

8.1 조인 (JOIN)

■ JOIN 정의

: 검색하고자 하는 컬럼이 한 개의 테이블이 아닌, 여러 개의 테이블에 존재하는 경우에 사용되는 기술.

FK

EMPNO	ENAME	SAL	DEPTNO
7369	SMITH	800	20
7499	ALLEN	1600	30
7521	WARD	1250	30
7566	JONES	2975	20
7654	MARTIN	1250	30
7698	BLAKE	2850	30
7782	CLARK	2450	10
7788	SCOTT	3000	20

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

PK

PK

↓ ↓

EMPNO	ENAME	DNAME	LOC
7369	SMITH	RESEARCH	DALLAS
7499	ALLEN	SALES	CHICAGO
7521	WARD	SALES	CHICAGO
7566	JONES	RESEARCH	DALLAS
7654	MARTIN	SALES	CHICAGO

8.2 조인 (JOIN)

■ JOIN 종류

1. ORACLE 조인 (Oracle 8i)

- 카테시안 프로덕트 (Cartesian Product)
- Equi-Join
- Non-EquiJoin
- Outer 조인
- Self 조인

2. ANSI조인(SQL:1999)(Oracle 9i 이후)

- Cross 조인
- Natural 조인
- Using 이용한 조인
- Outer 조인

8.3 ORACLE JOIN

■ Catasian Product

- : SELECT 문장에서 JOIN 조건을 생략하거나 , 잘못된 조인조건을 지정한 경우.
- : 첫번째 테이블의 모든 행과 두번째 테이블의 모든 행들이 JOIN 된다.
- : 조인의 결과는 데이터로서 유용하지 않다.

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

EMPNO	ENAME	DNAME
7369	SMITH	ACCOUNTING
7369	SMITH	RESEARCH
7369	SMITH	SALES
7369	SMITH	OPERATIONS
7499	ALLEN	ACCOUNTING
...		
7934	MILLER	OPERATIONS

8.4 ORACLE JOIN

- Equi-Join

: PK 와 FK 가 정확하게 일치하는 경우에만 검색되는 방식.

```
SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2.column2;
```

```
SQL> SELECT EMP.EMPNO , EMP.ENAME , DEPT.DNAME, DEPT.LOC
2  FROM EMP , DEPT
3  WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

EMPNO	ENAME	DNAME	LOC
7369	SMITH	RESEARCH	DALLAS
7499	ALLEN	SALES	CHICAGO
7521	WARD	SALES	CHICAGO

```
SQL> SELECT e.EMPNO , e.ENAME , d.DNAME, d.LOC
2  FROM EMP e, DEPT d
3  WHERE e.DEPTNO = d.DEPTNO;
```

EMPNO	ENAME	DNAME	LOC
7369	SMITH	RESEARCH	DALLAS
7499	ALLEN	SALES	CHICAGO
7521	WARD	SALES	CHICAGO

```
SQL> SELECT e.EMPNO , e.ENAME , d.DNAME, d.LOC
2  FROM EMP e, DEPT d
3  WHERE e.DEPTNO = d.DEPTNO
4  AND e.EMPNO = 7900;
```

EMPNO	ENAME	DNAME	LOC
7900	JAMES	SALES	CHICAGO

8.5 ORACLE JOIN

- Non-EquiJoin

: 정확하게 일치하는 경우가 아닌 조인 방식.

EMPNO	ENAME	SAL
7369	SMITH	800
7499	ALLEN	1600
7521	WARD	1250
7566	JONES	2975
7654	MARTIN	1250
7698	BLAKE	2850
7782	CLARK	2450
7788	SCOTT	3000
7839	KING	5000
7844	TURNER	1500
7876	ADAMS	1100

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

EMPNO	ENAME	SAL	GRADE
7369	SMITH	800	1
7900	JAMES	950	1
7876	ADAMS	1100	1
7521	WARD	1250	2
7654	MARTIN	1250	2
7934	MILLER	1300	2
7844	TURNER	1500	3
7499	ALLEN	1600	3
7782	CLARK	2450	4

```
SQL> SELECT e.EMPNO , e.ENAME , e.SAL , g.GRADE
2 FROM EMP e, SALGRADE g
3 WHERE e.SAL BETWEEN g.LOSAL AND g.HISAL;
```

8.6 ORACLE JOIN

■ Outer Join

: 조인조건에 만족하지 않은, 누락된 행까지 포함하여 출력.

EMPNO	ENAME	DEPTNO
7369	SMITH	20
7499	ALLEN	30
7521	WARD	30
7566	JONES	20
7654	MARTIN	30
7698	BLAKE	30
7782	CLARK	10
7788	SCOTT	20
7839	KING	10
7844	TURNER	30
7876	ADAMS	20

EMPNO	ENAME	DEPTNO
7900	JAMES	30
7902	FORD	20
7934	MILLER	10

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON



40 은 항상 누락된다.

8.7 ORACLE JOIN

```
SQL> SELECT e.EMPNO , e.ENAME , d.DNAME, d.LOC
2 FROM EMP e, DEPT d
3 WHERE e.DEPTNO (+) = d.DEPTNO;
```

EMPNO	ENAME	DNAME	LOC
7782	CLARK	ACCOUNTING	NEW YORK
7839	KING	ACCOUNTING	NEW YORK
7934	MILLER	ACCOUNTING	NEW YORK
7566	JONES	RESEARCH	DALLAS
7902	FORD	RESEARCH	DALLAS
7876	ADAMS	RESEARCH	DALLAS
7369	SMITH	RESEARCH	DALLAS
7788	SCOTT	RESEARCH	DALLAS
7521	WARD	SALES	CHICAGO
7844	TURNER	SALES	CHICAGO
7499	ALLEN	SALES	CHICAGO
EMPNO	ENAME	DNAME	LOC
7900	JAMES	SALES	CHICAGO
7698	BLAKE	SALES	CHICAGO
7654	MARTIN	SALES	CHICAGO
		OPERATIONS	BOSTON

15 개의 행이 선택되었습니다.

```
SQL> SELECT e.EMPNO , e.ENAME , d.DNAME, d.LOC
2 FROM EMP e, DEPT d
3 WHERE d.DEPTNO = e.DEPTNO (+)
4 /
```

EMPNO	ENAME	DNAME	LOC
7782	CLARK	ACCOUNTING	NEW YORK
7839	KING	ACCOUNTING	NEW YORK
7934	MILLER	ACCOUNTING	NEW YORK
7566	JONES	RESEARCH	DALLAS
7902	FORD	RESEARCH	DALLAS
7876	ADAMS	RESEARCH	DALLAS
7369	SMITH	RESEARCH	DALLAS
7788	SCOTT	RESEARCH	DALLAS
7521	WARD	SALES	CHICAGO
7844	TURNER	SALES	CHICAGO
7499	ALLEN	SALES	CHICAGO
EMPNO	ENAME	DNAME	LOC
7900	JAMES	SALES	CHICAGO
7698	BLAKE	SALES	CHICAGO
7654	MARTIN	SALES	CHICAGO
		OPERATIONS	BOSTON

15 개의 행이 선택되었습니다.

8.8 ORACLE JOIN

■ Self Join

: 특정 테이블 자신을 자신이 Join하는 방법이다.

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

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

EMPNO	ENAME	MGR
7900	JAMES	7698
7902	FORD	7566
7934	MILLER	7782

```
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

EMPNO	ENAME
7900	JAMES
7902	FORD
7934	MILLER

SMITH의 관리자는 FORD

8.9 ORACLE JOIN

```
SQL> SELECT a.ENAME 사원 , b.ENAME 관리자
2 FROM EMP a , EMP b
3 WHERE a.MGR = b.EMPNO;
```

사원	관리자
SMITH	FORD
ALLEN	BLAKE
WARD	BLAKE
JONES	KING
MARTIN	BLAKE
BLAKE	KING
CLARK	KING
SCOTT	JONES
TURNER	BLAKE
ADAMS	SCOTT
JAMES	BLAKE

사원	관리자
FORD	JONES
MILLER	CLARK

13 개의 행이 선택되었습니다.

```
SQL> SELECT a.ENAME 사원 , b.ENAME 관리자
2 FROM EMP a , EMP b
3 WHERE a.MGR = b.EMPNO(+);
```

사원	관리자
SMITH	FORD
ALLEN	BLAKE
WARD	BLAKE
JONES	KING
MARTIN	BLAKE
BLAKE	KING
CLARK	KING
SCOTT	JONES
KING	
TURNER	BLAKE
ADAMS	SCOTT

사원	관리자
JAMES	BLAKE
FORD	JONES
MILLER	CLARK

14 개의 행이 선택되었습니다.

8.10 ANSI JOIN (SQL:1999,SQL3)

■ SQL:1999 특징

- : Join의 형식이 FROM 절에서 지정된다.
- : Join 조건이 WHERE 절이 아닌 ON절에서 명시된다.

```
SELECT table1.column, table2.column
FROM table1
[CROSS JOIN table2] |
[NATURAL JOIN table2] |
[JOIN table2 USING (column_name)] |
[JOIN table2
  ON (table1.column_name = table2.column_name)] |
[LEFT|RIGHT|FULL OUTER JOIN table2
  ON (table1.column_name = table2.column_name)];
```

■ Cross Join

- : Catesian Product와 동일하다.

```
SQL> SELECT EMPNO , ENAME , DNAME
2 FROM DEPT
3 CROSS JOIN EMP;
```

EMPNO	ENAME	DNAME
7369	SMITH	ACCOUNTING
7499	ALLEN	ACCOUNTING
7521	WARD	ACCOUNTING

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

EMPNO	ENAME	DNAME
7369	SMITH	ACCOUNTING
7499	ALLEN	ACCOUNTING
7521	WARD	ACCOUNTING

8.11 ANSI JOIN(SQL:1999)

■ Natural Join

: Equi-Join 과 동일하다. 즉, 같은 이름을 가진 컬럼에 기반한다.

```
SQL> SELECT EMPNO , ENAME, DNAME, LOC  
2 FROM EMP  
3 NATURAL JOIN DEPT;
```

EMPNO	ENAME	DNAME	LOC
7369	SMITH	RESEARCH	DALLAS
7499	ALLEN	SALES	CHICAGO
7521	WARD	SALES	CHICAGO

```
SQL> SELECT e.EMPNO , e.ENAME , d.DNAME, d.LOC  
2 FROM EMP e, DEPT d  
3 WHERE e.DEPTNO = d.DEPTNO;
```

EMPNO	ENAME	DNAME	LOC
7369	SMITH	RESEARCH	DALLAS
7499	ALLEN	SALES	CHICAGO
7521	WARD	SALES	CHICAGO

■ join ~ using 이용한 Join

: 동일이름의 컬럼이 여러 개인 경우 조인 컬럼을 지정.

: Natural Join과는 상호배타적이다.

```
SQL> SELECT EMPNO, ENAME, DNAME, LOC  
2 FROM EMP  
3 JOIN DEPT  
4 USING (DEPTNO);
```

EMPNO	ENAME	DNAME	LOC
7369	SMITH	RESEARCH	DALLAS
7499	ALLEN	SALES	CHICAGO
7521	WARD	SALES	CHICAGO

8.12 ANSI JOIN(SQL:1999)

■ join~on 이용한 Join

: Non-EquiJoin 또는 임의의 조건으로 조인시 사용.

: 조인할 컬럼을 명시하기 위해서 사용.

: 복잡한 조건의 조인 가능하다. (서브쿼리 , AND/OR 연산자 , EXIST , IN 연산자)

```
SQL> SELECT EMPNO, ENAME, DNAME, LOC
2 FROM EMP
3 JOIN DEPT
4 ON EMP.DEPTNO = DEPT.DEPTNO
5 AND DEPT.DEPTNO = 10;
```

EMPNO	ENAME	DNAME	LOC
7782	CLARK	ACCOUNTING	NEW YORK
7839	KING	ACCOUNTING	NEW YORK
7934	MILLER	ACCOUNTING	NEW YORK

```
SQL> SELECT EMPNO , ENAME, DNAME, SAL
2 FROM EMP e
3 JOIN DEPT d
4 ON e.DEPTNO = d.DEPTNO
5 AND SAL IN ( 800 );
```

EMPNO	ENAME	DNAME	SAL
7369	SMITH	RESEARCH	800

```
SQL> SELECT EMPNO , ENAME, DNAME, SAL , GRADE
2 FROM EMP e
3 JOIN DEPT d
4 ON e.DEPTNO = d.DEPTNO
5 JOIN SALGRADE s
6 ON e.SAL BETWEEN s.LOSAL AND s.HISAL;
```

EMPNO	ENAME	DNAME	SAL	GRADE
7369	SMITH	RESEARCH	800	1
7900	JAMES	SALES	950	1
7876	ADAMS	RESEARCH	1100	1
7521	WARD	SALES	1250	2
7654	MARTIN	SALES	1250	2
7934	MILLER	ACCOUNTING	1300	2
7844	TURNER	SALES	1500	3

```
SQL> SELECT e.ename 사원 , d.ename 관리자
2 FROM emp e
3 join emp d
4 on e.mgr = d.empno;
```

사원	관리자
SMITH	FORD

8.13 ANSI JOIN(SQL:1999)

■ Outer Join

```
SQL> SELECT EMPNO ,ENAME , DNAME  
2 FROM DEPT LEFT OUTER JOIN EMP  
3 ON DEPT.DEPTNO = EMP.DEPTNO;
```

EMPNO	ENAME	DNAME
7369	SMITH	RESEARCH
7499	ALLEN	SALES
7521	WARD	SALES
7566	JONES	RESEARCH
7654	MARTIN	SALES
7698	BLAKE	SALES
7782	CLARK	ACCOUNTING
7788	SCOTT	RESEARCH
7839	KING	ACCOUNTING
7844	TURNER	SALES
7876	ADAMS	RESEARCH
<hr/>		
EMPNO	ENAME	DNAME
7900	JAMES	SALES
7902	FORD	RESEARCH
7934	MILLER	ACCOUNTING
		OPERATIONS

15 개의 행이 선택되었습니다.

좌측에 기술한 테이블 (DEPT)의 모든 행들이
우측에 기술한 테이블(EMP)내 행들과 일치 여부에
상관없이 모두 출력된다.

```
SQL> SELECT EMPNO ,ENAME , DNAME  
2 FROM EMP RIGHT OUTER JOIN DEPT  
3 ON DEPT.DEPTNO = EMP.DEPTNO;
```

EMPNO	ENAME	DNAME
7369	SMITH	RESEARCH
7499	ALLEN	SALES
7521	WARD	SALES
7566	JONES	RESEARCH
7654	MARTIN	SALES
7698	BLAKE	SALES
7782	CLARK	ACCOUNTING
7788	SCOTT	RESEARCH
7839	KING	ACCOUNTING
7844	TURNER	SALES
7876	ADAMS	RESEARCH
<hr/>		
EMPNO	ENAME	DNAME
7900	JAMES	SALES
7902	FORD	RESEARCH
7934	MILLER	ACCOUNTING
		OPERATIONS

15 개의 행이 선택되었습니다.

우측에 기술한 테이블 (DEPT)의 모든 행들이
좌측에 기술한 테이블(EMP)내 행들과 일치 여부에
상관없이 모두 출력된다.

8.14 ANSI JOIN(SQL:1999)

- Outer Join

```
SQL> INSERT INTO EMP  
2 VALUES ( 9000, 'TEST' , 'SALES', 7499, '90/01/01' , 400 , NULL , NULL );
```

1 개의 행이 만들어졌습니다.

```
SQL> SELECT EMPNO, ENAME, DNAME  
2 FROM DEPT FULL OUTER JOIN EMP  
3 ON DEPT.DEPTNO = EMP.DEPTNO;
```

EMPNO	ENAME	DNAME
7369	SMITH	RESEARCH
7499	ALLEN	SALES
7521	WARD	SALES
7566	JONES	RESEARCH
7654	MARTIN	SALES
7698	BLAKE	SALES
7782	CLARK	ACCOUNTING
7788	SCOTT	RESEARCH
7839	KING	ACCOUNTING
7844	TURNER	SALES
7876	ADAMS	RESEARCH
7900	JAMES	SALES
7902	FORD	RESEARCH
7934	MILLER	ACCOUNTING
9000	TEST	OPERATIONS

16 개의 행이 선택되었습니다.

8.15 ANSI JOIN(SQL:1999)

■ 실습 문제

1. 부서 테이블과 사원테이블에서 사번, 사원명, 부서코드, 부서명을 검색하시오.
(사원명 오름차순 정렬할 것)
2. 부서 테이블과 사원테이블에서 사번, 사원명 , 급여 , 부서명을 검색하시오. 단, 급여가 2000 이상인 사원에 대하여 급여기준으로 내림차순 정렬할 것.
3. 부서 테이블과 사원 테이블에서 사번, 사원명, 업무, 급여 , 부서명을 검색하시오. 단, 업무가 Manager이며 급여가 2500 이상인 사원에 대하여 사번을 기준으로 오름차순 정렬할 것.
4. 사원 테이블과 급여 등급 테이블에서 사번, 사원명, 급여, 등급을 검색하시오. 단, 등급은 급여가 하한값과 상한값 범위에 포함되고 등급이 4이며 급여를 기준으로 내림차순정렬할 것.
5. 부서 테이블, 사원 테이블, 급여등급 테이블에서 사번, 사원명, 부서명, 급여 , 등급을 검색하시오. 단, 등급은 급여가 하한값과 상한값 범위에 포함되며 등급을 기준으로 내림차순 정렬할 것.
6. 사원 테이블에서 사원명과 해당 사원의 관리자명을 검색하시오.
7. 사원 테이블에서 사원명, 해당 사원의 관리자명, 해당 사원의 관리자의 관리자명을 검색하시오
8. 7번 결과에서 상위 관리자가 없는 모든 사원의 이름도 사원명에 출력되도록 수정하시오.