```
REM Author :
REM Date :
REM Objective : Chapter 2. Basic SELECT
REM Environment : CentOS 6.5, SQLGate 2010 for Oracle, Oracle 11g Enterprise Edition 11.2.0

REM SELECT의 기능
1. Selection : 조건검색, Row에 대한 필터링
2. Projection : column 에 대한 필터링
3. Join : 여러 테이블에서의 검색

REM SELECT Syntax

    SELECT [DISTINCT | ALL]  {* | column1, column2 [AS [alias]] | expr}
    FROM table_name
    WHERE condition
    ORDER BY column [ASC | DESC];

1. SELECT 절과 FROM 절은 반드시 있어야 한다.
2. SELECT 절 다음에 질의하고 싶은 칼럼을 차례대로 나열한다. 이 때 여러 개의 칼럼 구분은 쉼표(,)로 한다.
3. FROM 절 다음에는 조회할 테이블 이름을 적는다.
4. * : 모든 칼럼을 조회한다.
5. ALL : 모든 결과 ROW를 보여준다.(기본값)
6. DISTINCT : 중복된 ROW 를 제외한 ROW를 보여준다.
7. expr : SQL 함수를 사용하거나, 수학 연산을 포함한 표현식
8. alias : 칼럼에 대한 별칭 사용.
9. Default Column Heading : column 명이 대문자로 Display 된다.
10. Default Data Justification : Number 값은 오른쪽 정렬, Character 와 Date 값은 왼쪽 정렬된다.

REM 모든 열 선택
    SELECT * FROM dept;

    SELECT *
    FROM emp;

    SELECT * FROM emp;

    SQL> SET pagesize 1000   -- 1000줄을 한 페이지로 설정하는 SQL*Plus 명령 실행
    SQL> /  -- SQL*Plus buffer 에 들어있는 SQL 문장을 다시 실행
    SQL> COL[UMN] mgr FOR[MAT] 9999  --숫자 칼럼의 크기를 4자리로 조정하는 SQL*Plus 명령 실행
    SQL> COL ename FORMAT A8  -- 문자칼럼의 크기를 8자리로 조정하는 SQL*Plus 명령 실행

REM 특정 열 선택
1. 각 열의 구분은 "," 로 한다.
    SELECT empno, ename, sal
    FROM emp;

    SELECT empno, ename, job, mgr FROM emp;

REM 산술연산자 : 수학 연산 표현식
1. +, - : 음수, 혹은 양수를 나타내는 기호. 단항 연산자.
2. *(multiply), /(divide) : 곱하기, 나누기를 의미. 이항 연산자.
3. +(add), -(subtract) : 더하기, 빼기를 의미. 이항 연산자.
4. 연산자의 우선순위가 있다.   1 --> 2 --> 3
5. 우선순위가 높은 연산 먼저 수행하며, 같은 우선순위의 연산자들을 왼쪽에서 오른쪽으로 순서대로 계산해 나간다.
6. 괄호를 사용하여 우선순위를 조절할 수 있다.

    SELECT empno, ename, sal, sal + 100
    FROM emp;

    SELECT sal, -sal FROM emp;
    SELECT sal, sal * 1.1 FROM emp;
    SELECT sal, comm, sal + comm FROM emp;
    SELECT sal, -sal + 100 * -2 FROM emp;
    SELECT sal, (-sal + 100) * -2 FROM emp;
    SELECT empno, ename, sal, sal * 12 FROM emp;
    SELECT empno, ename, sal, sal * 12 + comm FROM emp;
    SELECT empno, ename, sal, sal + comm * 12 FROM emp;
    SELECT empno, ename, sal, (sal + comm) * 12 FROM emp;
```

```
REM NULL value
1. NULL 이란?
   1) 특정 행, 특정 열에 대한 아직 값을 알 수 없는 상태, 의미가 없는 상태를 표현
   2) 이용할 수 없거나, 지정되지 않거나, 알 수 없거나, 적용할 수 없는 값
2. 0 또는 공백과 다르다.
3. 연산의 대상에 포함되지 않는다.
4. NULL 값을 포함한 산술 연산 식의 결과는 언제나 NULL 이다.
5. NOT NULL 또는 Primary Key 제약조건이 걸린 칼럼에서는 NULL value가 나타날 수 없다.
6. Oracle DB에서는 NULL 인 칼럼은 Length 가 0 이므로 data를 위한 물리적 공간을 차지하지 않는다.

    SELECT empno, job, comm FROM emp;
    --NULL 인 값은 비어있는 것으로 표현된다. job이 salesman인 사원들에게만 커미션이 적용되며, 사번 7844인 사원의 커미션은 0이다.

    SELECT empno, ename, sal * 12 + comm
    FROM emp;
    --comm 값이 NULL 인 경우 연봉은 얼마인가? 연봉 계산한 수식의 column heading은 어떻게 나타나는가?
    --comm 값이 NULL 인 row 의 경우 (sal + comm) * 12를 하면 결과도 모두 NULL 이 된다.
    --또한, expression 전체가 column heading 으로 나타난다.

REM NVL function
1. NULL 값을 어떤 특정한 값으로 치환할 때 사용
2. 치환할 수 있는 값의 형태는 숫자형, 문자형, 날자형 모두 가능
3. 치환된 값은 expr1칼럼의 데이타 타입과 일치해야 한다.
4. Syntax
    NVL(expr1, expr2)
        --expr1 : NULL
        --expr2 : 치환값
        --expr1값이 NULL 아니면 expr1 값을 그대로 사용
        --만약 expr1 값이 NULL이면, expr2 값으로 대체
5. 예
    NVL(comm, 0)
    NVL(hiredate, '12/09/04')
    NVL(job, 'No Job')

    --위에서 연봉을 구하는 Query 를 NVL 함수를 사용하여 제대로 나올 수 있도록 고쳐보자.
    SELECT empno, ename, sal, comm, sal * 12 + NVL(comm, 0)
    FROM emp;

    SELECT empno, comm, NVL(comm, 0)
    FROM emp;

    --다음은 매니저가 없는, 즉 최고 직급의 사원인 경우 'No Manager'라고 출력되도록 하는 문장이다. 실행하여 메시지를 적어보고, Error 가 나는 이유를 설명하시오.
    SQL> SELECT NVL(mgr, 'No Manager') FROM emp;

REM Alias 별칭
1. column header 에 별칭을 부여 할 수 있다.
2. SELECT 절에 expression 을 사용할 때 도움이 된다.
3. 열 이름 바로 뒤에 기술한다. 또는 열이름과 별칭 사이에 AS를 사용할 수 있다.
4. 별칭에 공백이나 특수문자나 한글사용할 때, 대소문자를 기술할 때(기본값은 모두 대문자)에는 "" 로 기술한다.

    SELECT empno 사번 FROM emp;
    SELECT sal * 12 연봉 FROM emp;
    SELECT sal * 12 annual_salary FROM emp;
    SELECT sal * 12 Annual_Salary FROM emp;
    SELECT sal * 12 Annual Salary FROM emp;  --Error 발생
    SELECT ename  "Name" , sal AS "Salary", sal * 12 + NVL(comm, 0)  AS "Annual Salary"
    FROM emp;

REM Concatenation Operator (연결 연산자)
1. 문자열 리터럴을 이을 때에는 '||' 를 사용한다.
2. 연결연산자(||)는 character string 들을 여결하여 하나의 결과 string 을 만들어 낸다.

    SELECT empno || ename FROM emp;
    SELECT empno || ename || hiredate FROM emp;
```

```
    --number 이나 date값은 default 형태의 character 값으로 자동 변환한 후 연결된다.

REM Literals (상수)
1. Literal 은 상수 값을 의미.
2. Character literal 은 작은 따옴표로 묶고, Number literal 은 따옴표 없이 표현한다.
3. Character literal을 작은 따옴표로 묶어 주어야 Oracle Server 는 keyword나 객체 이름을 구별할 수 있다.

    SELECT 'Emp# of ' || ename || ' is ' empno FROM emp;
    SELECT dname || ' is located at ' || loc FROM dept;
    SELECT ename || ' is a ' || job AS "Employee" FROM emp;
    SELECT ename || ' ' || sal FROM emp;
    SELECT ename || ' is working as a ' || job FROM emp;
    SELECT 'Java is a language.' FROM emp;  --14번 출력
    SELECT 'Java is a language.' FROM dept;  --4번 출력
    SELECT 'Java is a language.' FROM dual;

REM Duplicate Values(중복 행 제거하기)
1. 일반  Query는 ALL 을 사용하기 때문에 중복된 행이 출력된다.
2. DISTINCT 를 사용하면 중복된 행의 값을 제거한다.
3. DISTINCT 는 SELECT 바로 뒤에 기술한다.
4. DISTINCT 다음에 나타나는 column은 모두 DISTINCT 에 영향을 받는다.

    SELECT job FROM emp;
    SELECT ALL job FROM emp;
    SELECT DISTINCT job FROM emp;
    SELECT deptno FROM emp;
    SELECT DISTINCT deptno FROM emp;
    SELECT deptno, job FROM emp;
    SELECT DISTINCT deptno, job FROM emp;

REM WHERE 절
1. Syntax

    SELECT column...
    FROM table_name
    WHERE conditions;

2. WHERE 절을 사용하지 않으면 FROM 절에 명시된 table의 모든 row를 조회하게 된다.
3. table내의 특정 row만 선택하고 싶을 때 WHERE 절에 조건식을 사용한다.
4. Oracle Server 는 table의 row를 하나씩 읽어 WHERE 절의 조건식을 평가하여 TRUE로 만족하는 것만을 선택한다.
5. condition을 평가한 결과는 TRUE, FALSE, NULL 중의 하나이다.
6. condition 내에서 character 와 date 값의 literal은 작은 따옴표를 사용하고, number 값은 그대로 사용한다.
7. condition 에서 사용하는 character 값은 대소문자를 구별한다.
   1) WHERE ename = 'JAMES';
   2) WHERE ename = 'james';
8. date 타입은 현재 session 의 NLS_DATE_FORMAT 에 맞춰 표현한다.
9. WHERE 는 FROM 다음에 와야 한다.

/*
    날짜 형식 바꾸기
    ALTER SESSION SET NLS_DATE_FORMAT='YYYY-MM-DD';
    SELECT value FROM NLS_SESSION_PARAMETERS
    WHERE parameter = 'NLS_DATE_FORMAT';
*/

REM 비교연산자
--<, >, <=, >=, =, !=, <>(같지 않다)

    --직위가 CLERK 인 사원의 이름과 직위 및 부서번호를 출력하시오.
    SELECT ename, job, deptno
    FROM emp
    WHERE job = 'CLERK';

    SELECT empno, ename, job
    FROM emp
    WHERE empno = 7934;

    SELECT empno, ename, job, hiredate
    FROM emp
    WHERE hiredate = '1981-12-03';
```

```
204
205     SELECT empno, ename
206     FROM emp
207     WHERE ename = 'JAMES';
208
209     SELECT empno, ename
210     FROM emp
211     WHERE ename = 'james';
212
213     SELECT dname
214     FROM dept
215     WHERE deptno = 30;
216
217     SELECT ename, sal
218     FROM emp
219     WHERE sal >= 1500;
220
221     --1983년 이후에 입사한 사원의 사번, 이름, 입사일을 출력하시오.
222     SELECT empno, ename, hiredate
223     FROM emp
224     WHERE hiredate >= '1983-01-01';    //01-JAN-83
225
226     --급여가 보너스(comm) 이하인 사원의 이름, 급여 및 보너스를 출력하시오
227     SELECT ename, sal, comm
228     FROM emp
229     WHERE sal <= NVL(comm, 0);
230
231     --10번 부서의 모든 사람들에게 급여의 13%를 보너스로 지급하기로 했다. 이름,
        급여, 보너스 금액, 부서번호를 출력하시오.
232     SELECT ename, sal, sal * 0.13, deptno
233     FROM emp
234     WHERE deptno = 10;
235
236     --30번 부서의 연봉을 계산하여, 이름, 부서번호, 급여, 연봉을 출력하라. 단,
        년말에 급여의 150%를 보너스로 지급한다.
237     SELECT ename, deptno, sal, sal * 12 + NVL(comm, 0) + sal * 1.5 AS "년봉"
238     FROM emp
239     WHERE deptno = 30;
240
241     --부서번호가 20인 부서의 시간당 임금을 계산하시오. 단, 1달의 근무일수는
        12일이고, 1일 근무시간은 5시간이다. 출력양식은 이름, 급여, 시간당 임금을 출력하라.
242     SELECT ename, sal, sal / 12 / 5
243     FROM emp
244     WHERE deptno = 20;
245
246     --모든 사원의 실수령액을 계산하여 출력하시오. 단, 이름, 급여, 실수령액을
        출력하시오. (실수령액은 급여에 대해 10%의 세금을 뺀 금액)
247     SELECT ename, sal, sal - sal * 0.1 AS "실수령액"
248     FROM emp;
249
250     --사번이 7788인 사원의 이름과 급여를 출력하시오.
251     --급여가 3000이 넘는 직종을 선택하시오.
252     --PRESIDENT를 제외한 사원들의 이름과 직종을 출력하시오.
253     --BOSTON 지역에 있는 부서이 번호와 이름을 출력하시오.
254
255 REM 논리연산자
256 --AND, OR, NOT
257
258     --사원테이블에서 급여가 1000불이상이고, 부서번호가 30번인 사원의 사원번호,
        성명, 담당업무, 급여, 부서번호를 출력하시오.
259     SELECT empno, ename, job, sal, deptno
260     FROM emp
261     WHERE sal >= 1000 AND deptno = 30;
262
263     --사원테이블에서 급여가 2000불이상이거나 담당업무가 매니저인 사원의 정보중
        사원번호, 이름, 급여, 업무를 출력하시오.
264     SELECT empno, ename, sal, job
265     FROM emp
266     WHERE sal >= 2000  OR job = 'MANAGER';
267
268 REM SQL 연산자
269 1. BETWEEN A AND B : A보다 같거나 크고, B보다 작거나 같은
270 2. IN(list) : list 안에 있는 멤버들과 같은
```

3. **ANY(list)** : list에 있는 어느 한 멤버와 값을 비교, 반드시 **=, !=, <, >, <=, >=** 등과 함께 사용한다.
4. **ALL(list)** : list에 있는 모든 멤버와 값을 비교, 반드시 **=, !=, <, >, <=, >=** 등과 함께 사용한다.
5. A **LIKE** B **[ESCAPE** 'C'**]**: A가 B의 **패**턴과 **일**치하면 **TRUE**, 보통 **%, _** 연산자와 같이 사용, **escape** 을 사용하면 B의 **패**턴 중에서 C**를** 상수로 취급한다.
6. **IS NULL / IS NOT NULL : NULL** 여부**를** 테스트

1)**BETWEEN** A **AND** B
--사원테이블에서 월급이 1300불에서 1500불까지의 사원정보중 성명, 담당업무, 월급을 출력하시오.

```
SELECT ename, job, sal
FROM emp
WHERE sal >= 1300 AND sal <= 1500;
WHERE sal BETWEEN 1300 AND 1500;

SELECt ename, job, sal
FROM emp
WHERE sal BETWEEN 1500 AND 1300;
--반드시 작은 값이 먼저 나와야 한다.

SELECT ename FROM emp
WHERE hiredate BETWEEN '1982-01-01' AND '1982-12-31';

--급여가 2000 에서 3000 사이인 사원을 출력하시오.
SELECT ename, job, sal FROM emp
WHERE sal BETWEEN 2000 AND 3000;
```

2) **IN**

```
--사원테이블에서 업무가 회사원, 매니저, 분석가인 사원의 이름, 업무를 출력하시오.
SELECT ename AS "이름",  job
FROM emp
WHERE job = 'CLERK' OR job = 'MANAGER' OR job = 'ANALYST';
WHERE job IN('CLERK', 'MANAGER', 'ANALYST');

--관리자의 사원번호가 7902, 7566, 7788인 모든 사원의 사원번호, 이름, 급여 및 관리자의 사원번호를 출력하시오.

SELECT dname FROM emp WHERE deptno IN(10,20);

--ANY와 ALL연산자의 앞에는 비교연산자가 반드시 함께 사용되어야 한다.
--IN 연산자는 =ANY와 같다.
--NOT IN 연산자는 <>ALL 연산자와 같은 결과이다.

SELECT ename FROM emp
WHERE job IN('ANALYST', 'CLERK')
WHERE job =ANY('ANALYST', 'CLERK');

SELECT ename FROM emp
WHERE sal <ALL(2000, 3000);

--BOSTON 이나 DALLAS 에 위치한 부서를 출력하시오.
SELECT dname, loc FROM dept
WHERE loc IN('BOSTON', 'DALLAS');
WHERE log =ANY('BOSTON', 'DALLAS');

--30, 40번 부서에 속하지 않는 사원들을 출력하시오.
SELECT ename, deptno FROM emp
WHERE deptno NOT IN(30,40);
WHERE deptno <>ALL(30,40);

--DALLAS 의 20번 부서, 또는 CHICAGO의 30번 부서를 출력하시오.
SELECT * FROM dept
WHERE (deptno, loc) IN )(20, 'DALLAS'), (30, 'CHICAGO'));
```

3) **LIKE(%, _)**

```
--Wildcard : %(0개 이상의 문자 대표),  _ (1개의 문자 대표)
--Wildcard 문자를 일반 문자로 사용하고 싶을 때 ESCAPE 을 사용한다.
--ESCAPE 문자 바로 뒤에 사용된 Wildcard 문자는 일반 문자로 취급한다.

SELECT ename, job, hiredate FROM emp
WHERE hiredate LIKE '87%';
WHERE hiredate >= '87/01/01';
```

```
339
340         SELECT dname FROM dept
341         WHERE dname LIKE 'A%';
342
343         --이름이 A로 시작하는 사원을 출력하시오.
344         SELECT ename FROM emp
345         WHERE ename LIKE 'A%';
346
347         --사번이 8번으로 끝나는 사원을 출력하시오.
348         SELECT empno, ename FROM emp
349         WHERE empno LIKE '%8';
350
351         --1982에 입사한 사원을 출력하시오.
352         SELECT ename, hiredate FROM emp
353         WHERE hiredate LIKE '1982%';
354         WHERE hiredate BETWEEN '1982-01-01' AND '1982-12-31';
355         WHERE hiredate >= '1982-01-01' AND hiredate <= '1982-12-31';
356
357         SELECT empno, ename
358         FROM emp
359         WHERE ename LIKE 'MILLE_';
360
361         SELECT empno, ename
362         FROM emp
363         WHERE ename LIKE '%$_TEST' ESCAPE '$';
364
365       4) IS NULL / IS NOT NULL
366         --column 의 NULL 여부를 판단할 때에는 반드시 'IS NULL' 혹은 'IS NOT NULL'
          연산자를 사용한다.
367         SELECT ename FROM emp
368         WHERE comm IS NULL;
369         WHERE comm IS NOT NULL;
370
371         --comm 지급 대상인 사원을 출력하시오.
372         SELECT ename, comm FROM emp
373         WHERE comm IS NOT NULL;
374
375         SELECT ename, mgr
376         FROM emp
377         WHERE mgr IS NULL;
378
379     REM 연산자 우선순위
380     1. +, -, 괄호 ()
381     2. *, / : 산술연산자
382     3. +, - : 산술연산자,  ||
383     4. =, <>, <, >, <=, >=, IS NULL, LIKE, BETWEEN, IN
384     5. NOT
385     6. AND
386     7. OR
387
388         SELECT ename, job FROM emp
389         WHERE NOT job = 'ANALYST';
390
391         SELECT ename, sa, deptno FROM emp
392         WHERE sal > 2500 AND deptno = 20;
393
394         SELECT deptno, dname FROM dept
395         WHERE deptno = 10 OR deptno = 20;
396
397         --업무가 SALESMAN 이거나 업무가 MANAGER 이고, 급여가 1300불이상인 사람의
          사원번호, 이름, 업무, 급여를 출력하시오.
398         SELECT empno, ename, job, sal
399         FROM emp
400         WHERE (job LIKE 'S%' OR job LIKE 'M%') AND sal >= 1300;
401
402         --직종이 CLERK 인 사원 중에서 급여가 1000 이상인 사원을 출력하시오.
403         SELECT ename, job, sal FROM emp
404         WHERE job = 'CLERK' AND sal >= 1000;
405
406     REM ORDER BY
407     1. 기본적으로 오라클의 데이터는 정렬되지 않는다.
408     2. 같은 쿼리를 수행할 때마다 결과가 다르게 나올 수 있다.
409     3. 별칭을 사용할 수 있다.
```

4. Syntax

```
SELECT column_list
FROM table
[WHERE conditions]
[ORDER BY column[, column] {ASC | DESC};
```

5. 특징
   1) 기본적으로 오름차순정렬한다.
      --숫자인경우 (1 --> 999)
      --날짜인경우 (옛날 --> 최근)
      --문자인겨우 (알파벳순서, 유니코드순)
   2) NULL 은 오름차순일 경우는 제일 마지막에, 내림차순인 경우에는 제일 처음에 출력
6. ORDER BY 절에 정렬의 기준이 되는 column 을 여러개 지정할 수 있다. 첫번째 column 으로 정렬한 다음, 그 column 값이 같은 row 들에 대해서는 두 번째 column 값으로 정렬한다.
7. 오름차순(ASC) 정렬이 기본이며, 내림차순으로 정렬하고자 할 때에는 DESC를 사용한다.
8. ORDER BY 절에 column 이름 대신 positional notation 을 사용할 수도 있다. Position number 는 SELECT 절에서의 column 순서를 의미한다.

```
--입사일자 순으로 정렬하여 사원번호, 이름, 입사일자를 출력하시오.
SELECT empno, ename, hiredate
FROM emp
ORDER BY hiredate DESC;

--부서번호가 20번인 사원의 연봉 오름차순으로 출력하시오.
SELECT empno, ename, sal, comm, sal * 12 + NVL(comm, 0) AS "Annual"
FROM emp WHERE deptno = 20
ORDER BY "Annual" ASC;

--부서번호로 정렬한 후, 부서번호가 같을 경우 급여가 많은 순으로 사원번호,
사원이름, 업무, 부서번호, 급여를 출력하시오.
SELECT empno, ename, job, deptno, sal
FROM emp
ORDER BY deptno ASC, sal DESC;
```