분석 함수 활용

문제 1.

EMP 테이블에서 사원 정보와 소속 부서의 평균 급여를 다음과 같이 검색하시오.

출력 결과

EMPNO	ENAME	SAL	DEPTNO	AVG_SAL
7782	CLARK	2450	10	2916.67
7839	KING	5000	10	2916.67
7934	MILLER	1300	10	2916.67
7566	JONES	2975	20	2175
7902	FORD	3000	20	2175
7876	ADAMS	1100	20	2175
7369	SMITH	800	20	2175
7788	SCOTT	3000	20	2175
7521	WARD	1250	30	1566.67
7844	TURNER	1500	30	1566.67
7499	ALLEN	1600	30	1566.67
7900	JAMES	950	30	1566.67
7698	BLAKE	2850	30	1566.67
7654	MARTIN	1250	30	1566.67

14 rows selected.

해결 방법 1. Inline View 활용

```
SQL> SELECT e.empno, e.ename, e.sal, e.deptno, a.avg_sal
     FROM emp e, ( SELECT deptno, ROUND(AVG(sal),2) AS AVG_SAL
                   FROM emp GROUP BY deptno ) a
     WHERE e.deptno = a.deptno ORDER BY e.deptno ;
```

해결 방법 2. Scalar Subquery 이용

```
SQL> SELECT empno, ename, sal, deptno, (SELECT ROUND(AVG(sal),2)
                                        FROM emp
                                       WHERE deptno = e.deptno) AS AVG_SAL
     FROM emp e ORDER BY deptno ;
```

두 가지의 Subquery의 사용 방법을 통해 문제를 해결할 수 있다. 경우에 따라서는 위와 같은 문장의 사용이 필요할 수도 있으나 동일 테이블의 반복적인 접근이 오히려 성능을 저하시키는 경우도 있다.

해결 방법 3. 분석 함수 사용

```
SQL> SELECT empno, ename, sal, deptno,
            ROUND(AVG(sal) OVER(PARTITION BY deptno),2) AS AVG_SAL
     FROM emp ;
```

분석 함수의 사용이 항상 정답은 아니다. 경우에 따라 일반적인 Subquery를 이용하는 것이 필요할 수도 있다. 단, 다양한 문장을 통해 동일한 결과를 검색하도록 문장을 작성할 수 있다면 경우에 따라 필요한 문장의 선택을 보다 손쉽게 할 수 있다. 성능과 관련된 자세한 사항은 SQL Tuning 과정을 참고한다.

문제 2.

EMP 테이블에서 1981년에 입사한 사원 정보와 소속 부서의 평균 급여를 분석 함수를 이용하여 검색하시오.

출력 결과

```
     EMPNO ENAME             SAL HIREDATE     DEPTNO    AVG_SAL
---------- ---------- ---------- -------- ---------- ----------
      7782 CLARK            2450 81/06/09         10    2916.67
      7839 KING             5000 81/11/17         10    2916.67
      7566 JONES            2975 81/04/02         20       2175
      7902 FORD             3000 81/12/03         20       2175
      7698 BLAKE            2850 81/05/01         30    1566.67
      7844 TURNER           1500 81/09/08         30    1566.67
      7654 MARTIN           1250 81/09/28         30    1566.67
      7900 JAMES             950 81/12/03         30    1566.67
      7521 WARD             1250 81/02/22         30    1566.67
      7499 ALLEN            1600 81/02/20         30    1566.67

10 rows selected.
```

해결 방법 1. Scalar Subquery 이용

```
SQL> SELECT empno, ename, sal, hiredate, deptno, (SELECT ROUND(AVG(sal),2)
                                                  FROM emp
                                                  WHERE deptno = e.deptno) AS AVG_SAL
     FROM emp e
     WHERE hiredate BETWEEN TO_DATE('1981/01/01','YYYY/MM/DD')
                        AND TO_DATE('1981/12/31','YYYY/MM/DD')
     ORDER BY deptno ;
```

해결 방법 2. Inline View 이용

```
SQL> SELECT e.empno, e.ename, e.sal, e.hiredate, e.deptno, a.avg_sal
     FROM emp e, ( SELECT deptno, ROUND(AVG(sal),2) AS AVG_SAL
                   FROM emp
                   WHERE deptno IS NOT NULL
                   GROUP BY deptno ) a
     WHERE e.deptno = a.deptno
       AND hiredate BETWEEN TO_DATE('1981/01/01','YYYY/MM/DD')
                        AND TO_DATE('1981/12/31','YYYY/MM/DD')
     ORDER BY e.deptno ;
```

해결 방법 3. 분석 함수 이용

```
SQL> SELECT empno, ename, sal, hiredate, deptno,
            ROUND(AVG(sal) OVER(PARTITION BY deptno),2) AS AVG_SAL
     FROM emp
     WHERE hiredate BETWEEN TO_DATE('1981/01/01','YYYY/MM/DD')
                        AND TO_DATE('1981/12/31','YYYY/MM/DD') ;
```

EMPNO	ENAME	SAL	HIREDATE	DEPTNO	AVG_SAL
7782	CLARK	2450	81/06/09	10	3725
7839	KING	5000	81/11/17	10	3725
7566	JONES	2975	81/04/02	20	2987.5
7902	FORD	3000	81/12/03	20	2987.5
7698	BLAKE	2850	81/05/01	30	1566.67
7844	TURNER	1500	81/09/08	30	1566.67
7654	MARTIN	1250	81/09/28	30	1566.67
7900	JAMES	950	81/12/03	30	1566.67
7521	WARD	1250	81/02/22	30	1566.67
7499	ALLEN	1600	81/02/20	30	1566.67

```
10 rows selected.
```

분석 함수는 WHERE절의 조건식 비교 후 결과 값을 생성한다. 때문에 위와 같은 문장은 조건절이 먼저 비교되면서 부서별 평균 급여는 잘못된 값을 보여줄 수 있다.

```
SQL> SELECT empno, ename, sal, hiredate, deptno, avg_sal
     FROM (SELECT empno, ename, sal, hiredate, deptno,
                  ROUND(AVG(sal) OVER(PARTITION BY deptno),2) AS AVG_SAL
           FROM emp)
     WHERE hiredate BETWEEN TO_DATE('1981/01/01','YYYY/MM/DD')
                        AND TO_DATE('1981/12/31','YYYY/MM/DD') ;
```

문제 3.

EMP 테이블에서 사원들의 정보를 EMPNO 컬럼으로 정렬하고, 각 사원의 급여를 행 별로 누적하여 TOTAL 컬럼을 검색하시오.

출력 결과

```
     EMPNO ENAME             SAL      TOTAL
---------- ---------- ---------- ----------
      7369 SMITH             800        800
      7499 ALLEN            1600       2400
      7521 WARD             1250       3650
      7566 JONES            2975       6625
      7654 MARTIN           1250       7875
      7698 BLAKE            2850      10725
      7782 CLARK            2450      13175
      7788 SCOTT            3000      16175
      7839 KING             5000      21175
      7844 TURNER           1500      22675
      7876 ADAMS            1100      23775
      7900 JAMES             950      24725
      7902 FORD             3000      27725
      7934 MILLER           1300      29025

14 rows selected.
```

해결 방법 1. Self Join 이용

```
SQL> SELECT a.empno, a.ename, a.sal , SUM(b.sal) AS TOTAL
     FROM emp a , emp b
     WHERE a.empno >= b.empno
     GROUP BY a.empno, a.ename, a.sal
     ORDER BY a.empno ;
```

해결 방법 2. Scalar Subquery 이용

```
SQL> SELECT a.empno, a.ename, a.sal, (SELECT SUM(sal)
                                      FROM emp
                                      WHERE empno <= a.empno) AS TOTAL
     FROM emp a
     ORDER BY a.empno ;
```

해결 방법 3. 분석 함수 이용

```
SQL> SELECT empno, ename, sal, SUM(sal) OVER (ORDER BY empno) AS TOTAL
      FROM emp ;
```

JOIN, Subquery 등을 이용하여도 결과는 생성 가능하지만 동일 집합의 반복 접근은 I/O를 증가시키며 성능을 악화 시킬 수 있다. 필요에 따라 Join, Subquery를 이용해야 할 수도 있지만 분석 함수의 사용을 고려한다.

추가 실습

```
SQL> SELECT empno, ename, sal,
            SUM(sal) OVER(ORDER BY empno ROWS BETWEEN UNBOUNDED PRECEDING
                                                  AND CURRENT ROW) AS TOTAL
     FROM emp ;
```

EMPNO	ENAME	SAL	TOTAL
7369	SMITH	800	800
7499	ALLEN	1600	2400
7521	WARD	1250	3650
7566	JONES	2975	6625
7654	MARTIN	1250	7875
7698	BLAKE	2850	10725
7782	CLARK	2450	13175
7788	SCOTT	3000	16175

...

14 rows selected.

```
SQL> SELECT empno, ename, sal,
            SUM(sal) OVER(ORDER BY empno ROWS BETWEEN CURRENT ROW
                                                  AND UNBOUNDED FOLLOWING) AS TOTAL
     FROM emp ;
```

EMPNO	ENAME	SAL	TOTAL
7369	SMITH	800	29025
7499	ALLEN	1600	28225
7521	WARD	1250	26625
7566	JONES	2975	25375
7654	MARTIN	1250	22400
7698	BLAKE	2850	21150
7782	CLARK	2450	18300
7788	SCOTT	3000	15850

...

14 rows selected.

```
SQL> SELECT empno, ename, sal,
            SUM(sal) OVER(ORDER BY empno ROWS BETWEEN 1 PRECEDING
                                                 AND 1 FOLLOWING) AS TOTAL
     FROM emp ;

     EMPNO ENAME             SAL      TOTAL
---------- ---------- ---------- ----------
      7369 SMITH             800       2400
      7499 ALLEN            1600       3650
      7521 WARD             1250       5825
      7566 JONES            2975       5475
      7654 MARTIN           1250       7075
      7698 BLAKE            2850       6550
      7782 CLARK            2450       8300
      7788 SCOTT            3000      10450
      7839 KING             5000       9500
      7844 TURNER           1500       7600
      7876 ADAMS            1100       3550
      7900 JAMES             950       5050
      7902 FORD             3000       5250
      7934 MILLER           1300       4300

14 rows selected.
```

문제 4.

EMP 테이블에서 입사 일자를 기준으로 전후 3개월에 해당하는 사원들의 급여의 합계를 다음과 같이 출력하시오.

출력 결과

ENAME	HIREDATE	BEFORE	AFTER	SAL	SUM_SAL
SMITH	1980/12/17	1980/09/17	1981/03/17	800	3650
ALLEN	1981/02/20	1980/11/20	1981/05/20	1600	9475
WARD	1981/02/22	1980/11/22	1981/05/22	1250	9475
JONES	1981/04/02	1981/01/02	1981/07/02	2975	11125
BLAKE	**1981/05/01**	1981/02/01	1981/08/01	2850	11125
CLARK	1981/06/09	1981/03/09	1981/09/09	2450	9775
TURNER	1981/09/08	1981/06/08	1981/12/08	1500	14150
MARTIN	1981/09/28	1981/06/28	1981/12/28	1250	11700
KING	1981/11/17	1981/08/17	1982/02/17	5000	13000
JAMES	1981/12/03	1981/09/03	1982/03/03	950	13000
FORD	1981/12/03	1981/09/03	1982/03/03	3000	13000
MILLER	1982/01/23	1981/10/23	1982/04/23	1300	10250
SCOTT	1987/04/19	1987/01/19	1987/07/19	3000	4100
ADAMS	1987/05/23	1987/02/23	1987/08/23	1100	4100

14 rows selected.

해결 방법 1. Scalar Subquery 이용

```
SQL> SELECT ename, hiredate, ADD_MONTHS(hiredate, -3) AS BEFORE,
                             ADD_MONTHS(hiredate, +3) AS AFTER,
                    sal,
                    (SELECT SUM(sal)
                     FROM emp
                     WHERE hiredate BETWEEN ADD_MONTHS(e.hiredate, -3)
                                        AND ADD_MONTHS(e.hiredate, +3)) AS sum_sal
    FROM emp e
    ORDER BY e.hiredate ;
```

해결 방법 2. 분석 함수 이용

```
SQL> SELECT ename, hiredate, hiredate - INTERVAL '3' MONTH  AS BEFORE,
                             hiredate + INTERVAL '3' MONTH  AS AFTER ,
                    sal,
                    SUM(sal) OVER(ORDER BY hiredate
                                  RANGE BETWEEN INTERVAL '3' MONTH PRECEDING
                                            AND INTERVAL '3' MONTH FOLLOWING) AS sum_sal
     FROM emp ;
```

문제 5.

EMP 테이블에서 최대 급여, 최소 급여, 급여 합계를 다음과 같이 출력하시오.

출력 결과

ENAME	SAL	DEPTNO	MAXSAL	MINSAL	SUMSAL
SMITH	800	20	5000	800	29025
ALLEN	1600	30	5000	800	29025
WARD	1250	30	5000	800	29025
JONES	2975	20	5000	800	29025
MARTIN	1250	30	5000	800	29025
BLAKE	2850	30	5000	800	29025
CLARK	2450	10	5000	800	29025
SCOTT	3000	20	5000	800	29025
KING	5000	10	5000	800	29025
TURNER	1500	30	5000	800	29025
ADAMS	1100	20	5000	800	29025
JAMES	950	30	5000	800	29025
FORD	3000	20	5000	800	29025
MILLER	1300	10	5000	800	29025

14 rows selected.

해결 방법 1. Scalar Subquery 이용

```
SQL> SELECT ename, sal, deptno, ( SELECT MAX(sal) FROM emp ) AS maxsal,
                                ( SELECT MIN(sal) FROM emp ) AS minsal,
                                ( SELECT SUM(sal) FROM emp ) AS sumsal
      FROM emp ;
```

Subquery의 결과가 캐싱되어 각 Subquery의 실행 횟수는 한 번씩이지만 세 개의 Subquery가 각각 실행된다.

해결 방법 2. Subquery 사용 최소화

```
SQL> SELECT e1.ename, e1.sal, e2.max, e2.min, e2.sum
     FROM emp e1, (SELECT MAX(sal) AS max, MIN(sal) AS min, SUM(sal) AS sum
                   FROM emp
                   WHERE sal IS NOT NULL) e2 ;
```

해결 방법 3. 분석 함수 이용

```
SQL> SELECT ename, sal, deptno, MAX(sal) OVER() AS maxsal,
                                MIN(sal) OVER() AS minsal,
                                SUM(sal) OVER() AS sumsal
     FROm emp ;
```

문제 6.

EMP 테이블에서 '1981/12/25' 일에 입사한 사원의 입사 순위를 다음과 같이 검색하시오. 만약 해당 날짜의 데이터가 없다면 예상되는 순위를 검색하시오.

```
        RK HIREDATE
---------- --------------------
         1 1980/12/17
         2 1981/02/20
         3 1981/02/22
         4 1981/04/02
         5 1981/05/01
         6 1981/06/09
         7 1981/09/08
         8 1981/09/28
         9 1981/11/17
        10 1981/12/03
        10 1981/12/03
           1981/12/25
        12 1982/01/23
        13 1987/04/19
        14 1987/05/23
```

출력 결과

```
      RANK
----------
        11
```

해결 방법

```
SQL> SELECT DENSE_RANK(TO_DATE('81/12/25','RR/MM/DD'))
            WITHIN GROUP (ORDER BY hiredate) AS RANK
     FROM emp ;
```

함수 소개: Hypothetical Functions

가정에 근거하여 각 함수에 맞는 값을 확인

Q. 급여가 2000 이라면 각각 순위 및 백분율은 얼마인가?

```
SQL> SELECT RANK (2000)         WITHIN GROUP ( ORDER BY sal DESC ) rank,
            DENSE_RANK (2000)   WITHIN GROUP ( ORDER BY sal DESC ) dense_rank,
            CUME_DIST (2000)    WITHIN GROUP ( ORDER BY sal DESC ) cume_dist,
            PERCENT_RANK (2000) WITHIN GROUP ( ORDER BY sal DESC ) per_rank
     FROM emp ;
      RANK DENSE_RANK  CUME_DIST   PER_RANK
---------- ---------- ---------- ----------
         7          6 .466666667 .428571429
```

문제 7.

EMP 테이블에서 최대 급여, 최소 급여를 받는 사원의 이름을 다음과 같이 검색하시오.

출력 결과

```
MAX_ENAME  MIN_ENAME
---------- ----------
KING       SMITH
```

해결 방법 1. Subquery 이용

```
SQL> SELECT *
     FROM (SELECT ename AS MAX_ENAME
           FROM emp
           WHERE sal = ( SELECT MAX(sal) FROM emp ) ) a,
          (SELECT ename AS MIN_ENAME
           FROM emp
           WHERE sal = ( SELECT MIN(sal) FROM emp ) ) b;
```

해결 방법 2. 분석 함수 이용

```
SQL> SELECT MAX(ename) keep (dense_rank FIRST ORDER BY sal DESC) AS MAX_ENAME,
            MAX(ename) keep (dense_rank LAST  ORDER BY sal DESC) AS MIN_ENAME
     FROM emp ;
```

추가 실습

```
SQL> SELECT deptno,
            MAX(ename) keep (dense_rank FIRST ORDER BY sal DESC) AS MAX_ENAME,
            MAX(ename) keep (dense_rank LAST  ORDER BY sal DESC) AS MIN_ENAME
     FROM emp
     GROUP BY deptno ;
```

```
    DEPTNO MAX_ENAME  MIN_ENAME
---------- ---------- ----------
        10 KING       MILLER
        20 SCOTT      SMITH
        30 BLAKE      JAMES
```

함수 소개: FIRST / LAST

각 WINDOW 영역에서 FIRST/LAST 하나의 행만을 추출하려는 경우 사용 가능

ORDER BY 를 이용하여 WINDOW 내의 정렬을 진행하고 DENSE_RANK FIRST/LAST로 그들 중 하나의 행을 선택할 수 있다. 이때 동일한 ranking을 가지고 있는 집합들 중 Aggregate Function의 결과를 보여 준다.

문제 8.

30번 부서의 사원들 중에 가장 많은 커미션과 가장 적은 커미션을 받는 사원의 이름과 함께 검색하시오.

› 출력 결과

```
     EMPNO ENAME             SAL       COMM MAX        MIN
---------- ---------- ---------- ---------- ---------- ----------
      7698 BLAKE            2850            MARTIN     TURNER
      7900 JAMES             950            MARTIN     TURNER
      7654 MARTIN           1250       1400 MARTIN     TURNER
      7521 WARD             1250        500 MARTIN     TURNER
      7499 ALLEN            1600        300 MARTIN     TURNER
      7844 TURNER           1500          0 MARTIN     TURNER

6 rows selected.
```

해결 방법

```
SQL> SELECT empno, ename, sal, comm,
            FIRST_VALUE(ename) OVER(ORDER BY comm DESC NULLS LAST
                                    ROWS BETWEEN UNBOUNDED PRECEDING
                                            AND UNBOUNDED FOLLOWING) AS max,
            LAST_VALUE(ename)  OVER(ORDER BY comm DESC NULLS FIRST
                                    ROWS BETWEEN UNBOUNDED PRECEDING
                                            AND UNBOUNDED FOLLOWING) AS min
     FROM emp
     WHERE deptno = 30 ;
```

복잡한 Subquery 등을 활용하면 분석 함수의 사용 없이도 검색은 가능하지만 앞서 확인 했듯이 성능상 유리하지 않을 것이다. FIRST_VALUE/LAST_VALUE 함수는 FIRST / LAST와 비슷하게 WINDOW 내의 처음과 마지막 행의 값을 가져 올 수 있으며 WINDOWING 절을 지정하여 원하는 WINDOW의 정의가 가능하다. 또한 NULL을 제외 시키고 작업 가능하다. (FIRST/LAST는 WINDOWING 절 사용이 불가능하며 NULL이 포함되어 계산 됨)

문제 9.

EMP 테이블에서 HIREDATE, EMPNO 컬럼으로 정렬하여 다음과 같이 사원 정보를 검색한다.
이때 해당 사원보다 위, 아래(앞, 뒤)의 입사 일자를 함께 출력하시오.

출력 결과

EMPNO	ENAME	HIREDATE	PREV_HIRE	NEXT_HIRE
7369	SMITH	1980/12/17		1981/02/20
7499	ALLEN	**1981/02/20**	1980/12/17	1981/02/22
7521	**WARD**	**1981/02/22**	**1981/02/20**	**1981/04/02**
7566	JONES	**1981/04/02**	1981/02/22	1981/05/01
7698	BLAKE	1981/05/01	1981/04/02	1981/06/09
7782	CLARK	1981/06/09	1981/05/01	1981/09/08
7844	TURNER	1981/09/08	1981/06/09	1981/09/28
7654	MARTIN	1981/09/28	1981/09/08	1981/11/17
7839	KING	1981/11/17	1981/09/28	1981/12/03
7900	JAMES	1981/12/03	1981/11/17	1981/12/03
7902	FORD	1981/12/03	1981/12/03	1982/01/23
7934	MILLER	1982/01/23	1981/12/03	1987/04/19
7788	SCOTT	1987/04/19	1982/01/23	1987/05/23
7876	ADAMS	1987/05/23	1987/04/19	

14 rows selected.

해결 방법 1. ROWNUM을 이용한 Outer Join 사용

```
SQL> SELECT a.empno, a.ename, a.hiredate,b.hiredate AS PREV_HIRE,c.hiredate AS NEXT_HIRE
     FROM (SELECT ROWNUM no1, empno, ename, hiredate
           FROM (SELECT empno, ename, hiredate
                 FROM emp
                 ORDER BY hiredate,empno)) a,
          (SELECT ROWNUM+1 no2, empno, ename, hiredate
           FROM (SELECT empno, ename, hiredate
                 FROM emp
                 ORDER BY hiredate,empno)) b,
          (SELECT ROWNUM-1 no3, empno, ename, hiredate
           FROM (SELECT empno, ename, hiredate
                 FROM emp
                 ORDER BY hiredate,empno)) c
     WHERE a.no1 = b.no2 (+)
       AND a.no1 = c.no3 (+)
     ORDER BY a.hiredate, a.empno ;
```

해결 방법 2. 분석 함수 이용

```
SQL> SELECT empno, ename, hiredate,
            LAG(hiredate) OVER (ORDER BY hiredate,empno) AS PREV_HIRE,
            LEAD(hiredate) OVER (ORDER BY hiredate,empno) AS NEXT_HIRE
     FROM emp ;
```

함수 소개: LAG / LEAD

지정된 개수의 이전, 이후 행의 값 가져오기. WINDOWING 절을 지정하지 못하며 NULL 값을 대체하는 값을 지정할 수 있음. (NVL 불필요)

Q. 30번 부서의 사원을 이름순으로 정렬하여 검색하며 이전, 다음 행의 급여를 함께 표시

```
SQL> SELECT empno, ename, sal,
            LAG (sal,1,0) over ( order by ename ) prev_sal,
            LEAD (sal,1,0) over ( order by ename ) next_sal
     FROM emp
     WHERE deptno = 30 ;
```

EMPNO	ENAME	SAL	PREV_SAL	NEXT_SAL
7499	ALLEN	1600	0	2850
7698	BLAKE	2850	1600	950
7900	JAMES	950	2850	1250
7654	MARTIN	1250	950	1500
7844	TURNER	1500	1250	1250
7521	WARD	1250	1500	0

6 rows selected.

문제 10.

EMP 테이블에서 급여를 많이 받는 순서대로 정렬하였을 때 5개의 범위로 나누어 각 등급을 다음과 같이 출력하시오.

출력 결과

```
     EMPNO ENAME             SAL      GRADE
---------- ---------- ---------- ----------
      7839 KING             5000          1
      7902 FORD             3000          1
      7788 SCOTT            3000          1
      7566 JONES            2975          2
      7698 BLAKE            2850          2
      7782 CLARK            2450          2
      7499 ALLEN            1600          3
      7844 TURNER           1500          3
      7934 MILLER           1300          3
      7521 WARD             1250          4
      7654 MARTIN           1250          4
      7876 ADAMS            1100          4
      7900 JAMES             950          5
      7369 SMITH             800          5

14 rows selected.
```

해결 방법 1. Cartesian Product 활용

```
SQL> SELECT empno, ename, sal, grade
     FROM (SELECT empno, ename, sal,
                  CEIL(ROUND(rownum/CNT,1)) AS grade
           FROM ( SELECT empno, ename, sal
                  FROM emp
                  ORDER BY sal DESC ),
                ( SELECT CEIL(COUNT(*)/5) CNT
                  FROM emp) ) ;
```

해결 방법 2. 분석 함수 이용

```
SQL> SELECT empno, ename, sal, NTILE (5) OVER (ORDER BY sal DESC) grade
     FROM emp ;
```

함수 소개: NTILE

WINDOW 그룹의 행을 정렬 후 지정한 개수의 범위(등급)으로 나눈 후 각 값이 가지고 있는 등급 값을 보여준다.

문제 11.

EMP 테이블의 사원정보를 다음과 같이 검색하시오.

각 부서의 가장 큰 급여부터 3번째 값까지의 결과를 함께 출력

출력 결과

DEPTNO	ENAME	SAL	M1_SAL	M2_SAL	M3_SAL
10	**KING**	**5000**	**5000**	**2450**	**1300**
10	**CLARK**	**2450**	**5000**	**2450**	**1300**
10	**MILLER**	**1300**	**5000**	**2450**	**1300**
20	FORD	3000	3000	2975	1100
20	SCOTT	3000	3000	2975	1100
20	JONES	2975	3000	2975	1100
20	ADAMS	1100	3000	2975	1100
20	SMITH	800	3000	2975	1100
30	BLAKE	2850	2850	1600	1500
30	ALLEN	1600	2850	1600	1500
30	TURNER	1500	2850	1600	1500
30	WARD	1250	2850	1600	1500
30	MARTIN	1250	2850	1600	1500
30	JAMES	950	2850	1600	1500

14 rows selected.

해결 방법 1. 분석 함수 이용

```
SQL> SELECT deptno,
            ename,
            MIN(sal) AS sal,
            NTH_VALUE(MIN(sal),1)
            OVER(PARTITION BY deptno ORDER BY MIN(sal) DESC
                 ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS m1_sal,
            NTH_VALUE(MIN(sal),2)
            OVER(PARTITION BY deptno ORDER BY MIN(sal) DESC
                 ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS m2_sal,
            NTH_VALUE(MIN(sal),3)
            OVER(PARTITION BY deptno ORDER BY MIN(sal) DESC
                 ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS m3_sal
     FROM emp
    GROUP BY deptno, ename;
```

해결 방법 2. Subquery 이용

```
SQL> SELECT deptno, ename, sal, m1_sal, m2_sal, (SELECT MAX(sal) FROM EMP
                                                  WHERE sal != m1_sal
                                                    AND sal != m2_sal
                                                    AND deptno = 10) AS m3_sal
     FROM(SELECT deptno, ename, sal, m1_sal, (SELECT MAX(sal) FROM EMP
                                               WHERE sal != m1_sal
                                                 AND deptno = 10) AS m2_sal
           FROM EMP, (SELECT MAX(sal) AS m1_sal FROM EMP WHERE deptno = 10)
           WHERE deptno = 10)
     WHERE deptno = 10
     UNION ALL
     SELECT deptno, ename, sal, m1_sal, m2_sal, (SELECT MAX(sal) FROM EMP
                                                  WHERE sal != m1_sal
                                                    AND sal != m2_sal
                                                    AND deptno = 20) AS m3_sal
     FROM(SELECT deptno, ename, sal, m1_sal, (SELECT MAX(sal) FROM EMP
                                              WHERE sal != m1_sal
                                                AND deptno = 20) AS m2_sal
          FROM EMP, (SELECT MAX(sal) AS m1_sal FROM EMP
                     WHERE deptno = 20)
          WHERE deptno = 20)
     WHERE deptno = 20
     UNION ALL
     SELECT deptno, ename, sal, m1_sal, m2_sal, (SELECT MAX(sal) FROM EMP
                                                  WHERE sal != m1_sal
                                                    AND sal != m2_sal
                                                    AND deptno = 30) AS m3_sal
     FROM(SELECT deptno, ename, sal, m1_sal, (SELECT MAX(sal) FROM EMP
                                               WHERE sal != m1_sal
                                                 AND deptno = 30) AS m2_sal
          FROM EMP, (SELECT MAX(sal) AS m1_sal FROM EMP
                     WHERE deptno = 30)
          WHERE deptno = 30)
     WHERE deptno = 30
     ORDER BY deptno, sal DESC ;
```

문제 12.

EMP 테이블에서 다음과 같이 사원 정보를 검색하시오.

출력 결과

```
    DEPTNO EMPLOYEES
---------- ------------------------------------------------
        10 CLARK,KING,MILLER
        20 ADAMS,FORD,JONES,SCOTT,SMITH
        30 ALLEN,BLAKE,JAMES,MARTIN,TURNER,WARD
```

해결 방법 1. 분석 함수 및 계층 질의 이용

```
SQL> SELECT deptno,
            LTRIM(MAX(SYS_CONNECT_BY_PATH(ename,','))
                  KEEP(DENSE_RANK LAST ORDER BY curr),',') AS employee
     FROM   (SELECT deptno,
                    ename,
                    ROW_NUMBER() OVER (PARTITION BY deptno ORDER BY ename) AS curr,
                    ROW_NUMBER() OVER (PARTITION BY deptno ORDER BY ename) -1 AS prev
             FROM   emp)
     START WITH curr = 1
     CONNECT BY prev = PRIOR curr AND deptno = PRIOR deptno
     GROUP BY deptno ;
```

```
    DEPTNO EMPLOYEE
---------- ----------------------------------------
        10 CLARK,KING,MILLER
        20 ADAMS,FORD,JONES,SCOTT,SMITH
        30 ALLEN,BLAKE,JAMES,MARTIN,TURNER,WARD
```

해결 방법 2. 히든 함수 (WM_CONCAT) 사용

"$ORACLE_HOME/rdbms/admin/owminst.plb" 이용하여 WMSYS 스키마 생성 필요 (v10g)

```
SQL> SELECT deptno, WM_CONCAT(ename) AS employee
     FROM   emp
     GROUP BY deptno;
```

```
    DEPTNO EMPLOYEE
---------- ----------------------------------------
        10 CLARK,MILLER,KING
        20 SMITH,FORD,ADAMS,SCOTT,JONES
        30 ALLEN,JAMES,TURNER,BLAKE,MARTIN,WARD
```

ENAME 컬럼을 기준으로 정렬되지는 않았다. VM_CONCAT 함수는 정렬 작업을 지원하지 않는다.

정렬 작업이 필요하다면?

```
SQL> SELECT deptno, MAX(employee) AS employee
     FROM (SELECT deptno,
                  WM_CONCAT(ename) OVER(PARTITION BY deptno ORDER BY ename) AS employee
           FROM emp
           ORDER BY deptno)
     GROUP BY deptno ;

    DEPTNO EMPLOYEE
---------- ----------------------------------------
        10 CLARK,KING,MILLER
        20 ADAMS,FORD,JONES,SCOTT,SMITH
        30 ALLEN,BLAKE,JAMES,MARTIN,TURNER,WARD
```

해결 방법 3. LISTAGG 사용 (v11gR2)

```
SQL> SELECT deptno,
            LISTAGG(ename,',') WITHIN GROUP (ORDER BY ename) AS employee
     FROM emp
     GROUP BY deptno ;

    DEPTNO EMPLOYEE
---------- ----------------------------------------
        10 CLARK,KING,MILLER
        20 ADAMS,FORD,JONES,SCOTT,SMITH
        30 ALLEN,BLAKE,JAMES,MARTIN,TURNER,WARD
```

추가 실습

Q. 부서별 직업의 중복을 제거하고 출력

```
SQL> SELECT  deptno, WM_CONCAT(DISTINCT  job) AS employee
     FROM   emp GROUP BY deptno ;

    DEPTNO EMPLOYEE
---------- ----------------------------------------
        10 CLERK,MANAGER,PRESIDENT
        20 ANALYST,CLERK,MANAGER
        30 CLERK,MANAGER,SALESMAN
```

```
SQL> SELECT d.dname, LISTAGG(e.ename||'('||e.sal||')',',')
                     WITHIN GROUP (ORDER BY ename) AS employee
     FROM emp e, dept d
     WHERE e.deptno = d.deptno
     GROUP BY d.dname ;

DNAME          EMPLOYEE
-------------- --------------------------------------------------------------------------------
ACCOUNTING     CLARK(2450),KING(5000),MILLER(1300)
RESEARCH       ADAMS(1100),FORD(3000),JONES(2975),SCOTT(3000),SMITH(800)
SALES          ALLEN(1600),BLAKE(2850),JAMES(950),MARTIN(1250),TURNER(1500),WARD(1250)
```

문제 13.

EMP 테이블에서 DEPTNO, SAL, ENAME 컬럼을 기준으로 정렬된 정보를 검색하면서 부서별 누적된 급여, 부서별 급여의 백분율, 전체 사원의 급여 합계에서의 백분율을 검색한다.

출력 결과

DEPTNO	ENAME	SAL	CUM_SAL	PCT_DEPT	PCT_OVERALL
10	MILLER	1300	1300	14.9	4.5
10	CLARK	2450	3750	28	8.4
10	KING	5000	8750	57.1	17.2
20	SMITH	800	800	7.4	2.8
20	ADAMS	1100	1900	10.1	3.8
20	JONES	2975	4875	27.4	10.2
20	FORD	3000	7875	27.6	10.3
20	SCOTT	3000	10875	27.6	10.3
30	JAMES	950	950	10.1	3.3
30	MARTIN	1250	2200	13.3	4.3
30	WARD	1250	3450	13.3	4.3
30	TURNER	1500	4950	16	5.2
30	ALLEN	1600	6550	17	5.5
30	BLAKE	2850	9400	30.3	9.8

14 rows selected.

해결 방법 1. Subquery 이용

```
SQL> SELECT e1.deptno, e1.ename, e1.sal,
            SUM(e4.sal) AS cum_sal,
            ROUND(100*e1.sal/e2.sal_by_dept,1) AS pct_dept,
            ROUND(100*e1.sal/e3.sal_overall,1) AS pct_overall
     FROM emp e1,
          (SELECT deptno, SUM(sal) sal_by_dept
           FROM emp GROUP BY deptno) e2 ,
          (SELECT SUM(sal) sal_overall
           FROM emp) e3,
          emp e4
     WHERE e1.deptno = e2.deptno
       AND e1.deptno = e4.deptno
       AND (e1.sal   > e4.sal OR (e1.sal   = e4.sal AND e1.ename >= e4.ename))
     GROUP BY e1.deptno, e1.ename, e1.sal,
              ROUND(100*e1.sal/e2.sal_by_dept,1) ,
              ROUND(100*e1.sal/e3.sal_overall,1)
     ORDER BY deptno,sal,ename ;
```

해결 방법 2. 분석 함수 이용

```
SQL> SELECT deptno, ename, sal,
            SUM(sal) OVER(PARTITION BY deptno ORDER BY sal, ename)       AS cum_sal,
            ROUND(100*RATIO_TO_REPORT(sal) OVER(PARTITION BY deptno),1)  AS pct_dept,
            ROUND(100*RATIO_TO_REPORT(sal) OVER(),1)                     AS pct_overall
     FROM emp ;
```

함수 소개 : RATIO_TO_REPORT

WINDOW 영역의 합계 내에서 현재 값이 차지하는 백분율. 별도의 WINDOWING 절을 설정하는 것은 불가능하다.

Q. 사원 정보를 출력하면서 부서별 급여의 합계 중 해당 사원이 받는 급여의 백분율을 표시하고 부서별 급여의 합계 출력

```
SQL> break on deptno skip 1
SQL> compute sum label 'TOTAL' of sal on deptno
SQL> SELECT deptno, ename, sal,
            ROUND ( RATIO_TO_REPORT (sal) over ( partition BY deptno ) , 2) AS ratio
     FROM emp ;
```

```
    DEPTNO ENAME             SAL      RATIO
---------- ---------- ---------- ----------
        10 CLARK            2450        .28
           KING             5000        .57
           MILLER           1300        .15
********** ----------
TOTAL                       8750

        20 JONES            2975        .27
           FORD             3000        .28
           ADAMS            1100         .1
           SMITH             800        .07
           SCOTT            3000        .28
********** ----------
TOTAL                      10875

        30 WARD             1250        .13
           TURNER           1500        .16
           ALLEN            1600        .17
           JAMES             950         .1
           BLAKE            2850         .3
           MARTIN           1250        .13
********** ----------
TOTAL                       9400
```

```
SQL> clear compute break
```

문제 14.

각 부서별 사원 급여의 백분율 및 부서별 급여 합계와 부서별 급여 합계의 백분율을 검색한다.

출력 결과

```
    DEPTNO      EMPNO ENAME             SAL PCT
---------- ---------- ---------- ---------- --------------------
        10       7934 MILLER           1300 14.9%
        10       7782 CLARK            2450   28%
        10       7839 KING             5000 57.1%
                                       8750 30.1%
        20       7369 SMITH             800  7.4%
        20       7876 ADAMS            1100 10.1%
        20       7566 JONES            2975 27.4%
        20       7902 FORD             3000 27.6%
        20       7788 SCOTT            3000 27.6%
                                      10875 37.5%
        30       7900 JAMES             950 10.1%
        30       7654 MARTIN           1250 13.3%
        30       7521 WARD             1250 13.3%
        30       7844 TURNER           1500   16%
        30       7499 ALLEN            1600   17%
        30       7698 BLAKE            2850 30.3%
                                       9400 32.4%

17 rows selected.
```

해결 방법.

```
SQL> SELECT DECODE(GROUPING_ID(deptno,empno),0,deptno) AS deptno,
            empno,
            ename,
            DECODE(GROUPING_ID(deptno,empno), 0,SUM(sal),
                                              1,SUM(sal) OVER(PARTITION BY deptno
                                                              ORDER BY sal)) AS sal,
          LPAD(DECODE(GROUPING_ID(deptno,empno),
                      0,ROUND(RATIO_TO_REPORT(sal) OVER(PARTITION BY deptno) * 100,1)||'%',
                      1,ROUND(RATIO_TO_REPORT(DECODE(GROUPING_ID(deptno,empno),1,SUM(sal)))
                              OVER() * 100,1)||'%'),5,' ') AS pct
      FROM emp
      GROUP BY deptno, ROLLUP((empno,ename,sal)) ;
```

분석 함수

SQL은 Relationship 이 존재하는 테이블 구조의 데이터를 제어하기 위한 언어이다. SQL은 DB에 저장된 데이터를 제어하기 위한 매우 강력한 언어이지만, 다양한 Business Intelligence Calculation을 수행하기에는 부족함이 존재한다. 때문에 복잡한 형태의 분석 작업을 진행하려면 과도한 프로그래밍이 사용되고, 그로 인해 성능은 저하되기도 한다. Oracle Database 8i부터는 이러한 요구 사항들을 해결하기 위해 새로운 함수를 제공한다. 이 함수들은 분석 작업에 유용하기 때문에 Analytic Functions이라고 하며 DB 버전이 올라갈수록 계속 추가되고 있다.

```
SQL> SELECT empno, ename, sal, deptno, SUM(sal) OVER(PARTITION BY deptno) AS dept_tot
     FROM emp;

     EMPNO ENAME             SAL     DEPTNO   DEPT_TOT
---------- ---------- ---------- ---------- ----------
      7782 CLARK            2450         10       8750
      7839 KING             5000         10       8750
      7934 MILLER           1300         10       8750
      7566 JONES            2975         20      10875
      7902 FORD             3000         20      10875
...
```

분석 함수는 Aggregate Function 뒤에 Analytic Clause(OVER 절)을 통해 행 그룹의 정의를 지정하고 각 그룹당 결과 값을 반복하여 출력한다. 여기서 행 그룹의 범위를 WINDOW라 부르며 하나의 WINDOW가 계산을 수행하는데 사용되는 행들의 집합을 결정하게 되며 PARTITION BY, ORDER BY, WINDOWING을 통하여 조절하게 된다.

분석 함수는 Join 문장, WHERE, GROUP BY, HAVING 등과 함께 쓰일 때 가장 마지막에 연산(집계)을 진행하며 SELECT 절과 ORDER BY 절에서만 사용이 가능하다.

PARTITION BY 절은 GROUP BY 절과 동일한 작업 수행한다. 단, GROUP BY 절을 사용하지 않고 필요한 집합으로 (WINDOW) 행들을 그룹화 시킬 수 있다.

```
SQL> SELECT empno, ename, sal, AVG(sal) OVER()                    AS avg_overall,
                               AVG(sal) OVER(PARTITION BY deptno) AS avg_deptno
     FROM emp ;

     EMPNO ENAME             SAL AVG_OVERALL AVG_DEPTNO
---------- ---------- ---------- ----------- ----------
      7782 CLARK            2450  2073.21429 2916.66667
      7839 KING             5000  2073.21429 2916.66667
      7934 MILLER           1300  2073.21429 2916.66667
      7566 JONES            2975  2073.21429       2175
...
```

ORDER BY 절은 PARTITION BY로 정의된 WINDOW 내에서의 행들의 정렬 순서를 정의

```
SQL> SELECT empno, ename, sal, deptno,
            row_number( ) over ( ORDER BY sal ASC ) AS rnum
     FROM emp ;

     EMPNO ENAME             SAL     DEPTNO       RNUM
---------- ---------- ---------- ---------- ----------
      7369 SMITH             800         20          1
      7900 JAMES             950         30          2
      7876 ADAMS            1100         20          3
...

SQL> SELECT empno, ename, sal, deptno,
            row_number( ) over ( PARTITION BY deptno ORDER BY sal DESC ) AS rnum
     FROM emp ;

     EMPNO ENAME             SAL     DEPTNO       RNUM
---------- ---------- ---------- ---------- ----------
      7839 KING             5000         10          1
      7782 CLARK            2450         10          2
      7934 MILLER           1300         10          3
      7788 SCOTT            3000         20          1
      7902 FORD             3000         20          2
...

SQL> SELECT empno, ename, sal, comm,
             DENSE_RANK( ) over ( ORDER BY comm ASC )             AS no1,
             DENSE_RANK( ) over ( ORDER BY comm ASC NULLS FIRST ) AS no2
      FROM emp
      WHERE deptno = 30 ;

     EMPNO ENAME             SAL       COMM        NO1        NO2
---------- ---------- ---------- ---------- ---------- ----------
      7698 BLAKE            2850                     5          1
      7900 JAMES             950                     5          1
      7844 TURNER           1500          0          1          2
      7499 ALLEN            1600        300          2          3
      7521 WARD             1250        500          3          4
      7654 MARTIN           1250       1400          4          5

6 rows selected.
```

ORDER BY 절은 PARTITION BY에 의해 그룹화된 행들의 정렬 순서를 결정하며 NULL 값을 가지고 있는 행이 있을 경우 NULL에 대한 값을 FIRST, LAST로 보낼 수 있도록 조절 가능하다.

WINDOWING 절은 일부 Aggregate Function과 함께 쓰일 수 있으며 행들의 그룹을 물리적, 논리적으로 조절하여 Function이 적용될 WINDOW를 정의한다. 즉, PARTITON BY 절은 컬럼에 같은 값을 기준으로만 그룹화하지만 WINDOWING 절은 ROWS와 RANGE를 이용하여 하나의 WINDOW를 결정하는 범위를 보다 자유롭게 조정할 수 있다.

```
SQL> SELECT empno, ename, TO_CHAR(hiredate,'YYYY/MM/DD') hiredate, sal,
            SUM(sal) over ( ORDER BY empno
                            ROWS BETWEEN 3 PRECEDING
                                     AND 3 FOLLOWING ) AS physical,
            SUM(sal) over ( ORDER BY hiredate
                            RANGE BETWEEN INTERVAL '3' MONTH PRECEDING
                                      AND INTERVAL '3' MONTH FOLLOWING ) AS logical
     FROM emp
     WHERE deptno IN (10,20) ;
```

EMPNO	ENAME	HIREDATE	SAL	PHYSICAL	LOGICAL
7369	SMITH	1980/12/17	800	9225	800
7566	JONES	1981/04/02	2975	14225	5425
7782	CLARK	1981/06/09	2450	15325	5425
7839	KING	1981/11/17	5000	18825	9300
7902	FORD	1981/12/03	3000	13400	9300

ROWS는 WINDOW의 범위를 정의할 때 물리적인 행을 지정한다. 어떤 행에서 시작해서 어떤 행까지가 하나의 WINDOW 영역으로 정의 할지 범위를 BETWEEN을 통하여 정의 할 수 있다.

UNBOUNDED PRECEDING는 첫 번째 행,
UNBOUNDED FOLLOWING은 마지막 행,
CURRENT ROW는 현재 행 참조함.

RANGE는 논리적인 값을 근거로 WINDOW 범위를 설정할 수 있다. 모든 함수가 WINDOWING절을 사용할 수 있는 것은 아니다. (매뉴얼 참조)

추가 함수 소개

CUME_DIST (cumulative distribution) / PERCENT_RANK

둘 모두 계산식에 차이가 있으나 WINDOW 그룹 내에서 누적 분포를 계산할 때 사용

값의 범위는 0 ~ 1까지 사용되며 PERCENT_RANK는 항상 시작 값이 0부터 시작

Q. 급여를 내림차순 기준으로 정렬하였을 경우 각 사원의 누적 분포를 계산

```
SQL> SELECT ename, sal,
            ROUND ( PERCENT_RANK ( ) over ( order by sal DESC ),2) AS per_rank,
            ROUND ( CUME_DIST( ) over ( order by sal DESC ),2)     AS cume_dist,
            RANK( ) over ( order by sal DESC )                     AS rank,
            ROW_NUMBER ( ) over ( order by sal DESC )              AS row_num
     FROM emp ;
```

ENAME	SAL	PER_RANK	CUME_DIST	RANK	ROW_NUM
KING	5000	0	.07	1	1
FORD	3000	.08	.21	2	2
SCOTT	3000	.08	.21	2	3
JONES	2975	.23	.29	4	4
BLAKE	2850	.31	.36	5	5
CLARK	2450	.38	.43	6	6
ALLEN	1600	.46	.5	7	7
TURNER	1500	.54	.57	8	8
MILLER	1300	.62	.64	9	9
WARD	1250	.69	.79	10	10
MARTIN	1250	.69	.79	10	11
ADAMS	1100	.85	.86	12	12
JAMES	950	.92	.93	13	13
SMITH	800	1	1	14	14

PERCENT_RANK :(RANK - 1) / (COUNT(*) - 1)

CUME_DIST : (RANK or ROW_NUMBER) / COUNT(*)

동등 순위의 RANK 발생 시 해당 RANK의 마지막 ROW_NUMBER 사용

PERCENTILE_CONT / PERCENTILE_DISC

cume_dist, percent_rank의 결과가 누적 분포도를 계산 한다면 PERCENTILE_CONT, PERCENTILE_DISC는 분포도 값(지정되는 백분율)을 역으로 계산하여 실제의 값을 가져온다.

```
SQL> SELECT empno, ename, sal, deptno,
            ROUND( cume_dist ( ) over ( order by sal DESC ),2) AS cume_dist
     FROM emp
     WHERE deptno = 30 ;
```

EMPNO	ENAME	SAL	DEPTNO	CUME_DIST
7698	BLAKE	2850	30	.17
7499	ALLEN	1600	30	.33
7844	**TURNER**	**1500**	**30**	**.5**
7521	WARD	1250	30	.83
7654	MARTIN	1250	30	.83
7900	JAMES	950	30	1

30번 부서 번호의 사원 중 급여를 기준으로 내림차순 정렬 했을 때 1500의 급여가 전체 WINDOW 중 50% 범위에 해당하는 것을 확인

Q. 30번 부서 사원 중 급여를 기준으로 내림차순 정렬을 하였을 때 50% 범위에 해당하는 급여는 얼마인가?

```
SQL> SELECT PERCENTILE_CONT(0.5) within GROUP ( ORDER BY sal DESC ) AS CONT ,
            PERCENTILE_DISC(0.5) within GROUP ( ORDER BY sal DESC ) AS DISC
     FROM emp
     WHERE deptno = 30 ;
```

CONT	DISC
1375	1500

PERCENTILE_CONT는 선형 보간법을 이용하여 평균에 근거하는 결과를 보여주므로 실제의 값을 보여주는 PERCENTILE_DISC보다는 정확한 값을 보여주지는 않을 수 있다. 보다 자세한 계산 공식은 매뉴얼 참고 가능 (MEDIAN 함수도 같은 결과 확인 가능하다.)

```
SQL> SELECT MEDIAN(sal)
     FROM emp WHERE deptno = 30 ;
```

MEDIAN(SAL)
1375

MEDIAN은 PERCENTILE_CONT(0.5) 와 같은 결과