지금은 인덱스를 이용한 정렬의 제거가 필요하다

SQL을 작성하다 보면 대부분의 고객들은 특정 컬럼으로 데이터를 정렬하기를 원하는 경우가 대다수이다. 이와 같은 요청이 발생한다면 대다수의 개발자들은 해당 SQL의 마지막에 우선 정렬을 수행할 수 있는 ORDER BY 절을 추가하게 된다. 이렇게 추가된 ORDER BY 절은 해당 시스템을 구성하는 대부분의 애플리케이션에 추가된다. 이와 같은 ORDER BY 절을 제거할 수 있다면 기존의 시스템은 전혀 다른 새로운 시스템이 될 것이다.

ORDER BY 절에 의해 발생하는 정렬이 해당 시스템의 성능 저하를 유발한다는 것은 대부분의 사람들이 아는 상식 아닌 상식이다. 그럼에도 불구하고 우리 주변에는 모든 SQL에 ORDER BY 절을 설정해 정렬을 수행하고 있다. 이와 같은 정렬은 데이터베이스의 성능을 저하시키게 되는데, 이 가운데 주요한 SQL에 대해 우리가 정렬을 제거할 수 있다면 시스템은 놀랄만한 성능 향상을 얻을 것이다. 이제부터 SQL에서 ORDER BY 절에 의한 정렬이 발생하지 않게 하는 방법을 함께 확인해 보자.

정렬은 시스템의 성능을 저하시키는 주범

정렬은 시스템의 성능을 저하시키는 주범이라는 것을 모르는 사람은 없을 것이다. 하지만 정렬이 많은 경우 왜 성능 저하의 주범인지를 정확히 이해하는 사람도 거의 없다. 그렇다면 많은 정렬이 이뤄지면 왜 성능 저하가 발생하는 것인가? 단지 정렬을 수행하기 때문인가? 나를 알고 적을 알면 백전백승이라고 했듯이 정렬의 단점을 정확히 파악하는 것만으로도 우리는 정렬을 왜 제거해야 하는지를 이해할 수 있을 것이다.

정렬은 우선 데이터를 추출한 후 어떤 특정 컬럼으로 다시 추출하는 것을 의미한다. 이러한 점에서도 정렬은 성능을 저하시키게 된다. 이미 추출한 데이터를 다시 엑세스한다는 것은 누가 보아도 성능 저하를 발생시키게 될 것이다. 여기에 더욱 성능을 저하시키는 하나의 요소가 있다. 그것이 바로 디스크 I/O이다. 정렬은 기본적으로 해당 작업을 수행하는 프로세스의 메모리 영역을 사용하게 되어 있다. 하지만 메모리는 유한하며 충분치 않은 것이 아직까지의 현실이다. 그렇기 때문에 대용량의 데이터에 대해 정렬을 수행한다면 메모리에서 처리를 모두 하지 못하기 때문에 디스크에서 정렬을 수행하게 된다. 해당 디스크 영역이 TEMP 테이블스페이스가 되는 것이다. 이와 같이 디스크 영역을 이용해 정렬을 수행한다면 디스크 I/O의 증가로 성능은 저하된다. 성능은 디스크 I/O의 양과 밀접하다. 그렇기 때문에 정렬이 많이 발생한다면 디스크 I/O의 증가로 성능을 저하시키게 되는 것이다.

기본적인 SQL의 정렬을 제거해 보자

가장 기본적인 SQL의 정렬을 제거하는 것이 정렬 제거의 시작일 것이다. 아래의 예제를 확인해 보자.

```
SQL> SELECT 부서번호, 사원이름, 급여
       FROM 사원
ORDER BY 부서번호, 사원이름 ;
```

위의 SQL에서 ORDER BY 절에 의해 발생하는 정렬을 제거하고자 한다면 어떻게 해야 하는가? ORDER BY 절을 그냥 지우면 되는 것인가? ORDER BY 절을 무조건 지운다면 당연히 정렬된 데이터는 추출되지 않을 것이다. 그렇다면 어떻게 정렬을 제거하고 정렬된 데이터를 추출해야 하는 것인가?

정렬을 수행하지 않고 정렬된 데이터를 추출하기 위해서는 어디엔가 정렬된 데이터가 저장되어 있어야 한다. 그렇다면 테이블에 정렬된 데이터가 저장되어 있는가? 테이블에는 Insert되는 순서에 의해 저장되므로 정렬된 데이터가 저장될 수는 없다. 그렇다면 어디에서 우리는 정렬된 데이터를 찾을 수 있는가? 테이블의 주변을 확인해 본다면 쉽게 찾을 수 있다. 정렬된 데이터는 인덱스에 저장되어 있다. 그렇기 때문에 인덱스만 잘 이용한다면

위의 SQL은 ORDER BY에 의한 정렬 없이 정렬된 데이터가 추출될 수 있을 것이다. 사원 테이블에 부서번호+사원이름 인덱스를 생성해서 해당 인덱스를 이용한다면 위의 SQL은 정렬이 제거된다. 왜 이런 현상이 발생하는 것인가? 부서번호+사원이름 인덱스는 기본적으로 부서번호 인덱스로 정렬되어 있고 동일 부서번호에 대해서는 두 번째 컬럼인 사원이름 컬럼으로 정렬이 수행된다. 그렇기 때문에 해당 인덱스를 이용한다면 결과는 부서번호로 데이터가 정렬되며 동일한 부서번호 데이터는 사원이름 컬럼의 값으로 정렬이 수행될 것이다.

이와 같이 인덱스를 이용해 우리는 원하는 정렬된 데이터를 추출할 수 있다. 이와 같이 정렬된 데이터를 추출한다면 별도의 정렬은 발생하지 않게 된다. 물론, 해당 SQL에서 ORDER BY 절을 설정하는 경우나 제거하는 경우 모두 정렬된 데이터가 추출된다.

WHERE 조건이 존재하는 SQL의 정렬을 제거해 보자

앞에서는 WHERE 조건이 존재하지 않는 경우를 확인해 보았다. 그렇다면 WHERE 조건이 존재한다면 어떻게 정렬 없이 정렬된 데이터를 추출할 수 있겠는가? 아래의 예제를 확인해 보자.

```
SQL> SELECT 부서번호, 사원이름, 급여
       FROM 사원
       WHERE 위치 = '서울'
ORDER BY 부서번호, 사원이름 ;
```

위의 SQL과 같이 WHERE 절에 위치 조건이 존재한다면 인덱스를 이용해 어떻게 정렬된 데이터를 추출할 수 있겠는가? 인덱스를 위치+부서번호+사원이름으로 생성하는 경우를 확인해 보자. 이와 같이 인덱스를 생성한다면 위치 컬럼의 값으로 데이터가 정렬되며 동일한 위치 컬럼에 대해서는 부서번호 컬럼으로 정렬된다. 위치 컬럼의 값과 부서번호 컬럼이 값이 동일한 데이터에 대해서는 마지막 컬럼인 사원이름 컬럼의 값으로 정렬된다. 그렇기 때문에 위치+부서번호+사원이름 인덱스를 이용한다면 위의 SQL은 자동으로 정렬된 데이터가 추출될 것이다. 하나의 예제를 추가로 확인해 보자.

```
SQL> SELECT 부서번호, 사원이름, 급여
       FROM 사원
       WHERE 위치 LIKE '서울%'
ORDER BY 부서번호, 사원이름 ;
```

위와 같이 위치 컬럼의 조건이 동등(=) 조건이 아닌 LIKE 연산자라면 어떤 현상이 발생하는가? 위치+부서번호+사원이름 인덱스를 이용한다면 위치 컬럼의 값으로 먼저 정렬이 수행된다. 예를 들어 위치 컬럼의 값과 부서번호 컬럼의 값이 (서울북부,10)인 데이터와 (서울남부, 20)인 데이터는 어떻게 되겠는가? 해당 인덱스를 이용한다면 (서울남부, 20)인 데이터가 (서울북부,10)인 데이터보다 먼저 추출될 것이다. 결국, ORDER BY 절 없이는 부서번호 컬럼과 사원이름 컬럼으로 정렬을 수행할 수 없게 된다. 하지만 부서번호+사원이름 인덱스를 이용한다면 위의 SQL은 어떻게 되겠는가? 부서번호+사원이름 인덱스를 이용한다면 위의 SQL은 부서번호 컬럼과 사원번호 컬럼의 값으로 자동 정렬된다. 하지만, WHERE 조건의 컬럼이 인덱스에 존재하지 않으므로 인덱스를 전체 엑세스해야 한다. 잘못하면 정렬을 제거하기 위해 인덱스를 전체 스캔하는 부분이 더 많은 성능 저하를 발생시킬 수 있으므로 주의해야 한다. 결국, 인덱스를 이용해서 정렬을 수행하기 위해서는 아래와 같은 조건을 만족하는 인덱스가 존재해야 한다.

- 점 조건+……+점 조건+ORDER BY 절의 1번째 컬럼+…… ORDER BY 절의 n번째 컬럼 (점 조건은 동등(=) 조건을 의미)

정렬을 제거한다는 것은 데이터의 증가뿐 아니라 n-Row 처리의 아키텍처를 구현하기 위해 매우 중요한 요소임에 틀림없다. 그렇다고 해당 시스템의 모든 정렬을 제거할 필요는 없으며 해당 시스템에서 중요한 SQL에 대해 정렬을 제거하도록 노력해야 할 것이다. 정렬의 제거는 대용량화되는 데이터베이스의 성능을 최적화하기 위해 반드시 필요한 요소임을 명심하길 바란다.

권순용 kwontra@hanmail.net | Data Consulting 업무를 수행하는 (주)엑시엄의 대표이사이며 DBA로 시작하여 SQL 튜닝, 데이터베이스 아키텍처 및 모델링 업무를 주로 수행했다. 데이터베이스 교육에도 많은 관심을 가지고 있으며 저서로는 『Perfect! 오라클 실전 튜닝』, 『초보자를 위한 오라클 10g』 및 『INSIDE SQL』이 있다. 또한, 데이터 액세스 최적화에 대한 특허를 출원했다.