핵심정리

순수 관계 연산자와 SQL 문장 비교

- SELECT 연산은 WHERE 절로 구현
- PROJECT 연산은 SELECT 절로 구현
- (NATURAL) JOIN 연산은 다양한 JOIN 기능으로 구현
- DIVIDE 연산은 현재 사용되지 않음

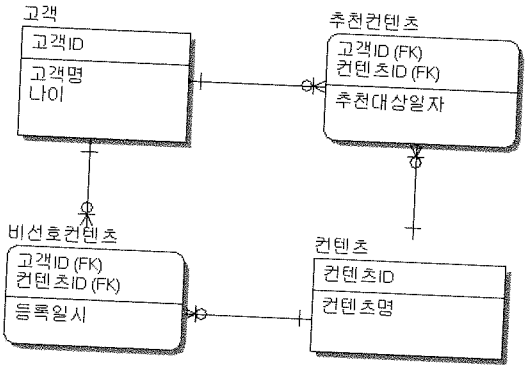
65 다음 중 순수 관계 연산자에 해당하지 않는 것은?

① SELECT
② UPDATE
③ JOIN
④ DIVIDE

66 다음 중 아래 데이터 모델을 참고하여 설명에 맞게 올바르게 작성한 SQL 문장을 2개 고르시오.

아 래

[데이터 모델]



[설명]

우리는 매일 배치작업을 통하여 고객에게 추천할 컨텐츠를 생성하고 고객에게 추천서비스를 제공한다.
추천 컨텐츠 엔터티에서 언제 추천을 해야 하는지를 정의하는 추천 대상일자가 있어 해당일자에만 컨텐츠를 추천해야 한다. 또한 고객이 컨텐츠를 추천 받았을 때 선호하는 컨텐츠가 아닌 경우에는 고객이 비선호 컨텐츠로 분류하여 더 이상 추천 받기를 원하지 않는다. 그러므로 우리는 비선호 컨텐츠 엔터티에 등록된 데이터에 대해서는 추천을 수행하지 않아야 한다.

※ 배치작업이란? 어떤 처리를 연속적으로 하는 것이 아니고 일정량씩 나누어 처리하는 경우 그 일정량을 배치(batch)라고 한다. 배치의 원뜻은 한 묶음이라는 의미다. [기계공학용어사전]
예) 상품을 주문하는 로직은 그당시에 발생하는 트랜잭션에 대한 처리이므로 배치작업이라 표현하지는 않는다. 하지만 상품별 주문량을 집계하는 로직의 경우 특정조건(기간등)으로 일괄처리를 해야함으로 배치작업이라 표현할 수 있다.

핵심정리

ANSI/ISO SQL에서 표시하는 FROM 절의 JOIN 형태

- INNER JOIN
- NATURAL JOIN
- USING 조건절
- ON 조건절
- CROSS JOIN
- OUTER JOIN(LEFT, RIGHT, FULL)

①
```
SELECT C.컨텐츠ID, C.컨텐츠명
FROM   고객 A INNER JOIN 추천컨텐츠 B
ON     (A.고객ID = B.고객ID) INNER JOIN 컨텐츠 C
ON     (B.컨텐츠ID = C.컨텐츠ID)
WHERE A.고객ID = #custId#
AND    B.추천대상일자 = TO_CHAR(SYSDATE, 'YYYY.MM.DD')
AND    NOT EXISTS (SELECT X.컨텐츠ID
                   FROM   비선호컨텐츠 X
                          WHERE  X.고객ID = B.고객ID);
```

②
```
SELECT C.컨텐츠ID, C.컨텐츠명
FROM   고객 A INNER JOIN 추천컨텐츠 B
ON     (A.고객ID = #custId# AND A.고객ID = B.고객ID) INNER JOIN 컨텐츠 C
ON     (B.컨텐츠ID = C.컨텐츠ID) RIGHT OUTER JOIN 비선호컨텐츠 D
ON     (B.고객ID = D.고객ID AND B.컨텐츠ID = D.컨텐츠ID)
WHERE B.추천대상일자 = TO_CHAR(SYSDATE, 'YYYY.MM.DD')
AND    B.컨텐츠ID IS NOT NULL;
```

③
```
SELECT C.컨텐츠ID, C.컨텐츠명
FROM   고객 A INNER JOIN 추천컨텐츠 B
ON     (A.고객ID = B.고객ID) INNER JOIN 컨텐츠 C
ON     (B.컨텐츠ID = C.컨텐츠ID) LEFT OUTER JOIN 비선호컨텐츠 D
ON     (B.고객ID = D.고객ID AND B.컨텐츠ID = D.컨텐츠ID)
WHERE A.고객ID = #custId#
AND    B.추천대상일자 = TO_CHAR(SYSDATE, 'YYYY.MM.DD')
AND    D.컨텐츠ID IS NULL;
```

④
```
SELECT C.컨텐츠ID, C.컨텐츠명
FROM   고객 A INNER JOIN 추천컨텐츠 B
ON     (A.고객ID = #custId# AND A.고객ID = B.고객ID) INNER JOIN 컨텐츠 C
ON     (B.컨텐츠ID = C.컨텐츠ID)
WHERE B.추천대상일자 = TO_CHAR(SYSDATE, 'YYYY.MM.DD')
AND    NOT EXISTS (SELECT X.컨텐츠ID
                   FROM   비선호컨텐츠 X
                          WHERE  X.고객ID = B.고객ID
                          AND    X.컨텐츠ID = B.컨텐츠ID);
```

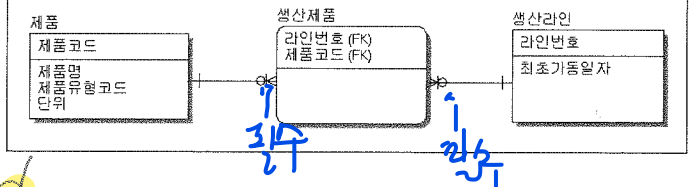
핵심정리

INNER JOIN

INNER JOIN은 OUTER (외부) JOIN과 대비하여 내부 JOIN이라고 하며 JOIN 조건에서 동일한 값이 있는 행만 반환한다.

67 아래는 어느 회사의 생산설비를 위한 데이터 모델의 일부에 대한 설명으로 가장 적절한 것을 2개 고르시오.

아 래



① 제품, 생산제품, 생산라인 엔터티를 Inner Join 하기 위해서 생산제품 엔터티는 WHERE절에 최소 2번이 나타나야 한다.

② 제품과 생산라인 엔터티를 Join시 적절한 Join조건이 없으므로 카티시안 곱(Cartesian Product)이 발생한다.

③ 제품과 생산라인 엔터티에는 생산제품과 대응되지 않는 레코드는 없다.

④ 특정 생산라인번호에서 생산되는 제품의 제품명을 알기위해서는 제품, 생산제품, 생산라인까지 2개 엔터티의 Inner Join인 필요하다.

68 아래의 테이블 스키마 정보를 참고하여, 다음 중 '구매 이력이 있는 고객 중 구매 횟수가 3회 이상인 고객의 이름과 등급을 출력하시오.'라는 질의에 대해 아래 SQL 문장의 ㉠, ㉡에 들어 갈 구문으로 가장 적절한 것은?

아 래

[테이블]
고객(고객번호(PK), 이름, 등급)
구매정보(구매번호(PK),구매금액,고객번호(FK))
* 구매정보 테이블의 고객번호는 고객 테이블의 고객번호를 참조하는 외래키(Foreign Key)이다.

[SQL 문장]
```
SELECT A.이름, A.등급
FROM 고객 A
㉠
GROUP BY A.이름, A.등급
㉡
```

① ㉠ : INNER JOIN 구매정보 B ON A.고객번호=B.고객번호
㉡ : HAVING SUM(B.구매번호) >=3

② ㉠ : INNER JOIN 구매정보 B ON A.고객번호=B.고객번호
㉡ : HAVING COUNT(B.구매번호) >=3

③ ㉠ : LEFT OUTER JOIN 구매정보 B ON A.고객번호=B.고객번호

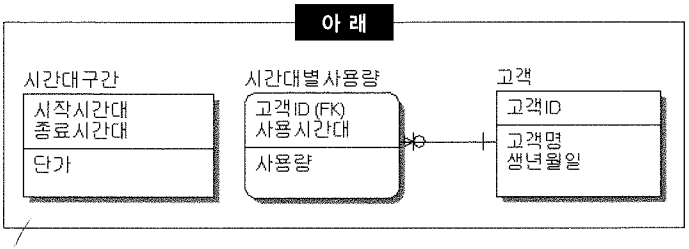
㉡ : HAVING SUM(B.구매번호) >=3

④ ㉠ : INNER JOIN 구매정보 B ON A.고객번호=B.고객번호

㉡ : WHERE B.구매번호 >=3

69 **아래는 어느 회사의 정산 데이터 모델의 일부이며 고객이 서비스를 사용한 시간대에 따라 차등 단가를 적용하려고 한다. 다음 중 시간대별사용량 테이블을 기반으로 고객별 사용금액을 추출하는 SQL으로 가장 적절한 것은?**

아 래



① SELECT A.고객ID, A.고객명, SUM(B.사용량 * C.단가) AS 사용금액
FROM 고객 A INNER JOIN 시간대별사용량 B
ON (A.고객ID = B.고객ID) INNER JOIN 시간대구간 C
ON (B.사용시간대 >= C.시작시간대 AND B.사용시간대 <= C.종료시간대)
GROUP BY A.고객ID, A.고객명
ORDER BY A.고객ID, A.고객명;

② SELECT A.고객ID, A.고객명, SUM(B.사용량 * C.단가) AS 사용금액
FROM 고객 A INNER JOIN 시간대별사용량 B INNER JOIN 시간대구간 C
ON (A.고객ID = B.고객ID AND B.사용시간대
BETWEEN C.시작시간대 AND C.종료시간대)
GROUP BY A.고객ID, A.고객명
ORDER BY A.고객ID, A.고객명;

③ SELECT A.고객ID, A.고객명, SUM(B.사용량 * C.단가) AS 사용금액
FROM 고객 A INNER JOIN 시간대별사용량 B
ON (A.고객ID = B.고객ID) INNER JOIN 시간대구간 C
ON B.사용시간대 BETWEEN C.시작시간대 AND C.종료시간대
GROUP BY A.고객ID, A.고객명
ORDER BY A.고객ID, A.고객명;

④ SELECT A.고객ID, A.고객명, SUM(B.사용량 * C.단가) AS 사용금액
FROM 고객 A INNER JOIN 시간대별사용량 B
ON (A.고객ID = B.고객ID) BETWEEN JOIN 시간대구간 C
GROUP BY A.고객ID, A.고객명
ORDER BY A.고객ID, A.고객명;

핵심정리

70 다음 중 팀(TEAM) 테이블과 구장(STADIUM) 테이블의 관계를 이용해서 소속팀이 가지고 있는 전용구장의 정보를 팀의 정보와 함께 출력하는 SQL을 작성할 때 결과가 다른 것은?

① SELECT T.REGION_NAME, T.TEAM_NAME, T.STADIUM_ID, S.STADIUM_NAME
FROM TEAM T INNER JOIN STADIUM S
USING (T.STADIUM_ID = S.STADIUM_ID);

② SELECT TEAM.REGION_NAME, TEAM.TEAM_NAME, TEAM.STADIUM_ID, STADIUM.STADIUM_NAME
FROM TEAM INNER JOIN STADIUM
ON (TEAM.STADIUM_ID = STADIUM.STADIUM_ID);

③ SELECT T.REGION_NAME, T.TEAM_NAME, T.STADIUM_ID, S.STADIUM_NAME
FROM TEAM T, STADIUM S
WHERE T.STADIUM_ID = S.STADIUM_ID;

④ SELECT TEAM.REGION_NAME, TEAM.TEAM_NAME, TEAM.STADIUM_ID, STADIUM.STADIUM_NAME
FROM TEAM, STADIUM
WHERE TEAM.STADIUM_ID = STADIUM.STADIUM_ID;

CROSS JOIN
테이블 간 JOIN 조건이 없는 경우 생길 수 있는 모든 데이터의 조합을 말한다. 결과는 양쪽 집합의 M*N 건의 데이터 조합이 발생한다.

71 아래의 사례1은 Cartesian Product를 만들기 위한 SQL 문장이며 사례1과 같은 결과를 얻기 위해 사례2 SQL 문장의 ㉠ 안에 들어갈 내용을 작성하시오.

아 래

[사례1]
```
SELECT ENAME, DNAME
FROM   EMP, DEPT
ORDER  BY ENAME;
```

[사례2]
```
SELECT ENAME, DNAME
FROM   EMP  ㉠  DEPT
ORDER  BY ENAME;
```

핵심정리

LEFT OUTER JOIN

조인 수행시 먼저 표기된 좌측 테이블에 해당하는 데이터를 먼저 읽은 후, 나중 표기된 우측 테이블에서 JOIN 대상 데이터를 읽어 온다. 즉, Table A와 B가 있을 때 (Table 'A'가 기준이 됨), A와 B를 비교해서 B의 JOIN 칼럼에서 같은 값이 있을 때 그 해당 데이터를 가져오고, B의 JOIN 칼럼에서 같은 값이 없는 경우에는 B 테이블에서 가져오는 칼럼들은 NULL 값으로 채운다.

72 다음 중 아래 테이블들을 대상으로 SQL 문장을 수행한 결과로 가장 적절한 것은?

아 래

[테이블 : OS]

OSID(PK)	OS명
100	Android
200	iOS
300	Bada

[테이블 : 단말기]

단말기ID(PK)	단말기명	OSID(FK)
1000	A1000	100
2000	B2000	100
3000	C3000	200
4000	D3000	300

[테이블 : 고객]

고객번호(PK)	고객명	단말기ID(FK)
11000	홍길동	1000
12000	강감찬	NULL
13000	이순신	NULL
14000	안중근	3000
15000	고길동	4000
16000	이대로	4000

[SQL]

```
SELECT A.고객번호, A.고객명, B.단말기ID, B.단말기명, C.OSID,
  C.OS명
FROM   고객 A LEFT OUTER JOIN 단말기 B
ON     (A.고객번호 IN (11000, 12000) AND A.단말기ID =
        B.단말기ID) LEFT OUTER JOIN OS C
ON     (B.OSID = C.OSID)
ORDER BY A.고객번호;
```

①

고객번호	고객명	단말기ID	단말기명	OSID	OS명
11000	홍길동	1000	A1000	100	Android
12000	강감찬	NULL	NULL	NULL	NULL
13000	이순신	NULL	NULL	NULL	NULL
14000	안중근	NULL	NULL	NULL	NULL
15000	고길동	NULL	NULL	NULL	NULL
16000	이대로	NULL	NULL	NULL	NULL

②

고객번호	고객명	단말기ID	단말기명	OSID	OS명
11000	홍길동	1000	A1000	100	Android
12000	강감찬	NULL	NULL	NULL	NULL

핵심정리

FULL OUTER JOIN
조인 수행시 좌측, 우측 테이블의 모든 데이터를 읽어 JOIN하여 결과를 생성한다. 즉, TABLE A와 B가 있을 때 (TABLE 'A', 'B' 모두 기준이 됨), RIGHT OUTER JOIN과 LEFT OUTER JOIN의 결과를 합집합으로 처리한 결과와 동일하다.

③

고객번호	고객명	단말기ID	단말기명	OSID	OS명
11000	홍길동	1000	A1000	100	Android

④

고객번호	고객명	단말기ID	단말기명	OSID	OS명
11000	홍길동	1000	A1000	100	Android
12000	강감찬	NULL	NULL	NULL	NULL
13000	이순신	NULL	NULL	NULL	NULL
14000	안중근	3000	C3000	200	iOS
15000	고길동	4000	D4000	300	Bada
16000	이대로	4000	D4000	300	Bada

73 다음 중 아래 (1), (2), (3)의 SQL에서 실행결과가 같은 것은?

아 래

```
(1)  SELECT A.ID, B.ID
     FROM TBL1 A FULL OUTER JOIN TBL2 B
     ON A.ID = B.ID

(2)  SELECT A.ID, B.ID
     FROM TBL1 A LEFT OUTER JOIN TBL2 B
     ON A.ID = B.ID
     UNION
     SELECT A.ID, B.ID
     FROM TBL1 A RIGHT OUTER JOIN TBL2 B
     ON A.ID = B.ID

(3)  SELECT A.ID, B.ID
     FROM TBL1 A, TBL2 B
     WHERE A.ID = B.ID
     UNION ALL
     SELECT A.ID, NULL
     FROM TBL1 A
     WHERE NOT EXISTS (SELECT 1 FROM TBL2 B WHERE A.ID = B.ID)
     UNION ALL
     SELECT NULL, B.ID
     FROM TBL2 B
     WHERE NOT EXISTS (SELECT 1 FROM TBL1 A WHERE B.ID = A.ID)
```

① 1, 2
② 1, 3
③ 2, 3
④ 1, 2, 3

핵심정리

OUTER JOIN 문장 예시

• LEFT OUTER JOIN

```
SELECT X.KEY1,
       Y.KEY2
FROM TAB1 X LEFT
       OUTER JOIN
       TAB2 Y
ON (X.KEY1=Y.KEY2)
```

• RIGHT OUTER JOIN

```
SELECT X.KEY1,
       Y.KEY2
FROM TAB1 X RIGHT
       OUTER JOIN
       TAB2 Y
ON (X.KEY1=Y.KEY2)
```

• FULL OUTER JOIN

```
SELECT X.KEY1,
       Y.KEY2
FROM TAB1 X FULL
       OUTER JOIN
       TAB2 Y
ON (X.KEY1=Y.KEY2)
```

74 아래의 EMP 테이블과 DEPT 테이블에서 밑줄 친 속성은 주키이며 EMP.C는 DEPT와 연결된 외래키이다. EMP 테이블과 DEPT 테이블을 LEFT, FULL, RIGHT 외부조인(outer join)하면 생성되는 결과 건수로 가장 적절한 것은?

아 래

EMP 테이블

A	B	C
1	b	w
3	d	w
5	y	y

DEPT 테이블

C	D	E
w	1	10
z	4	11
v	2	22

① 3건, 5건, 4건
② 4건, 5건, 3건
③ 3건, 4건, 4건
④ 3건, 4건, 5건

75 신규 부서의 경우 일시적으로 사원이 없는 경우도 있다고 가정하고 DEPT와 EMP를 조인하되 사원이 없는 부서 정보도 같이 출력하도록 할 때, 아래 SQL 문장의 ㉠ 안에 들어갈 내용을 기술하시오.

아 래

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

L, O, J

76 다음 중 아래와 같은 데이터 상황에서 SQL의 수행 결과로 가장 적절한 것은?

아 래

TAB1

C1	C2
A	1
B	2
C	3
D	4
E	5

TAB2

C1	C2
B	2
C	3
D	4

```
SELECT *
FROM TAB1 A LEFT OUTER JOIN TAB2 B
  ON ( A.C1 = B.C1 AND B.C2 BETWEEN 1 AND 3)
```

①

C1	C2	C1	C2
A	1		
B	2	B	2
C	3	C	3
D	4	D	4
E	5		

②

C1	C2	C1	C2
A	1		
B	2	B	2
C	3	C	3
D	4		
E	5		

③

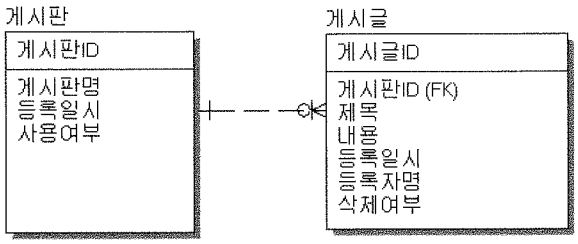
C1	C2	C1	C2
A	1		
B	2	B	2
C	3	C	3

④

C1	C2	C1	C2
A	1		
B	2	B	2
C	3	C	3
D	4	D	4

77 아래와 같은 데이터 모델에서 ORACLE을 기준으로 SQL을 작성하였다. 그러나 SQL Server에서도 동일한 결과를 보장할 수 있도록 ANSI 구문으로 SQL을 변경하려고 한다. 다음 중 아래의 SQL을 ANSI 표준 구문으로 변경한 것으로 가장 적절한 것은?

아 래




```
[SQL]
SELECT A.게시판ID, A.게시판명, COUNT(B.게시글ID) AS CNT
FROM   게시판 A, 게시글 B
WHERE  A.게시판ID = B.게시판ID(+)
AND    B.삭제여부(+) = 'N'
AND    A.사용여부 = 'Y'
GROUP BY A.게시판ID, A.게시판명
ORDER BY A.게시판ID;
```

LEFT OUTER JOIN 조건

①
```
SELECT A.게시판ID, A.게시판명, COUNT(B.게시글ID) AS CNT
FROM   게시판 A LEFT OUTER JOIN 게시글 B
ON     (A.게시판ID = B.게시판ID AND B.삭제여부 = 'N')
WHERE A.사용여부 = 'Y'
GROUP BY A.게시판ID, A.게시판명
ORDER BY A.게시판ID;
```

②
```
SELECT A.게시판ID, A.게시판명, COUNT(B.게시글ID) AS CNT
FROM   게시판 A LEFT OUTER JOIN 게시글 B
ON     (A.게시판ID = B.게시판ID AND A.사용여부 = 'Y')
WHERE B.삭제여부 = 'N'
GROUP BY A.게시판ID, A.게시판명
ORDER BY A.게시판ID;
```

③
```
SELECT A.게시판ID, A.게시판명, COUNT(B.게시글ID) AS CNT
FROM   게시판 A LEFT OUTER JOIN 게시글 B
ON     (A.게시판ID = B.게시판ID)
WHERE A.사용여부 = 'Y'
AND    B.삭제여부 = 'N'
GROUP BY A.게시판ID, A.게시판명
ORDER BY A.게시판ID;
```

④ SELECT A.게시판ID, A.게시판명, COUNT(B.게시글ID) AS CNT
FROM 게시판 A RIGHT OUTER JOIN 게시글 B
ON (A.게시판ID = B.게시판ID AND A.사용여부 = 'Y' AND
B.삭제여부 = 'N')
GROUP BY A.게시판ID, A.게시판명
ORDER BY A.게시판ID;

78 다음과 같은 2개의 릴레이션이 있다고 가정하자. student의 기본키는 st_num이고, department의 기본키는 dept_num이다. 또한 student의 d_num은 department의 dept_num을 참조하는 외래키이다. 아래 SQL문의 실행 결과 건수는?

아 래

```
SELECT count(st_name)
FROM   student s
WHERE  not exists
       (SELECT *
        FROM   department d
        WHERE  s.d_num   = d.dept_num
          and  dept_name = '전자계산학과');
```

Student

st_num	st_name	d_num
1001	Yoo	10
1002	Kim	30
1003	Lee	20
1004	Park	10
1005	Choi	20
1006	Jeong	10

Department

dept_num	dept_name
10	컴퓨터공학과
20	원자력공학과
30	전자계산학과

79 (SQL Server) 다음 중 아래의 SQL과 동일한 결과를 추출하는 SQL은? (단, 테이블 TAB1, TAB2의 PK 컬럼은 A, B 이다)

TAB1

A	B
1	2
3	4

TAB2

A	B
3	4

아 래

```
[SQL]
SELECT A, B
FROM   TAB1
EXCEPT
SELECT A, B
FROM   TAB2;
```

1, 2

④
```
SELECT TAB2.A, TAB2.B
    FROM   TAB1, TAB2
    WHERE TAB1.A ◇ TAB2.A
    AND    TAB1.B ◇ TAB2.B
```
3, 4

②
```
SELECT TAB1.A, TAB1.B
    FROM   TAB1
    WHERE  TAB1.A NOT IN (SELECT TAB2.A
                          FROM   TAB2)
    AND    TAB1.B NOT IN (SELECT TAB2.B
                          FROM   TAB2);
```
1, 2

③
```
SELECT TAB2.A, TAB2.B
    FROM   TAB1, TAB2
    WHERE TAB1.A = TAB2.A
    AND    TAB1.B = TAB2.B
```
3, 4

④
```
SELECT TAB1.A, TAB1.B
    FROM   TAB1
    WHERE  NOT EXISTS (SELECT 'X'
                       FROM   TAB2
                       WHERE TAB1.A = TAB2.A
                       AND    TAB1.B = TAB2.B);
```
3, 4

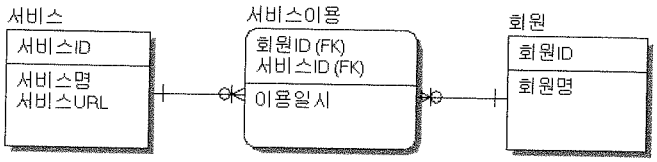
1, 2

집합 연산자의 종류

집합 연산자	연산자의 의미
UNION	여러 개의 SQL문의 결과에 대한 합집합으로 결과에서 모든 중복된 행은 하나의 행으로 만든다.
UNION ALL	여러 개의 SQL문의 결과에 대한 합집합으로 중복된 행도 그대로 결과로 표시된다. 즉, 단순히 결과만 합쳐놓은 것이다. 일반적으로 여러 질의 결과가 상호 배타적인(Exclusive)일 때 많이 사용한다. 개별 SQL문의 결과가 서로 중복되지 않는 경우, UNION과 결과가 동일하다. (결과의 정렬 순서에는 차이가 있을 수 있음)
INTERSECT	여러 개의 SQL문의 결과에 대한 교집합이다. 중복된 행은 하나의 행으로 만든다.
EXCEPT	앞의 SQL문의 결과에서 뒤의 SQL문의 결과에 대한 차집합이다. 중복된 행은 하나의 행으로 만든다. (일부 데이터베이스는 MINUS를 사용함)

80 아래와 같은 데이터 모델에 대해 SQL을 수행 하였다. 다음 중 수행된 SQL과 동일한 결과를 도출하는 SQL은?

아 래



[수행 SQL]

```
SELECT A.서비스ID, B.서비스명, B.서비스URL
FROM   (SELECT 서비스ID
        FROM   서비스
           INTERSECT
        SELECT 서비스ID
        FROM   서비스이용) A, 서비스 B
WHERE  A.서비스ID = B.서비스ID;
```

①
```
SELECT B.서비스ID, A.서비스명, A.서비스URL
FROM   서비스 A, 서비스이용 B
WHERE  A.서비스ID = B.서비스ID;
```

②
```
SELECT X.서비스ID, X.서비스명, X.서비스URL
FROM   서비스 X
WHERE  NOT EXISTS (SELECT 1
                   FROM   (SELECT 서비스ID
                           FROM   서비스
                              MINUS
                           SELECT 서비스ID
                           FROM   서비스이용) Y
                   WHERE X.서비스ID = Y.서비스ID);
```

③
```
SELECT B.서비스ID, A.서비스명, A.서비스URL
FROM   서비스 A LEFT OUTER JOIN 서비스이용 B
ON     (A.서비스ID = B.서비스ID)
WHERE B.서비스ID IS NULL
GROUP BY B.서비스ID, A.서비스명, A.서비스URL;
```

④
```
SELECT A.서비스ID, A.서비스명, A.서비스URL
FROM   서비스 A
WHERE 서비스ID IN (SELECT 서비스ID
                   FROM   서비스이용
                   MINUS
                   SELECT 서비스ID
                   FROM   서비스);
```

핵심정리

SELECT 칼럼명1,
　　칼럼명2, ...
FROM 테이블명1
[WHERE 조건식]
[[GROUP BY
　칼럼(Column)이나
　표현식
[HAVING 그룹조건식]]
집합 연산자
SELECT 칼럼명1,
　　칼럼명2, ...
FROM 테이블명2
[WHERE 조건식]
[[GROUP BY
　칼럼(Column)이나
　표현식
[HAVING 그룹조건식]]
[ORDER BY 1, 2
　[ASC또는 DESC] ;
→ ORDER BY는 집합 연산을 적용한 최종 결과에 대한 정렬 처리이므로 가장 마지막 줄에 한번만 기술한다.

81 SET OPERATOR 중에서 수학의 교집합과 같은 기능을 하는 연산자로 가장 적절한 것은?

① UNION
② INTERSECT
③ MINUS
④ EXCEPT

82 다음 중 아래의 EMP 테이블의 데이터를 참조하여 실행한 SQL의 결과로 가장 적절한 것은?

아 래

```
SELECT ENAME AAA, JOB AAB
FROM EMP
WHERE EMPNO = 7369
UNION ALL
SELECT ENAME BBA, JOB BBB
FROM EMP
WHERE EMPNO = 7566
ORDER BY 1, 2;
```

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	1980-12-17	800		20
7499	ALLEN	SALESMAN	7698	1981-02-20	1600	300	30
7521	WARD	SALESMAN	7698	1981-02-22	1250	500	30
7566	JONES	MANAGER	7839	1981-04-02	2975		20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250	1400	30
7698	BLAKE	MANAGER	7839	1981-05-01	2850		30
7782	CLARK	MANAGER	7839	1981-06-09	2450		10
7788	SCOTT	ANALYST	7566	1987-07-13	3000		20
7839	KING	PRESIDENT		1981-11-17	5000		10
7844	TURNER	SALESMAN	7698	1981-09-08	1500	0	30
7876	ADAMS	CLERK	7788	1987-07-13	1100		20
7900	JAMES	CLERK	7698	1981-12-03	950		30
7902	FORD	ANALYST	7566	1981-12-03	3000		20
7934	MILLER	CLERK	7782	1982-01-23	1300		10

①

AAA	AAB
SMITH	CLERK
JONES	MANAGER

②

AAA	AAB
JONES	MANAGER
SMITH	CLERK

③

BBA	BBB
SMITH	CLERK
JONES	MANAGER

④

BBA	BBB
JONES	MANAGER
SMITH	CLERK

83 다음 중 아래 TBL1, TBL2 테이블에 대해 SQL을 수행한 결과인 것은?

아 래

[테이블 : TBL1]

COL1	COL2
AA	A1
AB	A2

[테이블 : TBL2]

COL1	COL2
AA	A1
AB	A2
AC	A3
AD	A4

[SQL]

```
SELECT COL1, COL2, COUNT(*) AS CNT
FROM  (SELECT COL1, COL2
       FROM   TBL1
       UNION ALL
       SELECT COL1, COL2
       FROM   TBL2
       UNION
       SELECT COL1, COL2
       FROM   TBL1)
GROUP BY COL1, COL2;
```

①

COL1	COL2	CNT
AA	A1	1
AB	A2	1
AC	A3	1
AD	A4	1

②

COL1	COL2	CNT
AA	A1	2
AB	A2	2
AC	A3	1
AD	A4	1

③

COL1	COL2	CNT
AA	A1	3
AB	A2	3
AC	A3	1
AD	A4	1

④

COL1	COL2	CNT
AA	A1	3
AB	A2	3
AC	A3	2
AD	A4	2

핵심정리

84 다음 중 아래에서 테이블 T1, T2에 대한 가, 나 두 개의 쿼리 결과 조회되는 행의 수로 가장 적절한 것은?

아 래

T1(A,B,C)

A	B	C
A3	B2	C3
A1	B1	C1
A2	B1	C2

T2(A,B,C)

A	B	C
A1	B1	C1
A3	B2	C3

가.
```
SELECT A, B, C FROM R1
  UNION ALL
  SELECT A, B, C FROM R2
```

나.
```
SELECT A, B, C FROM R1
  UNION
  SELECT A, B, C FROM R2
```

① 가: 5개, 나: 3개
② 가: 5개, 나: 5개
③ 가: 3개, 나: 3개
④ 가: 3개, 나: 5개

일반 집합 연산자를 SQL과 비교

- UNION 연산은 UNION 기능으로,
- INTERSECTION 연산은 INTERSECT 기능으로,
- DIFFERENCE 연산은 EXCEPT(Oracle은 MINUS) 기능으로,
- PRODUCT 연산은 CROSS JOIN 기능으로 구현되었다.

85 다음 중 아래와 같은 집합이 존재 할 때, 집합 A와 B에 대하여 집합연산을 수행한 결과 집합 C가 되는 경우 이용되는 데이터베이스 집합연산은?

아 래

집합 A = {가, 나, 다, 라},
집합 B = {다, 라, 마, 바},
집합 C = {다, 라}

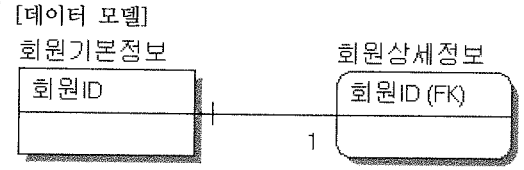
① Union
② Difference
③ Intersection
④ Product

핵심정리

86 아래와 같은 데이터 모델에 대한 설명으로 가장 적절한 것은? (단, 시스템적으로 회원기본정보와 회원상세정보는 1:1, 양쪽 필수 관계임을 보장한다.)

아 래

[데이터 모델]



① 회원ID 컬럼을 대상으로 (회원기본정보 EXCEPT 회원상세정보) 연산을 수행하면 회원상세정보가 등록되지 않은 회원ID가 추출된다.

② 회원ID 컬럼을 대상으로 (회원기본정보 UNION ALL 회원상세정보) 연산을 수행한 결과의 건수는 회원기본정보의 전체건수와 동일하다.

③ 회원ID 컬럼을 대상으로 (회원기본정보 INTERSECT 회원상세정보) 연산을 수행한 결과의 건수와 두 테이블을 회원ID로 JOIN 연산을 수행한 결과의 건수는 동일하다.

④ 회원ID 컬럼을 대상으로 (회원기본정보 INTERSECT 회원상세정보) 연산을 수행한 결과와 (회원기본정보 UNION 회원상세정보) 연산을 수행한 결과는 다르다.

PRIOR : CONNECT BY절에 사용되며, 현재 읽은 칼럼을 지정한다. PRIOR 자식 = 부모 형태를 사용하면 계층구조에서 부모 데이터에서 자식 데이터(부모 → 자식) 방향으로 전개하는 순방향 전개를 한다. 그리고 PRIOR 부모 = 자식 형태를 사용하면 반대로 자식 데이터에서 부모 데이터(자식 → 부모) 방향으로 전개하는 역방향 전개를 한다.

87 아래와 같은 데이터 상황에서 아래의 SQL을 수행할 경우 정렬 순서상 2번째 표시될 값을 적으시오.

아 래

TAB1

C1	C2	C3
1		A
2	1	B
3	1	C
4	2	D

```
SELECT C3
FROM TAB1
START WITH C2 IS NULL
CONNECT BY PRIOR C1 = C2
ORDER SIBLINGS BY C3 DESC
```

핵심정리

- START WITH절은 계층 구조 전개의 시작 위치를 지정하는 구문이다. 즉, 루트 데이터를 지정한다.(액세스)
- ORDER SIBLINGS BY : 형제 노드(동일 LEVEL) 사이에서 정렬을 수행한다.

88 다음 중 Oracle 계층형 질의에 대한 설명으로 가장 부적절한 것은?

① START WITH절은 계층 구조의 시작점을 지정하는 구문이다.
② ORDER SIBLINGS BY절은 형제 노드 사이에서 정렬을 지정하는 구문이다.
③ 순방향전개란 부모 노드로부터 자식 노드 방향으로 전개하는 것을 말한다.
④ 루트 노드의 LEVEL 값은 0이다.

89 다음 중 아래와 같은 사원 테이블에 대해서 SQL을 수행하였을 때의 결과로 가장 적절한 것은?

아 래

[테이블 : 사원]

사원번호(PK)	사원명	입사일자	매니저사원번호(FK)
001	홍길동	2012-01-01	NULL
002	강감찬	2012-01-01	001
003	이순신	2013-01-01	001
004	이민정	2013-01-01	001
005	이병헌	2013-01-01	NULL
006	안성기	2014-01-01	005
007	이수근	2014-01-01	005
008	김병만	2014-01-01	005

[SQL]
```
SELECT 사원번호, 사원명, 입사일자, 매니저사원번호
FROM   사원
START WITH 매니저사원번호 IS NULL
CONNECT BY PRIOR 사원번호 = 매니저사원번호
AND 입사일자 BETWEEN '2013-01-01' AND '2013-12-31'
ORDER SIBLINGS BY 사원번호;
```

①

사원번호(PK)	사원명	입사일자	매니저사원번호(FK)
001	홍길동	2012-01-01	NULL
003	이순신	2013-01-01	001
004	이민정	2013-01-01	001
005	이병헌	2013-01-01	NULL

②

사원번호(PK)	사원명	입사일자	매니저사원번호(FK)
003	이순신	2013-01-01	001
004	이민정	2013-01-01	001
005	이병헌	2013-01-01	NULL

③

사원번호(PK)	사원명	입사일자	매니저사원번호(FK)
001	홍길동	2012-01-01	NULL

④

사원번호(PK)	사원명	입사일자	매니저사원번호(FK)
001	홍길동	2012-01-01	NULL
005	이병헌	2013-01-01	NULL
006	안성기	2014-01-01	005
007	이수근	2014-01-01	005
008	김병만	2014-01-01	005