DML : Data Manipulation Language

데이터 조작어(Data Manipulation Language)

데이터 조작어(이하 DML)는 데이터베이스에 저장된 데이터에 대한 질의, 삽입, 갱신, 삭제를 수행하기 위한 SQL 문장

명령어	설명
SELECT	데이터를 조회한다.
INSERT	데이터를 삽입한다.
UPDATE	데이터를 변경한다.
DELETE	데이터를 삭제한다.

DISPLAY TABLE STRUCTURE

칼럼의 이름과 데이터 타입을 포함한 테이블 구조 확인

Syntax

DESC[RIBE] *tablename*

Example

```
SQL> DESCRIBE s_emp;
```

```
COLUMN_NAME                                TYPE            CONSTRAINT
---------------------------------------------------------- ----------------- --------------------------------
EMPNO                                      NUMBER(4)       PRIMARY KEY
ENAME                                      VARCHAR(10)
JOB                                        VARCHAR(9)
MGR                                        NUMBER(4)
HIREDATE                                   DATE
SAL                                        NUMBER(7,2)
COMM                                       NUMBER(7,2)
DEPTNO                                     NUMBER(2)       REFERENTIAL


INDEX_NAME                      TYPE               COLUMN_NAME
------------------------------------------- ------------------------ ----------------------------------------
PK_EMP                          NORMAL             EMPNO
```

Display Data with the SELECT Statement

SQL SELECT 문을 사용하여 데이터베이스 테이블의 데이터를 조회할 수 있다.

Syntax

```
SELECT column_informantion
FROM table(s)
WHERE condition
ORDER BY expression or keyword
```

where	SELECT	검색할 열, 식 또는 상수를 지정
	FROM	데이터를 가지고 올 테이블 지정
	WHERE	특정 행을 검색할 기준(선택사항)
	ORDER BY	조회된 데이터를 정렬

DISPLAY ALL DATA IN A TABLE

테이블의 모든 칼럼을 출력하려면 **SELECT** 키워드에 '*'를 입력

Syntax

```
SELECT *
FROM table_name
```

Example

```
SQL> SELECT *
  2  FROM s_dept;
```

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

4 rows selected.
```

DISPLAY ALL DATA IN A TABLE

조회하고 싶은 칼럼 이름과 해당 칼럼이 정의된 테이블을 입력하여 조회

Syntax

```
SELECT [DISTINCT] column_name [, column_name]
FROM table_name
```

Example

```
SQL> SELECT dname
  2     FROM s_dept;
```

```
DNAME
--------------
ACCOUNTING
RESEARCH
SALES
OPERATIONS

4 rows selected.
```

DISPLAY ALL DATA IN A TABLE

S_EMP 테이블의 칼럼과 구조를 조회하고, 사원 이름, 급여를 출력

Example

```
SQL> SELECT dname
  2      FROM s_dept;
```

```
SQL> DESC s_emp;

SQL> SELECT ename, sal
  2 FROM s_emp;

ENAME            SAL
---------- ----------
SMITH            800
ALLEN           1600
WARD            1250
JONES           2975
MARTIN          1250
BLAKE           2850
CLARK           2450
SCOTT           3000
KING            5000
TURNER          1500
ADAMS           1100
JAMES            950
FORD            3000
```

DISPLAY UNIQUE COLUMNS OF DATA

DISTINCT 절을 사용하여 고유한 데이터 행을 조회한다. **DISTINCT** 절은 **SELECT** 문 결과가 반환되기 전 중복된 행을 제거한다.

Syntax

```
SELECT [DISTINCT] column_name [, column_name]
FROM table_name
```

Example

```
SQL> SELECT job
  2 FROM s_emp;
JOB
---------
CLERK
SALESMAN
SALESMAN
MANAGER
SALESMAN
MANAGER
MANAGER
ANALYST
PRESIDENT
SALESMAN
CLERK
CLERK
ANALYST
STUDENT
```

```
SQL> SELECT DISTINCT job
  2 FROM s_emp;

JOB
---------
CLERK
ANALYST
STUDENT
PRESIDENT
SALESMAN
MANAGER
```

DISPLAY SPECIFIC COLUMN NAMES

별칭 - **SELECT** 문에서 칼럼 이름을 대체로 정의할 수 있다.

별칭에 공백, 특수문자가 포함되어 있거나, 대소문자 구분이 필요하면 별칭을 큰 따음표로 묶는다.

Example

```
SQL> SELECT DISTINCT job "Title"
  2 FROM s_emp;

Title
---------
CLERK
ANALYST
STUDENT
PRESIDENT
SALESMAN
MANAGER
```

```
SQL> SELECT ename EMPLOYEES
  2 FROM s_emp;

EMPLOYEES
----------
SMITH
ALLEN
WARD
JONES
MARTIN
BLAKE
CLARK
SCOTT
KING
TURNER
ADAMS
JAMES
FORD
```

DISPLAY SPECIFIC ROWS OF DATA

WHERE절을 사용해 조건에 맞는 행을 조회한다.

Syntax

```
SELECT { * | column_name [, column_name...] }
FROM table_name
WHERE condition
```

Example

```
SQL> SELECT ename
  2     FROM s_emp
  3     WHERE sal > 2000;
```

```
ENAME
----------
KING
BLAKE
CLARK
JONES
FORD
SCOTT

6 rows selected.
```

DISPLAY SPECIFIC ROWS OF DATA

Comparison Operators Overview

=	Equal to
<>	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to

DISPLAY SPECIFIC ROWS OF DATA

Comparison Operators Overview

BETWEEN…AND…	between two values
NOT BETWEEN… AND…	Not between two values
IN (list)	Equal to any member of the following list
NOT IN (list)	Not Equal to any member of the following list
LIKE	Match a character pattern using wildcard characters
IS NULL	Is a null
IN NOT NULL	Is not a null

DISPLAY SPECIFIC COLUMN NAMES

=

부서 번호가 **20**인 직원의 부서 번호, 사원 이름, 직무를 조회

Example

```
SQL> SELECT deptno, ename, job
  2     FROM s_emp
  3     WHERE deptno = 20;

DEPTNO ENAME      JOB
---------- ---------- ---------
        20 JONES      MANAGER
        20 FORD       ANALYST
        20 SMITH      CLERK
        20 SCOTT      ANALYST
        20 ADAMS      CLERK
```

DISPLAY SPECIFIC COLUMN NAMES

=

BLAKE의 사원 번호, 이름, 직무, 급여를 조회

Example

```
SQL> SELECT empno, ename, job, sal
  2 FROM s_emp
  3 WHERE ename = 'Blake';

0 row selected.

SQL> SELECT empno, ename, job, sal
  2 FROM s_emp
  3 WHERE ename = 'BLAKE';

    EMPNO ENAME      JOB              SAL
---------- ---------- --------- ----------
     7698 BLAKE      MANAGER          2850

1 row selected.
```

DISPLAY SPECIFIC COLUMN NAMES

=

입사 날짜가 **82/12/22**일인 사원의 이름, 부서 번호, 입사 날짜 조회

Example

```
SQL> SELECT ename, deptno, hiredate
  2  FROM s_emp
  3  WHERE hiredate = '82/12/22;

ENAME      DEPTNO HIREDATE
---------- ---------- ---------------------------------------------------------------
SCOTT          20 0082/12/22
```

DISPLAY SPECIFIC COLUMN NAMES

<>

직무가 **MANAGER**가 아닌 사원의 이름, 직무, 급여 조회

Example

```
SQL> SELECT ename, job, sal
  2     FROM s_emp
  3     WHERE job <> 'MANAGER';
```

```
ENAME      JOB            SAL
---------- --------- ----------
KING       PRESIDENT      5000
MARTIN     SALESMAN       1250
ALLEN      SALESMAN       1600
TURNER     SALESMAN       1500
JAMES      CLERK           950
WARD       SALESMAN       1250
FORD       ANALYST        3000
SMITH      CLERK           800
SCOTT      ANALYST        3000
ADAMS      CLERK          1100
MILLER     CLERK          1300

11 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

>

입사 날짜가 **82/11/01**보다 큰 사원의 이름, 입사 날짜 조회

Example

```
SQL> SELECT ename, hiredate
  2     FROM s_emp
  3     WHERE hiredate > '82/11/01';
ENAME      HIREDATE
---------- ----------------------------------------------------------------
SCOTT      82/12/22
ADAMS      83/01/15
```

급여가 **2,000**이상인 사원의 이름, 급여 조회

```
SQL> SELECT ename, sal
  2     FROM s_emp
  3     WHERE sal > 2000;
ENAME           SAL
---------- ----------
KING          5000
BLAKE         2850
CLARK         2450
JONES         2975
FORD          3000
SCOTT         3000
```

DISPLAY SPECIFIC COLUMN NAMES

>=

부서 번호가 **30**이상인 사원의 부서 번호, 이름 조회

Example

```
SQL> SELECT deptno, ename
  2 FROM s_emp
  3 WHERE deptno >= 30;

DEPTNO ENAME
---------- ----------
        30 BLAKE
        30 MARTIN
        30 ALLEN
        30 TURNER
        30 JAMES
        30 WARD
```

이름이 **SCOTT** 이상인 사원의 이름과 직무 조회

```
SQL> SELECT ename, job
  2 FROM s_emp
  3 WHERE ename >= 'SCOTT';

ENAME      JOB
---------- ---------
TURNER     SALESMAN
WARD       SALESMAN
SMITH      CLERK
SCOTT      ANALYST
```

DISPLAY SPECIFIC COLUMN NAMES

<

입사 날짜가 **81/05/01** 보다 작은 사원의 이름, 부서 번호, 급여, 입사 날짜 조회

Example

```
SQL> SELECT deptno, ename, sal, hiredate
  2 FROM s_emp
  3 WHERE hiredate < '81/05/01';
```

```
 DEPTNO ENAME          SAL HIREDATE
---------- ---------- ---------- --------------------------------------------------------------------
        20 JONES            2975 81/04/01
        30 ALLEN            1600 81/02/11
        30 WARD             1250 81/02/23
        20 SMITH             800 80/12/09

4 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

<=

부서 번호가 **10**이하인 사원의 이름, 부서 번호, 직무 조회

Example

```
SQL> SELECT deptno, ename, job
  2 FROM emp
  3 WHERE deptno <= 10;
```

```
 DEPTNO ENAME      JOB
---------- ---------- ---------
        10 KING       PRESIDENT
        10 CLARK      MANAGER
        10 MILLER     CLERK

3 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

BETWEEN

입사 날짜가 **80/12/01**과 **81/03/01**사이에 있는 사원의 이름과 입사 날짜 조회

Example

```
SQL> SELECT ename, hiredate
  2 FROM s_emp
  3 WHERE hiredate BETWEEN '80/12/01' AND '81/03/01';
ENAME      HIREDATE
---------- ---------------------------------------------------------------
ALLEN      81/02/11
WARD       81/02/23
SMITH      80/12/09
```

급여가 **1,500**과 **2,000** 사이인 사원의 이름과 급여 조회

```
SQL> SELECT ename, sal
  2 FROM s_emp
  3 WHERE sal BETWEEN 1500 AND 2000;

ENAME           SAL
---------- ----------
ALLEN          1600
TURNER          1500

2 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

NOT

BETWEEN

부서 번호가 **10**과 **20** 사이에 있지 않은 부서 번호 조회

Example

```
SQL> SELECT *
  2 FROM s_dept
  3 WHERE deptno NOT BETWEEN 10 AND 20;

 DEPTNO DNAME         LOC
---------- -------------- -------------
       30 SALES          CHICAGO
       40 OPERATIONS     BOSTON
```

사원 이름이 **ALLEN**과 **SMITH** 사이에 있지 않은 사원 이름과 부서 번호 조회

```
SQL> SELECT ename, deptno
  2 FROM s_emp
  3 WHERE ename NOT BETWEEN 'ALLEN' AND 'SMITH';

ENAME        DEPTNO
---------- ----------
TURNER           30
WARD            30
ADAMS            20
```

DISPLAY SPECIFIC COLUMN NAMES

IN

직무가 **MANAGER, SALESMAN**인 사원 이름, 직무, 부서 번호 조회

Example

```
SQL> SELECT ename, job, deptno
  2 FROM s_emp
  3 WHERE job IN ('MANAGER', 'SALESMAN');
```

```
ENAME      JOB          DEPTNO
---------- --------- ----------
BLAKE      MANAGER          30
CLARK      MANAGER          10
JONES      MANAGER          20
MARTIN     SALESMAN         30
ALLEN      SALESMAN         30
TURNER     SALESMAN         30
WARD       SALESMAN         30

7 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

NOT IN

부서 번호가 **10, 20**이 아닌 사원의 사원 번호, 이름, 부서 번호 조회

Example

```
SQL> SELECT empno, ename, deptno
  2 FROM s_emp
  3 WHERE deptno NOT IN (10,20);

 EMPNO ENAME      DEPTNO
---------- ---------- ----------
      7698 BLAKE          30
      7654 MARTIN         30
      7499 ALLEN          30
      7844 TURNER         30
      7900 JAMES          30
      7521 WARD           30

6 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

LIKE

사원 이름이 **'M'**으로 시작하는 사원 이름 조회

Example

```
SQL> SELECT ename
  2 FROM s_emp
  3 WHERE ename LIKE 'M%';

ENAME
----------
MARTIN
MILLER
```

입사 날짜가 **'01'**로 끝나는 사원 이름, 입사 날짜 조회

```
SQL> SELECT ename, hiredate
  2 FROM s_emp
  3 WHERE hiredate LIKE '%01';

ENAME      HIREDATE
---------- ---------------------------------------------------------------
BLAKE      81/05/01
JONES      81/04/01
```

DISPLAY SPECIFIC COLUMN NAMES

LIKE

사원 이름에 'A'를 포함하는 사원 조회

Example

```
SQL> SELECT ename
  2 FROM s_emp
  3 WHERE ename LIKE '%A%';

ENAME
----------
BLAKE
CLARK
MARTIN
ALLEN
JAMES
WARD
ADAMS

7 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

LIKE

사원 이름이 **'B'**로 시작해서 **'F'**끝나는 다섯 글자의 사원 이름 조회

Example

```
SQL> SELECT ename
  2 FROM s_emp
  3 WHERE ename LIKE 'B___E';
ENAME
----------
BLAKE
```

두 번째 문자가 **'L'**인 사원의 이름 조회

```
SQL> SELECT ename
  2 FROM s_emp
  3 WHERE ename LIKE '_L%';

ENAME
----------
BLAKE
CLARK
ALLEN

3 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

IS NULL

커미션이 **NULL** 값인 사원의 이름, 직무, 급여 조회

Example

```
SQL> SELECT ename, job, sal
  2 FROM s_emp
  3 WHERE comm IS NULL;

ENAME      JOB            SAL
---------- --------- ----------
KING       PRESIDENT      5000
BLAKE      MANAGER        2850
CLARK      MANAGER        2450
JONES      MANAGER        2975
JAMES      CLERK           950
FORD       ANALYST        3000
SMITH      CLERK           800
SCOTT      ANALYST        3000
ADAMS      CLERK          1100
MILLER     CLERK          1300

10 rows selected.
```

DISPLAY SPECIFIC COLUMN NAMES

IS NOT NULL

커미션이 **NULL** 값인 아닌 사원의 이름, 직무, 커미션 조회

Example

```
SQL> SELECT ename, job, comm
  2 FROM s_emp
  3 WHERE comm IS NOT NULL;
```

```
ENAME      JOB           COMM
---------- --------- ----------
MARTIN     SALESMAN        1400
ALLEN      SALESMAN         300
TURNER     SALESMAN           0
WARD       SALESMAN         500

4 rows selected.
```

DISPLAY ROWS WITH COMPLEX CONDITIONS

AND, OR을 이용해 **WHERE**절에서 여러 조건을 결합하여 테이블 조회

Sysntx

```
SELECT { * |  column_name [, column_name...] }
FROM table_name
WHERE condition {AND | OR} condition
```

DISPLAY ROWS WITH COMPLEX CONDITIONS

AND

부서 번호가 **20**이고, 급여가 **2,000** 보다 큰 사원의 이름, 급여, 부서 번호 조회

Example

```
SQL> SELECT ename, sal, deptno
  2 FROM s_emp
  3 WHERE deptno = 20
  4 AND sal > 2000;

ENAME           SAL     DEPTNO
---------- ---------- ----------
JONES          2975         20
FORD           3000         20
SCOTT           3000         20
```

부서 번호가 **10**이고, 직무가 **MANAGER**인 사원의 이름, 급여, 부서 번호, 직무 조회

```
SQL> SELECT ename, sal, deptno, job
  2 FROM s_emp
  3 WHERE deptno = 10
  4 AND job = 'MANAGER';

ENAME           SAL     DEPTNO JOB
---------- ---------- ---------- ---------
CLARK          2450         10 MANAGER
```

DISPLAY ROWS WITH COMPLEX CONDITIONS

OR

부서 번호가 **20**이거나, 직무가 **SALESMAN**인 사원의 이름, 급여, 부서 번호, 직무 조회

Example

```
SQL> SELECT ename, sal, deptno, job
  2 FROM s_emp
  3 WHERE deptno = 20
  4 OR job = 'SALESMAN';

ENAME           SAL     DEPTNO JOB
---------- ---------- ---------- ---------
JONES          2975         20 MANAGER
MARTIN         1250         30 SALESMAN
ALLEN          1600         30 SALESMAN
TURNER         1500         30 SALESMAN
WARD           1250         30 SALESMAN
FORD           3000         20 ANALYST
SMITH           800         20 CLERK
SCOTT          3000         20 ANALYST
ADAMS          1100         20 CLERK

9 rows selected.
```

DISPLAY ROWS WITH COMPLEX CONDITIONS

우선 순위는 모든 비교 연산자, **AND** 그리고 **OR** 순서다.

Rules of Precedence

ORDER Evaluated	Operator
1	All Comparison Operators (=, <>, >, >=, <, <=, IN, LIKE, IN NULL, BETWEEN…AND)
2	AND
3	OR

DISPLAY ROWS WITH COMPLEX CONDITIONS

급여가 **1,000** 이상이고, 직무 번호가 **10** 또는 **20**인 사원의 이름, 급여, 직무 번호 조회

Example

```
SQL> SELECT ename, sal, deptno
   FROM s_emp
   WHERE sal >= 1000 AND deptno = 10 OR deptno = 20;

ENAME          SAL    DEPTNO
---------- ---------- ----------
SMITH           800        20
JONES          2975        20
CLARK          2450        10
SCOTT          3000        20
KING           5000        10
ADAMS          1100        20
FORD           3000        20

7 rows selected.
```

DISPLAY ROWS WITH COMPLEX CONDITIONS

Example

```
SQL> SELECT ename, sal, deptno
  2 FROM s_emp
  3 WHERE sal >= 1000 AND (deptno = 10 OR deptno = 20);
```

```
ENAME           SAL    DEPTNO
---------- ---------- ----------
KING          5000        10
CLARK          2450         10
JONES         2975        20
FORD          3000        20
SCOTT          3000         20
ADAMS          1100         20
MILLER         1300         10

7 rows selected.
```

ORDER THE ROWS DISPLAYED

특정 칼럼을 기준으로 정렬이 가능하다.

Sysntx

```
SELECT { * | column_name [, column_name...] }
FROM table_name
WHERE condition
ORDER BY   column_name  {ASC | DESC} [, column_name [ASC | DESC] …]
```

where	column_name	칼럼 이름
	table_name	테이블 이름
	ASC	행을 오름차순으로 정렬(Default)
	DESC	행을 내림차순으로 정렬

ORDER THE ROWS DISPLAYED

급여를 기준으로 오름차순 정렬

Example

```
SQL> SELECT ename, sal
  2 FROM s_emp
  3 WHERE deptno = 30
  4 ORDER BY sal;
```

```
ENAME          SAL
---------- ----------
JAMES          950
MARTIN         1250
WARD          1250
TURNER         1500
ALLEN         1600
BLAKE         2850

6 rows selected.
```

ORDER THE ROWS DISPLAYED

급여를 기준으로 내림차순 정렬

Example

```
SQL> SELECT ename, sal
  2 FROM s_emp
  3 WHERE deptno = 30
  4 ORDER BY sal DESC;
```

```
ENAME           SAL
---------- ----------
BLAKE          2850
ALLEN          1600
TURNER          1500
MARTIN          1250
WARD           1250
JAMES           950

6 rows selected.
```

ORDER THE ROWS DISPLAYED

사원 번호, 사원 이름, 부서 번호를 사원 번호 및 부서 번호로 오름차순 정렬

Example

```
SQL> SELECT empno, ename, deptno
  2 FROM s_emp
  3 ORDER BY empno, deptno;
```

```
 EMPNO ENAME         DEPTNO
---------- ---------- ----------
      7369 SMITH            20
      7499 ALLEN            30
      7521 WARD             30
      7566 JONES            20
      7654 MARTIN           30
      7698 BLAKE            30
      7782 CLARK            10
      7788 SCOTT            20
      7839 KING             10
      7844 TURNER           30
      7876 ADAMS            20
      7900 JAMES            30
      7902 FORD             20
      7934 MILLER           10

14 rows selected.
```

ORDER THE ROWS DISPLAYED

급여가 **1,500**이상인 사원의 이름, 부서 번호, 급여를 조회하고, 부서 번호로 오름차순, 급여로 내림차순하여 정렬

Example

```
SQL> SELECT ename, deptno, sal
  2 FROM s_emp
  3 WHERE sal >= 1500
  4 ORDER BY deptno, sal DESC;
```

```
ENAME        DEPTNO       SAL
---------- ---------- ----------
KING             10       5000
CLARK             10       2450
FORD             20       3000
SCOTT             20       3000
JONES             20       2975
BLAKE             30       2850
ALLEN             30       1600
TURNER            30       1500

8 rows selected.
```

INSERT DATA

INSERT 문을 이용해 테이블에 새로운 행을 추가할 수 있다.

Syntax

INSERT INTO *table* [*column*] VALUES (*value, value*…)

where	*table*	테이블 이름
	column	칼럼 이름
	value	열에 해당하는 값

INSERT DATA

INSERT 절에 칼럼을 나열하지 않고 테이블의 모든 칼럼에 대한 값을 삽입

Example

```
SQL> DESC s_dept;

COLUMN_NAME          TYPE           CONSTRAINT
------------------------------------------------------------------------------
DEPTNO               NUMBER(2)      PRIMARY KEY
DNAME                VARCHAR(14)
LOC                  VARCHAR(13)

INDEX_NAME           TYPE           COLUMN_NAME
------------------------------------------------------------------------------
PK_DEPT              NORMAL           DEPTNO
```

S_DEPT 테이블에 **'HR'** 부서 정보 입력

```
SQL> INSERT INTO s_dept
  2 VALUES (50, 'HR', 'SEOUL');

 DEPTNO DNAME         LOC
---------- -------------- -------------
       10 ACCOUNTING     NEW YORK
       20 RESEARCH       DALLAS
       30 SALES          CHICAGO
       40 OPERATIONS     BOSTON
       50 HR             SEOUL
```

INSERT DATA

INSERT 절에 칼럼을 나열하고 테이블의 모든 칼럼에 대한 값을 삽입

Example

```
SQL> DESCRIBE s_emp;

COLUMN_NAME                    TYPE          CONSTRAINT
------------------------------ ------------- ----------
EMPNO                          NUMBER(7)     PRIMARY KEY
                                             NOT NULL
ENAME                          VARCHAR(10)   NOT NULL
JOB                            VARCHAR(9)
MGR                            NUMBER(4)     REFERENTIAL
HIREDATE                       DATE
SAL                            NUMBER(7)
COMM                           NUMBER(7)
DEPTNO                         NUMBER(2)     REFERENTIAL

INDEX_NAME                     TYPE          COLUMN_NAME
------------------------------ ------------- ----------
S_EMP_ID_PK                    NORMAL        EMPNO
```

INSERT DATA

S_EMP 테이블에 새로운 직원 정보 입력

Example

```
SQL> INSERT INTO s_emp (empno, ename, job, mgr, hiredate, sal, comm, deptno)
  VALUES (1234,'ALEX', 'DESIGNER', 7839, SYSDATE, 3000, NULL, 20);

1 row inserted.

SQL> SELECT *
  2 FROM s_emp
  3 WHERE ename = 'ALEX';

    EMPNO    ENAME      JOB           MGR    HIREDATE                                  SAL       COMM     DEPTNO
---------- ---------- --------- ---------- ---------------------------------------------------------- ---------- ---------- ----------
     1234       ALEX      DESIGNER       7839    2022/10/07                                 3000                  20

1 row selected.
```

INSERT SPECIAL VALUES

새로운 사원 **CHRIS**에 대한 정보를 입력하되, 직무, 담당 매니저, 급여 정보를 제공하지 않는다. 빈 문자열로 **NULL**을 명시적으로 삽입

Example

```
SQL> INSERT INTO emp
  2 VALUES (7633, 'CHRIS', '', '', '', 2500, '', 50);

1 row inserted.
```

새로운 사원 **HENRY**에 대한 정보를 입력하되, 직무, 담당 매니저, 급여 정보를 제공하지 않는다. **NULL**을 암시적으로 삽입

```
SQL>  INSERT INTO s_emp(empno, ename, sal, deptno)
  2 VALUES (7634, 'HENRY', 1300, 50);

1 row inserted.
```

```
SQL> SELECT *
  2 FROM s_emp
  3 WHERE empno IN(7633, 7634);

    EMPNO ENAME      JOB           MGR HIREDATE                                                                SAL       COMM     DEPTNO
---------- ---------- --------- ---------- ------------------------------------------------------------------------ ---------- ---------- ----------
      7634 HENRY                                                                                              1300                     50
      7633 CHRIS                                                                                              2500                     50

2 rows selected.
```

INSERT SPECIAL VALUES

데이터베이스 사용자의 이름으로 **S_EMP** 테이블에 정보 입력하기

Example

```
SQL> INSERT INTO s_emp (empno, ename, hiredate, sal, deptno)
  2 VALUES (7636, USER, SYSDATE, 2000, 10);

1 row inserted.
```

```
SQL> SELECT *
  2 FROM s_emp
  3 WHERE empno = 7636;

    EMPNO ENAME      JOB          MGR HIREDATE                                                                SAL       COMM     DEPTNO
---------- ---------- --------- ---------- ---------------------------------------------------------------------- ---------- ---------- ----------
     7636 SYS                    2022/10/09                                                                  2000                  10

1 row selected.
```

UPDATE DATA

UPDATE 문으로 기존 행 수정

Syntax

```
UPDATE table
SET VALUES (column = value)
[WHERE condition]
```

where	*table*	업데이트 할 테이블 이름
	column	업데이트 할 칼럼 이름
	value	새로운 값
	condition	조건에 맞는 행을 업데이트

UPDATE DATA

사원번호 **7636**의 부서 번호와 **ALEX**의 부서 번호와 급여 변경

Example

```
SQL> UPDATE s_emp
  2 SET deptno = 20
  3 WHERE empno = 7636;

1 row updated.

SQL> UPDATE s_emp
  2 SET deptno = 20, sal = 4000
  3 WHERE ename = 'ALEX';

1 row updated.
```

```
SQL> SELECT empno, ename, sal, deptno
  2 FROM s_emp
  3 WHERE empno =7636 OR ename = 'ALEX';

    EMPNO ENAME           SAL     DEPTNO
---------- ---------- ---------- ----------
     1234 ALEX            4000         20
     7636 SYS             2000         20

2 rows selected.
```

UPDATE DATA

회사에서 부서 번호 **10**인 사원들에게 커미션을 **1,000**씩 지급한다.

Example

```
SQL> UPDATE s_emp
  2 SET comm = 1000
  3 WHERE deptno = 10;

3 rows updated.

SQL> SELECT ename, comm, deptno
  2 FROM s_emp
  3 WHERE deptno= 10;

ENAME          COMM     DEPTNO
---------- ---------- ----------
CHRIS          1000         10
CLARK           1000         10
KING           1000         10

3 rows selected.
```

DELETE DATA

DELETE 문으로 기존 행 삭제

Syntax

```
DELETE FROM table
[WHERE condition]
```

where	*table*	테이블 이름
	condition	조건에 맞는 행을 삭제

DELETE DATA

S_EMP 테이블에서 ALEX 사원 삭제

Example

```
SQL> DELETE FROM s_emp
  2 WHERE ename = 'ALEX';

1 row deleted.
```

```
SQL> SELECT ename
  2 FROM s_emp
  3 WHERE ename = 'ALEX';

0 row selected.
```

DELETE DATA

S_EMP 테이블에서 부서 번호 **50**인 사원들 삭제

Example

```
SQL> DELETE FROM s_emp
  2 WHERE deptno = 50;

2 row deleted.
```

```
SQL> SELECT ename
  2 FROM s_emp
  3 WHERE deptno = 50;

0 row selected.
```

조건이 없이 테이블 이름만 입력한 경우, 테이블 내 모든 데이터가 삭제된다. 전체 테이블을 삭제하는 경우가 아니면 **WHERE** 절을 생략하면 안된다.

CONTROL TRANSACTIONS

- 트랜잭션(TRANSACTION) - INSERT, UPDATE, DELETE
- 데이터 조작 작업은 데이터베이스 버퍼에 영향을 준다.
- 현재 사용자는 SELECT 문으로 데이터 조작 작업의 결과를 검토할 수 있다.
- 다른 사용자는 현재 사용자에 대한 데이터 조작 작업의 결과를 볼 수 없다.
- 영향을 받은 행은 LOCK이 걸리게 되고, 다른 사용자는 행을 변경할 수 없다.

Control Transaction Logic

Statement	Description
COMMIT	현재 트랜잭션을 종료하고 트랜잭션의 갱신된 내용을 데이터베이스에 반영
ROLLBACK	현재 트랜잭션을 종료하고 트랜잭션에서 갱신된 내용 모두를 취소

CONTROL TRANSACTIONS

State of the Data After COMMIT

- **COMMIT** 문을 사용하여 보류중인 모든 변경 내용**(INSERT, UPDATE, DELETE)**을 영구적으로 만든다.
- **COMMIT** 후 데이터 변경 사항이 데이터베이스 파일에 기록된다.
- 영향을 받은 행은 **LOCK**이 해제되고, 다른 사용자가 행을 변경할 수 있다.

S_DEPT 테이블에 새로운 부서 추가하고, **COMMIT**하기

Example

```
SQL> INSERT INTO dept (deptno, dname, loc)
  2 VALUES (60, 'LAW', 'LA');

SQL> SELECT *
  2 FROM dept
  3 WHERE dname = 'LAW';

    DEPTNO DNAME         LOC
---------- -------------- -------------
        60 LAW           LA

SQL> COMMIT;

Commit completed.
```

CONTROL TRANSACTIONS

State of the Data After ROLLBACK

- 데이터의 변경이 취소되고, 데이터의 이전 상태가 복원
- 영향을 받은 행은 **LOCK**이 해제되고, 다른 사용자가 행을 변경할 수 있다.

CONTROL TRANSACTIONS

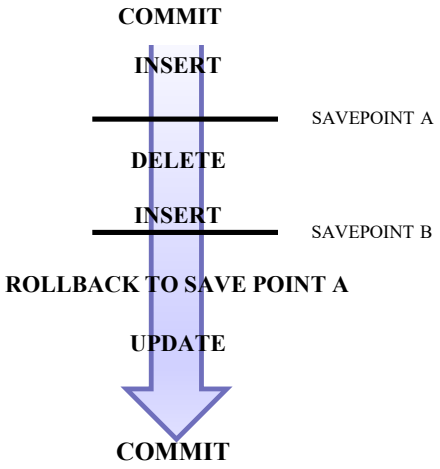
S_EMP 테이블에 부서 번호 **20**인 사원들의 행을 실수로 삭제했다. **ROLLBACK**을 이용해 복원한다.

Example

```
SQL> DELETE FROM s_emp
  2 WHERE deptno = 20;

5 rows deleted.

SQL> SELECT *
  2 FROM s_emp
  3 WHERE deptno = 20;

0 row selected.

SQL> ROLLBACK;

Rollback completed.

SQL> SELECT *
  2 FROM s_emp
  3 WHERE deptno = 20;

     EMPNO ENAME      JOB           MGR HIREDATE                                SAL       COMM     DEPTNO
---------- ---------- --------- ---------- ---------------------------------------- ---------- ---------- ----------
      7369 SMITH      CLERK           7902 0080/12/09                             800                    20
      7566 JONES      MANAGER         7839 0081/04/01                            2975                    20
      7788 SCOTT      ANALYST         7566 0082/12/22                            3000                    20
      7876 ADAMS      CLERK           7788 0083/01/15                            1100                    20
      7902 FORD       ANALYST         7566 0081/12/11                            3000                    20
```

CONTROL TRANSACTIONS

- **SAVEPOINT** 문을 이용하여 현재 트랜잭션에 저장 지점을 만든다.
- 부분 롤백을 수행하기 위해선 저장 지점을 미리 만들어야 한다.
- 동일한 이름의 저장 지점을 설정하면 이전의 저장 지점은 삭제된다.



Alter Transaction Logic

Statement	Description
SAVEPOINT	현재 트랜잭션 내에서 저장 지점 표시
ROLLBACK TO SAVEPOINT	저장 지점이 표시된 후 보류 중인 변경 내용 삭제

CONTROL TRANSACTIONS

S_EMP 테이블에 새로운 사원의 정보를 추가하고 SAVEPOINT a, b 만들기

Example

```
SQL> INSERT INTO s_emp(empno, ename, hiredate, sal)
  2 VALUES (3790, 'GOODMAN', SYSDATE, 2000);

1 row inserted.

SQL> SAVEPOINT a;

Savepoint created.

SQL> INSERT INTO s_emp(empno, ename, hiredate, sal)
  2 VALUES (3791, 'BADMAN', SYSDATE, 1000);

1 row inserted.

SQL> SAVEPOINT b;

Savepoint created.

SQL> INSERT INTO s_emp(empno, ename, hiredate, sal)
  2 VALUES (3792, 'YESMAN', SYSDATE, 3000);
1 row inserted.
```

```
SQL> SELECT empno, ename
  2 FROM s_emp;

     EMPNO ENAME
---------- ----------
      3790 GOODMAN
      3791 BADMAN
      3792 YESMAN
      7369 SMITH
      7499 ALLEN
      7521 WARD
      7566 JONES
      7654 MARTIN
      7698 BLAKE
      7782 CLARK
      7788 SCOTT
      7839 KING
      7844 TURNER
      7876 ADAMS
      7900 JAMES
      7902 FORD
```

CONTROL TRANSACTIONS

마지막 두 사원을 실행 취소하되, 첫 번째 사원은 그대로 둔다. **SAVEPOINT**를 이용해 영구적으로 변경한다.

Example

```
SQL> ROLLBACK TO SAVEPOINT a;

Rollback completed.

SQL> COMMIT;

Commit completed.
```

```
SQL> SELECT empno, ename
  2 FROM s_emp;

     EMPNO ENAME
---------- ----------
      3790 GOODMAN
      7369 SMITH
      7499 ALLEN
      7521 WARD
      7566 JONES
      7654 MARTIN
      7698 BLAKE
      7782 CLARK
      7788 SCOTT
      7839 KING
      7844 TURNER
      7876 ADAMS
      7900 JAMES
      7902 FORD
```

CONTROL TRANSACTIONS

- **COMMIT 또는 ROLLBACK을 암묵적으로 수행하는 상황에 주의해야 한다.**

Implicit Transaction Processing

Circumstance	Result
CREATE TABLE과 같은 DDL 명령어 실행	자동 COMMIT
명시적 COMMIT 또는 ROLLBACK을 하지 않고 데이터베이스 정상 종료	자동 COMMIT
비정상적인 종료 또는 시스템 오류	자동 ROLLBACK