Advanced Group Operations

- **ROLLUP, CUBE, GROUPING SETS** 이 중요한 이유

1. 대부분의 **UNION ALL + GROUP BY** 쿼리들을 **SQL** 하나로 통합하여 튜닝할수 있다.
2. **1**번을 적용하면 **SQL** 도 간단해 진다.
3. **DW** 나 **OLAP** 업무에서는 이런 **SQL** 유형이 대부분을 차지한다.

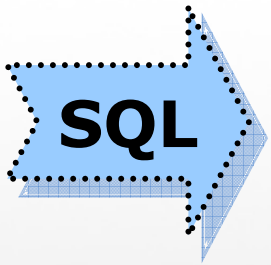
- (주) 오픈메이드 컨설팅
- **오 동 규 수석 컨설턴트**

Check List Chapter 13 -Advanced Group Operations

•ROLLUP : 오른쪽부터 GROUP BY 컬럼을 삭제하여 집합을 계속 생성하라.

```
SELECT A, B, SUM(C)
  FROM t
GROUP BY ROLLUP(A, B)
```


```
GROUP BY A, B
UNION ALL
GROUP BY A
UNION ALL
GROUP BY NULL
```




```
SELECT A, B, SUM(C)
  FROM t
GROUP BY A, B
UNION ALL
SELECT A, NULL, SUM(C)
  FROM t
GROUP BY A
UNION ALL
SELECT NULL, NULL, SUM(C)
 FROM t
GROUP BY NULL
```

•Group by 컬럼갯수 + 1 만큼 집합이 강제로 생성된다.

Check List Chapter 13 -Advanced Group Operations

•CUBE : 나올 수 있는 모든 경우의 GROUP BY 절을 생성 하라.

```
SELECT A, B, SUM(C)
  FROM t
GROUP BY CUBE(A, B)
```

개념

```
GROUP BY NULL
UNION ALL
GROUP BY B
UNION ALL
GROUP BY A
UNION ALL
GROUP BY A, B
```

SQL

```
SELECT NULL, NULL, SUM(C)
 FROM t
GROUP BY NULL
UNION ALL
SELECT NULL, B, SUM(C)
  FROM t
GROUP BY B
UNION ALL
SELECT A, NULL, SUM(C)
  FROM t
GROUP BY A
UNION ALL
SELECT A, B, SUM(C)
  FROM t
GROUP BY A, B
```

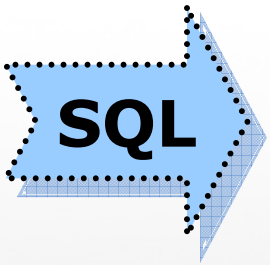
•2 의 N 승(Group by 컬럼갯수) 만큼 집합이 강제로 생성된다.

Check List Chapter 13 -Advanced Group Operations

•GROUPING SETS: comma(,) 를 UNION 으로 바꿔서 GROUP BY 절을 계속 생성 하라.

```
SELECT A, B, SUM(C)
  FROM t
GROUP BY
GROUPING SETS(A, B)
```


```
GROUP BY A
UNION ALL
GROUP BY B
```




```
SELECT A, NULL, SUM(C)
  FROM t
GROUP BY A
UNION ALL
SELECT NULL, B, SUM(C)
  FROM t
GROUP BY B
```

•Comma 로 분리된 Group by 절의 컬럼수 만큼 집합이 생성된다.

Check List Chapter 13 -Advanced Group Operations

•GROUPING: group by 가 생략 되면 1로 표시하라

```
SELECT o.year,
       TO_CHAR(TO_DATE(o.month, 'MM'), 'Month') month,
       r.name region,
       SUM(o.tot_sales),
       GROUPING(o.year) y, GROUPING(o.month) m, GROUPING(r.name) r
 FROM all_orders o JOIN region r ON r.region_id = o.region_id
WHERE o.month BETWEEN 1 AND 3
GROUP BY ROLLUP (o.year, o.month, r.name);
```

YEAR	MONTH	REGION	SUM(O.TOT_SALES)	Y	M	R
2000	January	New England	1018430	0	0	0
2000	January	Mid-Atlantic	1221394	0	0	0
2000	January	Southeast US	758042	0	0	0
2000	January		2997866	0	0	1
2000	February	New England	1231492	0	0	0
2000	February	Mid-Atlantic	857352	0	0	0
2000	February	Southeast US	1236846	0	0	0
2000	February		3325690	0	0	1
2000	March	New England	1132966	0	0	0
2000	March	Mid-Atlantic	1274062	0	0	0
2000	March	Southeast US	1311986	0	0	0
2000	March		3719014	0	0	1
2000			10042570	0	1	1

… 이하생략

Check List Chapter 13 -Advanced Group Operations

•GROUPING: group by 가 생략 되면 1로 표시하라

```
SELECT o.year,
          TO_CHAR(TO_DATE(o.month, 'MM'), 'Month') month,
           r.name region,
           SUM(o.tot_sales),
           GROUPING(o.year) y, GROUPING(o.month) m, GROUPING(r.name) r
 FROM all_orders o JOIN region r ON r.region_id = o.region_id
WHERE o.month BETWEEN 1 AND 3
GROUP BY ROLLUP (o.year, o.month, r.name);
```

YEAR	MONTH	REGION	SUM(O.TOT_SALES)	Y	M	R
2000	February	New England	1231492	0	0	0
2000	February	Mid-Atlantic	857352	0	0	0
2000	February	Southeast US	1236846	0	0	0
2000	February		3325690	0	0	1
2000	March	New England	1132966	0	0	0
2000	March	Mid-Atlantic	1274062	0	0	0
2000	March	Southeast US	1311986	0	0	0
2000	March		3719014	0	0	1
2000			10042570	0	1	1

… 이하생략

Check List Chapter 13 -Advanced Group Operations

•GROUPING_ID: group by 된 모든컬럼을 grouping 함수를 이용하여 2진수로 나타낸 다음 10진수로 다시 계산하라.

```
SELECT o.year,
          TO_CHAR(TO_DATE(o.month, 'MM'), 'Month') month,
           r.name region,
           SUM(o.tot_sales) total,
           GROUPING(o.year) y, GROUPING(o.month) m, GROUPING(r.name) r,
           GROUPING_ID (o.year, o.month, r.name) gid
 FROM all_orders o JOIN region r ON r.region_id = o.region_id
WHERE o.month BETWEEN 1 AND 3
GROUP BY CUBE (o.year, o.month, r.name);
```

YEAR	MONTH	REGION	TOTAL	Y	M	R	GID
2001	March		1859507	0	0	1	1
2001		Southeast US	1653437	0	1	0	2
2001			5021285	0	1	1	3
	March	Southeast US	1967979	1	0	0	4
	March		5578521	1	0	1	5
		Southeast US	4960311	1	1	0	6
			15063855	1	1	1	7

…이후생략

2진법 계산

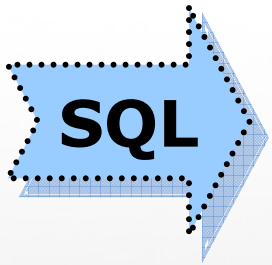
→ 0 + 0 + 1
→ 0 + 2 + 0
→ 0 + 2 + 1
→ 4 + 0 + 0
→ 4 + 1 + 0
→ 4 + 2 + 0
→ 4 + 2 + 1

Check List Chapter 13 -Advanced Group Operations

•Partial GROUPING_SETS:Group by 절에 있는 컬럼은 고정시키고 GROUPING SETS를 적용시켜라.

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C,
GROUPING SETS(A, B)
```


```
GROUP BY C,A
UNION ALL
GROUP BY C,B
```




```
SELECT A, NULL, C, SUM(D)
  FROM t
GROUP BY C,A
UNION ALL
SELECT NULL, B, C, SUM(D)
  FROM t
GROUP BY C,B
```

•**GROUP BY C 는 고정이다.**

Check List Chapter 13 -Advanced Group Operations

•Partial ROLLUP:Group by 절에 있는 컬럼은 고정시키고 각각의 ROLLUP을 적용시켜라.

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C,ROLLUP(A, B)
```

개념

```
GROUP BY C, A, B
UNION ALL
GROUP BY C, A
UNION ALL
GROUP BY C,NULL
```

SQL

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C, A, B
UNION ALL
SELECT A, NULL, C, SUM(D)
  FROM t
GROUP BY C, A
UNION ALL
SELECT NULL, NULL, C,SUM(D)
 FROM t
GROUP BY C, NULL
```

•GROUP BY C 는 고정이다.

Check List Chapter 13 -Advanced Group Operations

•DUPLICATE Partial ROLLUP 시 GROUP_ID 함수 적용:중복데이터를 걸러낸다.

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C,ROLLUP(C, B)
```


```
GROUP BY C, C, B
UNION ALL
GROUP BY C, C
UNION ALL
GROUP BY C
```

GROUP BY C,C 와 GROUP BY C 는 데이터가 같으므로 중복이다.
GROUP BY C → 0
GROUP BY C,C → 1
GROUP BY C,C,C → 2

Check List Chapter 13 -Advanced Group Operations

•Partial CUBE : Group by 절에 있는 컬럼은 고정시키고 각각의 cube를 적용시켜라.

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C, CUBE(A, B)
```

SQL

```
SELECT NULL, NULL, C, SUM(D)
 FROM t
GROUP BY C, NULL
UNION ALL
SELECT NULL, B, C, SUM(D)
  FROM t
GROUP BY C, B
UNION ALL
SELECT A, NULL, C, SUM(D)
  FROM t
GROUP BY C, A
UNION ALL
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C, A, B
```

개념

```
GROUP BY C, NULL
UNION ALL
GROUP BY C, B
UNION ALL
GROUP BY C, A
UNION ALL
GROUP BY C, A, B
```

•GROUP BY C 는 고정이다.

Check List Chapter 13 -Advanced Group Operations

•Composite Grouping :괄호로 묶여진 컬럼들은 하나의 컬럼으로 생각해라. (ROLLUP 이나 GROUPING SETS 도 동일함)

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY CUBE((A, B), C)
```

SQL

```
SELECT NULL, NULL,NULL, C, SUM(D)
 FROM t
GROUP BY NULL
UNION ALL
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY (A,B), C
UNION ALL
SELECT A, B, NULL, SUM(D)
  FROM t
GROUP BY A,B
UNION ALL
SELECT NULL, NULL, C, SUM(D)
  FROM t
GROUP BY C
```

개념

```
GROUP BY NULL
UNION ALL
GROUP BY (A,B), C
UNION ALL
GROUP BY (A,B)
UNION ALL
GROUP BY C
```

•(A,B) 를 하나의 컬럼으로 보아야 한다.

Check List Chapter 13 -Advanced Group Operations

•Concatenated Grouping : ROLLUP(C) 를 먼저 전개하면 NULL 인 경우와 NULL 이 아닌 경우로 나뉜다. 각각 ROLLUP을 진행하라.

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY ROLLUP(A,B), ROLLUP ( C)
```

C 를 NULL 과 NULL 이 아닌 것으로 나눈 이유는 ROLLUP(C) 를 전개하면
GROUP BY C → NULL 아님
UNION ALL
GROUP BY NULL → NULL 임 이 되기 때문이다.

개념

```
GROUP BY A, B
UNION ALL
GROUP BY A
UNION ALL
GROUP BY NULL
UNION ALL
```

C 가 NULL 인경우

```
GROUP BY A,B,C
UNION ALL
GROUP BY A,C
UNION ALL
GROUP BY C
```

C 가 NOT NULL 인경우

Check List Chapter 13 -Advanced Group Operations

•Concatenated Grouping : GROUPING SETS

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY GROUPING SETS(A,B),
         GROUPING SETS(C)
```


```
GROUP BY A, C
UNION ALL
GROUP BY B, C
```

•드 모르강 법칙 적용

Check List Chapter 13 -Advanced Group Operations

•Concatenated Grouping : GROUPING SETS

```
SELECT A, B, C, D, SUM(E)
  FROM t
GROUP BY GROUPING SETS(A,B),
         GROUPING SETS(C,D)
```


```
GROUP BY (A, C)
UNION ALL
GROUP BY (A, D)
UNION ALL
GROUP BY (B, C)
UNION ALL
GROUP BY (B, D)
```

- 드 모르강 법칙 적용

Check List Chapter 13 -Advanced Group Operations

•ROLLUP and CUBE 가 GROUPING SETS 의 인자로 되는경우: ROLLUP 과 CUBE 를 UNION ALL 로 분리하라

```
SELECT A, B, C, SUM(D)
  FROM t
GROUP BY C,
GROUPING SETS(ROLLUP(A), ROLLUP(B))
```


```
GROUP BY ROLLUP(A)
UNION ALL
GROUP BY ROLLUP(B)
```

•GROUPING SETS 내에 아무리 많은 ROLLUP 과 CUBE 가 있어도 UNION ALL 로 분리하면 된다.

Check List Chapter 13

Advanced Group Operations-정리

- ROLLUP
- CUBE
- GROUPING SETS
- The GROUPING_ID and GROUP_ID Functions
- Complex Grouping

1. Partial Grouping :
   Group By A , ROLLUP(B,C)
2. Grouping on Composite Columns :
   GROUP BY ROLLUP ((o.year, o.month),r.name)
3. Concatenated Groupings :
   GROUP BY ROLLUP (o.year, o.month), ROLLUP(r.name);
4. ROLLUP and CUBE as arguments to GROUPING SETS :
   GROUP BY GROUPING SETS (ROLLUP (o.year, o.month), ROLLUP(r.name))