```
REM Author : Henry
REM Date : 2024.06.12
REM Objective :
REM Environment : Ubuntu Server 22.04 LTS, MySQL Workbench 8.0 CE, MySQL Community Server 8.0.37-0ubuntu0.22.04.3 (ubuntu)

REM  데이터 조작(DML)
-데이터베이스에 데이터를 추가, 갱신, 삭제할 때 사용


REM INSERT 문
1. 테이블에 새 행 삽입할 때 사용
2. Syntax

    INSERT INTO table_name [ (column1[, column2[,...]])]
    VALUES (value1, [, value2[,...]]);

3. GuideLine
    1)VALUES 절에서는 동시에 한개의 레코드만 삽입된다.
    2)테이블 이름 뒤에 특정 칼럼을 지정하지 않으면 반드시, 스키마 순서대로 VALUES에서 사용해야 한다.
    3)명확한 Query문을 위해 가급적이면 컬럼이름을 지정하는 것이 좋다.(권장)
    4)문자형과 날짜형은 반드시 단일 따옴표를 사용하고, 숫자형은 단일 따옴표를 사용하지 않는다.
    5)각 열의 값을 포함하는 새 행을 삽입할 수 있으므로 INSERT 절에 열 목록이 필요 없지만 열 목록을 사용하지 않는 경우에는 값을 테이블의 기본 열 순서에 따라 나열해야 한다.
    6)테이블 리스트에 있는 칼럼 갯수와 VALUES 절의 값 갯수는 일치해야 한다.
        INSERT INTO dept(deptno, dname)
        VALUES (99, '총무과', '서울');

    7)입력될 값의 데이터 타입은 칼럼의 데이터 타입과 일치해야 한다.
        INSERT INTO dept(deptno, dname)
        VALUES ('99', 총무과);

    8)입력될 값의 크기는 칼럼의 크기보다 크지 않아야한다.
        INSERT INTO dept(deptno, dname)
        VALUES (999, '총무과');

    9)NULL 값에 주의하자.
    10)Foreign Key에 주의하자.
    11)오직 한번에 하나의 행만 입력할 수 있다.
        INSERT INTO dept
        VALUES (99, '총무과', '서울', 98, '인사과', '대전');

        INSERT INTO dept(deptno, dname, loc)
        VALUES(50, 'DEVELOPMENT', 'SEOUL');



REM NULL값을 갖는 행 삽입
1. 암시적방법
```

VALUES (99, '총무과', '서울', 98, '인사과', '대전');
와 같이 한번에 두개의 레코드를 입력할 수 없다.

```
48      INSERT INTO dept(deptno, dname)
49      VALUES (60, 'MIS');
50
51  2. 명시적 방법
52      INSERT INTO dept
53      VALUES (70, 'FINANCE', NULL);
54
55
56
57  REM NOW 함수 또는 CURDATE(), CURTIME()를 사용하여 현재 날짜 및 시간 삽입
58      INSERT INTO emp (empno, ename, job, mgr, hiredate, sal, comm, detpno)
59      VALUES (7196, 'GREEN', 'SALESMAN', 7782, CURDATE(), 2000, NULL, 10);
60
61      CREATE TABLE emp_copy
62      AS
63      SELECT *
64      FROM emp;
65
66      --Database의 Character set 변경, Table Collation 변경 --> utf8_general_ci
67
68      INSERT INTO emp_copy(empno, ename, job, mgr, hiredate, sal, comm, deptno)
69      VALUES(9999, 'Sujan', '', NULL, SYSDATE, 3000,NULL , 10);
70
71      INSERT INTO emp_copy
72      VALUES(8888, '홍길동', '', NULL, CURDATE(), 4000,NULL , 20)
73
74      INSERT INTO emp_copy(empno, ename, hiredate, deptno)
75      VALUES(7777, '백두산', CURDATE(), 40);
76
77      INSERT INTO emp_copy(empno, ename, hiredate, deptno)
78      VALUES (6666, '한라산', STR_TO_DATE('20080501', '%Y%m%d'), 30);
79
80
81
82  REM 다른 테이블로부터 행 복사
83      CREATE TABLE emp_clone
84      AS
85      SELECT empno, ename, sal, hiredate
86      FROM emp
87      WHERE 1 < 0;
88
89
90      --사원테이블에서 부서번호 10번의 레코드만 emp_clone 으로 복사하시오.
91      INSERT INTO emp_clone(empno, ename, sal, hiredate)
92      SELECT empno, ename, sal, hiredate
93      FROM emp
94      WHERE deptno = 10;
95
96
97
98  REM  UPDATE 문
```

NOW() 함수는 사용 X
NOW()는 연-월-일 시-분-초 모두 출력되는 함수라
입력 오류가 발생할 가능성이 높다.(잘려서 들어감)
=> CURDAET() 많이 사용

서버에 따라 한글 입력을 위해
UTF8mb4로 설정이 필요할 수 있다.

날짜를 형식을 지정해서 입력받을 수 있다.

레코드를 수정해야 하는 경우 UPDATE 문 사용

1. 기본 행 수정
2. 필요한 경우 한번에 여러 행 갱신 가능
3. Syntax

```
UPDATE table_name
SET column = value [, column = value, ...]
[WHERE condition];
```

한번에 여러 컬럼을 수정할 수 있다.

단, 어떤 레코드를 바꿀 지 반드시 조건을 지정해야 한다.

```
UPDATE emp
SET deptno = 20
WHERE empno = 7782;

UPDATE emp
SET deptno = 20;
```

REM 무결성 제약 조건 오류
-무결성 제약 조건의 영향을 받는 값을 포함하는 레코드를 갱신하면 오류가 발생

```
UPDATE emp
SET deptno = 55
WHERE deptno = 10;
```

REM **DELETE 문**
1. 기존 행 제거
2. Syntax

```
DELECT [FROM] table_name
[WHERE condition];

DELECT FROM dept
WHERE dname = 'DEVELOPMENT';

DELECT FROM emp;
```

REM 무결성 제약 조건 오류
-무결성 제약 조건의 영향을 받는 값을 포함하는 레코드를 삭제하면 오류가 발생

```
DELETE FROM dept
WHERE deptno = 10;
```

사원 테이블은 부서 번호와 연결되어 있다.
이 상태에서 해당 레코드를 삭제하면 연결이 깨진다.

```
--emp_copy테이블에서 1987년에 입사한 사원을 제거하시오.
DELETE FROM emp_copy
WHERE YEAR(hiredate) = '1987';
```

150
151 REM DML과 **TRANSACTION**
152 1. DDL 명령어인 경우에는 직접 **Database**의 **table**에 영향을 미치기 때문에 DDL명령어**를** 입**력**하는 순간 명령어에 해당하는 작업이 즉시**(**AUTO **COMMIT)** 완료된**.**
153 2. 하지만**,** DML 명령어의 경우**,** 조작하려는 **table**을 memory buffer에 **올**려놓고 작업을 하기 때문에 실시간**으**로 **table**에 영향을 미치는 것이 아니다**.**
154 3. 따**라**서 buffer에서 처리한 DML 명령어가 실제 **table**에 반영되기 위해서는 **COMMIT** 명령어**를** 입**력**하여 **Transaction**을 종료해**야** 한다**.**
155 4. **table**의 전체 **data를** 삭제하는 경우**,** System 활용 **측**면에서는 삭제된 **data를 log**로 저장하는 **DELETE TABLE** 보다는 System 부하가 적은 **TRUNCATE TABLE**을 권고한다**.**
156 5. 단**, TRUNCATE TABLE**의 경우 삭제된 **data**의 **log**가 없**으**므로 **ROLLBACK**이 불가능하므로 주의해**야** 한다**.**