰라고 함.
중첩질의	WHERE 절	nested subquery, predicate subquery	WHERE 절에 술어와 같이 사용되며 결과를 한정시키기 위해 사용됨. 상관 혹은 비상관 형태.