KH정보교육원

GROUP BY & HAVING

GROUP BY & HAVING

KH정보교육원

ORDER BY절

SELECT한 컬럼에 대해 정렬을 할 때 작성하는 구문이다.

SELECT 구문의 가장 마지막에 작성하며, 실행순서도 가장 마지막에 수행된다.

표현식

ORDER BY 컬럼명 | 별칭 | 컬럼순번 정렬방식 [NULLS FIRST | LAST]

GROUP BY & HAVING

KH정보교육원

GROUP BY절

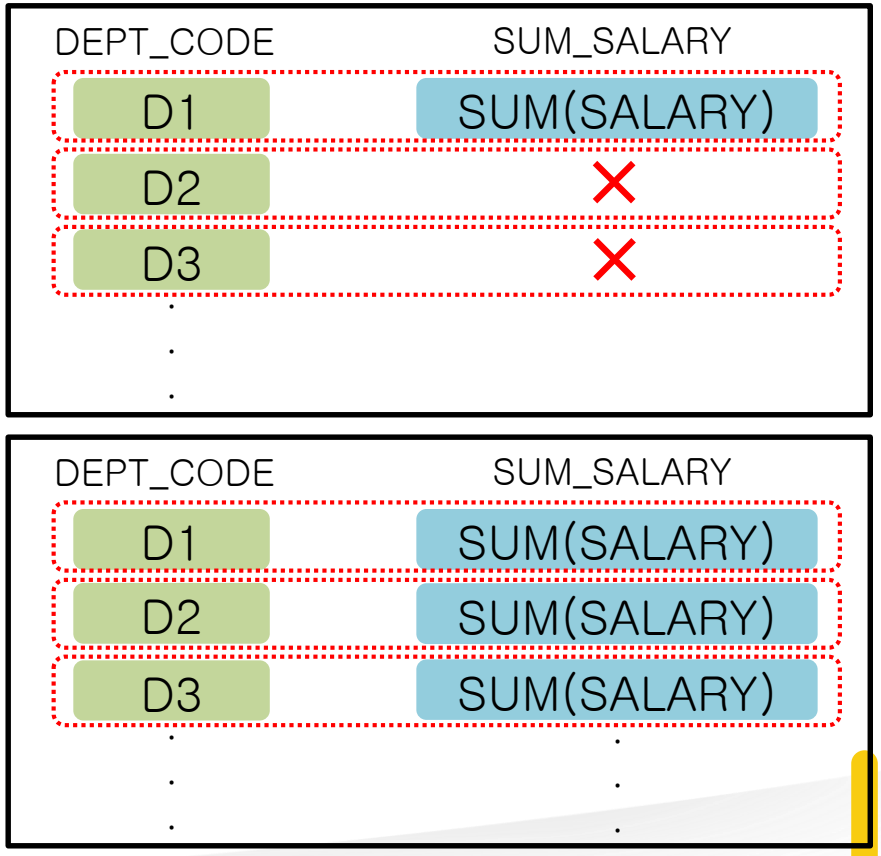
그룹함수는 단 한 개의 결과값만 산출하기 때문에, 그룹함수를 이용하여 여러 개의 결과값을 산출하기 위해서는 그룹함수가 적용될 그룹의 기준을 GROUP BY절에 기술하여 사용해야 한다.

```
SELECT DEPT_CODE,
       SUM(SALARY)
FROM EMPLOYEE;
```

** 에러 발생

해결

```
SELECT DEPT_CODE,
       SUM(SALARY)
FROM EMPLOYEE
GROUP BY DEPT_CODE;
```



GROUP BY & HAVING

KH정보교육원

GROUP BY절

[EMPLOYEE 테이블에서 부서코드, 그룹별 급여의 합계, 그룹별 급여의 평균(정수처리), 인원수를 조회하고, 부서코드 순으로 정렬하세요]

```
SELECT DEPT_CODE,
       SUM(SALARY) AS 합계,
       AVG(SALARY) AS 평균,
       COUNT(*) AS 인원수
FROM EMPLOYEE
GROUP BY DEPT_CODE
ORDER BY DEPT_CODE ASC;
```

	DEPT_CODE	합계	평균	인원수
1	D1	7820000	2606666	3
2	D2	6520000	2173333	3
3	D5	15760000	2626666	6
4	D6	10100000	3366666	3
5	D8	6986240	2328746	3
6	D9	17700000	5900000	3
7	(null)	5210000	2605000	2

[EMPLOYEE 테이블에서 부서코드, 보너스를 지급받는 사원 수를 조회하고 부서코드 순으로 정렬하세요]

```
SELECT DEPT_CODE,
       COUNT(BONUS)
FROM EMPLOYEE
WHERE BONUS IS NOT NULL
GROUP BY DEPT_CODE
ORDER BY DEPT_CODE ASC;
```

	DEPT_CODE	COUNT(BONUS)
1	D1	2
2	D5	2
3	D6	1
4	D8	2
5	D9	1
6	(null)	1

GROUP BY & HAVING

KH정보교육원

GROUP BY절

[EMPLOYEE 테이블에서 EMP_NO의 8번째 자리가 1이면 '남', 2이면 '여'로 결과를 조회하고, 성별별 급여의 평균(정수처리), 급여의 합계, 인원수를 조회한 뒤, 인원수로 내림차순 정렬하세요]

```
SELECT DECODE(SUBSTR(EMP_NO, 8, 1), 1, '남', 2, '여') AS 성별,
       FLOOR(AVG(SALARY)) AS 평균,
       SUM(SALARY) AS 합계
       COUNT(*) AS 인원수
FROM EMPLOYEE
GROUP BY DECODE(SUBSTR(EMP_NO, 8, 1), 1, '남', 2, '여')
ORDER BY COUNT(*) DESC;
```

	성별	평균	합계	인원수
1	남	3317333	49760000	15
2	여	2542030	20336240	8

GROUP BY & HAVING

KH정보교육원

HAVING절

그룹함수로 값을 구해올 그룹에 대해 조건을 설정할 때는 HAVING절에 기술한다. (WHERE절은 SELECT에 대한 조건)

```
SELECT DEPT_CODE,
       FLOOR(AVG(SALARY)) 평균
FROM EMPLOYEE
WHERE SALARY > 3000000
GROUP BY DEPT_CODE
ORDER BY 1;
```

[급여 3000000원 이상인 직원의 그룹별 평균]

	DEPT_CODE	평균
1	D1	3660000
2	D5	3630000
3	D6	3650000
4	D9	5900000

```
SELECT DEPT_CODE,
       FLOOR(AVG(SALARY)) 평균
FROM EMPLOYEE
GROUP BY DEPT_CODE
HAVING FLOOR(AVG(SALARY)) > 3000000
ORDER BY 1;
```

[급여 평균이 3000000원 이상인 그룹에 대한 평균]

	DEPT_CODE	평균
1	D6	3366666
2	D9	5900000

GROUP BY & HAVING

KH정보교육원

ROLLUP과 CUBE

단일함일 경우 같다

그룹별 산출한 결과값의 집계를 계산하는 함수이다.

```
SELECT JOB_CODE, SUM(SALARY)
FROM EMPLOYEE
GROUP BY ROLLUP(JOB_CODE)
ORDER BY 1;
```

```
SELECT JOB_CODE, SUM(SALARY)
FROM EMPLOYEE
GROUP BY CUBE(JOB_CODE)
ORDER BY 1;
```

	JOB_CODE	SUM(SALARY)
1	J1	8000000
2	J2	9700000
3	J3	10800000
4	J4	9320000
5	J5	8460000
6	J6	15746240
7	J7	8070000
8	(null)	70096240

진짜 NULL인지

ROLLUP의 NULL 인지 구분

필요

누계

GROUP BY & HAVING

KH정보교육원

ROLLUP

인자로 전달받은 그룹 중에 가장 먼저 지정한 그룹별 합계와 총 합계를 구한다.

```
SELECT DEPT_CODE,
       JOB_CODE,
       SUM(SALARY)
FROM EMPLOYEE
GROUP BY ROLLUP(DEPT_CODE, JOB_CODE)
ORDER BY 1;
```

	DEPT_CODE	JOB_CODE	SUM(SALARY)
1	D1	J6	6440000
2	D1	J7	1380000
3	D1	(null)	7820000
4	D2	J4	6520000
5	D2	(null)	6520000
6	D5	J3	3500000
7	D5	J5	8460000
8	D5	J7	3800000
9	D5	(null)	15760000
10	D6	J3	7300000
11	D6	J4	2800000
12	D6	(null)	10100000
13	D8	J6	6986240
14	D8	(null)	6986240
15	D9	J1	8000000
16	D9	J2	9700000
17	D9	(null)	17700000
18	(null)	J6	2320000
19	(null)	J7	2890000
20	(null)	(null)	5210000
21	(null)	(null)	70096240

GROUP BY & HAVING

KH정보교육원

CUBE

그룹으로 지정된 모든 그룹에 대한 합계와 총 합계를 구한다.

```
SELECT DEPT_CODE,
       JOB_CODE,
       SUM(SALARY)
FROM EMPLOYEE
GROUP BY CUBE(DEPT_CODE, JOB_CODE)
ORDER BY 1;
```

DEPT_CODE (D1,D2,D5,D6,D8,D9) 6개

JOB_CODE (J1,J2,J3,J4,J5,J6,J7) 7개

소계 : 13개

총계 : 1개

	DEPT_CODE	JOB_CODE	SUM(SALARY)
1	D1	J6	6440000
2	D1	J7	1380000
3	D1	(null)	7820000
4	D2	J4	6520000
5	D2	(null)	6520000
6	D5	J3	3500000
7	D5	J5	8460000
8	D5	J7	3800000
9	D5	(null)	15760000
10	D6	J3	7300000
11	D6	J4	2800000
12	D6	(null)	10100000
13	D8	J6	6986240
14	D8	(null)	6986240
15	D9	J1	8000000
16	D9	J2	9700000
17	D9	(null)	17700000
18	(null)	J1	8000000
19	(null)	J2	9700000
20	(null)	J3	10800000
21	(null)	J4	9320000
22	(null)	J5	8460000
23	(null)	J6	2320000
24	(null)	J6	15746240
25	(null)	J7	2890000
26	(null)	J7	8070000
27	(null)	(null)	5210000
28	(null)	(null)	70096240

정렬 시 상황 (null 정렬)

GROUP BY & HAVING

KH정보교육원

ROLLUP과 CUBE

```
SELECT DEPT_CODE,
       JOB_CODE,
       SUM(SALARY)
FROM EMPLOYEE
GROUP BY ROLLUP(DEPT_CODE, JOB_CODE)

UNION

SELECT '',
       JOB_CODE,
       SUM(SALARY)
FROM EMPLOYEE
GROUP BY ROLLUP(DEPT_CODE, JOB_CODE)
ORDER BY 1;
```

아무 모든 행에 고정값

기준

```
SELECT DEPT_CODE,
       JOB_CODE,
       SUM(SALARY)
FROM EMPLOYEE
GROUP BY CUBE(DEPT_CODE, JOB_CODE)
ORDER BY 1;
```

기준 = 2개

GROUP BY & HAVING

KH정보교육원

GROUPING

ROLLUP이나 CUBE에 의한 집계 산출물이 인자로 전달받은 컬럼 집합의 산출물이면 0을 반환하고, 아니면 1을 반환하는 함수이다.

```
SELECT DEPT_CODE,
       JOB_CODE,
       SUM(SALARY),
       CASE WHEN GROUPING(DEPT_CODE) = 0
                 AND GROUPING(JOB_CODE) = 1
            TEHN '부서별합계'
            WHEN GROUPING(DEPT_CODE) = 1
                 AND GROUPING(JOB_CODE) = 0
            THEN '직급별합계'
            WHEN GROUPING(DEPT_CODE) = 1
                 AND GROUPING(JOB_CODE) = 1
            THEN '총합계'
            ELSE '그룹별합계'
       END AS '구분'
FROM EMPLOYEE
GROUP BY CUBE(DEPT_CODE, JOB_CODE)
ORDER BY 1;
```

예제 엄청 중요 !!!

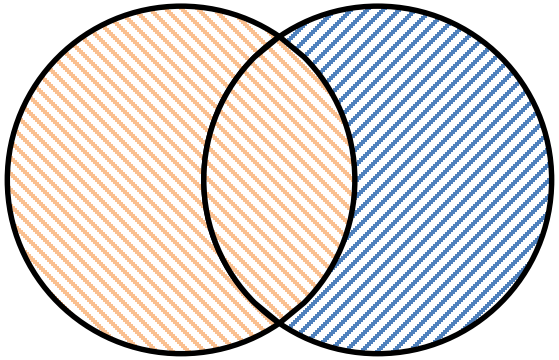
	DEPT_CODE	JOB_CODE	합계	구분
1	D1	J6	6440000	그룹별합계
2	D1	J7	1380000	그룹별합계
3	D2	J4	6520000	그룹별합계
4	D5	J3	3500000	그룹별합계
5	D5	J5	8460000	그룹별합계
6	D5	J7	3800000	그룹별합계
7	D6	J3	7300000	그룹별합계
8	D6	J4	2800000	그룹별합계
9	D8	J6	6986240	그룹별합계
10	D9	J1	8000000	그룹별합계
11	D9	J2	9700000	그룹별합계
12	(null)	J6	2320000	그룹별합계
13	(null)	J7	2890000	그룹별합계
14	D1	(null)	7820000	부서합계
15	D2	(null)	6520000	부서합계
16	D5	(null)	15760000	부서합계
17	D6	(null)	10100000	부서합계
18	D8	(null)	6986240	부서합계
19	D9	(null)	17700000	부서합계
20	(null)	(null)	5210000	부서합계
21	(null)	J1	8000000	직급합계
22	(null)	J2	9700000	직급합계
23	(null)	J3	10800000	직급합계
24	(null)	J4	9320000	직급합계
25	(null)	J5	8460000	직급합계
26	(null)	J6	15746240	직급합계
27	(null)	J7	8070000	직급합계
28	(null)	(null)	70096240	총합계

GROUP BY & HAVING

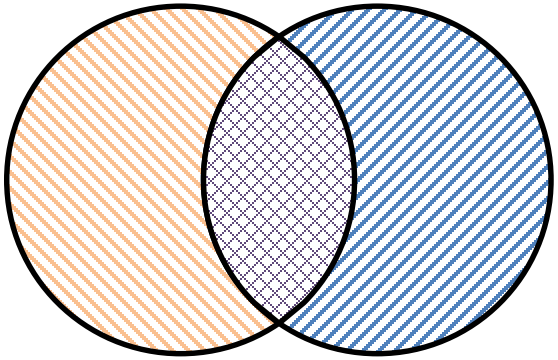
KH정보교육원

SET OPERATION

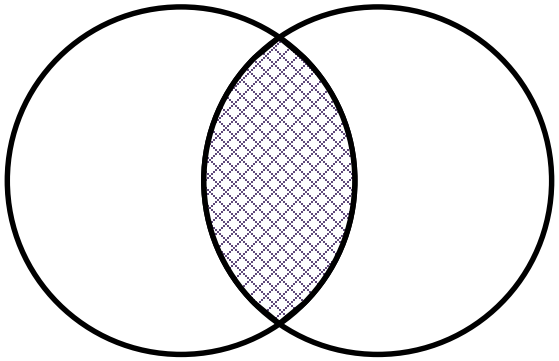
여러 개의 SELECT 결과물을 하나의 쿼리로 만드는 연산자이다.



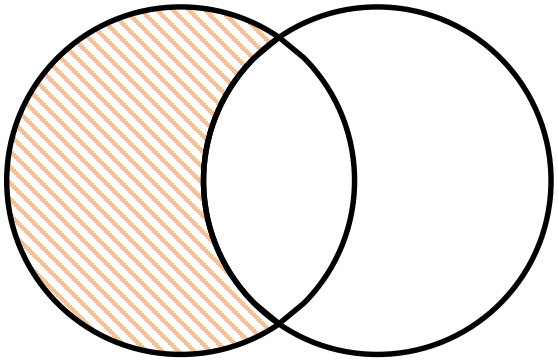
UNION



UNION ALL



INTERSECT



MINUS

GROUP BY & HAVING

KH정보교육원

UNION

UNION과 UNION ALL은 여러 개의 쿼리 결과를 하나로 합치는 연산자이다.
그 중 UNION은 중복된 영역을 제외하여 하나로 합치는 연산자이다.

```
SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE DEPT_CODE = 'D5'

UNOIN

SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE SALARY > 3000000;
```

차이

	EMP_ID	EMP_NAME	DEPT_CODE	SALARY
1	200	선동일	D9	8000000
2	201	송종기	D9	6000000
3	202	노옹철	D9	3700000
4	204	유재식	D6	3400000
5	205	정중하	D6	3900000
6	206	박나라	D5	1800000
7	207	하이유	D5	2200000
8	208	김해술	D5	2500000
9	209	심봉선	D5	3500000
10	210	윤은해	D5	2000000
11	215	대북혼	D5	3760000
12	217	전지연	D1	3660000

GROUP BY & HAVING

KH정보교육원

UNION ALL

UNION ALL은 UNION과 같은 여러 쿼리 결과물에 대한 합집합을 의미하며,
UNION과의 차이점은 중복된 영역을 모두 포함시키는 연산자이다.

```
SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE DEPT_CODE = 'D5'

UNOIN ALL

SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE SALARY > 3000000;
```

	EMP_ID	EMP_NAME	DEPT_CODE	SALARY
1	206	박나라	D5	1800000
2	207	하이유	D5	2200000
3	208	김해술	D5	2500000
4	209	심봉선	D5	3500000
5	210	윤은해	D5	2000000
6	215	대북혼	D5	3760000
7	200	선동일	D9	8000000
8	201	송종기	D9	6000000
9	202	노옹철	D9	3700000
10	204	유재식	D6	3400000
11	205	정중하	D6	3900000
12	209	심봉선	D5	3500000
13	215	대북혼	D5	3760000
14	217	전지연	D1	3660000

GROUP BY & HAVING

KH정보교육원

INTERSECT

여러 개의 SELECT 결과에서 공통된 부분만 결과로 추출한다. 즉, 수행 결과에 대한 교집합이라고 볼 수 있다.

```
SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE DEPT_CODE = 'D5'

INTERSECT

SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE SALARY > 3000000;
```

	EMP_ID	EMP_NAME	DEPT_CODE	SALARY
1	209	심봉선	D5	3500000
2	215	대북혼	D5	3760000

GROUP BY & HAVING

KH정보교육원

MINUS

선행 SELECT 결과에서 다음 SELECT 결과와 겹치는 부분을 제외한 나머지 부분만 추출한다. 즉, 두 쿼리 결과물의 차집합이라고 볼 수 있다.

개준

```
SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE DEPT_CODE = 'D5'

MINUS

SELECT EMP_ID,
       EMP_NAME,
       DEPT_CODE,
       SALARY
FROM EMPLOYEE
WHERE SALARY > 3000000;
```

	EMP_ID	EMP_NAME	DEPT_CODE	SALARY
1	206	박나라	D5	1800000
2	207	하이유	D5	2200000
3	208	김해술	D5	2500000
4	210	윤은해	D5	2000000

GROUP BY & HAVING

KH정보교육원

GROUPING SETS

그룹별로 처리된 여러 개의 SELECT문을 하나로 합친 결과를 원할 때 사용한다. SET OPERATOR(집합연산자) 사용한 결과와 동일한 결과를 얻을 수 있다.

```
SELECT DEPT_CODE,
       JOB_CODE,
       MANAGER_ID,
       FLOOR(AVG(SALARY))
FROM EMPLOYEE
GROUP BY GROUPING SETS(
       (DEPT_CODE, JOB_CODE, MANAGER_ID),
       (DEPT_CODE, MANAGER_ID),
       (JOB_CODE, MANAGER_ID)
);
```

SQL | 인출된 모든 행: 53(0.016초)

	DEPT_CODE	JOB_CODE	MANAGER_ID	FLOOR(AVG(SALARY))
1	D5	J5	207	2500000
2	D6	J4	204	2800000
3	D5	J3	207	3500000
4	D9	J2	200	6000000
5	D6	J3	200	3400000
6	D8	J6	211	2550000
7	(null)	J7	(null)	2890000
8	D8	J6	100	2436240
9	(null)	J6	(null)	2320000
10	D1	J6	214	3220000
11	D6	J3	204	3900000
12	D5	J7	207	1900000
13	D5	J5	200	2200000
14	D1	J7	200	1380000
15	D2	J4	(null)	2173333

.
.
.

51	(null)	J5	207	2500000
52	(null)	J5	(null)	3760000
53	(null)	J6	214	3220000