**[ II ] Select문 가장 많이 쓰이는 DML(데이터 조작 언어) 질의어**

1. SQL 문장 작성법
   * SQL 문장은 대소문자를 구별하지 않습니다.
   * SQL 문장은 한 줄 또는 여러 줄에 입력될 수 있습니다.
   * SQL에서 문자열이나 날짜는 반드시 작은따옴표안에 표시해야 합니다.
   * 절은 보통 읽고 편집하기 쉽게 줄을 나누도록 합니다.(권장)
   * 탭과 줄 넣기(들여쓰기)는 코드를 보다 읽기 쉽게 하기 위해 사용됩니다.(권장)
   * 일반적으로 키워드는 대문자로 입력합니다
   * 키워드를 제외한 다른 모든 단어, 즉 테이블 이름, 열 이름은 소문자로 입력하기도 합니다.
   * 명령어의 끝은 ;
   * 주석처리는 –
2. SQL 문장 실행
   * SELECT \* FROM TAB; --scott이 소유하고 있는 table을 전부 보여줌
     + scott(사용자)가 소유한 테이블
   * SELECT \* FROM dept; -- dept테이블 데이터 전체 보기
   * DESC dept; -- dept테이블의 구조보기
     + not null : null값이 있으면 안 된다.
     + number(4) : 부서코드는 정수 4자리
     + number(7,2) ; 총자리수 7자리 중 소수점 이하는 2자리
   * SELECT \* FROM emp;
   * SELECT empno, ename, sal, job FROM emp;
   * SELECT empno AS "사번", ename AS "직원 이름", job AS "직업" FROM emp; -- 별명 사용
   * SELECT empno 사번, ename 이름, job 직업 FROM emp;
   * SELECT empno AS NO, ename AS NAME, job FROM emp;
   * SELECT empno, ename, job FROM emp;
3. WHERE 조건과 비교연산자 : 같다(=), 크거나같다(>=) 작거나같다(<=) 다르다(<> ^= !=)

* 날짜포맷은 반드시 다음과 같다는 전제를 한다
* ALTER SESSION SET NLS\_DATE\_FORMAT=’RR/MM/DD’;

1. 논리연산자(AND, OR, NOT)
2. 산술표현식
   * SELECT ename, sal, sal+300 AS “올릴월급” FROM emp;
     + emp 테이블의 sal을 300 증가 시키기 위해 덧셈연산자를 사용하여 sal+300을 출력.

계산된 결과인 열[SAL+300]은 emp 테이블의 새로운 열이 아니고 단지 디스플레이를 위한 것.

* + SELECT ename, sal, comm, sal\*12 월급합, sal\*12+comm 연봉 FROM emp;
    - null값의 처리. 행이 특정열에 대한 데이터값이 없다면 그 값은 null이 됩니다.

null값은 이용할 수 없거나 지정되지 않았거나 알 수 없거나 적용할 수 없는 값. Null은 1이나공백과는 다르다. 0은 숫자이며 공백은 문자로 볼 수 있다. 어떠한 datatype도 null값을 포함할 수 있으나 주키는 null값이 허용되지 않는다.

* + - null값을 포함한 산술표현식 결과는 NULL이 된다. 그래서 다음과 같이 NVL함수 를 이용한다.
  + SELECT ename, sal, comm, sal+comm FROM emp;
  + SELECT ename, sal, com, sal\*12, sal\*12+NVL(comm,0) FROM emp;
    - NVL(expr1, expr2) expr1은 null값을 포함하고 있는 column이나 표현식

expr2는 null변환을 위한 목표값

expr1과 expr2는 같은 타입이여야 한다

* + - NVL(hiredate , ‘99/01/01’) DATA형에서의 예
    - NVL(job, ‘업무없음’) char형에서의 예
  + SELECT ename, sal, NVL(comm, 0) FROM emp;

1. 연결연산자 ( || ) ; 열이나 문자를 다른 열에 연결하는 연산

* SELECT ename || ‘ ‘ || job as “employees” FROM emp;
* SELECT ename || ‘ is a ‘ || job as “employees Details” FROM emp;
* SELECT ename || ‘ : 1 year salary = ‘ || sal\*12 FROM emp;
  + 컬럼명에 별칭은 프로그램 영역에서 중요한 의미를 가진다. 대다수 프로그램은 별칭으로 간편하게 사용하는 경우가 많다. Ex. rs.getString(“별칭/컬럼명”);
* SELECT ‘사번은 ‘ || empno || ‘입니다’ show FROM emp;
  + 실제 어플리케이션에서는 rs.getString(“show”)로 처리
* SELECT ename ||'은 ' || job

FROM emp

WHERE sal>=2000 AND SAL<3000;

1. 중복제거 DISTINCT ; DISTINCT를 사용하여 나타나는 결과는 기본적으로 오름차순 정렬

Ex) SELECT DISTINCT DEPTNO, JOB FROM emp;

(emp테이블에서 deptno별로 job를 한번씩 출력)

1. SQL 연산자
   * DISTINCT 필드; 필드 중복제거

* BETWEEN a AND b : a부터 b (a,b값 포함. a가 작은 값)
* IN(list) : list의 값 중 어느 하나와 일치한다
* LIKE *pattern* : 검색하고자 하는 값을 정확히 모를 경우 pattern과 일치하는지를 검색하며 pattern에는 다음의 두가지 와일드카드가 사용된다.
  + % : **문자가 없거나, 하나 이상의 문자**가 어떤 값이 와도 상관없다.
  + \_ : **하나의 문자**가 어떤 값이 와도 상관없다
* 이름에 %가 들어간 이름
  + SELECT ename FORM emp WHERE ename LIKE ‘%\%%’ ESCAPE ‘\’;
* IS NULL : null값을 가졌다
* NOT BETWEEN a AND b : a와 b 사이에 있지 않다(a,b값 포함되지 않음)
* NOT IN (list) : list의 값과 일치하지 않는다.
* NOT LIKE *pattern*  : 문자형태가 pattern과 일치하지 않는다.
* NOT IS NULL : null값을 갖지 않다
  + SELECT DISTINCT deptno FROM emp; -- 부서코드들 중복제거해서 출력
  + SELECT DISTINCT job FROM emp;
  + SELECT DISTINCT deptno, job FROM emp;
  + SELECT ename, sal FROM emp WHERE sal between 100 and 1500;
  + SELECT empno, ename, job, sal, hiredate FROM emp WHERE empno IN (7902,7788,7566);
* EMP 테이블에서 hiredate가 1982년인 사원의 empno,ename,job,sal,hiredate,deptno 를 출력하시오
  + SELECT ename FROM emp WHERE hiredate >= '82/01/01' AND hiredate <= '82/12/31';
  + SELECT empno,ename,job,sal,hiredate,deptno FROM emp where hiredate>=to\_date('1982/01/01', 'yyyy/mm/dd') and hiredate <= to\_date('1982/12/31', 'yyyy/mm/dd');
  + SELECT empno,ename,job,sal,hiredate,deptno FROM emp where hiredate between to\_date('1982/01/01', 'yyyy/mm/dd') and to\_date('1982/12/31', 'yyyy/mm/dd');

1. 정렬을 위한 ORDER BY 절
   * ASC 오름차순. Default
   * DESC 내림차순
   * (큰, 작은) 값부터 순서대로 출력(같은 값이 있으면 날짜가 나중인 사람 순으로 정함)