UNION ALL과 UNION

`UNION ALL` 또는 `UNION` 구문을 사용하면 복수의 질의문을 하나로 합친 결과를 얻을 수 있다. 테이블의 내용이나 이름이 똑같지 않더라도 구조가 같다면 합칠 수 있다.

UNION ALL

먼저 `UNION ALL` 부터 알아보자. 15장에서 만든 `음반` 테이블과 `노래` 테이블의 제목을 한 번에 조회해 보자.

```
SELECT 제목 FROM 음반
UNION ALL
SELECT 제목 FROM 노래
```

위 질의문을 실행한 결과는 다음과 같다.[1]

제목
Girl's Day Party #1
Everyday
Expectation
여자대통령
갸우뚱
Shuppy Shuppy
Control
영러브
한번만 안아줘
반짝반짝
기대해
I Don't Mind
Easy go
여자대통령

컬럼명이 달라도 `UNION ALL` 을 수행하는 데 문제가 없다. 따라서 위의 질의문을 다음과 같이 바꿔도 같은 결과를 얻을 수 있다.

```
SELECT 제목 FROM 음반
UNION ALL
SELECT 제목 AS Title FROM 노래
```

실제 테이블에 있는 데이터와 위 질의문의 실행 결과를 비교해 보면, 음반의 제목들이 먼저 나오고 그 뒤에 노래 제목들이 나온 것을 알 수 있다. 그리고 '여자대통령'이라는 노래는 동명의 음반에 실렸으므로 결과에 두 번 나타난 것을 볼 수 있다.

UNION

`UNION` 구문은 `UNION ALL` 과 비슷하지만, 중복되는 데이터를 제외한 결과를 돌려준다는 점에서 차이가 있다.

```
SELECT 제목 FROM 음반
UNION
SELECT 제목 FROM 노래
```

위의 질의를 실행하면 '여자대통령'이 한 번만 나온다.

제목
Control
Easy go
Everyday
Expectation
Girl's Day Party #1
I Don't Mind
Shuppy Shuppy
갸우뚱
기대해
반짝반짝
여자대통령
영러브
한번만 안아줘

`UNION ALL` 과 `UNION` 의 또다른 차이점으로, `UNION` 은 질의 결과가 정렬된다는 점을 들 수 있다. 질의 결과에서 중복을 제거하기 위해 먼저 정렬을 수행하기 때문이다. 그래서 제목을 기준으로 오름차순 정렬이 되어, 음반과 노래 제목이 뒤섞인 것을 볼 수 있다.

정렬과 중복 제거가 굳이 필요하지 않을 때는 `UNION ALL` 을 사용하는 것이 메모리와 속도 측면에서 유리할 것이다.

UNION ALL과 JOIN을 이용해 조건절을 대체

조건절의 예로 들었던 질의를 다음과 같이 바꿀 수 있다.

```
SELECT Name, bdate, Birthday.MM, MonthAbb
FROM Birthday, (
    SELECT '01' AS MM, 'Jan.' AS MonthAbb
    UNION ALL
    SELECT '02' AS MM, 'Feb.' AS MonthAbb
    UNION ALL
    SELECT '03' AS MM, 'Mar.' AS MonthAbb
    UNION ALL
    SELECT '04' AS MM, 'Apr.' AS MonthAbb
    UNION ALL
    SELECT '05' AS MM, 'May.' AS MonthAbb
    UNION ALL
    SELECT '06' AS MM, 'Jun.' AS MonthAbb
    UNION ALL
    SELECT '07' AS MM, 'Jul.' AS MonthAbb
    UNION ALL
    SELECT '08' AS MM, 'Aug.' AS MonthAbb
    UNION ALL
    SELECT '09' AS MM, 'Sep.' AS MonthAbb
    UNION ALL
    SELECT '10' AS MM, 'Oct.' AS MonthAbb
    UNION ALL
    SELECT '11' AS MM, 'Nov.' AS MonthAbb
    UNION ALL
    SELECT '12' AS MM, 'Dec.' AS MonthAbb
) AS Months
WHERE Birthday.MM = Months.MM
```

필자가 생각하기에 RDBMS는 테이블 연산에 최적화되어 있고 조건절은 부가적인 기능인 것 같다. 그러므로 이와 같이 테이블 간의 조인으로 계산을 할 수 있는 경우에는 조건절을 사용하는 것보다 조인으로 처리하는 것이 낫다. 실행 시간도 조건절을 사용했을 때보다 이 방식이 더 적게 걸렸다.[2]

절차지향적인 프로그래밍 언어에 익숙하면 아무래도 조건절에 먼저 손이 가기 때문에 발상의 전환이 필요하다.

연습 문제

16.1 위 질의를 실행한 결과는 조건절을 사용했을 때와 비슷하지만 약간 차이가 있다. 결과를 비교하고 그 이유를 생각해보라.

---

1. 이 장의 설명을 위해 예제 데이터를 추가했다. 실행 결과에 '여자대통령' 행이 없으면 `음반`, `노래`, `수록곡` 테이블에 `INSERT` 문으로 데이터를 추가하거나, 테이블을 삭제한 다음 앞 장의 `CREATE TABLE` 문을 실행해 다시 생성하라. ↩

2. 필자의 컴퓨터에서 조건절을 사용한 질의는 9ms, `UNION ALL` 과 `JOIN` 을 사용한 질의는 5~6ms가 걸렸다. ↩