

Chapter 17. ROLLUP과 CUBE

부분 집계에 사용되는 GROUP BY 절에 ROLLUP과 CUBE를 사용하면 다양한 통계 자료를 출력할 수 있다. 예를 들어, 사원 테이블에서 업무별 평균 급여, 부서별 평균 급여, 업무 및 부서별 평균 급여 등을 검색하려면, 각각의 결과를 별도의 쿼리 문장으로 작성해야 하지만 ROLLUP과 CUBE 옵션을 사용하면 한번에 모든 결과를 얻어낼 수 있다. 이번 장에서는 GROUP BY 절의 ROLLUP과 CUBE에 대해서 설명한다.

ROLLUP

ROLLUP은 GROUP BY에 의해서 출력되는 부분집계 결과에 누적된 부분 집계를 추가해준다. ROLLUP을 사용한 GROUP BY의 문법은 다음과 같다.

```
SELECT [column,] group_function(column) ...
FROM table
[WHERE condition]
[GROUP BY [ROLLUP] group_by_expression]
[HAVING having_expression]
[ORDER BY column];
```

ROLLUP은 뒤에 기술된 컬럼들을 좌측 컬럼에서 우측 컬럼으로 진행하면서 그룹화하고, 각각의 그룹에 대하여 GROUP BY에 의해 부분집계를 수행한다. 예를 들면, 사원 테이블의 부서번호 및 업무별 부분집계에 ROLLUP을 추가하면 결과는 다음과 같다.

```
SQL> SELECT DEPTNO, JOB, SUM(SAL)
2 FROM EMP
3 GROUP BY ROLLUP(DEPTNO, JOB);
```

DEPTNO	JOB	SUM(SAL)	
10	CLERK	1300	
10	MANAGER	2450	
10	PRESIDENT	5000	
10		8750	← 10번 부서의 급여 합계
20	CLERK	800	
20	ANALYST	3000	
20	MANAGER	2975	
20		6775	← 20번 부서의 급여 합계
30	CLERK	950	
30	MANAGER	2850	
30	SALESMAN	5600	
30		9400	← 30번 부서의 급여 합계
		24925	← 전체 부서의 급여 합계

13 개의 행이 선택되었습니다.

결과를 살펴보면 부서번호 및 업무별 급여 합계에 부서별 급여 합계와 전체 급여 합계 결과가 추가되었다. 즉, ROLLUP(DEPTNO, JOB)은 (DEPTNO, JOB), (DEPTNO), () 그룹으로 분류되어 각각의 그룹이 GROUP BY에 의해 부분 집계 된다. ROLLUP에 의해 생성되는 그

룹의 수는 ROLLUP에 기술된 컬럼의 개수가 n 이면 $n+1$ 개의 그룹이 만들어진다.

```
SELECT DEPTNO, JOB, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB
UNION ALL
SELECT DEPTNO, NULL, SUM(SAL)
FROM EMP
GROUP BY DEPTNO
UNION ALL
SELECT NULL, NULL, SUM(SAL)
FROM EMP
GROUP BY ();
```

CUBE

CUBE도 ROLLUP과 마찬가지로 GROUP BY에 의해서 출력되는 부분집계 결과에 누적된 부분 집계를 추가해준다. CUBE를 사용한 GROUP BY의 문법은 다음과 같다.

```
SELECT [column,] group_function(column) ...
FROM table
[WHERE condition]
[GROUP BY [CUBE] group_by_expression]
[HAVING having_expression]
[ORDER BY column];
```

CUBE는 뒤에 기술된 컬럼들에 대한 모든 조합을 그룹화하고, 각각의 그룹에 대하여 GROUP BY에 의해 부분집계를 수행한다. 예를 들면, 사원 테이블의 부서번호 및 업무별 부분집계에 CUBE를 추가하면 결과는 다음과 같다.

```
SQL> SELECT DEPTNO, JOB, SUM(SAL)
2 FROM EMP
3 GROUP BY CUBE(DEPTNO, JOB);
```

DEPTNO	JOB	SUM(SAL)		
		24925	← 전체	부서의 급여 합계
	CLERK	3050	← CLERK	업무의 급여 합계
	ANALYST	3000	← ANALYST	업무의 급여 합계
	MANAGER	8275	← MANAGER	업무의 급여 합계
	SALESMAN	5600	← SALESMAN	업무의 급여 합계
	PRESIDENT	5000	← PRESIDENT	업무의 급여 합계
10		8750	← 10번	부서의 급여 합계
10	CLERK	1300		
10	MANAGER	2450		
10	PRESIDENT	5000		
20		6775	← 20번	부서의 급여 합계
20	CLERK	800		
20	ANALYST	3000		
20	MANAGER	2975		
30		9400	← 30번	부서의 급여 합계
30	CLERK	950		
30	MANAGER	2850		
30	SALESMAN	5600		

18 개의 행이 선택되었습니다.

결과를 살펴보면 부서번호 및 업무별 급여 합계에 부서별 급여 합계, 업무별 급여 합계와 전체 급여 합계 결과가 추가되었다. 즉, CUBE(DEPTNO, JOB)은 (DEPTNO, JOB), (DEPTNO), (JOB), () 그룹으로 분류되어 각각의 그룹이 GROUP BY에 의해 부분 집계 된다. CUBE에 의해 생성되는 그룹의 수는 CUBE에 기술된 컬럼의 개수가 n 이면 2^n 개의 그룹이 만들어진다.

```
SELECT DEPTNO, JOB, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB
UNION ALL
SELECT DEPTNO, NULL, SUM(SAL)
FROM EMP
GROUP BY DEPTNO
UNION ALL
SELECT NULL, JOB, SUM(SAL)
FROM EMP
GROUP BY JOB
UNION ALL
SELECT NULL, NULL, SUM(SAL)
FROM EMP;
```

GROUPING 함수

GROUPING 함수는 ROLLUP과 CUBE를 사용했을 때, 각각의 행들이 부분집계에 참여했는지를 표시해준다. GROUPING 함수를 사용하는 SQL 문장의 문법은 다음과 같다.

```

SELECT [column,] group_function(column) ... ,
       GROUPING(expr)
FROM table
[WHERE condition]
[GROUP BY [ROLLUP][CUBE] group_by_expression]
[HAVING having_expression]
[ORDER BY column];

```

GROUPING 함수는 반드시 ROLLUP 또는 CUBE와 함께 사용하여야 하며, GROUPING 함수의 결과를 보면 출력 결과에서 부분집계의 누적집계를 쉽게 찾을 수 있다. 또한, GROUPING 함수를 사용할 때, 테이블에 저장된 NULL과 ROLLUP 또는 CUBE에 의해서 생성된 NULL값을 구별할 수 있어야 한다.

GROUPING 함수는 0 또는 1을 리턴하며 각각의 의미는 다음과 같다.

- 0을 리턴하는 경우
 - 해당 행이 집계 연산에 포함되었음.
 - 컬럼에 표시되는 NULL은 해당 컬럼에 저장된 NULL 값을 의미함.
- 1을 리턴하는 경우
 - 해당 행이 집계 연산에서 제외됨.
 - 컬럼에 표시되는 NULL은 ROLLUP 또는 CUBE가 추가한 값을 의미함.

다음은 GROUPING 함수를 사용하는 방법이다.

```

SQL> SELECT DEPTNO, JOB, SUM(SAL),
2  GROUPING(DEPTNO),
3  GROUPING(JOB)
4  FROM EMP
5  GROUP BY ROLLUP(DEPTNO, JOB);

```

DEPTNO	JOB	SUM(SAL)	GROUPING(DEPTNO)	GROUPING(JOB)
10	CLERK	1300	0	0
10	MANAGER	2450	0	0
10	PRESIDENT	5000	0	0
10		8750	0	1
20	CLERK	800	0	0
20	ANALYST	3000	0	0
20	MANAGER	2975	0	0
20		6775	0	1
30	CLERK	950	0	0
30	MANAGER	2850	0	0
30	SALESMAN	5600	0	0
30		9400	0	1
		24925	1	1

13 개의 행이 선택되었습니다.

위의 결과를 보면 GROUPING 함수의 결과에 1이 표시된 부분은 부분집계의 누적치이며, 해당 컬럼에는 NULL이 표시된다.

GROUPING SETS

GROUPING SETS는 GROUP BY를 확장한 것으로서 ROLLUP 또는 CUBE가 지정된 원칙에 따라 컬럼들을 그룹화하는 반면 GROUPING SETS을 이용하면 사용자가 원하는 컬럼들로 구성된 그룹을 만들 수 있다. Oracle 서버는 GROUPING SETS에 정의된 컬럼들을 그룹화하고 각 그룹에 대하여 GROUP BY를 수행 한 후, 그 결과를 UNION ALL 연산하게 된다. GROUPING SETS는 테이블을 한번만 검색하므로 GROUPING SETS에 많은 컬럼들을 기술 할수록 처리 효율은 향상된다. GROUPING SETS를 사용하는 방법은 다음과 같다.

```
SQL> SELECT DEPTNO, JOB, MGR, SUM(SAL)
2  FROM EMP
3  GROUP BY GROUPING SETS
4  ((DEPTNO, JOB, MGR),
5  (DEPTNO, MGR), (JOB, MGR));
```

DEPTNO	JOB	MGR	SUM(SAL)
10	PRESIDENT		5000
10	CLERK	7782	1300
10	MANAGER	7839	2450
20	ANALYST	7566	3000
20	MANAGER	7839	2975
20	CLERK	7902	800
30	CLERK	7698	950
30	SALESMAN	7698	5600
30	MANAGER	7839	2850
10			5000
10		7782	1300
10		7839	2450
20		7566	3000
20		7839	2975
20		7902	800
30		7698	6550
30		7839	2850
	CLERK	7698	950
	CLERK	7782	1300
	CLERK	7902	800
	ANALYST	7566	3000
	MANAGER	7839	8275
	SALESMAN	7698	5600
	PRESIDENT		5000

24 개의 행이 선택되었습니다.

위 문장은 다음 문장과 동일하다.

```

SELECT DEPTNO, JOB, MGR, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB, MGR
UNION ALL
SELECT DEPTNO, NULL, MGR, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, MGR
UNION ALL
SELECT NULL, JOB, MGR, SUM(SAL)
FROM EMP
GROUP BY JOB, MGR;

```

그러면, 다음과 같은 CUBE 문장과 ROLLUP 문장을 GROUPING SETS로 변환해보자.

CUBE(a, b, c)	GROUPING SETS ((a, b, c), (a, b), (a, c), (b, c), (a), (b), (c), ())
ROLLUP(a, b, c)	GROUPING SETS ((a, b, c), (a, b), (a), ())

복합 컬럼의 처리

복합 컬럼이란 ROLLUP, CUBE에 기술된 일부 컬럼이 ()에 의해 하나의 단위로 처리되는 컬럼이다. 예를 들면, ROLLUP (a, (b, c), d)에서 (b, c)는 하나의 컬럼으로 처리되기 때문에 GROUP BY에 의해 처리되는 그룹은 (a, b, c, d), (a, b, c), (a), ()이 된다. 복합 컬럼을 사용한 예는 다음과 같다.

```

SQL> SELECT DEPTNO, JOB, MGR, SUM(SAL)
2 FROM EMP
3 GROUP BY ROLLUP (DEPTNO, (JOB, MGR));

```

DEPTNO	JOB	MGR	SUM(SAL)
10	CLERK	7782	1300
10	MANAGER	7839	2450
10	PRESIDENT		5000
10			8750
20	CLERK	7902	800
20	ANALYST	7566	3000
20	MANAGER	7839	2975
20			6775
30	CLERK	7698	950
30	MANAGER	7839	2850
30	SALESMAN	7698	5600
30			9400
			24925

13 개의 행이 선택되었습니다.

위 문장은 아래 문장과 동일하다.

```

SELECT DEPTNO, JOB, MGR, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB, MGR
UNION ALL
SELECT DEPTNO, NULL, NULL, SUM(SAL)
FROM EMP
GROUP BY DEPTNO
UNION ALL
SELECT NULL, NULL, NULL, SUM(SAL)
FROM EMP
GROUP BY ();

```

그러면, 이와 같은 방식으로 다음과 같은 경우는 어떤 그룹들이 생성 될 것인지를 생각해 보자.

GROUPING SETS	GROUP BY
GROUP BY GROUPING SETS (a, b, c)	GROUP BY a UNION ALL GROUP BY b UNION ALL GROUP BY c
GROUP BY GROUPING SETS (a, b, (b, c))	GROUP BY a UNION ALL GROUP BY b UNION ALL GROUP BY b, c
GROUP BY GROUPING SETS ((a, b, c))	GROUP BY a, b, c
GROUP BY GROUPING SETS (a, (b), ())	GROUP BY a UNION ALL GROUP BY b UNION ALL GROUP BY ()
GROUP BY GROUPING SETS (a, ROLLUP (b, c))	GROUP BY a UNION ALL GROUP BY ROLLUP (b, c)

연결된 그룹의 처리

연결된 그룹이란 GROUP BY 뒤에 ROLLUP, CUBE, GROUPING SETS가 콤마로 분리되어 같이 기술된 것이다. 이런 경우에 각각의 그룹 집단이 다른 그룹 집단과 서로 조합되어 전개된다. 예를 들어, GROUP BY GROUPING SETS (a, b), GROUPING SETS (c, d)에 의해 만들어지는 그룹은 (a, c), (a, d), (b, c), (b, d)가 된다. 결합된 그룹을 사용하는 방법은 다음과 같다.

```
SQL> SELECT DEPTNO, JOB, MGR, SUM(SAL)
2 FROM EMP
3 GROUP BY DEPTNO, ROLLUP(JOB), CUBE(MGR);
```

DEPTNO	JOB	MGR	SUM(SAL)
10	PRESIDENT		5000
10	CLERK	7782	1300
10	MANAGER	7839	2450
20	ANALYST	7566	3000
20	MANAGER	7839	2975
20	CLERK	7902	800
30	CLERK	7698	950
30	SALESMAN	7698	5600
30	MANAGER	7839	2850
10			5000
10		7782	1300
10		7839	2450
20		7566	3000
20		7839	2975
20		7902	800
30		7698	6550
30		7839	2850
10	CLERK		1300
10	MANAGER		2450
10	PRESIDENT		5000
10			8750
20	CLERK		800
20	ANALYST		3000
20	MANAGER		2975
20			6775
30	CLERK		950
30	MANAGER		2850
30	SALESMAN		5600
30			9400

29 개의 행이 선택되었습니다.

위 문장은 아래 문장과 동일하다.

```
SELECT DEPTNO, JOB, MGR, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB, MGR
UNION ALL
SELECT DEPTNO, JOB, NULL, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, JOB
UNION ALL
SELECT DEPTNO, NULL, MGR, SUM(SAL)
FROM EMP
GROUP BY DEPTNO, MGR
UNION ALL
SELECT DEPTNO, NULL, NULL, SUM(SAL)
FROM EMP
GROUP BY DEPTNO;
```


복습

1. ROLLUP (a, b)에 의해서 만들어지는 그룹을 기술하시오.
2. CUBE (a, b)에 의해서 만들어지는 그룹을 기술하시오.
3. ROLLUP (a, (b, c, d), e)에 의해서 만들어지는 그룹을 기술하시오.
4. CUBE ((a, b), (c, d), e)에 의해서 만들어지는 그룹을 기술하시오.
5. 사원 테이블에서 업무 및 부서코드별 평균 급여, 업무별 평균 급여, 전체 평균 급여를 출력하는 SQL 문장을 작성하시오.

JOB	DEPTNO	AVG(SAL)
CLERK	10	1300
CLERK	20	800
CLERK	30	950
CLERK		1016.66667
ANALYST	20	3000
ANALYST		3000
MANAGER	10	2450
MANAGER	20	2975
MANAGER	30	2850
MANAGER		2758.33333
SALESMAN	30	1400
SALESMAN		1400
PRESIDENT	10	5000
PRESIDENT		5000
		2077.08333

15 개의 행이 선택되었습니다.

