

제06장

DML 활용

MySQL

학습목표

1. DML에 대해서 알 수 있다.
2. 트랜잭션에 대해서 알 수 있다.

```
each: function(e, t, n) {  
  var r, i = 0,  
      o = e.length,  
      a = M(e);  
  if (n) {  
    if (a) {  
      for (; o > i; i++)  
        if (r = t.apply(e[i], n), r ===  
    } else  
      for (i in e)  
        if (r = t.apply(e[i], n), r ===  
  } else if (a) {  
    for (; o > i; i++)  
      if (r = t.call(e[i], i, e[i]))  
    } else  
      for (i in e)  
        if (r = t.call(e[i], i, e[i]))  
  return e  
},  
trim: b && !b.call("\uffff\u00a0") ?  
  return null == e ? "" : b.call(  
} : function(e) {  
  return null == e ? "" : (e + "  
},  
makeArray: function(e, t) {  
  var n = t || [];  
  return null != e && (M(Obj  
}),  
isArray: function(e, t, n) {  
  var r;  
  if (t) {  
    if (n) return n.c  
    for (n = t.length;  
      if (n in t)  
  }  
}
```

목차

1. DML
2. 트랜잭션

```

each: function(e, t, n) {
    var r, i = 0,
        o = e.length,
        a = M(e);
    if (n) {
        if (a) {
            for (; o > i; i++)
                if (r = t.apply(e[i], n), r === !1) break;
        } else
            for (i in e)
                if (r = t.apply(e[i], n), r === !1) break;
    } else if (a) {
        for (; o > i; i++)
            if (r = t.call(e[i], e[i]), r === !1) break;
    } else
        for (i in e)
            if (r = t.call(e[i], e[i]), r === !1) break;
    return e;
},
trim: b && !b.call("\uffff\u00a0") ? function(e) {
    return null == e ? "" : b.call(e);
} : function(e) {
    return null == e ? "" : (e + "").replace(C, "");
},
makeArray: function(e, t) {
    var n = t || [];
    return null != e && (M(Object(e)) ? x.merge(n, "string"
),
inArray: function(e, t, n) {
    var r;
    if (t) {
        if (m) return m.call(t, e, n);
        for (r = t.length, r = r ? 0 > n ? Math.max(0, r + n) : n; r in t && t[r] === e) return r;
    }
}

```

1. DML

■ DML

- Data Manipulation Language (데이터 조작어)
- 정의된 데이터베이스에 데이터를 삽입하거나 수정하거나 삭제하는 역할을 수행하는 SQL문
- 테이블(Table), 뷰(View) 등 데이터베이스 객체에 행(Row)을 삽입/수정/삭제하는 기능을 담당하는 SQL문
- 실행 후에 작업의 완료 또는 취소 처리를 추가로 해야함

■ DML 종류

- INSERT : 행(Row) 삽입
- UPDATE : 행(Row) 수정
- DELETE : 행(Row) 삭제

INSERT

■ 구문

INSERT INTO 테이블_이름

[(칼럼1, 칼럼2, ...)]

칼럼 목록을 생략하면 모든 칼럼에 데이터를 입력해야 한다.

VALUES

(값1, 값2, ...)

■ nation_tbl 테이블의 code, name, continent 칼럼에 데이터 삽입

```
INSERT INTO nation_tbl(code, name, continent) VALUES(1, 'JAPAN', 'ASIA');
```

실행 전

code	name	continent
------	------	-----------

실행 후

code	name	continent
1	JAPAN	ASIA

UPDATE

■ 구문

UPDATE 테이블_이름

SET 칼럼1 = 값1, 칼럼2 = 값1

[WHERE 조건식]

WHERE절을 생략하면 전체 행을 수정한다.

■ nation_tbl 테이블에서 code=1인 국가의 name을 'KOREA'로 수정

```
UPDATE nation_tbl SET name = 'KOREA' WHERE code = 1;
```

실행 전

code	name	continent
1	JAPAN	ASIA

실행 후

code	name	continent
1	KOREA	ASIA

DELETE

■ 구문

DELETE

FROM 테이블_이름

[WHERE 조건식]

WHERE절을 생략하면 전체 행을 삭제한다.

■ nation_tbl 테이블에서 code=1인 데이터를 삭제

```
DELETE FROM nation_tbl WHERE code = 1;
```

실행 전

code	name	continent
1	KOREA	ASIA

실행 후

code	name	continent
------	------	-----------


```

each: function(e, t, n) {
    var r, i = 0,
        o = e.length,
        a = M(e);
    if (n) {
        if (a) {
            for (; o > i; i++)
                if (r = t.apply(e[i], n), r === !1) break;
        } else
            for (i in e)
                if (r = t.apply(e[i], n), r === !1) break;
    } else if (a) {
        for (; o > i; i++)
            if (r = t.call(e[i], e[i]), r === !1) break;
    } else
        for (i in e)
            if (r = t.call(e[i], e[i]), r === !1) break;
    return e;
},
trim: b && !b.call("\uffff\u00a0") ? function(e) {
    return null == e ? "" : b.call(e);
} : function(e) {
    return null == e ? "" : (e + "").replace(C, "");
},
makeArray: function(e, t) {
    var n = t || [];
    return null != e && (M(Object(e)) ? x.merge(n, "string"
),
isArray: function(e, t, n) {
    var r;
    if (t) {
        if (n) return m.call(t, e, n);
        for (r = t.length, r = n ? 0 > n ? Math.max(0, r + n) : r; r-- && t[r] !== e) return n;
    }
}

```

2. 트랜잭션

트랜잭션

■ 트랜잭션 (Transaction)

- 데이터베이스에서 처리되는 여러 SQL 명령들을 하나의 논리적 작업 단위로 처리하는 것
- 작업이 시작되면 중간에 멈추지 않고 반드시 종료해야 하는 작업 단위
- 작업 도중 하나라도 실패하면 아무 일도 하지 않은 상태로 되돌아가야 함

■ 트랜잭션이 필요한 상황

- 은행 이체 거래
내 통장의 출금과 상대 통장의 입금이 하나의 작업 단위로 묶여야 한다.
- 상품 구매
상품 구매에 따른 고객 포인트 증가, 상품 재고 감소, 매출 증가 등이 하나의 작업 단위로 묶여야 한다.

트랜잭션의 특징

■ 원자성 (Atomicity)

- 트랜잭션을 구성하는 연산들이 모두 정상적으로 실행되거나 하나도 실행되지 않아야 한다는 all-or-nothing 방식을 의미한다.

■ 일관성 (Consistency)

- 트랜잭션이 성공적으로 수행된 후에도 데이터베이스가 일관성 있는 상태를 유지해야 함을 의미한다.

■ 격리성 (Isolation)

- 현재 수행 중인 트랜잭션이 완료될 때까지 트랜잭션이 생성한 중간 연산 결과에 다른 트랜잭션들이 접근할 수 없음을 의미한다.

■ 지속성 (Durability)

- 트랜잭션이 성공적으로 완료된 후 데이터베이스에 반영한 수행 결과는 어떠한 경우에도 손실되지 않고 영구적이어야 함을 의미한다.

- Transaction Control Language
- 트랜잭션 처리를 위한 SQL문을 의미함
- 모든 DML(INSERT, UPDATE, DELETE)은 TCL이 반드시 필요함
- 종류
 - COMMIT
트랜잭션내의 모든 SQL 실행으로 인해 변경된 작업 내용을 디스크에 영구적으로 저장하고 트랜잭션을 종료한다.
 - ROLLBACK
트랜잭션내의 모든 SQL 실행으로 인해 변경된 작업 내용을 모두 취소하고 트랜잭션을 종료한다.

실습1. db_company

■ tbl_department

칼럼명	데이터타입	설명
dept_id	INT	부서아이디, PK
dept_name	VARCHAR(30)	부서명
location	VARCHAR(50)	위치

■ tbl_employee

칼럼명	데이터타입	설명
emp_id	INT	사원아이디, PK
dept_id	INT	부서아이디, FK(부서 테이블의 부서아이디 칼럼을 참조)
emp_name	VARCHAR(15)	사원명
position	CHAR(10)	직급
gender	CHAR(1)	성별
hire_date	DATE	입사일자
salary	INT	연봉

실습1. db_company

■ tbl_department

dept_id	dept_name	location
1	영업부	대구
2	인사부	서울
3	총무부	대구
4	기획부	서울

■ tbl_employee

emp_id	dept_id	emp_name	position	gender	hire_date	salary
1001	1	구창민	과장	M	1995-05-01	5000000
1002	1	김민서	사원	M	2017-09-01	2500000
1003	2	이은영	부장	F	1990-09-01	5500000
1004	2	한성일	과장	M	1993-04-01	5000000

실습1. db_company

■ tbl_department : 한 행씩 삽입하기

```
INSERT INTO tbl_department VALUES (null, '영업부', '대구');
INSERT INTO tbl_department VALUES (null, '인사부', '서울');
INSERT INTO tbl_department VALUES (null, '총무부', '대구');
INSERT INTO tbl_department VALUES (null, '기획부', '서울');
COMMIT;
```

■ tbl_employee : 동시에 여러 행 삽입하기

```
INSERT INTO
    tbl_employee
VALUES
    (null, '구창민', 1, '과장', 'M', '1995-05-01', 5000000),
    (null, '김민서', 1, '사원', 'M', '2017-09-01', 2500000),
    (null, '이은영', 2, '부장', 'F', '1990-09-01', 5500000),
    (null, '한성일', 2, '과장', 'M', '1993-04-01', 5000000);
COMMIT;
```

실습2. db_menu

■ db_menu.sql 스크립트 실행하기

- 명령 프롬프트 실행
 - C:> mysql -u 사용자 -p
 - mysql> source 경로/db_menu.sql