

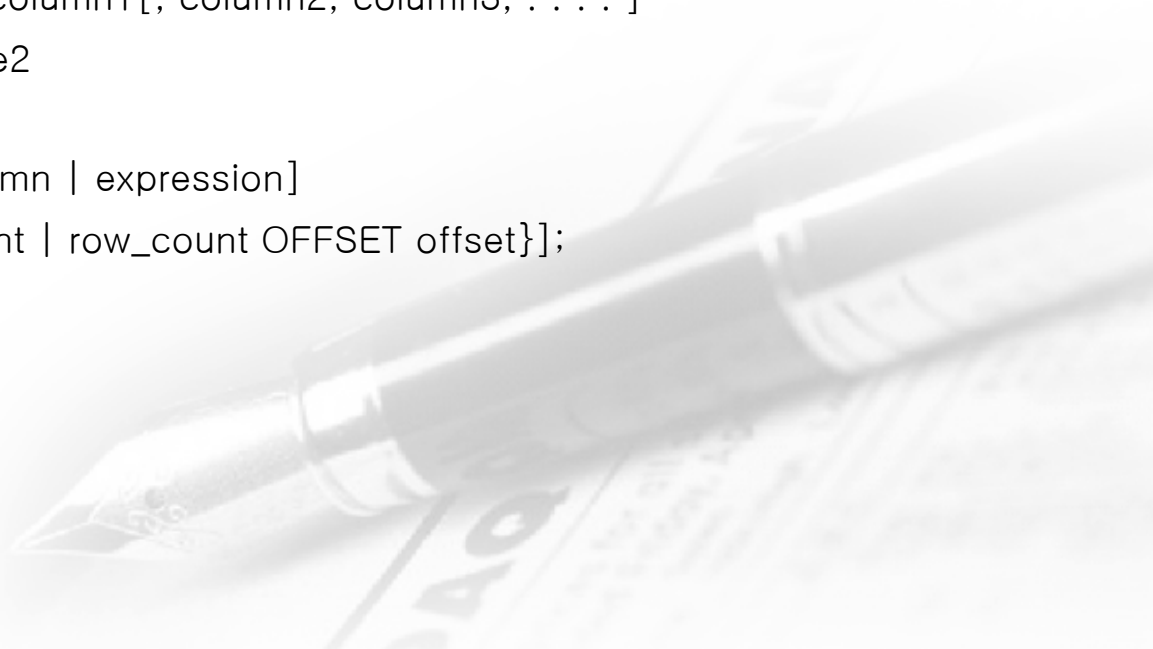
Sub Query & Join

SET OPERATOR

❖ SET 연산

- ✓ 1개 이상의 테이블에서 데이터를 추출하는 방법 중 하나
- ✓ SET 연산자를 이용하여 여러 개의 SELECT 문장을 연결하여 작성
- ✓ Syntax

```
SELECT          * | column1[, column2, column3, . . . . ]
FROM    table1
. . . . .
SET              operator
SELECT          * | column1[, column2, column3, . . . . ]
FROM            table2
. . . . .
[ORDER BY       column | expression]
[LIMIT {[offset,] row_count | row_count OFFSET offset}];
```



SET OPERATOR

❖ Guidelines

- ✓ 첫번째 SELECT 구문에서 기술된 열과 두번째 SELECT 구문에서 기술된 열들은 좌측부터 1대1 대응하며 그 개수와 타입이 일치해야 함
- ✓ FROM 절 뒤에 기술되는 테이블은 같을 수도 있고 다를 수도 있음
- ✓ 출력되는 HARDING은 첫번째 SELECT 구문에서 기술된 열이 출력
- ✓ ORDER BY는 한번만 기술 가능하고 SELECT 구문의 마지막에 기술
- ✓ BLOB, CLOB, BFILE, LONG 형 컬럼에는 사용할 수 없음



SET OPERATOR

❖ SET 연산자의 종류

- ✓ UNION :각 결과의 합(합집합:중복되는 값은 한번 출력)
- ✓ UNION ALL :각 결과의 합(합집합)
- ✓ INTERSECT :각 결과의 중복되는 부분만 출력(교집합)
- ✓ EXCEPT :첫번째 결과에서 두번째 결과를 뺀(차집합 - MINUS)



SET OPERATOR

❖ UNION과 UNION ALL의 차이

- ✓ 양쪽에서 검색된 결과를 모두 출력

```
SELECT DEPTNO  
FROM DEPT  
UNION  
SELECT DEPTNO  
FROM EMP;
```

```
SELECT DEPTNO  
FROM DEPT  
UNION ALL  
SELECT DEPTNO  
FROM EMP;
```



SET OPERATOR

❖ INTERSECT

- ✓ 양쪽에서 모두 검색된 자료만 출력

```
SELECT DEPTNO  
FROM DEPT  
INTERSECT  
SELECT DEPTNO  
FROM EMP;
```



SET OPERATOR

❖ EXCEPT

- ✓ 두번째 SELECT문장에서 검색되지 않았던 데이터를 첫번째 SELECT 문장에서 출력
- ✓ 첫번째 SELECT문장에서 두번째 SELECT문장에서의 값을 뺀 것을 출력
- ✓ MySQL에서는 지원하지 않음

```
SELECT DEPTNO  
FROM DEPT  
EXCEPT  
SELECT DEPTNO  
FROM EMP;
```



Sub Query

❖ Sub Query

- ✓ 하나의 SQL 문장의 절 안에 포함된 SQL
- ✓ Sub Query를 포함하고 있는 쿼리문을 Main Query 라고 하고 포함된 쿼리가 Sub Query
- ✓ Sub Query는 연산자의 오른쪽에 기술해야 하고 반드시 괄호로 감싸야 함
- ✓ Sub Query는 메인 쿼리가 실행되기 이전에 한번만 실행됨
- ✓ 분류
 - FROM 절에 SELECT문을 사용하면 인라인 뷰(View)라고 하고 WHERE 절에 SELECT문을 사용하면 서브 쿼리(Sub Query)
 - Sub Query 가 리턴하는 행의 개수에 따라 단일 행 Sub Query 와 다중 행 Sub Query로 나눔



Sub Query

❖ Sub Query

MILLER의 부서명을 알아내기 위한 Sub Query문

메인 쿼리

서브 쿼리

```
SELECT DNAME  
FROM DEPT  
WHERE DEPTNO = ( SELECT DEPTNO  
                  FROM EMP  
                  WHERE ENAME='MILLER' )
```

Sub Query

❖ 단일 행(Single Row) Sub Query

- ✓ 수행 결과가 하나의 데이터(행, row)만을 반환하는 Sub Query
- ✓ 단일 행 Sub Query문에서는 오직 하나의 데이터(행, row)를 메인 쿼리에 보내게 되는데 메인 쿼리의 WHERE 절에서는 단일 행 비교 연산자인 =, >, >=, <, <=, <> 사용 가능
- ✓ 인구 수가 최대인 도시 이름 구하기

```
SELECT MAX(POPU), NAME  
FROM tCity;
```

- 위의 문장은 에러 - 그룹 함수는 그룹화 하지 않은 컬럼 과 같이 조회할 수 없음
- MySQL 이나 Maria DB에서는 에러가 아님

```
SELECT NAME  
FROM tCity  
WHERE POPU = MAX(POPU);
```

- 위의 문장은 그룹 함수를 where 절에 사용해서 에러
- 인구의 최대값을 먼저 구해서 조회한 뒤 인구 수가 최대값과 동일한 데이터를 조회

```
SELECT MAX(POPU)  
FROM tCity;
```

```
SELECT NAME FROM tCity  
WHERE POPU = 974;
```

Sub Query

❖ 단일 행(Single Row) Sub Query

- ✓ 인구 수가 최대인 도시 이름 구하기 – Sub Query 로 해결

```
SELECT NAME
```

```
FROM tCity
```

```
WHERE POPU = (SELECT MAX(POPU) FROM tCity);
```



Sub Query

❖ 단일 행(Single Row) Sub Query

- ✓ 평균 급여를 구하는 쿼리문을 Sub Query로 사용하여 평균 급여보다 더 많은 급여를 받는 사원을 검색하는 문장

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



Sub Query

❖ 연습문제

- ✓ EMP 테이블에서 ENAME이 MILLER인 데이터와 같은 부서(DEPTNO)에서 근무하는 사원의 이름(ENAME)과 부서 번호(DEPTNO)를 출력하는 SQL 문을 작성

- ✓ EMP 테이블에서 ENAME이 MILLER인 데이터와 동일한 JOB을 가진 사원의 모든 컬럼을 출력하는 SQL 문을 작성



Sub Query

❖ 연습문제

- ✓ EMP 테이블에서 ENAME이 MILLER인 데이터의 급여(SAL)와 동일하거나 더 많이 받는 사원명(ENAME)과 급여(SAL)를 출력
- ✓ EMP 테이블에서 DEPT 테이블의 LOC가 DALLAS인 사원의 이름(ENAME), 부서 번호(DEPTNO)를 출력



Sub Query

❖ 연습문제

- ✓ EMP 테이블에서 DEPT 테이블의 DNAME이 SALES(영업부)인 부서에서 근무하는 사원의 이름(ENAME)과 급여(SAL)를 출력
- ✓ EMP 테이블에서 직속상관(MGR)의 이름이 KING인 사원의 이름(ENAME)과 급여(SAL)를 출력



Sub Query

❖ 다중 열 Sub Query

- ✓ Sub Query 의 결과가 여러 개의 열인 경우
- ✓ tStaff 테이블에서 안중근 과 같은 부서에 근무하고 성별이 같은 직원의 모든 정보를 조회

```
SELECT *  
FROM tStaff  
WHERE DEPART = (SELECT DEPART FROM tStaff WHERE NAME = '안중근')  
AND GENDER = (SELECT GENDER FROM tStaff WHERE NAME = '안중근');
```

```
SELECT *  
FROM tStaff  
WHERE (DEPART, GENDER) = (SELECT DEPART, GENDER FROM tStaff WHERE  
NAME = '안중근');
```



Sub Query

❖ 다중 행 Sub Query

- ✓ Sub Query에서 반환되는 결과가 하나 이상의 행일 때 사용하는 Sub Query
- ✓ 다중 행 Sub Query는 다중 행 연산자(Multiple Row Operator)와 함께 사용

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

Sub Query

❖ IN 연산자

- ✓ IN은 반환되는 여러 개의 행 중에서 하나만 참이 되어도 참이 되는 연산
- ✓ EMP 테이블에서 부서(DEPTNO)별로 가장 급여를 많이 받는 직원들과 동일한 급여를 받는 직원 번호(EMPNO), 직원 이름(ENAME), 급여(SAL), 부서번호(DEPTNO)를 출력

```
SELECT EMPNO, ENAME, SAL, DEPTNO  
FROM EMP  
WHERE SAL = (SELECT MAX(SAL)  
              FROM EMP  
              GROUP BY DEPTNO);
```



SQL Error [1427] [21000]: ORA-01427: 단일 행 하위 질의에 2개 이상의 행이 리턴되었습니다.

세부사항(D) >>



Sub Query

❖ IN 연산자

- ✓ EMP 테이블에서 부서(DEPTNO)별로 가장 급여를 많이 받는 직원들과 동일한 급여를 받는 직원 번호(EMPNO), 직원 이름(ENAME), 급여(SAL), 부서번호(DEPTNO)를 출력(IN 연산자 이용)

```
SELECT EMPNO, ENAME, SAL, DEPTNO
FROM EMP
WHERE SAL IN (SELECT MAX(SAL)
              FROM EMP
              GROUP BY DEPTNO);
```



Sub Query

❖ IN 연산자

- ✓ 결과가 2개 이상 구해지는 쿼리문을 Sub Query로 기술할 경우에는 다중 행 연산자와 함께 사용
- ✓ EMP 테이블에서 SAL이 3000 이상 받는 사원이 소속된 부서(10번, 20번)와 동일한 부서에서 근무하는 사원의 ENAME, SAL, DEPTNO 를 검색

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



Sub Query

❖ IN 연산자

- ✓ EMP 테이블에서 JOB이 MANAGER인 사람이 속한 부서의 부서 번호(DEPTNO)와 부서명(DNAME)과 지역(LOC)을 출력



Sub Query

❖ ALL 연산자

- ✓ ALL 조건은 메인 쿼리의 비교 조건이 Sub Query의 검색 결과와 모든 값이 일치하면 참
- ✓ 찾아진 값에 대해서 AND 연산을 해서 모두 참이면 참
- ✓ > ALL 은 “모든 비교값 보다 크냐”고 묻는 것이 되므로 최대값보다 더 크면 참
- ✓ DEPTNO가 30번인 소속 직원들 중에서 급여를 가장 많이 받는 직원보다 더 많은 급여를 받는 사람의 이름, 급여를 출력하는 쿼리문을 작성

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



Sub Query

❖ ALL 연산자

- ✓ DEPTNO가 30번인 소속 직원들 중에서 급여를 가장 많이 받는 직원보다 더 많은 급여를 받는 사람의 이름, 급여를 출력하는 쿼리문을 작성

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



Sub Query

❖ ANY 연산자

- ✓ ANY 조건은 메인 쿼리의 비교 조건이 Sub Query의 검색 결과와 하나 이상만 일치하면 참
- ✓ > ANY는 찾아진 값에 대해서 하나라도 크면 참인데 찾아진 값 중에서 가장 작은 값 즉 최소 값 보다 크면 참
- ✓ EMP 테이블에서 DEPTNO가 30번인 사원들의 SAL 중 가장 작은 값(950)보다 많은 급여를 받는 사원의 이름, 급여를 조회

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

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


Sub Query

❖ ANY 연산자

- ✓ EMP 테이블에서 DEPTNO가 30번인 직원들의 SAL 중 가장 작은 값(950)보다 많은 급여를 받는 직원의 이름, 급여를 조회



Sub Query

❖ 연습문제

- ✓ EMP 테이블에서 JOB이 SALESMAN 인 사원의 최소 SAL 보다 SAL이 많은 사원들의 ENAME 과 SAL, JOB를 출력하되 영업 사원(SALESMAN)은 출력하지 않도록 작성



Sub Query

❖ EXISTS 연산자

- ✓ EXISTS는 Sub Query로 어떤 데이터 존재 여부를 확인하는 연산자
- ✓ EXISTS의 결과는 참 과 거짓이 반환
- ✓ EMP 테이블에서 SAL이 2000 이 넘는 사원이 있으면 ENAME, SAL을 조회

```
SELECT ENAME, SAL
```

```
FROM EMP
```

```
WHERE EXISTS (SELECT 1 FROM EMP WHERE SAL > 2000);
```



Sub Query

❖ 다중 열 다중 행 서브 쿼리

- ✓ 최고 급여자의 목록 조사하기

```
SELECT *
```

```
FROM tStaff
```

```
WHERE (DEPART, SALARY) IN (SELECT DEPART, MAX(SALARY) FROM tStaff GROUP  
    BY DEPART);
```



Sub Query

❖ 연습문제

1. EMP 테이블에서 ENAME이 BLAKE 인 데이터와 같은 부서(DEPTNO)에 있는 모든 사원의 이름(ENAME)과 입사일자(HIREDATE)를 출력하는 SELECT문을 작성
2. EMP 테이블에서 평균 급여(SAL) 이상을 받는 모든 종업원에 대해서 종업원 번호(EMPNO)와 이름(ENAME)을 출력하는 SELECT문을 작성하시오. 단 급여가 많은 순으로 출력
3. EMP 테이블에서 이름(ENAME)에 “T”가 있는 사원이 근무하는 부서에서 근무하는 모든 종업원에 대해 사원 번호(EMPNO),이름(ENAME),급여(SAL)를 출력하는 SELECT문을 작성하시오. 단 사원번호(EMPNO) 순으로 출력
4. EMP 테이블에서 DEPT 테이블의 LOC 가 DALLAS인 종업원에 대해 이름(ENAME),업무(job), 급여(SAL)를 출력하는 SELECT문을 작성
5. EMP 테이블에서 MGR의 이름(ENAME)이 KING 인 사원의 이름(ENAME)과 급여(SAL)를 출력하는 SELECT문을 작성
6. EMP 테이블에서 월급(SAL)이 DEPTNO가 30인 데이터의 최저 월급보다 높은 사원의 모든 정보를 출력하는 SELECT문을 작성

Sub Query

❖ 연습문제

✓ 다음 중 다중 행 서브 쿼리와 관련이 없는 것은?

- ① ANY
- ② ALL
- ③ =
- ④ IN



JOIN

❖ JOIN이 필요한 상황

- ✓ 특정 부서 번호에 대한 부서 이름이 무엇인지는 부서(DEPT) 테이블에 있는 경우 특정 사원에 대한 부서명을 알아내기 위해서는 부서 테이블에서 정보를 가져와야 함

```
C:\Windows\system32\cmd.exe
SQL> SELECT ENAME, DEPTNO
2 FROM EMP
3 ORDER BY DEPTNO;
```

ENAME	DEPTNO
CLARK	10
KING	10
MILLER	10
JONES	20
FORD	20
ADAMS	20
SMITH	20
SCOTT	20
WARD	30
TURNER	30
ALLEN	30
JAMES	30
BLAKE	30
MARTIN	30

14 개의 행이 선택되었습니다.

```
SQL>
```

```
C:\Windows\system32\cmd.exe
SQL> SELECT DEPTNO, DNAME
2 FROM DEPT;
```

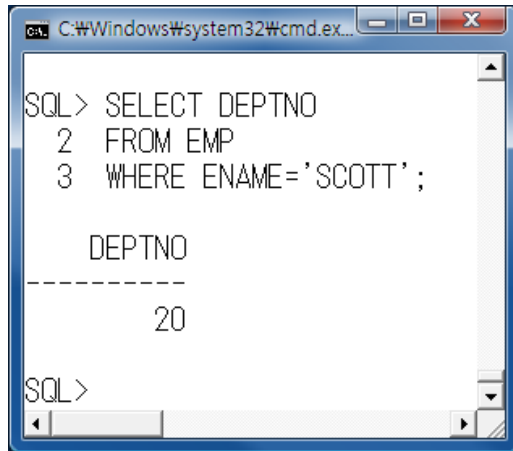
DEPTNO	DNAME
10	ACCOUNTING
20	RESEARCH
30	SALES
40	OPERATIONS

```
SQL>
```

JOIN

❖ JOIN이 필요한 상황

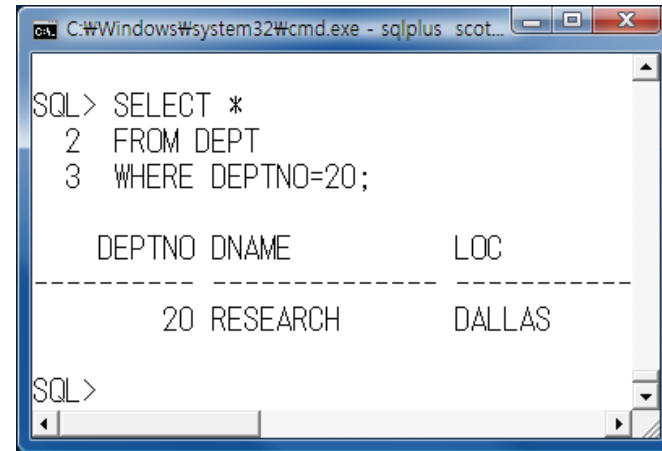
- ✓ MILLER인 사원이 소속되어 있는 부서의 이름이 무엇인지 알아보고자 하는 경우 MILLER란 사원의 부서명을 알아내는 작업 역시 사원 테이블에서 MILLER이 소속된 부서 번호를 알아낸 후에 부서 테이블에서 해당 부서 번호에 대한 부서명을 가져와야 함



```
SQL> SELECT DEPTNO
2 FROM EMP
3 WHERE ENAME='SCOTT';
```

DEPTNO
20

```
SQL>
```



```
SQL> SELECT *
2 FROM DEPT
3 WHERE DEPTNO=20;
```

DEPTNO	DNAME	LOC
20	RESEARCH	DALLAS

```
SQL>
```

- ✓ 원하는 정보가 두 개 이상의 테이블에 나누어져 있다면 위와 같이 여러 번 질의를 하지 않고 SQL에서는 두 개 이상의 테이블을 결합해야만 원하는 결과를 얻을 수 있을 때 한 번의 질의로 원하는 결과를 얻을 수 있는 조인 기능을 제공
- ✓ JOIN은 복수 개의 테이블에서 조건에 맞는 데이터를 조회하는 방법
- ✓ 구문이 길고 처리 시간이 오래 걸려 정확성과 효율성을 항상 고려해야 함

JOIN

❖ 조인의 종류

종 류	설 명
CROSS JOIN	2개 테이블의 모든 조합
EQUI JOIN	동일한 의미를 갖는 컬럼의 값이 같을 때 만 조인
NON-EQUI JOIN	동일한 의미를 갖는 컬럼에 상관없이 = 이외의 연산자를 이용해서 조인
OUTER JOIN	조인 조건에 만족하지 않는 행도 나타남 어느 한쪽 테이블에만 존재하는 데이터도 조인에 참여
SELF JOIN	하나의 테이블을 가지고 조인
SEMI JOIN	서브쿼리를 이용해서 조인을 수행

JOIN

❖ 조인을 위한 샘플 데이터 생성

```
CREATE TABLE TCAR  
(  
    car VARCHAR(30) NOT NULL,          -- 이름  
    capacity INT NOT NULL,             -- 배기량  
    price INT NOT NULL,                -- 가격  
    maker VARCHAR(30) NOT NULL        -- 제조사  
);
```

```
INSERT INTO TCAR (car, capacity, price, maker) VALUES ('소나타', 2000, 2500, '현대');  
INSERT INTO TCAR (car, capacity, price, maker) VALUES ('티볼리', 1600, 2300, '쌍용');  
INSERT INTO TCAR (car, capacity, price, maker) VALUES ('A8', 3000, 4800, 'Audi');  
INSERT INTO TCAR (car, capacity, price, maker) VALUES ('SM5', 2000, 2600, '삼성');
```



JOIN

❖ 조인을 위한 샘플 데이터 생성

```
CREATE TABLE TMAKER
```

```
(
```

```
    maker VARCHAR(30) NOT NULL,
```

```
    factory CHAR(10) NOT NULL,
```

```
    domestic CHAR(1) NOT NULL
```

```
);
```

```
-- 회사
```

```
-- 공장
```

```
-- 국산 여부. Y/N
```

```
INSERT INTO TMAKER (maker, factory, domestic) VALUES ('현대', '부산', 'y');
```

```
INSERT INTO TMAKER (maker, factory, domestic) VALUES ('쌍용', '청주', 'y');
```

```
INSERT INTO TMAKER (maker, factory, domestic) VALUES ('Audi', '독일', 'n');
```

```
INSERT INTO TMAKER (maker, factory, domestic) VALUES ('기아', '서울', 'y');
```



JOIN

❖ CROSS JOIN

- ✓ 특별한 키워드 없이 SELECT 문의 FROM 절에 사원(EMP) 테이블과 부서(DEPT) 테이블을 콤마로 연결하여 연속하여 기술하는 것으로 Cartesian Product 라고도 함

예 SELECT *
FROM EMP, DEPT;

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DEPTNO DNAME	LOC
7369	SMITH	CLERK	7902	80/12/17	800		20	10 ACCOUNTING	NEW YORK
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30	10 ACCOUNTING	NEW YORK
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30	10 ACCOUNTING	NEW YORK
7566	JONES	MANAGER	7839	81/04/02	2975		20	10 ACCOUNTING	NEW YORK
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30	10 ACCOUNTING	NEW YORK
7698	BLAKE	MANAGER	7839	81/05/01	2850		30	10 ACCOUNTING	NEW YORK
7782	CLARK	MANAGER	7839	81/06/08	2450		10	10 ACCOUNTING	NEW YORK
7788	SCOTT	ANALYST	7566	87/04/19	3000		20	10 ACCOUNTING	NEW YORK
7839	KING	PRESIDENT		81/11/17	5000		10	10 ACCOUNTING	NEW YORK
7844	TURNER	SALESMAN	7698	81/09/08	1500		30	10 ACCOUNTING	NEW YORK
7876	ADAMS	CLERK	7788	87/05/23	1100		20	10 ACCOUNTING	NEW YORK
7900	JAMES	CLERK	7698	81/12/03	950		30	10 ACCOUNTING	NEW YORK
7902	FORD	ANALYST	7566	81/12/03	3000		20	10 ACCOUNTING	NEW YORK
7934	MILLER	CLERK	7782	82/01/23	1300		10	10 ACCOUNTING	NEW YORK

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DEPTNO DNAME	LOC
7698	BLAKE	MANAGER	7839	81/05/01	2850		30	30 SALES	CHICAGO
7782	CLARK	MANAGER	7839	81/06/08	2450		10	30 SALES	CHICAGO
7788	SCOTT	ANALYST	7566	87/04/19	3000		20	30 SALES	CHICAGO
7839	KING	PRESIDENT		81/11/17	5000		10	30 SALES	CHICAGO
7844	TURNER	SALESMAN	7698	81/09/08	1500		30	30 SALES	CHICAGO
7876	ADAMS	CLERK	7788	87/05/23	1100		20	30 SALES	CHICAGO
7900	JAMES	CLERK	7698	81/12/03	950		30	30 SALES	CHICAGO
7902	FORD	ANALYST	7566	81/12/03	3000		20	30 SALES	CHICAGO
7934	MILLER	CLERK	7782	82/01/23	1300		10	30 SALES	CHICAGO
7369	SMITH	CLERK	7902	80/12/17	800		20	40 OPERATIONS	BOSTON
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30	40 OPERATIONS	BOSTON
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30	40 OPERATIONS	BOSTON
7566	JONES	MANAGER	7839	81/04/02	2975		20	40 OPERATIONS	BOSTON
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30	40 OPERATIONS	BOSTON
7698	BLAKE	MANAGER	7839	81/05/01	2850		30	40 OPERATIONS	BOSTON
7782	CLARK	MANAGER	7839	81/06/08	2450		10	40 OPERATIONS	BOSTON
7788	SCOTT	ANALYST	7566	87/04/19	3000		20	40 OPERATIONS	BOSTON
7839	KING	PRESIDENT		81/11/17	5000		10	40 OPERATIONS	BOSTON
7844	TURNER	SALESMAN	7698	81/09/08	1500		30	40 OPERATIONS	BOSTON
7876	ADAMS	CLERK	7788	87/05/23	1100		20	40 OPERATIONS	BOSTON
7900	JAMES	CLERK	7698	81/12/03	950		30	40 OPERATIONS	BOSTON
7902	FORD	ANALYST	7566	81/12/03	3000		20	40 OPERATIONS	BOSTON
7934	MILLER	CLERK	7782	82/01/23	1300		10	40 OPERATIONS	BOSTON

JOIN

❖ CROSS JOIN

✓ 조인 결과

- CROSS JOIN의 결과 얻어지는 컬럼의 수는 사원 테이블의 컬럼의 수(8)와 부서 테이블의 컬럼의 수(3)를 더한 것이므로 11
- 행의 수는 사원 한 명에 대해서 DEPT 테이블의 모든 행과 결합되기에 2개 테이블의 행의 개수를 곱한 것 만큼 조회
- CROSS JOIN의 결과를 보면 사원 테이블에 부서에 대한 상세정보가 결합되긴 했지만 조인 될 때 아무런 조건을 제시하지 않았기에 사원 한 명에 대해서 DEPT 테이블의 모든 행의 데이터와 결합된 형태이기에 CROSS JOIN의 결과는 아무런 의미를 갖지 못하는 경우가 많음
- 조인 결과가 의미를 갖으려면 조인할 때 조건을 지정해야 하는 경우가 대부분

✓ 연습문제

- TCAR 테이블 과 TMAKER 테이블을 CROSS JOIN을 수행



JOIN

❖ EQUI JOIN

- ✓ 가장 많이 사용하는 조인 방법으로 조인 대상이 되는 두 테이블에서 동일한 의미를 갖는 컬럼의 값이 일치되는 행을 연결하여 결과를 생성하는 조인 방법
- ✓ 사원 정보를 출력할 때 각 사원들이 소속된 부서의 상세 정보를 출력하기 위해서 두 개의 테이블을 EQUI JOIN 한 경우

예

```
SELECT *  
FROM EMP, DEPT  
WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

- 사원(EMP) 테이블과 부서(DEPT) 테이블의 공통 컬럼인 DEPTNO의 값이 일치(=)되는 조건을 WHERE 절에 기술하여 사용
- 테이블을 조인하려면 일치되는 동일한 의미를 갖는 컬럼을 사용해야 함
- 컬럼의 이름이 같은 경우 컬럼 이름을 구분하기 위해서 컬럼 이름 앞에 테이블 이름을 기술해야 함

JOIN

❖ EQUI JOIN

- ✓ 두 테이블을 조인한 결과로 살펴보면 다음과 같이 부서 번호를 기준으로 같은 값을 가진 사원 테이블의 컬럼과 부서 테이블의 컬럼이 결합

C:\WINDOWS\system32\cmd.exe - sqlplus scott/tiger

```
SQL> SELECT *  
2 FROM EMP, DEPT  
3 WHERE EMP.DEPTNO=DEPT.DEPTNO;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DEPTNO	DNAME	LOC
7782	CLARK	MANAGER	7839	81/06/09	2450		10	10	ACCOUNTING	NEW YORK
7839	KING	PRESIDENT		81/11/17	5000		10	10	ACCOUNTING	NEW YORK
7934	MILLER	CLERK	7782	82/01/23	1300		10	10	ACCOUNTING	NEW YORK
7566	JONES	MANAGER	7839	81/04/02	2975		20	20	RESEARCH	DALLAS
7902	FORD	ANALYST	7566	81/12/03	3000		20	20	RESEARCH	DALLAS
7876	ADAMS	CLERK	7788	87/05/23	1100		20	20	RESEARCH	DALLAS
7369	SMITH	CLERK	7902	80/12/17	800		20	20	RESEARCH	DALLAS
7788	SCOTT	ANALYST	7566	87/04/19	3000		20	20	RESEARCH	DALLAS
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30	30	SALES	CHICAGO
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30	30	SALES	CHICAGO
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30	30	SALES	CHICAGO
7900	JAMES	CLERK	7698	81/12/03	950		30	30	SALES	CHICAGO
7698	BLAKE	MANAGER	7839	81/05/01	2850		30	30	SALES	CHICAGO
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30	30	SALES	CHICAGO

14 개의 행이 선택되었습니다.

JOIN

❖ EQUI JOIN

- ✓ TCAR 테이블 과 TMAKER 테이블을 maker 컬럼을 기준으로 EQUI JOIN을 수행



JOIN

❖ EQUI JOIN

- ✓ EMP 테이블의 ENAME이 MILLER인 사원의 ENAME과 DEPT 테이블의 DNAME을 출력

예

```
SELECT ENAME, DNAME  
FROM EMP, DEPT  
WHERE EMP.DEPTNO=DEPT.DEPTNO  
AND ENAME='MILLER';
```

	ENAME	DNAME
1	MILLER	ACCOUNTING

JOIN

❖ EQUI JOIN

- ✓ 양쪽 테이블에 동일한 컬럼이 존재하는 경우 컬럼 이름을 기재할 때 테이블 이름을 기재해야 하는데 그렇지 않으면 애매한 컬럼 이름이라고 예러가 발생

예

```
SELECT ENAME, DNAME, DEPTNO  
FROM EMP, DEPT  
WHERE EMP.DEPTNO = DEPT.DEPTNO  
AND ENAME='MILLER';
```

SQL Error [1052] [23000]: (conn=12) Column 'DEPTNO' in field list is ambiguous



JOIN

❖ EQUI JOIN

- ✓ 동일한 이름의 컬럼은 컬럼 이름 앞에 테이블 명을 명시적으로 기술함으로써 컬럼이 어느 테이블 소속인지 구분할 수 있음

예

```
SELECT EMP.ENAME, DEPT.DNAME, EMP.DEPTNO  
FROM EMP, DEPT  
WHERE EMP.DEPTNO=DEPT.DEPTNO  
AND ENAME='MILLER';
```



JOIN

❖ 테이블에 다른 이름 부여

- ✓ 테이블 이름을 변경해서 사용할 수 있는데 FROM 절 다음에 테이블 이름을 명시하고 공백을 둔 다음에 다른 이름을 지정
- ✓ 이후 절에서 테이블 이름을 사용할 때는 다른 이름을 사용해야 함

예

```
SELECT E.ENAME, D.DNAME, E.DEPTNO, D.DEPTNO  
FROM EMP E, DEPT D  
WHERE E.DEPTNO = D.DEPTNO  
AND E.ENAME='MILLER';
```



JOIN

❖ 연습문제

1. DEPT 테이블의 LOC가 'NEW YORK' 인 사원의 EMP 테이블의 이름(ENAME)과 급여(SAL)를 조회
2. DEPT 테이블의 DNAME 컬럼의 값이 'ACCOUNTING' 인 사원의 EMP 테이블의 이름(ENAME)과 입사일(HIREDATE)을 조회



JOIN

❖ 연습문제

3. EMP 테이블의 JOB이 'MANAGER'인 사원의 EMP 테이블의 ENAME, DEPT 테이블의 DNAME을 조회



JOIN

❖ HASH JOIN

- ✓ 해시 함수는 테이블을 해시 메모리에 적재한 후에 해시 함수로써 연결하는 방법
- ✓ 해시 조인은 EQUI 조인 만 사용 가능한 방법
- ✓ 해시 조인은 먼저 선행 테이블을 결정하고 선행 테이블에서 주어진 조건(Where 절)에 해당하는 행을 선택하는 방식
- ✓ 해당 행이 선택되면 조인 키(Join Key)를 기준으로 해시 함수를 사용해서 해시 테이블을 메인 메모리(Main Memory)에 생성하고 후행 테이블에서 주어진 조건에 만족하는 행을 찾는 방식
- ✓ 후행 테이블의 조인 키를 사용해서 해시 함수를 적용하여 해당 버킷을 검색하는 방식



JOIN

❖ NON EQUI-JOIN

- ✓ 특정 범위 내에 있는지를 조사하기 위해서 WHERE 절에 조인 조건을 = 연산자 이외의 비교 연산자를 사용
- ✓ 급여 등급 테이블(SALGRADE)을 데이터 확인

예

```
SELECT * FROM SALGRADE;
```



JOIN

❖ NON EQUI-JOIN

- ✓ 급여 등급 테이블(salgrade)에는 급여에 대한 등급을 나누어 놓음
- ✓ 급여의 등급은 총 5등급으로 나누어져 있으며, 1등급은 급여가 700부터 1200 사이이고, 2등급은 1201부터 1400 사이이고, 3등급은 1401부터 2000 사이이고, 4등급은 2001부터 3000 사이이고, 5등급이면 3001부터 9999사이
- ✓ 급여 등급을 5개로 나누어 놓은 salgrade에서 정보를 얻어 와서 각 사원의 급여 등급을 조회하고자 하는 경우에 사원(emp) 테이블과 급여 등급(salgrade) 테이블을 조인해야 함

예

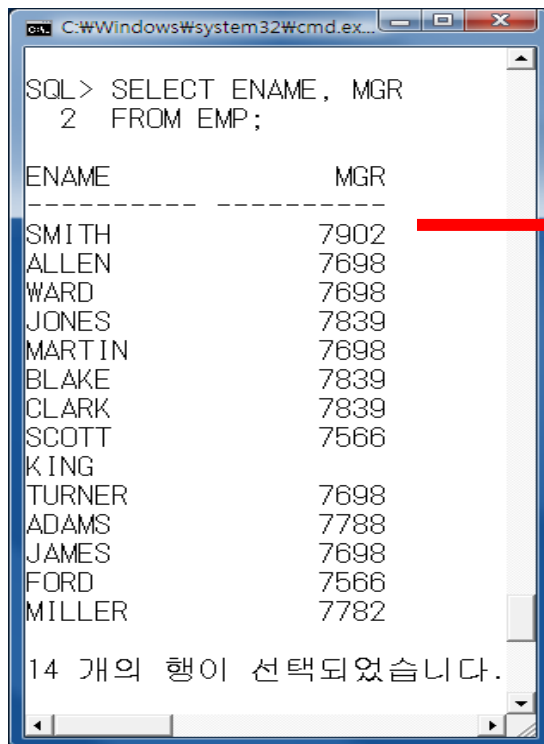
```
SELECT ENAME, SAL, GRADE  
FROM EMP, SALGRADE  
WHERE SAL BETWEEN LOSAL AND HISAL;
```



JOIN

❖ SELF JOIN

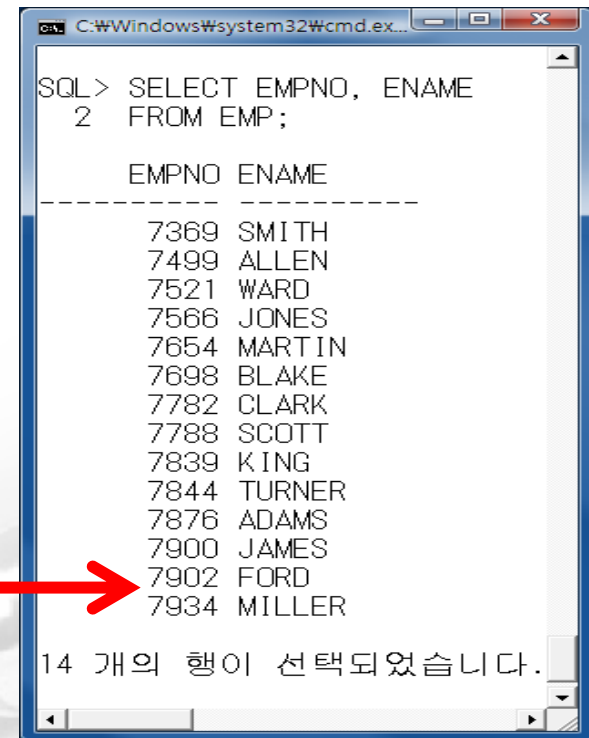
- ✓ 조인은 두 개 이상의 서로 다른 테이블을 서로 연결하는 것뿐만 아니라 하나의 테이블 내에서 조인을 해야만 원하는 자료를 얻는 경우가 발생할 수 있음 - 동일한 의미를 갖는 컬럼이 하나의 테이블에 2개 이상 존재하는 경우 : EMP 테이블에서는 MGR 이 관리자의 EMPNO
- ✓ 자기 자신과 조인을 맺는 것
- ✓ SMITH의 매니저 이름이 무엇인지 알아내려면 어떻게?



```
SQL> SELECT ENAME, MGR
2 FROM EMP;
```

ENAME	MGR
SMITH	7902
ALLEN	7698
WARD	7698
JONES	7839
MARTIN	7698
BLAKE	7839
CLARK	7839
SCOTT	7566
KING	
TURNER	7698
ADAMS	7788
JAMES	7698
FORD	7566
MILLER	7782

14 개의 행이 선택되었습니다.



```
SQL> SELECT EMPNO, ENAME
2 FROM EMP;
```

EMPNO	ENAME
7369	SMITH
7499	ALLEN
7521	WARD
7566	JONES
7654	MARTIN
7698	BLAKE
7782	CLARK
7788	SCOTT
7839	KING
7844	TURNER
7876	ADAMS
7900	JAMES
7902	FORD
7934	MILLER

14 개의 행이 선택되었습니다.

JOIN

❖ SELF JOIN

예

```
SELECT CONCAT(employee.ename, '의 매니저는 ', manager.ename)  
FROM EMP employee, EMP manager  
WHERE employee.mgr = manager.empno;
```



JOIN

❖ 연습문제

- ✓ EMP 테이블에서 MANAGER 가 'KING'인 직원들의 이름(ENAME)과 직급(JOB)을 조회



JOIN

❖ ANSI CROSS JOIN

- ✓ ANSI 표준은 단순 조인을 크로스 조인으로 정의하여 공식화해 두었는데 이름만 다를 뿐 같은 조인
- ✓ 콤마 대신 CROSS JOIN이라고 쓰면 됨

SELECT *

FROM EMP CROSS JOIN DEPT;

```

C:\Windows\system32\cmd.exe - sqlplus scott/tiger
SQL> SELECT *
  2  FROM EMP CROSS JOIN DEPT;

EMPNO  ENAME      JOB          MGR HIREDATE          SAL         COMM         DEPTNO  DEPTNO DNAME          LOC
-----
7369 SMITH      CLERK        7902 80/12/17          800          0             20      10 ACCOUNTING     NEW YORK
7499 ALLEN      SALESMAN     7698 81/02/20         1600         300           30      10 ACCOUNTING     NEW YORK
7521 WARD        SALESMAN     7521 81/02/22         1250         500           30      40 OPERATIONS     BOSTON
7566 JONES      MANAGER     7839 81/04/02         2975         0             20      40 OPERATIONS     BOSTON
7654 MARTIN    SALESMAN     7698 81/09/28         1250         1400          30      40 OPERATIONS     BOSTON
7698 BLAKE      CLERK        7902 80/12/17          800          0             20      40 OPERATIONS     BOSTON
7782 CLARK      MANAGER     7839 81/06/09         2450         0             10      40 OPERATIONS     BOSTON
7788 SCOTT      ANALYST     7566 87/04/19         3000         0             20      40 OPERATIONS     BOSTON
7839 KING      PRESIDENT   7839 81/11/17         5000         0             10      40 OPERATIONS     BOSTON
7844 TURNER    SALESMAN     7698 81/09/08         1500         0             30      40 OPERATIONS     BOSTON
7876 ADAMS      CLERK        7788 87/05/23         1100         0             20      40 OPERATIONS     BOSTON
7900 JAMES      CLERK        7698 81/12/03         950          0             30      40 OPERATIONS     BOSTON

56 개의 행이 선택되었습니다.
  
```

JOIN

❖ ANSI INNER JOIN

- ✓ ANSI 조인은 FROM 다음에 INNER JOIN 이란 단어를 사용하여 조인할 테이블 이름을 명시하고 ON 절을 사용하여 조인 조건을 명시하여 다음과 같이 작성

```
SELECT * FROM table1 INNER JOIN table2  
ON table1.column1 = table2.column2
```

- ✓ ANSI 조인에서는 조인 정보를 ON절에 기술하여 조인 조건을 명확하게 지정하고 다른 조건에 대해서는 WHERE 구문에서 지정

```
SELECT ENAME, DNAME  
FROM EMP INNER JOIN DEPT  
ON EMP.DEPTNO=DEPT.DEPTNO  
WHERE ENAME='MILLER';
```



JOIN

❖ USING을 이용한 조인 조건 지정

- ✓ 두 테이블에 각각 조인을 정의한 컬럼의 이름이 동일하다면 USING 절에서 조인할 컬럼을 지정하여 구문을 더 간단하게 표현

```
SELECT * FROM table1 JOIN table2  
USING (공통컬럼)
```

- ✓ EMP와 DEPT에 DEPTNO 라는 같은 이름의 컬럼이 있기 때문에 다음과 같이 간단하게 조인문을 기술

```
SELECT EMP.ENAME, DEPT.DNAME  
FROM EMP INNER JOIN DEPT  
USING (DEPTNO);
```



JOIN

❖ NATURAL JOIN

- ✓ 두 테이블에 각각 조인을 정의한 컬럼의 이름이 동일하다면 USING 절에서 조인할 컬럼을 지정하여 구문을 더 간단하게 표현

```
SELECT * FROM table1 NATURAL JOIN table2
```

- ✓ EMP와 DEPT에 DEPTNO 라는 같은 이름의 컬럼이 있기 때문에 다음과 같이 간단하게 조인문을 기술 – 중복된 컬럼은 한 번 만 조회됨

```
SELECT EMP.ENAME, DEPT.DNAME  
FROM EMP NATURAL JOIN DEPT;
```



JOIN

❖ OUTER JOIN

- ✓ SELF JOIN을 이용해서 특정 사원의 매니저 이름을 구했는데 결과를 살펴보면 이름이 KING인 사원 한 사람의 정보가 빠져 있음을 확인할 수 있음
- ✓ KING은 이 회사의 사장(PRESIDENT)으로 매니저가 존재하지 않으므로 MGR 컬럼 값이 NULL
- ✓ 사원 번호(EMPNO)가 NULL인 사원은 없으므로 조인 조건에 만족하지 않아서 KING은 SELF JOIN의 결과에서 배제
- ✓ 조인 조건에 만족하지 못하였더라도 해당 행을 나타내고 싶을 때에 사용하는 것이 외부 조인 (OUTER JOIN)



JOIN

❖ OUTER JOIN

- ✓ ANSI 구문에서 OUTER JOIN은 LEFT OUTER JOIN, RIGHT OUTER JOIN 그리고 FULL OUTER JOIN 세 가지 타입의 조인을 제공

SELECT *

FROM table1 [LEFT | RIGHT | FULL] OUTER JOIN table2

- ✓ OUTER JOIN은 어느 한쪽 테이블에는 해당하는 데이터가 존재하는데 다른 쪽 테이블에는 데이터가 존재하지 않을 경우 그 데이터가 출력되지 않는 문제점을 해결하기 위해 사용하는 조인 기법



JOIN

❖ OUTER JOIN

✓ 샘플 데이터 생성

1. 부서번호 와 부서명을 컬럼으로 갖는 DEPT01 테이블을 생성

```
CREATE TABLE DEPT01(  
  DEPTNO INT(2),  
  DNAME VARCHAR(14));
```

2. 샘플 데이터 추가 및 확인

```
INSERT INTO DEPT01 VALUES(10, 'ACCOUNTING');  
INSERT INTO DEPT01 VALUES (20, 'RESEARCH');
```

```
SELECT *  
FROM DEPT01;
```



JOIN

❖ OUTER JOIN

✓ 샘플 데이터 생성

3. 동일한 방법으로 DEPT02 테이블을 생성

```
CREATE TABLE DEPT02(  
  DEPTNO INT(2),  
  DNAME VARCHAR(14));
```

4. 샘플 데이터 추가 및 확인

```
INSERT INTO DEPT02 VALUES(10, 'ACCOUNTING');  
INSERT INTO DEPT02 VALUES (30, 'SALES');
```

```
SELECT * FROM DEPT02;
```



JOIN

❖ OUTER JOIN

- ✓ DEPT01 테이블의 20번 부서와 조인할 부서번호가 DEPT02에는 없지만 20번 부서도 출력되도록 하기 위해서 DEPT01 테이블이 왼쪽에 존재하기에 LEFT OUTER JOIN을 사용

SELECT *

FROM DEPT01 LEFT OUTER JOIN DEPT02

ON DEPT01.DEPTNO = DEPT02.DEPTNO;



JOIN

❖ OUTER JOIN

- ✓ DEPT02 테이블에만 있는 30번 부서까지 출력되도록 하기 위해서 RIGHT OUTER JOIN을 사용

```
SELECT *  
FROM DEPT01 RIGHT OUTER JOIN DEPT02  
USING(DEPTNO);
```



JOIN

❖ OUTER JOIN

- ✓ Maria DB는 FULL OUTER JOIN을 지원하지 않음

```
SELECT *  
FROM DEPT01 FULL OUTER JOIN DEPT02  
USING(DEPTNO);
```

SQL Error [1064] [42000]: (conn=34) You have an error in your SQL syntax; check the manual that corresponds to your MariaDB server version for the right syntax to use near 'OUTER JOIN DEPT02

USING(DEPTNO)
LIMIT 0, 200' at line 2



JOIN

❖ OUTER JOIN

- ✓ LEFT OUTER JOIN, RIGHT OUTER JOIN 을 합쳐서 FULL OUTER JOIN 구현

```
SELECT *
```

```
FROM DEPT01 LEFT OUTER JOIN DEPT02
```

```
ON DEPT01.DEPTNO = DEPT02.DEPTNO
```

```
UNION
```

```
SELECT *
```

```
FROM DEPT01 RIGHT OUTER JOIN DEPT02
```

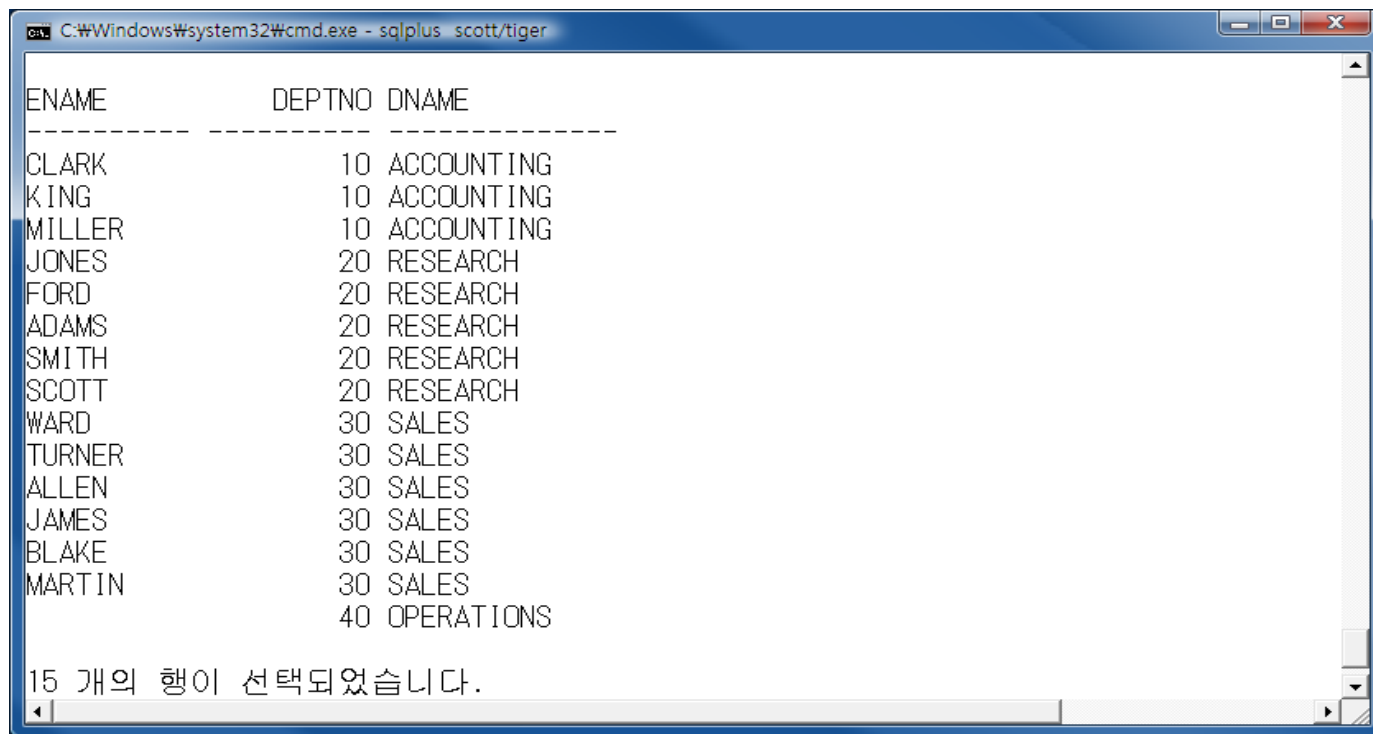
```
ON DEPT01.DEPTNO = DEPT02.DEPTNO
```



JOIN

❖ 연습문제

- ✓ 사원(EMP) 테이블과 부서(DEPT) 테이블을 조인하여 사원이름(ename)과 부서번호(deptno)와 부서명(dname)을 출력하도록 하는데 부서 테이블의 40번 부서와 조인할 사원 테이블의 부서번호가 없지만 아래 그림과 같이 40번 부서의 부서 이름도 출력되도록 쿼리문을 작성



ENAME	DEPTNO	DNAME
CLARK	10	ACCOUNTING
KING	10	ACCOUNTING
MILLER	10	ACCOUNTING
JONES	20	RESEARCH
FORD	20	RESEARCH
ADAMS	20	RESEARCH
SMITH	20	RESEARCH
SCOTT	20	RESEARCH
WARD	30	SALES
TURNER	30	SALES
ALLEN	30	SALES
JAMES	30	SALES
BLAKE	30	SALES
MARTIN	30	SALES
	40	OPERATIONS

15 개의 행이 선택되었습니다.

JOIN

❖ 다중 조인

- ✓ 조인을 중첩하면 여러 개의 테이블을 조인할 수 있음
- ✓ 조인한 결과도 하나의 테이블이므로 그 결과와 다른 테이블을 조인하는 것이 가능한데 이를 다중 조인이라고 함
- ✓ 제조사의 공장 필드와 도시 목록인 tCity의 도시명이 논리적인 FK여서 3중 조인해 볼 수 있음

```
SELECT * FROM TCAR C
```

```
INNER JOIN TMAKER M ON C.maker = M.maker
```

```
INNER JOIN tCity T ON M.factory = T.name;
```

	ABC CAR	123 CAPACITY	123 PRICE	ABC MAKER	ABC MAKER	ABC FACTORY	ABC DOMESTIC	ABC NAME	123 AREA	123 POPU	ABC METRO	ABC REGION
1	소나타	2,000	2,500	현대	현대	부산	y	부산	765	342	y	경상
2	터볼리	1,600	2,300	쌍용	쌍용	청주	y	청주	940	83	n	충청

JOIN

❖ 다중 조인

```
SELECT C.car, M.factory, T.area  
FROM TCAR C INNER JOIN TMAKER M ON C.maker = M.maker  
INNER JOIN  
tCity T ON M.factory = T.name;
```

	ABC CAR 	ABC FACTORY 	123 AREA 
1	소나타	부산	765
2	티볼리	청주	940



JOIN

❖ 다중 조인

```
SELECT * FROM TMAKER M
```

```
LEFT OUTER JOIN tCity T ON M.factory = T.name LEFT OUTER JOIN TCAR C ON M.maker  
= C.maker;
```



JOIN

❖ 연습문제

- EMP 테이블에서 모든 사원에 대한 이름(ename), 부서번호(deptno) DEPT 테이블에서 부서명(dname)을 가져와서 출력하는 SELECT 문장을 작성 - 2개의 테이블에는 DEPTNO 가 같이 존재
- DEPT 테이블의 LOC가 NEW YORK에서 근무하고 있는 사원에 대하여 EMP 테이블의 이름(ename), 업무(job), 급여(sal), DEPT 테이블의 부서명(dname)을 출력하는 SELECT 문장을 작성
- EMP 테이블에서 보너스(comm)가 null 이 아닌 사원에 대하여 이름(ename), DEPT 테이블의 부서명(dname), 위치(loc)를 출력하는 SELECT 문장을 작성
- EMP 테이블에서 이름(ename) 중 L자가 있는 사원에 대하여 이름(ename), 업무(job), DEPT 테이블의 부서명(dname), 위치(loc)를 출력하는 SELECT 문장을 작성
- EMP 테이블에서 그들의 관리자(mgr) 보다 먼저 입사한 사원에 대하여 이름(ename), 입사일(hiredate), 관리자(mgr) 이름, 관리자(mgr) 입사일을 출력하는 SELECT 문장을 작성

JOIN

❖ 연습문제

- EMP 테이블에 10개의 행이 있고 DEPT 테이블에 행이 2개 있을 때 다음의 SQL을 실행하면 조회되는 행 수는?

SELECT * FROM EMP, DEPT

- ① 20
- ② 40
- ③ 80
- ④ 160

- 다음 중 UNION과 UNION ALL'에 대한 설명으로 옳바르지 않은 것은?

- ① UNION은 두 개의 테이블에 대해서 합집합을 만들 수 있다
- ② UNION과 UNION ALL을 사용할 때에 두 개의 SELECT문에서 칼럼의 수와 데이터 타입이 일치해야 한다.
- ③ UNION은 중복을 제거한다.
- ④ UNION ALL은 정렬을 유발하지만. UNION은 정렬을 유발하지 않는다.