

Java Web Programming 입문 12

(Oracle PL/SQL #04)

오늘의 키워드

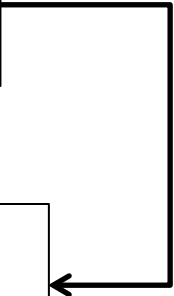
- ▶ 1. 함수 (Function)
 - 형 변환(Type Conversion) 함수
 - Null 제어 함수
 - DECODE 함수
 - CASE
- ▶ 2. 조인 (Join)

형 변환(Type Conversion)

- ▶ 자바에서는 불가능하지만...

```
SQL> SELECT ENAME, SAL  
      FROM EMP  
      WHERE DEPTNO = 10;
```

```
SQL> SELECT ENAME, SAL  
      FROM EMP  
      WHERE DEPTNO = '10';
```



형 변환(Type Conversion)

▶ 형 변환의 종류

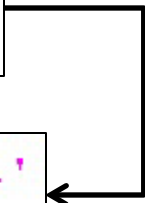
- 암시적 형 변환 (Implicit Type Conversion)
 - 자동으로 형 변환이 일어남
- 명시적 형 변환 (Explicit Type Conversion)
 - 직접 형을 지정해주어 변환을 강제함

형 변환(Type Conversion)

- ▶ 숫자형, 문자형, 날짜형 의 변환

```
SQL> SELECT SYSDATE - HIREDATE  
        FROM EMP ;
```

```
SQL> SELECT SYSDATE - '84/10/01'  
        FROM DUAL ;
```



형 변환(Type Conversion)

▶ 숫자형, 문자형, 날짜형의 변환

<-TO_NUMBER TO_CHAR->		TO_DATE-> <-TO_CHAR		
Number	<->	Character	<->	Date
10		'10'		84/01/01
		'84/01/01'		

▶ TO_변환형(Input, Form)

- Input : 변환될 데이터의 값
- Form : 데이터가 변환될 때의 형식

형 변환(Type Conversion)

▶ 숫자형, 문자형, 날짜형의 변환

```
SQL> SELECT SYSDATE - TO_DATE('1984/10/01', 'YYYY/MM/DD')  
        FROM DUAL;
```

```
SQL> SELECT TO_CHAR(SYSDATE, 'CC YYYY/MM/DD:HH24:MI:SS')  
        FROM DUAL;
```

- CC : CENTURY (21세기)
- HH24 : 24시간으로 표시되는 시간
- MI : 분
- SS : 초

형 변환(Type Conversion)

▶ 숫자형, 문자형, 날짜형의 변환

```
SQL> SELECT ENAME, SAL  
      FROM EMP;
```

```
SQL> SELECT ENAME, TO_CHAR(SAL, '$999,999.99')  
      FROM EMP;
```

```
SQL> SELECT ENAME, TO_CHAR(SAL, 'L999,999.99')  
      FROM EMP;
```


Null 제어

- ▶ Null 에는 “=” 사용이 불가능
 - 수치연산에 Null 값이 들어가면 결과를 알 수 없음

```
SQL> SELECT ENAME, SAL*12 + COMM  
      FROM EMP;
```

- NVL 함수를 사용

```
SQL> SELECT ENAME, SAL*12 + NVL(COMM, 0), COMM  
      FROM EMP;
```

- NVL (COMM, 0)
 - NULL이 아니면 COMM 컬럼의 값, NULL 이면 0

Null 제어

- NVL2 라는 녀석도 있다

```
SQL> SELECT NVL2 (COMM, 1, 0) , NVL (COMM, 0) , COMM  
FROM EMP;
```

- NVL(COMM, 1, 0)
 - COMM이 값을 가지고 있다면 1, NULL 이라면 0
- ▶ Null 값을 처리하는 부분은 눈여겨봐두자

DECODE

- ▶ NULL이 아닌값의 결과를 정해주고 싶을 때

```
SQL> SELECT ENAME, JOB, SAL,  
          DECODE(JOB,  
                'MANAGER', SAL*1.1,  
                'CLERK',   SAL*1.05,  
                SAL) "UPSAL"  
  
FROM EMP;
```

DECODE(JOB 1)	'MANAGER', SAL*1.1 2)	'CLERK', SAL*1.05 3)	SAL 4)	"UPSAL"
------------------	--------------------------	-------------------------	-----------	---------

- 1) Job Column의 값에 대해 다음 조건에 따라 결과값을 반환함을 명시
- 2) Job Column의 값이 MANAGER라면 Row의 SAL Column의 값에서 1.1 을 곱한 값을 반환
- 3) Job Column의 값이 CLERK이라면 Row의 SAL Column의 값에서 1.05 를 곱한 값을 반환
- 4) 그외 나머지 Job Column의 값들은 그냥 SAL 그대로 반환

CASE

▶ CASE문

```
SQL> SELECT ENAME, JOB, SAL,  
          DECODE(JOB, 'MANAGER', SAL*1.1, 'CLERK', SAL*1.05, SAL) "UPSAL",  
          CASE JOB                                -- 1)  
            WHEN 'MANAGER' THEN SAL*1.1          -- 2)  
            WHEN 'CLERK'   THEN SAL*1.05         -- 3)  
            ELSE SAL                                -- 4)  
          END "UPSAL2"                             -- 5)  
FROM EMP;
```

- 1) CASE JOB
 - JOB Column의 값에 따라 수행함을 명시
- 2) WHEN 'MANAGER' THEN SAL*1.1
 - 위에서 명시된 JOB Column의 값이 'MANAGER'일 경우 SAL Column의 값에서 1.1을 곱한값을 반환
- 3) WHEN 'CLERK' THEN SAL*1.05
 - JOB Column의 값이 'CLERK'일 경우 SAL Column의 값에서 1.05를 곱한값을 반환
- 4) ELSE SAL
 - 위의 WHEN 조건을 만족하지 않는 나머지 값들을 가질경우 SAL Column 값을 그대로 반환
- 5) END "UPSAL2"
 - CASE 문을 종료하고 이 CASE문의 결과로 나올 Column 의 ALIAS를 "UPSAL2"로 줌

CASE

▶ CASE문

```
SQL> SELECT ENAME, SAL,  
          CASE  
            WHEN SAL >= 3000 THEN 'O'  
            ELSE 'X'  
          END "OVER_3000"  
        FROM EMP;
```

- CASE에 대상을 명시하지 않고도 사용이 가능

함수 (Function)

▶ 정리

단일 행 함수	복수 행 함수
하나의 행당 하나의 결과값을 반환하는 함수	여러 개의 행당 하나의 결과값을 반환하는 함수
문 자 함수	COUNT
숫 자 함수	SUM
날 자 함수	MIN / MAX
변 환 함수	AVG

함수 (Function)

▶ 문자 함수

구 분	함 수	내 용
문자 함수	LOWER UPPER INITCAP CONCAT SUBSTR LENGTH NVL NVL2 SUBSTR RTRIM LTRIM TRIM RPAD LPAD TRANSLATE REPLACE SOUNDX LENGTH LENGTHB INSTR NULLIF COALESCE	<ul style="list-style-type: none"> · 모든 문자를 소문자로 · 모든 문자를 대문자로 · 첫 글자는 대문자, 나머지는 소문자로 · 첫 번째 문자와 두 번째 문자를 연결 · 문자값 중 해당 위치의 길이 만큼만 리턴할 때 · 문자의 길이를 리턴할 때 · 널값을 다른 값으로 대체할 때 · 조건에 의해 널값을 다른 값으로 대체할 때 · 특정 문자의 문자열중 필요 부분만 선별하여 사용 · 서브 스트링의 정확한 위치와 길이를 요구(오른쪽) · 서브 스트링의 정확한 위치와 길이를 요구(왼쪽) · RTRIM과 LTRIM의 복합기능 · 문자열을 제외한 공간에 지정한 문자열로 대체(오) · 문자열을 제외한 공간에 지정한 문자열로 대체(왼) · 첫 문자는 탐색집합의 첫 문자로 대체(2번째도 동일) · 특정 문자열을 다른 문자열로 대체 · 같은 단어 또는 유사한 사운드 단어를 음성학적으로 · 문자의 실제 길이를 변환할 때 · 문자열의 실제 길이를 변환할 때 · 문자열 내의 특정 스트링의 위치 · 조건이 같으면 NULL, 다르면 지정된 값을 리턴할 때 · 조건에 따라 여러 가지 값을 리턴할 때

함수 (Function)

▶ 숫자 함수

구 분	함 수	내 용
숫자 함수	ROUND TRUNC MOD(m/n) ABS SIGN FLOOR CEIL POWER LOG SIN COS TAN	<ul style="list-style-type: none"> · 해당 소수점 자리에서 반올림할 때 · 해당 소수점 자리에서 절삭할 때 · m을 n으로 나누고 남은 나머지를 리턴할 때 · 숫자 값을 절대값으로 바꾼다. · 숫자가 양수:+1, 음수:-1, 0:0 · 실수값을 정수값으로 · 그 수보다 가장 크거나 작은값을 리턴 · 해당 수에 대한 지수값을 표현 · 로그값으로 변환 · SIN값 · COS값 · TAN값
날짜 함수	SYSDATE ADD_MONTHS LAST_DAY NEW_TIME NEXT_DAY MONTH_BETWEEN	<ul style="list-style-type: none"> · 현재 시스템 날짜를 보여줄 때 · 지정된 날짜에 몇 월을 추가한 결과의 월을 계산할 때 · 해당 월의 마지막 날짜를 알고자 할 때 · 해당 표준시로 시간을 변환할 때 · 해당 날짜의 다음 지정된 날짜로 변환할 때 · 지정된 월 간의 월수를 알고자 할 때

함수 (Function)

▶ 숫자 함수

구 분	함 수	내 용
숫자 함수	ROUND TRUNC MOD(m/n) ABS SIGN FLOOR CEIL POWER LOG SIN COS TAN	<ul style="list-style-type: none"> · 해당 소수점 자리에서 반올림할 때 · 해당 소수점 자리에서 절삭할 때 · m을 n으로 나누고 남은 나머지를 리턴할 때 · 숫자 값을 절대값으로 바꾼다. · 숫자가 양수:+1, 음수:-1, 0:0 · 실수값을 정수값으로 · 그 수보다 가장 크거나 작은값을 리턴 · 해당 수에 대한 지수값을 표현 · 로그값으로 변환 · SIN값 · COS값 · TAN값
날짜 함수	SYSDATE ADD_MONTHS LAST_DAY NEW_TIME NEXT_DAY MONTH_BETWEEN	<ul style="list-style-type: none"> · 현재 시스템 날짜를 보여줄 때 · 지정한 날짜에 몇 월을 추가한 결과의 월을 계산할 때 · 해당 월의 마지막 날짜를 알고자 할 때 · 해당 표준시로 시간을 변환할 때 · 해당 날짜의 다음 지정한 날짜로 변환할 때 · 지정된 월 간의 월수를 알고자 할 때

▶ 시스템 함수

시스템 함수	USER USERID	<ul style="list-style-type: none"> · 현재 DB 사용자 · 현재 DB 사용자에게 할당되는 사용자번호
--------	----------------	---

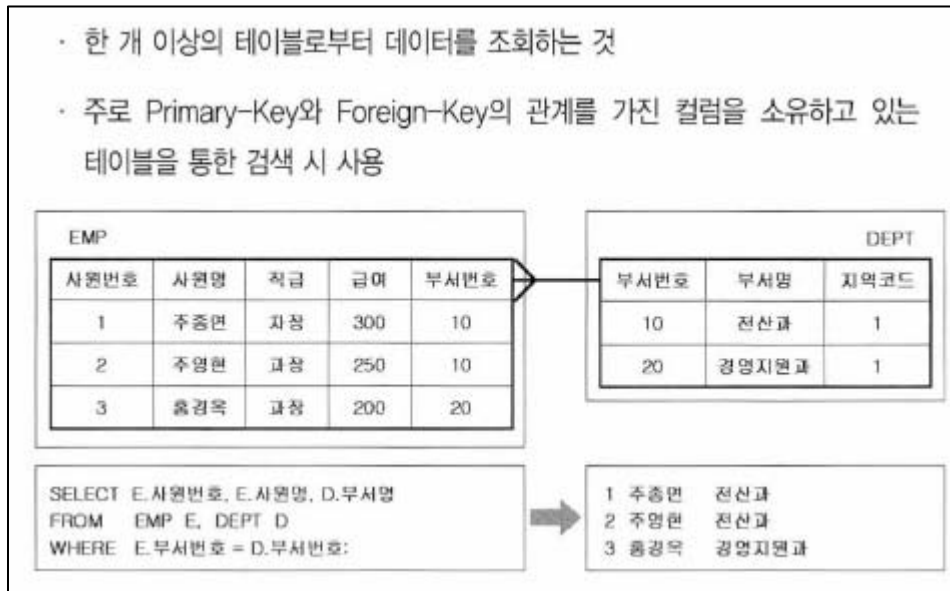
함수 (Function)

▶ 복수 행 함수 (Multiple Row Function)

1) COUNT	조건을 만족하는 모든 행의 수를 보여준다.
2) SUM	조건을 만족하는 모든 행의 합계를 보여준다.
3) AVG	조건을 만족하는 모든 행의 평균을 보여준다.
4) MAX	조건을 만족하는 모든 행의 최대값을 보여준다.
5) MIN	조건을 만족하는 모든 행의 최소값을 보여준다.
6) STDENV	조건을 만족하는 모든 행의 표준 편차를 보여준다.
7) VARIANCE	조건을 만족하는 모든 행의 분산값을 보여준다.

JOIN

- ▶ 한 개 이상의 Table(같은 녀석)로부터 Data를 조회



- ▶ JOIN 을 하지않는 SELECT문은 없다!!
 - (조금은 있다 사실)

JOIN

▶ JOIN을 해보자

```
SQL> SELECT ENAME, DNAME  
       FROM EMP, DEPT  
       ORDER BY ENAME;
```


- Cartesian Product (데카르트 곱)
- Table N, M 의 Row 개수가 곱해진 수만큼 결과 발생

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	1980-12-17	800		20
7499	ALLEN	SALESMAN	7698	1981-02-20	1600	300	30
7521	WARD	SALESMAN	7698	1981-02-22	1250	500	30
7566	JONES	MANAGER	7839	1981-04-02	2975		20
7654	MARTIN	SALESMAN	7698	1981-09-28	1250	1400	30
7698	BLAKE	MANAGER	7839	1981-05-01	2850		30
7782	CLARK	MANAGER	7839	1981-06-09	2450		10
7788	SCOTT	ANALYST	7566	1987-04-19	3000		20
7839	KING	PRESIDENT		1981-11-17	5000		10
7844	TURNER	SALESMAN	7698	1981-09-08	1500	0	30
7876	ADAMS	CLERK	7788	1987-05-23	1100		20
7900	JAMES	CLERK	7698	1981-12-03	950		30
7902	FORD	ANALYST	7566	1981-12-03	3000		20
7934	MILLER	CLERK	7782	1982-01-23	1300		10

JOIN

- ▶ 특수한 경우를 제외하고, JOIN을 사용한 SELECT 문에서 Cartesian Product (데카르트 곱) 결과가 조회되면 맞고 시작하는 거다.
- ▶ 조건이 필요하다. (Query에서 조건을 맡고 있는 WHERE)



```
SQL> SELECT ENAME, DNAME  
      FROM EMP, DEPT  
      ORDER BY ENAME;
```

```
SQL> SELECT ENAME, DNAME  
      FROM EMP, DEPT  
      WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

JOIN

- ▶ JOIN 을 이용한 결과조회시, JOIN에 사용된 각 Table의 컬럼명이 동일한 경우 문제가 생길 수 있다.

```
SQL> SELECT ENAME, DNAME, DEPTNO  
      FROM EMP, DEPT  
      WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

```
SQL> SELECT ENAME, DNAME, EMP.DEPTNO, DEPT.DEPTNO  
      FROM EMP, DEPT  
      WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

JOIN

- ▶ Table에도 별칭(Alias)을 줄 수 있다.

```
SQL> SELECT ENAME, DNAME, EMP.DEPTNO, DEPT.DEPTNO  
       FROM EMP, DEPT  
       WHERE EMP.DEPTNO = DEPT.DEPTNO;
```

```
SQL> SELECT E.ENAME, D.DNAME, E.DEPTNO  
       FROM EMP E, DEPT D  
       WHERE E.DEPTNO = D.DEPTNO; -- JOIN CONDITION
```