SQL – join

문혜영

JOIN

- **정규화와 JOIN**
  - 정규화
    - 이상현상(Anomaly) 발생을 피하기 위해 테이블을 분할
    - 학계/실무에서 주로 3NF(3차 정규형) 사용
  - JOIN
    - 데이터의 통합 조회를 위해 여러 테이블들을 연결
    - 실제 JOIN 연산은 두 개의 테이블에 대해서만 적용됨
    - 일반적인 경우 PK/FK의 연관에 의해 JOIN이 성립
      - 그 외에도 논리적인 값들의 연관성만으로 JOIN 성립 가능

JOIN

- 조인의 종류

1. 결합 결과의 행에 따른 분류:
    1. 내부 조인 (Inner Join) 두 테이블이 조인 조건을 만족하는 행만 반환
    2. 외부 조인 (Outer Join)
        - Left Outer Join : 왼쪽 테이블의 모든 행과 오른쪽 테이블의 일치하는 행을 반환
        - Right Outer Join : 오른쪽 테이블의 모든 행과 왼쪽 테이블의 일치하는 행을 반환
        - Full Outer Join: 두 테이블의 모든 행을 반환하며, 일치하지 않는 행은 NULL 값을 가짐

- 조인 조건의 유형에 따른 분류:
    1. 세타 조인 (Theta Join): 두 테이블 간에 임의의 비교 조건(예: =, <, >, <=, >= 등)을 사용하여 조인하는 방식.
    2. 동등 조인 (Equi Join): 두 테이블을 동등 비교 연산자(=)를 사용하여 조인. 세타 조인의 특별한 경우.
    3. 자연 조인 (Natural Join): 두 테이블 간에 동일한 이름을 가진 모든 열을 기준으로 동등 조인을 수행

COMPANY.SQL

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

JOIN

- **JOIN의 예**

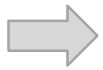
```
SELECT    ENAME, DNAME
FROM      EMP, DEPT
WHERE     EMP.DEPTNO = DEPT.DEPTNO;
```

EMP

	ENAME	DEPTNO
1	SMITH	20
2	ALLEN	30
3	WARD	30
4	JONES	20
5	MARTIN	30
6	BLAKE	30

DEPT

	DEPTNO	DNAME
1	10	ACCOUNTING
2	20	RESEARCH
3	30	SALES
4	40	OPERATIONS



	ENAME	DNAME
1	SMITH	RESEARCH
2	ALLEN	SALES
3	WARD	SALES
4	JONES	RESEARCH
5	MARTIN	SALES
6	BLAKE	SALES
7	CLARK	ACCOUNTING
8	SCOTT	RESEARCH
9	KING	ACCOUNTING
10	TURNER	SALES
11	ADAMS	RESEARCH
12	JAMES	SALES
13	FORD	RESEARCH
14	MILLER	ACCOUNTING

- **JOIN의 유형**
  - Equi Join / Non-Equi Join
  - 암시적 조인 / 명시적 조인
  - Inner Join / Outer Join / Cross Join / Self Join

EQUI JOIN

- **EQUI JOIN(동등 조인)**
  - 조인 조건으로 Equal (=) 연산 사용

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

ERROR!!!

  - 중복 칼럼의 경우, 칼럼명 앞에 테이블명을 붙여야 함
    - 중복되지 않는 칼럼도 칼럼명 앞에 테이블명을 붙이는 것을 권장
    - 예) 선수명, 팀ID, 팀명을 출력

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

→ 테이블명이 긴 경우 ALIAS 사용

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

EQUI JOIN

- **EQUI JOIN(동등 조인)**

```
SELECT   EMP.ENAME, EMP.DEPTNO, DEPT.DNAME
FROM     EMP, DEPT
WHERE    EMP.DEPTNO = DEPT.DEPTNO;
```
ALIAS 미사용

```
SELECT   E.ENAME, E.DEPTNO, D.DNAME
FROM     EMP E, DEPT D
WHERE    E.DEPTNO = D.DEPTNO;
```
ALIAS 사용
EMP는 E로 사용
DEPT는 D로 사용

```
SELECT   ENAME, E.DEPTNO, DNAME
FROM     EMP E, DEPT D
WHERE    E.DEPTNO = D.DEPTNO;
```
접두어 일부생략

```
SELECT    EMP.ENAME, EMP.DEPTNO, DEPT.DNAME
FROM     EMP E, DEPT D
WHERE    E.DEPTNO = D.DEPTNO;
```
ERROR!!!

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

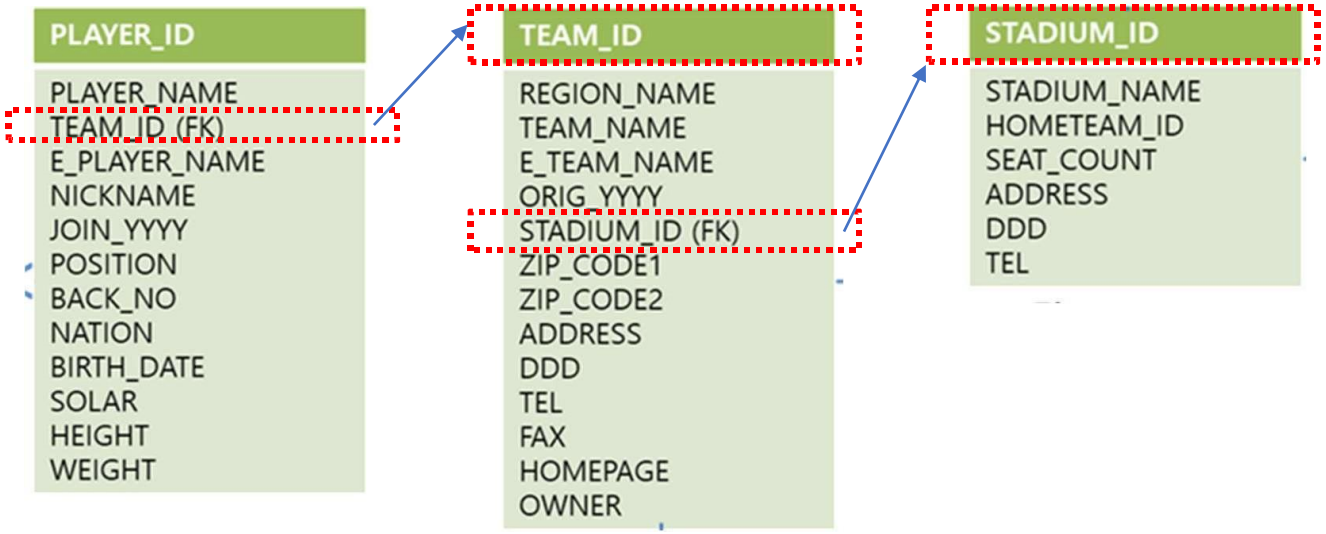
ENAME	DEPTNO	DNAME
SMITH	20	RESEARCH
ALLEN	30	SALES
WARD	30	SALES
JONES	20	RESEARCH
MARTIN	30	SALES
BLAKE	30	SALES
CLARK	10	ACCOUNTING
SCOTT	20	RESEARCH
KING	10	ACCOUNTING
TURNER	30	SALES
ADAMS	20	RESEARCH
JAMES	30	SALES
FORD	20	RESEARCH
MILLER	10	ACCOUNTING

- FROM 절에서 ALIAS 정의 후에는 WHERE/SELECT절에서 테이블명 사용 불가

EQUI JOIN

- **EQUI JOIN(동등 조인)**
  - 셋 이상 테이블의 조인은 실제로는 두 테이블 간 조인이 연쇄적으로 일어남

```
SELECT   P.PLAYER_NAME, P.POSITION, T.REGION_NAME, T.TEAM_NAME, S.STADIUM_NAME
FROM     PLAYER P, TEAM T, STADIUM S
WHERE    P.TEAM_ID = T.TEAM_ID   AND   T.STADIUM_ID = S.STADIUM_ID;
```



Non EQUI JOIN

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	null)	20
7900	JAMES	CLERK	7698	81/12/03	950	null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	null)	10

SALGRADE

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

DEPT

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

- **Non EQUI JOIN**
  - 조인 조건으로 Equal (=) 이외의 연산 사용
    - Between, >, >=, <, <= 등
    - 예) 사원별 급여 등급 조회
  - Q) EMP 테이블과 SALGRADE 테이블로부터, 사원명, 급여, 급여등급을 출력하는 질의를 완성하시오.

```
SELECT E.ENAME 사원명, E.SAL 급여, S.GRADE 급여등급, S.LOSAL, S.HISAL
FROM   EMP E, SALGRADE S;
```

```
SELECT E.ENAME 사원명, E.SAL 급여, S.GRADE 급여등급, S.LOSAL, S.HISAL
FROM   EMP E, SALGRADE S
WHERE E.SAL BETWEEN S.LOSAL  AND S. HISAL ;
```

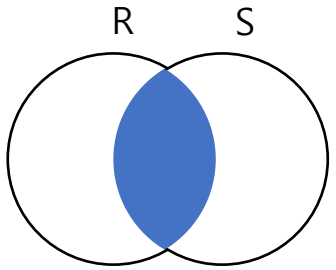
```
SELECT E.ENAME 사원명, E.SAL 급여, S.GRADE 급여등급
FROM       EMP E, SALGRADE S
WHERE E.SAL BETWEEN S.LOSAL  AND S. HISAL ;
```

사원명	급여	급여등급	LOSAL	HISAL
SMITH	800	1	700	1200
ALLEN	1600	1	700	1200
WARD	1250	1	700	1200
JONES	2975	1	700	1200
MARTIN	1250	1	700	1200
BLAKE	2850	1	700	1200
CLARK	2450	1	700	1200
SCOTT	3000	1	700	1200
KING	5000	1	700	1200
TURNER	1500	1	700	1200
ADAMS	1100	1	700	1200
JAMES	950	1	700	1200
FORD	3000	1	700	1200
MILLER	1300	1	700	1200
SMITH	800	2	1201	1400
ALLEN	1600	2	1201	1400
WARD	1250	2	1201	1400
JONES	2975	2	1201	1400
MARTIN	1250	2	1201	1400
BLAKE	2850	2	1201	1400

사원명	급여	급여등급	LOSAL	HISAL
SMITH	800	1	700	1200
ADAMS	1100	1	700	1200
JAMES	950	1	700	1200
WARD	1250	2	1201	1400
MARTIN	1250	2	1201	1400
MILLER	1300	2	1201	1400
ALLEN	1600	3	1401	2000
TURNER	1500	3	1401	2000
JONES	2975	4	2001	3000
BLAKE	2850	4	2001	3000
CLARK	2450	4	2001	3000
SCOTT	3000	4	2001	3000
FORD	3000	4	2001	3000
KING	5000	5	3001	9999

INNER JOIN

- **INNER JOIN (내부 조인)**
  - 서로 대응되는 내용만 검색하는 조인
    - 조건절을 필수로 사용
  - 조인의 Default이므로 "INNER" 생략 가능
    - INNER JOIN = JOIN



R

번호	이름
100	정수현
200	김치국
300	고아라

⋈

S

번호	이름
100	정수현
101	윤서윤
102	이수진

R∩S

번호	이름
100	정수현

INNER JOIN

- **INNER JOIN (내부 조인)**

```
SELECT E.ENAME , E.DEPTNO, E.SAL, D.DNAME
FROM    EMP E, DEPT D
WHERE E.DEPTNO=D.DEPTNO AND E.SAL >2000;
```

암시적

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

- 조인 조건과 일반 조건이 혼용되어 가독성이 떨어짐

  → 명시적 조인 (= 표준 조인)의 필요성

- 명시적 조인에서는 조인 관련 조건은 ON 절에, 그 외의 조건은 WHERE 절에 기술함
- 대부분의 DBMS는 명시적 조인을 표준으로 채택하지만, 기존의 암시적 조인도 허용

```
SELECT E.ENAME , E.DEPTNO, E.SAL, D.DNAME
FROM    EMP E INNER JOIN DEPT D
ON E.DEPTNO=D.DEPTNO
WHERE E.DEPTNO=D.DEPTNO AND E.SAL >2000;
```

명시적(표준)

	ENAME	DEPTNO	SAL	DNAME
1	JONES	20	2975	RESEARCH
2	BLAKE	30	2850	SALES
3	CLARK	10	2450	ACCOUNTING
4	SCOTT	20	3000	RESEARCH
5	KING	10	5000	ACCOUNTING
6	FORD	20	3000	RESEARCH

- 위의 INNER JOIN에서 INNER는 생략 가능

NATURAL JOIN

- **NATURAL JOIN**
  - INNER JOIN의 특수한 경우
    - NATURAL INNER JOIN = NATURAL JOIN
  - **<u>두 테이블 간 동일한 이름을 갖는 모든 칼럼들에 대해 EQUI JOIN 수행</u>**
    - 칼럼 간 데이터 타입도 동일해야 함
    - 별도의 조인 칼럼 및 조건을 지정할 수 없음
  - **<u>조인의 대상이 되는 칼럼에는</u>** 접두사(테이블 명 또는 ALIAS)를 사용할 수 없음
  - Q) 다음 NATURAL JOIN과 동일한 출력을 갖는 INNER JOIN 질의를 작성하시오.

```
SELECT   EMPNO, ENAME, DEPTNO, DNAME
FROM     EMP NATURAL JOIN DEPT;
```

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

	EMPNO	ENAME	DEPTNO	DNAME
1	7369	SMITH	20	RESEARCH
2	7499	ALLEN	30	SALES
3	7521	WARD	30	SALES
4	7566	JONES	20	RESEARCH
5	7654	MARTIN	30	SALES
6	7698	BLAKE	30	SALES

NATURAL JOIN

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

- **NATURAL JOIN**
  - Q) INNER JOIN과 NATURAL JOIN의 실행 결과를 비교하시오.
    - 칼럼 출력 순서 및 칼럼 개수

```
SELECT    *
FROM      EMP  NATURAL JOIN  DEPT;
```

	DEPTNO	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DNAME	LOC
1	20	7369	SMITH	CLERK	7902	80/12/17	800	(null)	RESEARCH	DALLAS
2	30	7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	SALES	CHICAGO

```
SELECT    *
FROM      EMP INNER JOIN DEPT
ON        EMP.DEPTNO = DEPT.DEPTNO;
```

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DEPTNO_1	DNAME	LOC
1	7369	SMITH	CLERK	7902	80/12/17	800	(null)	20	20	RESEARCH	DALLAS
2	7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30	30	SALES	CHICAGO

JOIN 조건절

- **ON 조건절**
  - 암시적 JOIN
    - 모든 조건을 WHERE절에 기술
  - 명시적 JOIN
    - JOIN 기준 조건은 ON절에 기술
      - ON절의 괄호는 생략 가능
    - JOIN과 무관한 일반 조건은 WHRER절에 기술
  - 예) 이름에 'S'가 포함된 사원의 사원이름, 부서코드, 부서명 출력

```
SELECT E.EMPNO, E.ENAME, E.DEPTNO, D.DNAME
FROM EMP E , DEPT D
WHERE E.DEPTNO = D.DEPTNO AND E.SAL >2000;
```

암시적

```
SELECT E.EMPNO, E.ENAME, E.DEPTNO, D.DNAME
FROM EMP E JOIN DEPT D
ON (E.DEPTNO = D.DEPTNO)
WHERE E.ENAME LIKE '%S%';
```

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

EMPNO	ENAME	DEPTNO	DNAME
7369	SMITH	20	RESEARCH
7566	JONES	20	RESEARCH
7788	SCOTT	20	RESEARCH
7876	ADAMS	20	RESEARCH
7900	JAMES	30	SALES

JOIN 조건절

- **USING 조건절**
  - ON절의 "=" 연산자 대신 USING 절 사용 가능
    - ON 절에서는 괄호 생략 가능, USING 에서는 괄호 생략 불가
  - 접두사(테이블 명 또는 ALIAS)를 사용할 수 없음

```
SELECT E.ENAME, E.DEPTNO, D.DNAME
FROM EMP E JOIN DEPT D
ON (E.DEPTNO = D.DEPTNO);
```

=

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

JOIN 조건절

- **WHERE, ON, USING절 조건 기술 비교**

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

암시적조인

```
SELECT ENAME, EMP.DEPTNO, DNAME
 FROM EMP JOIN DEPT
ON EMP.DEPTNO = DEPT.DEPTNO;
```

명시적조인

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

USING

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

두 테이블간 동일한 칼럼명이 DEPTNO밖에 없는경우

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

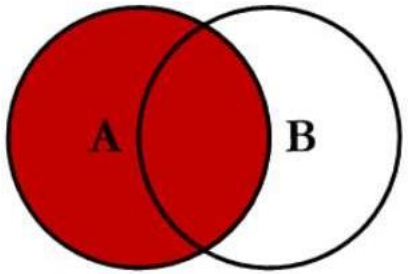
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

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

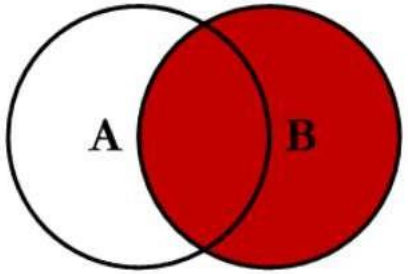
OUTER JOIN

- **OUTER JOIN (외부 조인)**
  - 서로 대응되지 않는 행도 출력하는 조인
  - 조건절을 필수로 사용
  - 성능 저하의 원인이 될 수 있으므로 필요한 경우만 사용

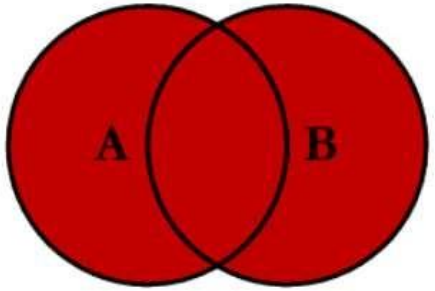
LEFT OUTER JOIN
= LEFT JOIN



RIGHT OUTER JOIN
= RIGHT JOIN



FULL OUTER JOIN
= FULL JOIN



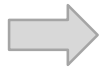
OUTER JOIN

- **OUTER JOIN (외부 조인)**
  - LEFT OUTER JOIN
    - 왼쪽 테이블의 데이터를 모두 읽은 후, 오른쪽 테이블에서 JOIN 데이터를 가져옴
    - 오른쪽 테이블이 JOIN 조건에 해당되지 않는 경우, 해당 칼럼은 NULL로 채움

PLAYER_NAME	TEAM_ID
홍길동	K01
강감찬	K01
김유신	K02
유관순	
최무선	

⟕

TEAM_ID	TEAM_NAME
K01	KIA
K02	두산
K03	LG
K04	넥센



PLAYER_NAME	TEAM_ID	TEAM_NAME
홍길동	K01	KIA
강감찬	K01	KIA
김유신	K02	두산
유관순		
최무선		

OUTER JOIN

- **OUTER JOIN (외부 조인)**
  - RIGHT OUTER JOIN
    - 오른쪽 테이블의 데이터를 모두 읽은 후,
      왼쪽 테이블에서 JOIN 데이터를 가져옴
    - 왼쪽 테이블이 JOIN 조건에 해당되지 않는 경우,
      해당 칼럼은 NULL로 채움

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

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

PLAYER_NAME	TEAM_ID
홍길동	K01
강감찬	K01
김유신	K02
유관순	
최무선	

⋈

TEAM_ID	TEAM_NAME
K01	KIA
K02	두산
K03	LG
K04	넥센

⇒

PLAYER_NAME	TEAM_ID	TEAM_NAME
홍길동	K01	KIA
강감찬	K01	KIA
김유신	K02	두산
	K03	LG
	K04	넥센

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

- Q) 다음 INNER JOIN과 RIGHT OUTER JOIN의 연산 결과를 비교하시오.

```
SELECT E.ENAME, E.DEPTNO, D.DNAME
FROM EMP E INNER JOIN DEPT D
ON E.DEPTNO=D.DEPTNO;
```

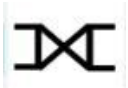
```
SELECT E.ENAME, E.DEPTNO, D.DNAME
FROM EMP E RIGHT OUTER JOIN DEPT D
ON E.DEPTNO=D.DEPTNO;
```

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
(null)	(null)	OPERATIONS

OUTER JOIN

- **OUTER JOIN (외부 조인)**
  - FULL OUTER JOIN
    - 양쪽 테이블의 데이터를 모두 읽은 후, 상대 테이블에서 JOIN 데이터를 가져옴
    - JOIN 조건에 해당되지 않는 경우, 해당 칼럼은 NULL로 채움

PLAYER_NAME	TEAM_ID
홍길동	K01
강감찬	K01
김유신	K02
유관순	
최무선	



TEAM_ID	TEAM_NAME
K01	KIA
K02	두산
K03	LG
K04	넥센

PLAYER_NAME	TEAM_ID	TEAM_NAME
홍길동	K01	KIA
강감찬	K01	KIA
김유신	K02	두산
유관순		
최무선		
	K03	LG
	K04	넥센

OUTER JOIN

- **OUTER JOIN (외부 조인)**
  - FULL OUTER JOIN
    - RIGHT OUTER JOIN과 LEFT OUTER JOIN의 합집합과 동일 (중복 제거 후)
      - 즉 UNION ALL이 아닌 UNION과 동일

```
SELECT E.ENAME, E.DEPTNO, D.DNAME
FROM   EMP E FULL OUTER JOIN DEPT D
ON       E.DEPTNO = D.DEPTNO;
```

```
SELECT      E.ENAME, E.DEPTNO, D.DNAME
FROM        EMP E LEFT OUTER JOIN DEPT D
ON          E.DEPTNO = D.DEPTNO
UNION
SELECT      E.ENAME, E.DEPTNO, D.DNAME
FROM        EMP E RIGHT OUTER JOIN DEPT D
ON          E.DEPTNO = D.DEPTNO;
```

	ENAME	DEPTNO	DNAME
1	MILLER	10	ACCOUNTING
2	KING	10	ACCOUNTING
3	CLARK	10	ACCOUNTING
4	FORD	20	RESEARCH
5	ADAMS	20	RESEARCH
6	SCOTT	20	RESEARCH
7	JONES	20	RESEARCH
8	SMITH	20	RESEARCH
9	JAMES	30	SALES
10	TURNER	30	SALES
11	BLAKE	30	SALES
12	MARTIN	30	SALES
13	WARD	30	SALES
14	ALLEN	30	SALES
15	(null)	(null)	OPERATIONS

CROSS JOIN

- **CROSS JOIN (교차 조인)**
  - 두 테이블의 곱집합(Cartesian Product)을 출력하는 조인
  - 별도의 조인 조건이 없음

번호	이름
1	홍길동
2	임꺽정

생산품코드	상품명
A	면도기
B	칫솔
C	치약

번호	이름	생산품코드	상품명
1	홍길동	A	면도기
2	임꺽정	A	면도기
1	홍길동	B	칫솔
2	임꺽정	B	칫솔
1	홍길동	C	치약
2	임꺽정	C	치약

CROSS JOIN

- **CROSS JOIN (교차 조인)**
  - Q) 다음 질의의 결과로 출력되는 행의 수는?

```
SELECT E.ENAME, E.DEPTNO, D.DEPTNO, D.DNAME
FROM EMP E CROSS JOIN DEPT D;
```

	ENAME	DEPTNO
1	SMITH	20
2	ALLEN	30
3	WARD	30
4	JONES	20
5	MARTIN	30
6	BLAKE	30
7	CLARK	10
8	SCOTT	20
9	KING	10
10	TURNER	30
11	ADAMS	20
12	JAMES	30
13	FORD	20
14	MILLER	10

	DNAME	DEPTNO
1	ACCOUNTING	10
2	RESEARCH	20
3	SALES	30
4	OPERATIONS	40

	ENAME	DEPTNO	DEPTNO_1	DNAME
35	KING	10	30	SALES
36	KING	10	40	OPERATIONS
37	TURNER	30	10	ACCOUNTING
38	TURNER	30	20	RESEARCH
39	TURNER	30	30	SALES
40	TURNER	30	40	OPERATIONS
41	ADAMS	20	10	ACCOUNTING
42	ADAMS	20	20	RESEARCH
43	ADAMS	20	30	SALES
44	ADAMS	20	40	OPERATIONS
45	JAMES	30	10	ACCOUNTING
46	JAMES	30	20	RESEARCH
47	JAMES	30	30	SALES
48	JAMES	30	40	OPERATIONS
49	FORD	20	10	ACCOUNTING
50	FORD	20	20	RESEARCH
51	FORD	20	30	SALES
52	FORD	20	40	OPERATIONS
53	MILLER	10	10	ACCOUNTING
54	MILLER	10	20	RESEARCH
55	MILLER	10	30	SALES
56	MILLER	10	40	OPERATIONS

JOIN 결과 비교

- **JOIN의 결과 비교**
  - Q) 다음 5가지 JOIN으로 생성되는 각 결과의 레코드 수는?

STUDENT

S_NAME	S_ID	DEPT
KIM	111	B
LEE	222	C
CHOI	333	D
PARK	444	B

DEPT

D_ID	D_NAME
A	MIS
B	CS
C	BIO

```
SELECT   S.S_NAME, D.D_NAME
FROM     STUDENT S INNER JOIN DEPT D    ON   (S.DEPT = D.D_ID);
```

```
SELECT   S.S_NAME, D.D_NAME
FROM     STUDENT S CROSS JOIN DEPT D;
```

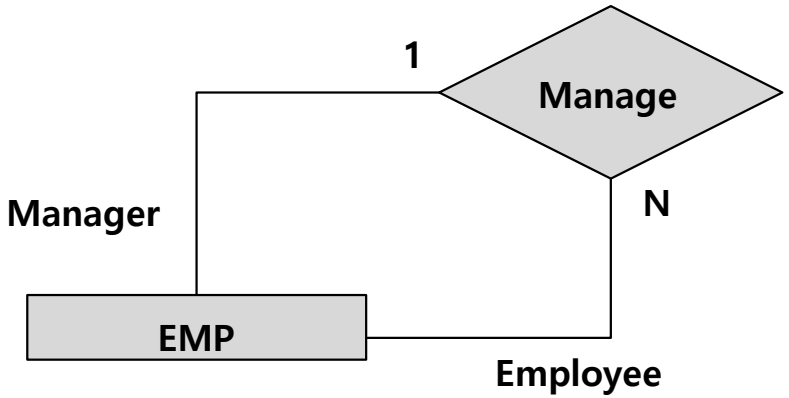
```
SELECT   S.S_NAME, D.D_NAME
FROM     STUDENT S LEFT OUTER JOIN DEPT D    ON   (S.DEPT = D.D_ID);
```

```
SELECT   S.S_NAME, D.D_NAME
FROM     STUDENT S RIGHT OUTER JOIN DEPT D    ON   (S.DEPT = D.D_ID);
```

```
SELECT   S.S_NAME, D.D_NAME
FROM     STUDENT S FULL OUTER JOIN DEPT D    ON   (S.DEPT = D.D_ID);
```

SELF JOIN

- **SELF JOIN (셀프 조인)**
  - 동일 테이블 사이의 조인
    - FROM 절에 동일 테이블이 두 번 이상 나타남
  - 테이블 식별을 위해 반드시 별칭(Alias)를 사용해야 함
    - 동일한 테이블을 **개념적으로** 서로 다른 두 개의 테이블로 사용함
    - 예) FROM EMP E INNER JOIN EMP M



SELF JOIN

- **SELF JOIN (셀프 조인)**
  - Q) EMP 테이블로부터 사원의 사번과 이름, 그리고 매니저의 사번과 이름을 출력하기 위한 질의를 작성하시오. (단 매니저가 없는 사원의 정보도 출력되어야 함)

```
SELECT E.EMPNO, E.ENAME, M.EMPNO MGRNO, M.ENAME MNAME
FROM EMP E LEFT JOIN EMP M
ON E.MGR=M.EMPNO;
```

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800	(null)	20
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7566	JONES	MANAGER	7839	81/04/02	2975	(null)	20
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850	(null)	30
7782	CLARK	MANAGER	7839	81/06/09	2450	(null)	10
7788	SCOTT	ANALYST	7566	87/07/13	3000	(null)	20
7839	KING	PRESIDENT	(null)	81/11/17	5000	(null)	10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7876	ADAMS	CLERK	7788	87/07/13	1100	(null)	20
7900	JAMES	CLERK	7698	81/12/03	950	(null)	30
7902	FORD	ANALYST	7566	81/12/03	3000	(null)	20
7934	MILLER	CLERK	7782	82/01/23	1300	(null)	10

SALGRADE

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

DEPT

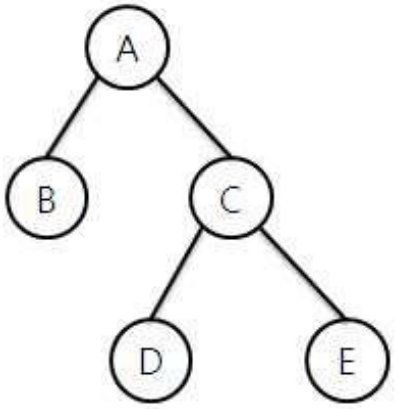
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

	EMPNO	ENAME	MGRNO	MNAME
1	7369	SMITH	7902	FORD
2	7499	ALLEN	7698	BLAKE
3	7521	WARD	7698	BLAKE
4	7566	JONES	7839	KING
5	7654	MARTIN	7698	BLAKE
6	7698	BLAKE	7839	KING
7	7782	CLARK	7839	KING
8	7788	SCOTT	7566	JONES
9	7839	KING	(null)	(null)
10	7844	TURNER	7698	BLAKE
11	7876	ADAMS	7788	SCOTT
12	7900	JAMES	7698	BLAKE
13	7902	FORD	7566	JONES
14	7934	MILLER	7782	CLARK

계층형 질의

- **계층형 데이터**
  - 동일 테이블에 계층적으로 상위와 하위 데이터가 포함된 데이터
  - 엔터티가 순환관계 모델로 설계된 경우 발생
  - 계층형 질의(Hierarchical Query)를 통해 접근 가능

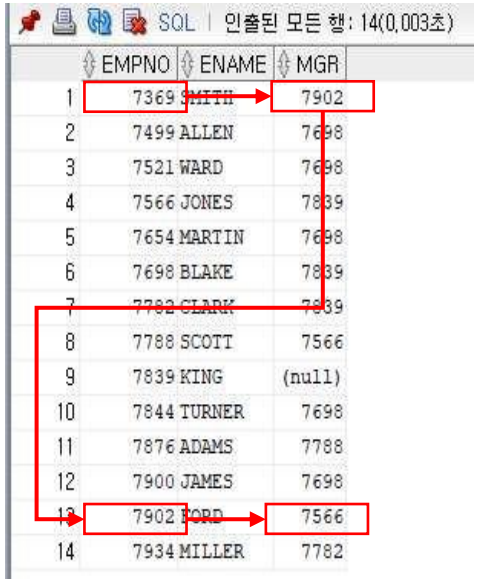
(1)순환관계 데이터 모델



(2)계층형 구조

사원	관리자
A	
B	A
C	A
D	C
E	C

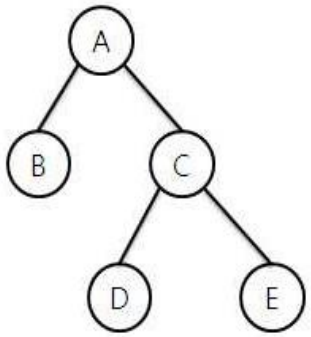
(3)샘플 데이터

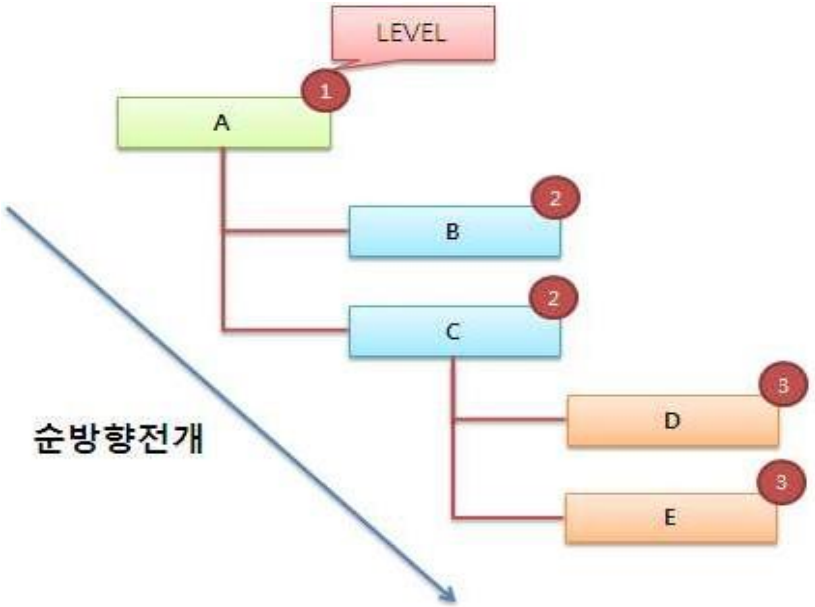


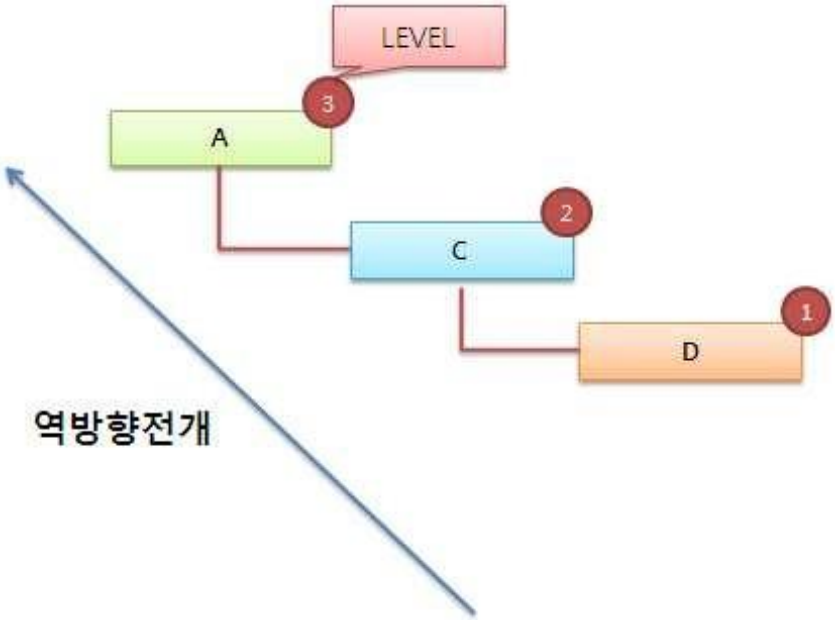
	EMPNO	ENAME	MGR
1	7369	SMITH	7902
2	7499	ALLEN	7698
3	7521	WARD	7698
4	7566	JONES	7839
5	7654	MARTIN	7698
6	7698	BLAKE	7839
7	7782	CLARK	7839
8	7788	SCOTT	7566
9	7839	KING	(null)
10	7844	TURNER	7698
11	7876	ADAMS	7788
12	7900	JAMES	7698
13	7902	FORD	7566
14	7934	MILLER	7782

계층형 질의

- **계층형 질의의 방향**





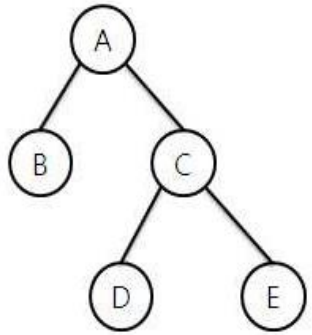


계층형 질의

- **계층형 질의의 구조**
  - START WITH – 시작 조건 지정
    - 예) START WITH MGR IS NULL
    - 예) START WITH EMPNO = 'D'
  - CONNECT BY – 다음에 전개될 방향 지정
    - (순방향) PRIOR 자식 = 부모
      - 예) CONNECT BY PRIOR EMPNO = MGR
    - (역방향) PRIOR 부모 = 자식
      - 예) CONNECT BY PRIOR MGR = EMPNO

```
SELECT        칼럼명...
FROM          테이블명
WHERE         조건...
START WITH    시작 조건
CONNECT BY    PRIOR  방향;
```

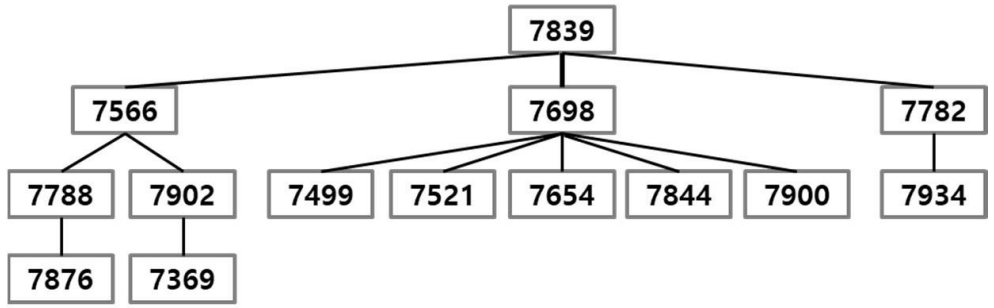
사원	관리자
A	
B	A
C	A
D	C
E	C



계층형 질의

COMMPANY파일

- **순방향 계층형 질의 예**




```
SELECT LEVEL, EMPNO 사원, MGR 관리자, CONNECT_BY_ISLEAF ISLEAF
FROM EMP
START WITH MGR IS NULL
CONNECT BY PRIOR EMPNO = MGR;
```

SQL | 인출된 모든 행: 14(0,003초)

	EMPNO	MGR
1	7369	7902
2	7499	7698
3	7521	7698
4	7566	7839
5	7654	7698
6	7698	7839
7	7782	7839
8	7788	7566
9	7839	(null)
10	7844	7698
11	7876	7788
12	7900	7698
13	7902	7566
14	7934	7782

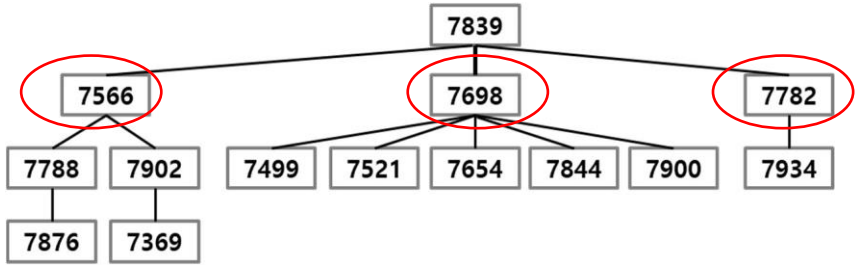
SQL | 인출된 모든 행: 14(0,002초)

	LEVEL	사원	관리자	ISLEAF
1	1	7839	(null)	0
2	2	7566	7839	0
3	3	7788	7566	0
4	4	7876	7788	1
5	3	7902	7566	0
6	4	7369	7902	1
7	2	7698	7839	0
8	3	7499	7698	1
9	3	7521	7698	1
10	3	7654	7698	1
11	3	7844	7698	1
12	3	7900	7698	1
13	2	7782	7839	0
14	3	7934	7782	1

PSEUDO COLUMN
[LEVEL] 시작 노드 = 1, Leaf까지 1씩 증가
[CONNECT_BY_ISLEAF] 해당 노드의 후속 노드가 없으면 1 즉 Leaf이면 1, 그렇지 않으면 0

계층형 질의

- **순방향 계층형 질의 예 (cont'd)**




```
SELECT CONNECT_BY_ROOT EMPNO 시작사원,  SYS_CONNECT_BY_PATH(EMPNO, '/') 경로, EMPNO 사원, MGR 관리자
FROM EMP
START WITH EMPNO in (7566, 7698, 7782)
CONNECT BY PRIOR EMPNO = MGR;
```

PSEUDO COLUMN
[CONNECTED_BY_ROOT] 시작 노드의 해당 칼럼 표시
[SYS_CONNECTED_BY_PATH] 시작 노드부터 현재 노드까지 경로 표시

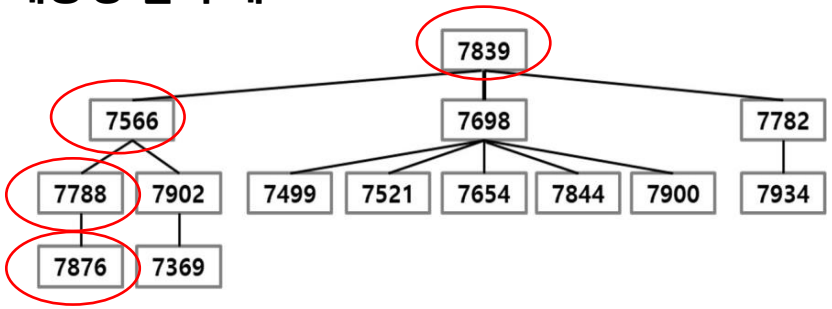
SQL | 인출된 모든 행: 13(0.003초)

	시작사원	경로	사원	관리자
1	7566	/7566	7566	7839
2	7566	/7566/7788	7788	7566
3	7566	/7566/7788/7876	7876	7788
4	7566	/7566/7902	7902	7566
5	7566	/7566/7902/7369	7369	7902
6	7698	/76	7698	7839
7	7698	/76	7499	7698
8	7698	/7698/7521	7521	7698
9	7698	/7698/7654	7654	7698
10	7698	/7698/7844	7844	7698
11	7698	/7698/7900	7900	7698
12	7782	/7782	7782	7839
13	7782	/7782/7934	7934	7782

경로

계층형 질의

- **역방향 계층형 질의 예**




```
SELECT      LEVEL, EMPNO 사원, MGR 관리자, CONNECT_BY_ISLEAF ISLEAF
FROM        EMP
START       WITH        EMPNO = '7876'
CONNECT BY PRIOR MGR = EMPNO;
```

PSEUDO COLUMN
[LEVEL] 시작 노드 = 1, Root까지 1씩 증가
[CONNECT_BY_ISLEAF] 해당 노드의 후속 노드가 없으면 1 즉 Root이면 1, 그렇지 않으면 0

SQL | 인출된 모든 행: 4(0.003초)

	LEVEL	사원	관리자	ISLEAF
1	1	7876	7788	0
2	2	7788	7566	0
3	3	7566	7839	0
4	4	7839	(null)	1

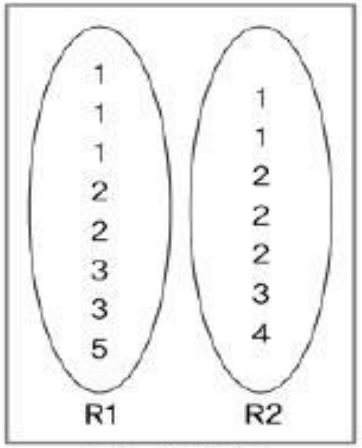
집합 연산자

- **집합 연산자 개요**
  - 여러 질의(Select 문) 결과를 하나로 결합하기 위해 사용
  - 집합 연산의 대상이 되는 두 질의는..
    - SELECT 절의 칼럼 수가 동일해야 하고
    - SELECT 절의 동일 위치에 존재하는 칼럼의 데이터 타입이 상호 호환 가능해야 함
      - 반드시 동일한 데이터 타입일 필요는 없음

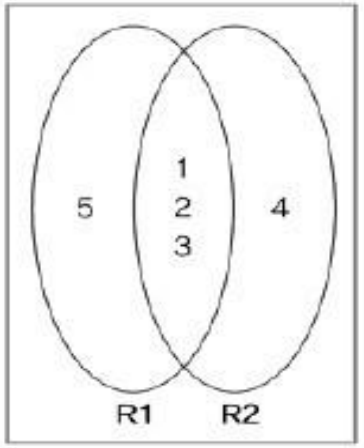
집합 연산자	연산자 의미
UNION	여러 SQL문의 결과에 대한 합집합 (중복된 행은 제거한 후 하나의 행만 출력)
UNION ALL	여러 SQL문의 결과에 대한 합집합 (중복된 행도 삭제하지 않고 모두 출력 → 속도가 빠르므로 우선 고려)
INTERSECT	여러 SQL문의 결과에 대한 교집합 (중복된 행은 제거한 후 하나의 행만 출력)
MINUS (Oracle) / EXCEPT (MS-SQL)	앞의 SQL문의 결과에서 뒤의 SQL문의 결과를 뺀 차집합 (중복된 행은 제거한 후 하나의 행만 출력)

집합 연산자

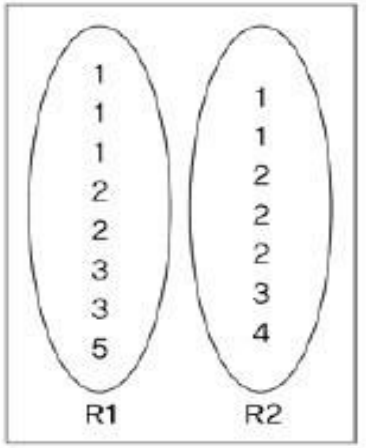
- 집합 연산 예



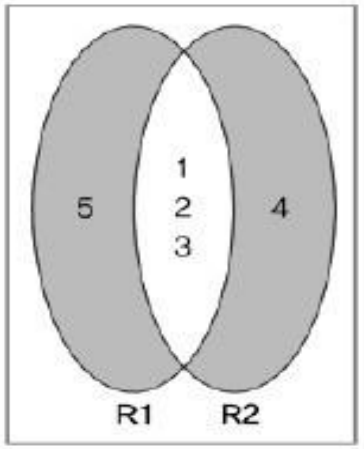
개별 결과집합



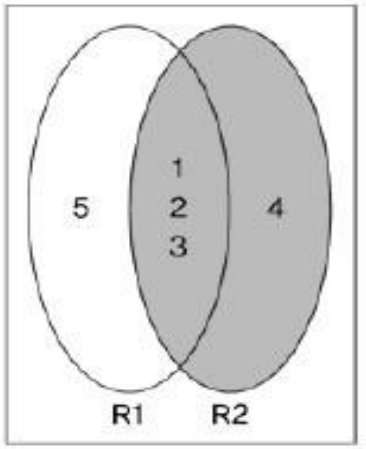
UNION



UNION ALL



INTERSECT



EXCEPT or EXCEPT(MINUS)

집합 연산자

- **집합 연산 질의 예**
  - 집합 연산은 둘 이상의 SELECT 문을 결합하는 것
  - ORDER BY는 집합 연산을 적용한 최종 결과에 대한 정렬
    - 맨 마지막 줄에 한 번만 기술함

- **UNION ALL**

```
SELECT      PLAYER_NAME, BACK_NO, TEAM_ID
FROM        PLAYER
WHERE       TEAM_ID = 'K04'
UNION ALL
SELECT      PLAYER_NAME, BACK_NO, TEAM_ID
FROM        PLAYER
WHERE       TEAM_ID = 'K06'
ORDER BY PLAYER_NAME;
```

집합 연산자

- **UNION ALL**
  - 이질적 성격의 데이터를 한꺼번에 출력하는 연산도 가능

```
SELECT      'T' 구분코드, PLAYER_NAME, TEAM_ID
FROM        PLAYER
WHERE       TEAM_ID = 'K06'
UNION ALL
SELECT      'P' 구분코드, PLAYER_NAME, POSITION
FROM        PLAYER
WHERE       POSITION = 'GK'
ORDER BY    구분코드, TEAM_ID;
```

    - 'P'와 'T'는 상수값

  - 출력 칼럼의 칼럼명은 첫 SELECT문의 칼럼명이 적용됨
    - ORDER BY 구분코드, POSITION → ERROR!!!

집합 연산자

- **INTERSECT**

```
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     TEAM_ID  = 'K06'
INTERSECT
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     POSITION = 'GK';
```

=

```
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     TEAM_ID  = 'K06' AND POSITION = 'GK';
```

– INTERSECT 연산자는 IN 서브쿼리, EXISTS 서브쿼리로도 표현 가능

→ 서브쿼리에서 설명

집합 연산자

- **MINUS**

```
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     TEAM_ID  = 'K06'
MINUS
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     POSITION = 'MF';
```

```
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     TEAM_ID = 'K06' AND POSITION <> 'MF';
```

```
SELECT    TEAM_ID 팀코드, PLAYER_NAME 선수명, POSITION 포지션
FROM      PLAYER
WHERE     TEAM_ID = 'K06'
AND       PLAYER_ID NOT IN (SELECT PLAYER_ID FROM PLAYER WHERE POSITION = 'MF');
```

집합 연산자

- **집합 연산과 ALIAS**

```
SELECT      PLAYER_NAME, TEAM_ID AS TP
FROM        PLAYER
WHERE       TEAM_ID = 'K06'
ORDER BY    TP;
```

```
SELECT      PLAYER_NAME, TEAM_ID AS TP
FROM        PLAYER
WHERE       TEAM_ID = 'K06'
ORDER BY    TEAM_ID;
```

```
SELECT      PLAYER_NAME, TEAM_ID AS ETC
FROM        PLAYER
WHERE       TEAM_ID = 'K06'
UNION ALL
SELECT      PLAYER_NAME, POSITION
FROM        PLAYER
WHERE       POSITION = 'GK'
ORDER BY    ETC;
```

```
SELECT      PLAYER_NAME, TEAM_ID AS ETC
FROM        PLAYER
WHERE       TEAM_ID = 'K06'
UNION ALL
SELECT      PLAYER_NAME, POSITION
FROM        PLAYER
WHERE       POSITION = 'GK'
ORDER BY    TEAM_ID;          ERROR!!!
```

수고하셨습니다