Chapter 4

그룹 함수

GROUP BY

HAVING

ROLLUP, CUBE, GROUPING

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4800
60	4800
60	4200
100	12008
100	9000
100	8200
100	7700
100	7800
...	...

EMPLOYEES 테이블에서 SALARY 최대 값

MAX(SALARY)
24000

함 수	설 명
AVG([DISTINCT|ALL] n)	Null 값을 무시한 n의 평균을 출력합니다.
SUM([DISTINCT|ALL] n)	Null 값을 무시한 n의 합계를 출력합니다.
MIN([DISTINCT|ALL] expr)	Null 값을 무시한 expr의 최솟값을 출력합니다.
MAX([DISTINCT|ALL] expr)	Null 값을 무시한 expr의 최댓값을 출력합니다.
COUNT({*|[DISTINCT|ALL] expr})	행의 수, expr은 Null 값을 제외하고 계산합니다. *를 사용하여 중복되거나 Null인 행들을 포함하여 모든 행을 계산합니다.

```
SQL> SELECT  AVG(salary), MAX(salary), MIN(salary), SUM(salary)
  2  FROM    employees
  3  WHERE   job_id LIKE 'SA%';
```

	AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
1	8900	14000	6100	311500

```
SQL> SELECT   MIN(hire_date), MAX(hire_date)
  2  FROM     employees;
```

	MIN(HIRE_DATE)	MAX(HIRE_DATE)
1	01/01/13	08/04/21

```
SQL> SELECT MIN(first_name), MAX(last_name)
  2  FROM   employees;
```

	MIN(FIRST_NAME)	MAX(LAST_NAME)
1	Adam	Zlotkey

```
SQL> SELECT MAX(salary)
  2  FROM employees;
```

	MAX(SALARY)
1	24000

COUNT 함수는 두 가지 형식이 있습니다.
COUNT(*)
COUNT(expr)

많이 사용됩니다. 반드시 알아두세요

COUNT(*) 는 중복되는 행과 null 값을 포함하는 행을 포함하여 테이블 행의 수를 리턴합니다.

COUNT(expr) 는 expr 에 의해 인식된 열에서 Null 이 아닌 행의 수를 리턴합니다.

다음 구문은 모든 사원의 수를 출력합니다.

```
SQL> SELECT COUNT(*) FROM employees;
```

	COUNT(*)
1	107

다음 구문은 커미션을 받는 사원의 수를 출력합니다.

```
SQL> SELECT COUNT(commission_pct)
  2  FROM employees;
```

	COUNT(COMMISSION_PCT)
1	35

Chapter 4

그룹 함수

GROUP BY

HAVING

ROLLUP, CUBE, GROUPING

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4800
60	4800
60	4200
100	12008
100	9000
100	8200
100	7700
100	7800
...	...

EMPLOYEES 테이블에서 salary 평균 값

DEPARTMENT_ID	AVG(SALARY)
90	19333.3333
60	5760
100	8601.3333
...	...

```
SELECT column, group_function(column)
FROM table
[WHERE    condition(s)]
[GROUP BY group_by_expression]
[ORDER BY {column|expr, ...} [[ASC]|DESC]];
```

GROUP BY 절은 where절 다음 order절 사이에 쓰입니다

```
SQL> SELECT    department_id,  AVG(salary)
  2  FROM      employees
  3  GROUP BY  department_id;
```

	DEPARTMENT_ID	AVG(SALARY)
1	100	8601.33333333333333333333333333333333333
2	30	4150
3	(null)	7000
4	90	19333.33333333333333333333333333333333
5	20	9500
6	70	10000
7	110	10154
8	50	3475.55555555555555555555555555555555556
9	80	8955.88235294117647058823529411764705824
10	40	6500
11	60	5760
12	10	4400

EMPLOYEES

DEPARTMENT_ID	JOB_ID	SALARY
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
30	PU_MAN	11000
30	PU_CLERK	3100
30	PU_CLERK	2900
30	PU_CLERK	2800
30	PU_CLERK	2600
30	PU_CLERK	2500
40	HR_REP	6500
50	ST_MAN	8000
50	ST_MAN	8200
50	ST_MAN	7900
50	ST_MAN	6500
50	ST_MAN	5800
...	...	...

EMPLOYEES 테이블에서 부서별, 직업별 SALARY 합

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
30	PU_CLERK	13900
30	PU_MAN	11000
40	HR_REP	6500
50	ST_MAN	36400
...	...	...

다음 구문은 각 부서내의 직무별로 급여 합계를 출력하는 리포트를 보여 줍니다. EMPLOYEES 테이블은 먼저 부서번호(DEPARTMENT_ID)로 그룹화한 다음에 직무(JOB_ID)로 그룹화합니다.

```
SQL> SELECT    department_id, job_id, SUM(salary)
  2  FROM      employees
  3  GROUP BY  department_id, job_id;
```

	DEPARTMENT_ID	JOB_ID	SUM(SALARY)
1	110	AC_ACCOUNT	8300
2	90	AD_VP	34000
3	50	ST_CLERK	55700
4	80	SA_REP	243500
5	50	ST_MAN	36400
6	80	SA_MAN	61000
7	110	AC_MGR	12008
8	90	AD_PRES	24000
9	60	IT_PROG	28800
10	100	FI_MGR	12008
11	30	PU_CLERK	13900
12	50	SH_CLERK	64300
13	20	MK_MAN	13000
14	100	FI_ACCOUNT	39600
15	(null)	SA_REP	7000
16	70	PR_REP	10000
17	30	PU_MAN	11000
18	10	AD_ASST	4400
19	20	MK_REP	6000
20	40	HR_REP	6500

다음은 GROUP BY 절을 포함하는 위의 SELECT 문장이 계산되는 단계를 보여 줍니다.

- SELECT 절은 검색할 열을 명시합니다.
  . EMPLOYEES 테이블의 부서번호 열, 직무 열
  . GROUP BY 절에 명시된 그룹의 모든 급여를 더합니다. SUM 함수는 각각의 부서번호 그룹 내의 모든 직무에 대한 급여 열에 적용됩니다.
- FROM 절에 데이터베이스가 액세스 할 테이블을 명시합니다.
- GROUP BY절은 행을 그룹화하는 방법을 명시합니다.
  . 먼저, 부서번호로 행을 그룹화 합니다.
  . 두 번째로 부서번호 그룹 내에서 직무로 행을 그룹화 합니다.

같은 SELECT 문장에 개별적인 열(DEPARTMENT_ID)과 그룹 함수(COUNT)를 혼합해서 사용할 때, 개별적인 열(이 경우에는 DEPARTMENT_ID)을 명시하는 GROUP BY 절을 포함해야 합니다. GROUP BY 절이 없으면 "ORA-00937: not a single-group group function" 에러가 발생합니다.

```
SQL> SELECT  department_id, COUNT(first_name)
  2  FROM    employees;
```

```
ORA-00937: not a single-group group function
00937, 00000 -  "not a single-group group function"
*Cause:
*Action:
1행, 9열에서 오류 발생
```

WHERE 절에 그룹함수를 사용하여 제한할 수 없습니다. 다음 구문은 WHERE 절에 그룹 함수를 사용했기 때문에 실행 시 오류가 발생합니다.

```
SQL> SELECT    department_id, AVG(salary)
  2  FROM      employees
  3  WHERE     AVG(salary) > 2000
  4  GROUP BY  department_id;
```

```
ORA-00934: group function is not allowed here
00934, 00000 -  "group function is not allowed here"
*Cause:
*Action:
3행, 11열에서 오류 발생
```

Chapter 4

그룹 함수
GROUP BY
HAVING
ROLLUP, CUBE, GROUPING

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4800
60	4800
60	4200
100	12008
100	9000
100	8200
100	7700
100	7800
...	...

19333.3333

5760

8601.3333

평균 급여가 6000 이상인 부서들의 정보만 출력함

DEPARTMENT_ID	AVG(SALARY)
90	19333.3333
100	8601.3333
...	...

GROUP BY절의 조건 HAVING

SELECT *column*, *group_function(column)*
FROM *table*
[WHERE *condition(s)*]
[GROUP BY *group_by_expression*]
[HAVING *group_condition*]
[ORDER BY {*column*|*expr* [[ASC]|DESC], ...};

다음 구문은 부서의 급여 평균이 8000을 초과하는 부서의 번호와 급여 평균을 출력합니다.

```
SQL> SELECT   department_id, ROUND(AVG(salary), 2)
  2  FROM     employees
  3  GROUP BY department_id
  4  HAVING   AVG(salary) > 8000;
```

	DEPARTMENT_ID	ROUND(AVG(SALARY),2)
1	100	8601.33
2	90	19333.33
3	20	9500
4	70	10000
5	110	10154
6	80	8955.88

다음 구문은 급여 평균이 8000을 초과하는 각 직무에 대하여 직무와 급여 평균을 출력합니다. 예에서는 Sales 직무를 담당하는 사원은 제외하고 급여 평균으로 결과를 정렬합니다.

```
SQL> SELECT   job_id, AVG(salary) PAYROLL
  2  FROM     employees
  3  WHERE    job_id NOT LIKE 'SA%'
  4  GROUP BY job_id
  5  HAVING   AVG(salary) > 8000
  6  ORDER BY AVG(salary);
```

	JOB_ID	PAYROLL
1	AC_ACCOUNT	8300
2	PR_REP	10000
3	PU_MAN	11000
4	AC_MGR	12008
5	FI_MGR	12008
6	MK_MAN	13000
7	AD_VP	17000
8	AD_PRES	24000

Chapter 4

그룹 함수
GROUP BY
HAVING
ROLLUP, CUBE, GROUPING

ROLLUP구문은 GROUP BY 절과 같이 사용 되며, GROUP BY절에 의해서 그룹 지어진 집합 결과에 대해서 좀 더 상세한 정보를 반환하는 기능을 수행 합니다.

SELECT절에 ROLLUP을 사용함으로써 보통의 SELECT된 데이터와 그 데이터의 총계를 구할 수 있다.

ROLLUP

deptno	JOB	COUNT	SUM
10	A	1	10
10	B	1	20
		2	30
20	A	1	50
		1	50
30	C	4	40
		4	40
계		7	120

```
SQL> SELECT department_id, job_id, ROUND(AVG(salary),2), COUNT(*)
  2  FROM employees
  3  GROUP BY ROLLUP(department_id, job_id)
  4  ORDER BY department_id, job_id;
```

ROLLUP

SQL | 인출된 모든 행: 33(0.003초)

	DEPARTMENT_ID	JOB_ID	ROUND(AVG(SALARY),2)	COUNT(*)
1	10	AD_ASST	4400	1
2	10	(null)	4400	1
3	20	MK_MAN	13000	1
4	20	MK_REP	6000	1
5	20	(null)	9500	2
6	30	PU_CLERK	2780	5
7	30	PU_MAN	11000	1
8	30	(null)	4150	6
9	40	HR_REP	6500	1
10	40	(null)	6500	1
11	50	SH_CLERK	3215	20
12	50	ST_CLERK	2785	20

CUBE는 서브 그룹에 대한 Summary를 추출하는데 사용 된다.

즉 ROLLUP에 의해 나타 내어지는 Item Total값과 Column Total값을 나타 낼 수 있다.

```
SQL> SELECT department_id, job_id, ROUND(AVG(salary),2), COUNT(*)
  2  FROM employees
  3  GROUP BY CUBE(department_id, job_id)
  4  ORDER BY department_id, job_id;
```

SQL | 인출된 모든 행: 52(0.005초)

	DEPARTMENT_ID	JOB_ID	ROUND(AVG(SALARY),2)	COUNT(*)
1	10	AD_ASST	4400	1
2	10	(null)	4400	1
3	20	MK_MAN	13000	1
4	20	MK_REP	6000	1
5	20	(null)	9500	2
6	30	PU_CLERK	2780	5
7	30	PU_MAN	11000	1
8	30	(null)	4150	6
...				
43	(null)	PU_CLERK	2780	5
44	(null)	PU_MAN	11000	1
45	(null)	SA_MAN	12200	5
46	(null)	SA_REP	7000	1
47	(null)	SA_REP	8350	30
48	(null)	SH_CLERK	3215	20
49	(null)	ST_CLERK	2785	20
50	(null)	ST_MAN	7280	5
51	(null)	(null)	7000	1
52	(null)	(null)	6461.83	107

CUBE

GROUPING 함수는 ROLLUP, CUBE에 모두 사용할 수 있다.
GROUPING 함수는 해당 Row가 GROUP BY에 의해서 산출된 Row인 경우에는 0을 반환하고, ROLLUP이나 CUBE에 의해서 산출된 Row인 경우에는 1을 반환하게 된다.

따라서 해당 Row가 결과집합에 의해 산출된 Data 인지, ROLLUP이나 CUBE에 의해서 산출된 Data 인지를 알 수 있도록 지원하는 함수이다.

```
SQL> SELECT
  2    NVL2(department_id, department_id||'',
  3          DECODE(GROUPING(department_id), 1, '소계')) AS 부서,
  4    NVL(job_id, DECODE(GROUPING(job_id), 1, '소계')) AS 직무,
  5    ROUND(AVG(salary),2) AS 평균,
  6    COUNT(*) AS 사원의수
  7  FROM
  8    employees
  9  GROUP BY
 10    CUBE(department_id, job_id)
 11  ORDER BY
 12    부서, 직무;
```

	부서	직무	평균	사원의수
1	10	AD_ASST	4400	1
2	10	소계	4400	1
3	100	FI_ACCOUNT	7920	5
4	100	FI_MGR	12008	1
5	100	소계	8601.33	6
6	110	AC_ACCOUNT	8300	1
7	110	AC_MGR	12008	1
8	110	소계	10154	2
9	20	MK_MAN	13000	1
10	20	MK_REP	6000	1
11	20	소계	9500	2
12	30	PU_CLERK	2780	5
13	30	PU_MAN	11000	1
14	30	소계	4150	6
15	40	HR_REP	6500	1
16	40	소계	6500	1
17	50	SH_CLERK	3215	20
18	50	ST_CLERK	2785	20
19	50	ST_MAN	7280	5
20	50	소계	3475.56	45
21	60	IT_PROG	5760	5
22	60	소계	5760	5
23	70	PR_REP	10000	1
24	70	소계	10000	1
25	80	SA_MAN	12200	5
26	80	SA_REP	8396.55	29
27	80	소계	8955.88	34
28	90	AD_PRES	24000	1
29	90	AD_VP	17000	2
30	90	소계	19333.33	3
31	소계	AC_ACCOUNT	8300	1
32	소계	AC_MGR	12008	1
33	소계	AD_ASST	4400	1
34	소계	AD_PRES	24000	1
35	소계	AD_VP	17000	2
36	소계	FI_ACCOUNT	7920	5
37	소계	FI_MGR	12008	1
38	소계	HR_REP	6500	1
39	소계	IT_PROG	5760	5
40	소계	MK_MAN	13000	1
41	소계	MK_REP	6000	1
42	소계	PR_REP	10000	1
43	소계	PU_CLERK	2780	5
44	소계	PU_MAN	11000	1
45	소계	SA_MAN	12200	5
46	소계	SA_REP	8350	30
47	소계	SH_CLERK	3215	20
48	소계	ST_CLERK	2785	20
49	소계	ST_MAN	7280	5
50	소계	소계	6461.83	107
51	(null)	SA_REP	7000	1
52	(null)	소계	7000	1

문제 1.
사원 테이블에서 JOB_ID별 사원 수를 구하세요.
사원 테이블에서 JOB_ID별 월급의 평균을 구하세요. 월급의 평균 순으로 내림차순 정렬하세요

문제 2.
사원 테이블에서 입사 년도 별 사원 수를 구하세요.

문제 3.
급여가 1000 이상인 사원들의 부서별 평균 급여를 출력하세요. 단 부서 평균 급여가 2000이상인 부서만 출력

문제 4.
사원 테이블에서 commission_pct(커미션) 컬럼이 null이 아닌 사람들의
department_id(부서별) salary(월급)의 평균, 합계, count를 구합니다.

조건 1) 월급의 평균은 커미션을 적용시킨 월급입니다.
조건 2) 평균은 소수 2째 자리에서 절삭 하세요.

문제 5.
직업별 월급합, 총합계를 출력하세요

	JOB_ID	SUM(SALARY)
1	AC_ACCOUNT	8300
2	AC_MGR	12008
3	AD_ASST	4400
4	AD_PRES	24000
5	AD_VP	34000
6	FI_ACCOUNT	39600
7	FI_MGR	12008
8	HR_REP	6500
9	IT_PROG	28800
10	MK_MAN	13000
11	MK_REP	6000
12	PR_REP	10000
13	PU_CLERK	13900
14	PU_MAN	11000
15	SA_MAN	61000
16	SA_REP	250500
17	SH_CLERK	64300
18	ST_CLERK	55700
19	ST_MAN	36400
20	합계	691416

문제 6.
부서별, JOB_ID를 그룹핑 하여
토탈, 합계를 출력하세요.
GROUPING() 을 이용하여
소계 합계를 표현하세요

	DEPARTMENT_ID	JOB_ID	TOTAL	SUM
1	10	AD_ASST	1	4400
2	10	소계	1	4400
3	20	MK_REP	1	6000
4	40	소계	1	6500
5	40	HR_REP	1	6500
6	(null)	SA_REP	1	7000
7	(null)	소계	1	7000
8	110	AC_ACCOUNT	1	8300
9	70	소계	1	10000
10	70	PR_REP	1	10000
11	30	PU_MAN	1	11000
12	110	AC_MGR	1	12008
13	100	FI_MGR	1	12008
14	20	MK_MAN	1	13000
15	30	PU_CLERK	5	13900
16	20	소계	2	19000
17	110	소계	2	20308
18	90	AD_PRES	1	24000
19	30	소계	6	24900
20	60	소계	5	28800
21	60	IT_PROG	5	28800
22	90	AD_VP	2	34000
23	50	ST_MAN	5	36400
24	100	FI_ACCOUNT	5	39600
25	100	소계	6	51608
26	50	ST_CLERK	20	55700
27	90	소계	3	58000
28	80	SA_MAN	5	61000
29	50	SH_CLERK	20	64300
30	50	소계	45	156400
31	80	SA_REP	29	243500
32	80	소계	34	304500
33	합계	소계	107	691416