sql-study-day12

08 - 3 SQL-99 표준 문법으로 배우는 조인

NATURAL JOIN

- NATURAL JOIN은 등가 조인을 대신해 사용할 수 있는 조인 방식으로 조인대상이 되는 두 테이블에 이름과 자료형이 같은 열을 찾은 후 그열을 기준으로 등가 조인을 해 주는 방식이다.

JOIN ~ USING

JOIN ~ USING 키워드를 사용한 조인 역시 기존 등가 조인을 대신하는 조인 방식이다.

```
FROM TABLE1 JOIN TABLE2 USING (조인에 사용한 기준열)
```

다른 조인 방식과 마찬가지로 조인된 결과 행을 추가로 제한할 때 WHERE절에 조건식을 추가하여 함께 사용할 수 있다.

JOIN ~ ON

가장 범용성 있는 JOIN ~ ON 키워드를 사용한 조인 방식에서는 기존 WHERE절에 있는 조인 조건식을 ON 키워드 옆에 작성한다. 조인 기준 조건식은 ON에 명시하고 그 밖의 출력 행을 걸러 내기 위해 WHERE 조건식을 따로 사용하는 방식이다.

```
FROM TABLE JOIN TABLE2 ON (조인 조건식)
```

OUTER JOIN

OUTER JOIN 키워드는 외부 조인에 사용한다. WHERE절이 아닌 FROM절에서 외부 조인을 선언한다.

왼쪽 외부 조인 (Left Outer Join)	기존	WHERE TABLE1.COL1 = TABLE2.COL1(+)
	SQL-99	FROM TABLE1 LEFT OUTER JOIN TABLE2 ON (조인 조건식)
오른쪽 외부 조인 (Right Outer Join)	기존	WHERE TABLE1.COL1(+) = TABLE2.COL1
	SQL-99	FROM TABLE1 RIGHT OUTER JOIN TABLE2.ON (조인 조건식)
전체 외부 조인 (Full Outer Join)	기존	기본 문법은 없음 (UNION 집합 연산자를 활용)
	SQL-99	FROM TABLE1 FULL OUTER JOIN TABLE2 ON (조인 조건식)

SQL-99 조인 방식에서 세 개 이상의 테이블을 조인할 때

기존 조인 방식은 FROM절에 조인 테이블을 명시하고 조인 관련 조건식을 WHERE절에 명시하기 때문에 테이블 수가 두 개를 넘더라도 다음과 같이 작성하면 아무 문제가 없다

```
FROM TABLE1, TABLE2 TABLE3
WHERE TABLE1.COL = TABLE2.COL
ABD TABLE2.COL = TABLE3.COM
```

하지만 FROM절에 조인 관련 내용을 작성해야 하는 SQL-99 방식에서는 FROM절에 두 개 테이블을 키워드로 조인한 바로 옆에 SQL-99 방식의 조인 내용을 추가로 작성하면 세 개 이상의 테이블도 조인할 수 있다.

```
FROM TABLE1 JOIN TABLE2 ON (조인 조건식)
JOIN TABLE3 ON (조건식)
```

잊기 전에 한 번 더!

- Q1. 급여(SAL)가 2000 초과인 사원들의 부서 정보, 사원 정보를 오른쪽과 같이 출력해 보세요. (단 SQL-99 이전 방식과 SQL-99 방식을 각각 사용하여 작성하세요.

```
// SQL-99 이전 방식
SELECT D.DEPTNO, D.DNAME, E.EMPNO, E.ENAME, E.SAL
FROM EMP E, DEPT D
WHERE E.DEPTNO = D.DEPTNO
AND SAL > 2000;
```

```
// SQL-99 방식
SELECT DEPTNO, D.DNAME, E.EMPNO, E.ENAME, E.SAL
FROM EMP E JOIN DEPT D USING (DEPTNO)
WHERE SAL > 2000;
```

- Q2. 오른쪽과 같이 각 부서별 평균 급여, 최대 급여, 최소 급여, 사원수를 출력해 보세요. (단 SQL-99 이전 방식과 SQL-99 방식을 각각 사용하여 작성하세요.)

```
// SQL-99 이전 방식
SELECT D.DEPTNO, D.DNAME,
TRUNC(AVG(E.SAL)) AS AVG_SAL,
MAX(E.SAL) AS MAX_SAL,
MIN(E.SAL) AS MIN_SAL,
COUNT(*) AS CNT
FROM EMP E, DEPT D
WHERE E.DEPTNO = D.DEPTNO
GROUP BY D.DEPTNO, D.DNAME;
```

```
// SQL-99 방식
SELECT DEPTNO, D.DNAME, TRUNC(AVG(E.SAL)) AS AVG_SAL,
MAX(E.SAL) AS MAX_SAL,
MIN(E.SAL) AS MIN_SAL,
COUNT(*) AS CNT
FROM EMP E JOIN DEPT D USING (DEPTNO)
GROUP BY DEPTNO, D.DNAME;
```

- Q3. 모든 부서 정보와 사원 정보를 오른쪽과 같이 부서 번호, 사원 이름순으로 정렬하여 출력해 보세요. (단 SQL-99 이전 방식과 SQL-99 방식을 각각 사용하여 작성하세요.)

```
// SQL-99 이전 방식
SELECT D.DEPTNO, D.DNAME, E.EMPNO, E.ENAME, E.JOB, E.SAL
FROM EMP E, DEPT D
WHERE E.DEPTNO(+) = D.DEPTNO
ORDER BY D.DEPTNO, E.ENAME;
```

```
// SQL-99 방식
SELECT D.DEPTNO, D.DNAME, E.EMPNO, E.ENAME, E.JOB, E.SAL
```

```
FROM EMP E RIGHT OUTER JOIN DEPT D ON (D.DEPTNO = E.DEPTNO)
ORDER BY DEPTNO, E.ENAME;

// DEPTNO, D.DEPTNO ㅠㅠ ?
```

- Q4. 다음과 같이 모든 부서 정보, 사원 정보, 급여 등급 정보, 각 사원의 직속 상관의 정보를 부서 번호, 사원 번호 순서로 정렬하여 출력해 보세요. (단 SQL-99 이전 방식과 SQL-99 방식을 각각 사용하여 작성하세요.)

```
// SQL-99 이전 방식
SELECT D.DEPTNO, D.DNAME,
       E.EMPNO, E.ENAME, E.MGR, E.SAL, E.DEPTNO,
       S.LOSAL, S.HISAL, S.GRADE,
       E2.EMPNO AS MGR_EMPNO, E2.ENAME AS MGR_ENAME
  FROM EMP E, DEPT D, SALGRADE S, EMP E2
 WHERE E.DEPTNO(+) = D.DEPTNO
   AND E.SAL BETWEEN S.LOSAL(+) AND S.HISAL(+)
   AND E.MGR = E2.EMPNO(+)
ORDER BY D.DEPTNO, E.EMPNO;
```

```
// SQL-99 방식
SELECT D.DEPTNO, D.DNAME,
       E.EMPNO, E.ENAME, E.MGR, E.SAL, E.DEPTNO,
       S.LOSAL, S.HISAL, S.GRADE,
       E2.EMPNO AS MGR_EMPNO, E2.ENAME AS MGR_ENAME
  FROM EMP E RIGHT OUTER JOIN DEPT D
               ON (E.DEPTNO = D.DEPTNO)
             LEFT OUTER JOIN SALGRADE S
               ON (E.SAL BETWEEN S.LOSAL AND S.HISAL)
             LEFT OUTER JOIN EMP E2
               ON (E.MGR = E2.EMPNO)
ORDER BY D.DEPTNO, E.EMPNO;
```