

MySQL 활용

데이터의 입력과 조회

목차

- 실습 환경 구축
- 데이터의 입력
- 데이터의 변경
- 데이터의 삭제
- 데이터의 조회

실습환경 구축

- 쇼핑몰을 가정한 테이블을 만들고 데이터를 입력한다.
 - 회원테이블 (userTbl)

컬럼	아이디	이름	생년	지역	국번	전화번호	키	가입일
필드명	userID	name	birthYear	addr	mobile1	mobile2	height	mDate
데이터타입	CHAR(20)	VARCHAR(20)	INT	CHAR(2)	CHAR(3)	CHAR(8)	SMALLINT	DATE
제약조건	PK, NN	NN	NN	NN				

- 구매테이블 (butTbl)

컬럼	순번	아이디	품명	분류	단가	수량
필드명	num	userID	prodName	groupName	price	amount
데이터타입	INT	CHAR(20)	VARCHAR(20)	CHAR(4)	INT	SMALLINT
제약조건	PK, AI, NN	FK	NN		NN	NN

실습환경 구축

- 테이블 생성

```
CREATE TABLE usertbl -- 회원 테이블
( userID    CHAR(20) NOT NULL PRIMARY KEY, -- 사용자 아이디(PK)
  name      VARCHAR(20) NOT NULL, -- 이름
  birthYear INT NOT NULL, -- 출생년도
  addr      CHAR(2) NOT NULL, -- 지역(경기, 서울, 경남 식으로 2글자만입력)
  mobile1   CHAR(3), -- 휴대폰의 국번(011, 016, 017, 018, 019, 010 등)
  mobile2   CHAR(8), -- 휴대폰의 나머지 전화번호(하이픈제외)
  height    SMALLINT, -- 키
  mDate     DATE -- 회원 가입일
);

CREATE TABLE buytbl -- 회원 구매 테이블(Buy Table의 약자)
( num        INT AUTO_INCREMENT NOT NULL PRIMARY KEY, -- 순번(PK)
  userID     CHAR(20) NOT NULL, -- 아이디(FK)
  prodName   VARCHAR(20) NOT NULL, -- 상품명
  groupName  CHAR(4) , -- 분류
  price      INT NOT NULL, -- 단가
  amount     SMALLINT NOT NULL, -- 수량
  FOREIGN KEY (userID) REFERENCES usertbl(userID)
);
```

실습환경 구축

• 데이터의 입력

```
INSERT INTO usertbl VALUES('LSG', '이승기', 1987, '서울', '011', '1111111', 182, '2008-8-8');
```

```
INSERT INTO usertbl VALUES('KBS', '김범수', 1979, '경남', '011', '2222222', 173, '2012-4-4');
```

```
INSERT INTO usertbl VALUES('KKH', '김경호', 1971, '전남', '019', '3333333', 177, '2007-7-7');
```

```
INSERT INTO usertbl VALUES('JYP', '조용필', 1950, '경기', '011', '4444444', 166, '2009-4-4');
```

```
INSERT INTO usertbl VALUES('SSK', '성시경', 1979, '서울', NULL, NULL, 186, '2013-12-12');
```

```
INSERT INTO usertbl VALUES('LJB', '임재범', 1963, '서울', '016', '6666666', 182, '2009-9-9');
```

```
INSERT INTO usertbl VALUES('YJS', '윤종신', 1969, '경남', NULL, NULL, 170, '2005-5-5');
```

```
INSERT INTO usertbl VALUES('EJW', '은지원', 1972, '경북', '011', '8888888', 174, '2014-3-3');
```

```
INSERT INTO usertbl VALUES('JKW', '조관우', 1965, '경기', '018', '9999999', 172, '2010-10-10');
```

```
INSERT INTO usertbl VALUES('BBK', '바비킴', 1973, '서울', '010', '0000000', 176, '2013-5-5');
```

실습환경 구축

- 데이터의 입력

```
INSERT INTO buytbl VALUES(NULL, 'KBS', '운동화', NULL, 30, 2);
INSERT INTO buytbl VALUES(NULL, 'KBS', '노트북', '전자', 1000, 1);
INSERT INTO buytbl VALUES(NULL, 'JYP', '모니터', '전자', 200, 1);
INSERT INTO buytbl VALUES(NULL, 'BBK', '모니터', '전자', 200, 5);
INSERT INTO buytbl VALUES(NULL, 'KBS', '청바지', '의류', 50, 3);
INSERT INTO buytbl VALUES(NULL, 'BBK', '메모리', '전자', 80, 10);
INSERT INTO buytbl VALUES(NULL, 'SSK', '책', '서적', 15, 5);
INSERT INTO buytbl VALUES(NULL, 'EJW', '책', '서적', 15, 2);
INSERT INTO buytbl VALUES(NULL, 'EJW', '청바지', '의류', 50, 1);
INSERT INTO buytbl VALUES(NULL, 'BBK', '운동화', NULL, 30, 2);
INSERT INTO buytbl VALUES(NULL, 'EJW', '책', '서적', 15, 1);
INSERT INTO buytbl VALUES(NULL, 'BBK', '운동화', NULL, 30, 2);
```

데이터의 입력




- INSERT 문
 - 테이블에 데이터를 삽입하는 명령어
 - INSERT [INTO] 테이블명[(열1, 열2, ...)] VALUES (값1, 값2 ..);
 - 열은 생략이 가능하나 생략할 경우 VALUES 다음에 나오는 값들의 순서 및 개수가 테이블에 정의된 열의 순서 및 개수와 동일해야 함.

데이터의 입력

- 데이터의 입력

- 테이블이 아래와 같을때 데이터 입력의 예

member

Column Name	Datatype	PK	NN
 id	VARCHAR(20)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
 name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
 addr	VARCHAR(150)	<input type="checkbox"/>	<input checked="" type="checkbox"/>

INSERT INTO member(id, name, addr) VALUES ('1', '정성훈', '서울 동대문구');

INSERT member(id, name, addr) VALUES ('1', '정성훈', '서울 동대문구');

INSERT member VALUES ('1', '정성훈', '서울 동대문구');

데이터의 입력

- AUTO INCREMENT

- 컬럼이 AUTO_INCREMENT로 설정되어 있는 경우 해당열의 값이 없다고 여기고 INSERT문 작성

INSERT member VALUES (NULL, '정성훈', '서울 동대문구');

- AUTO_INCREMENT 설정시 주의사항

- PRIMARY KEY 또는 UNIQUE로 설정
- 데이터형은 숫자형만 가능
- 입력시 NULL 지정

데이터의 입력

- 데이터 입력 실습

- testdb에 아래와 같은 학생 테이블을 생성하시오
- 테이블명 : student
- 컬럼 : 학번, 학과, 번호, 이름, 주소, 이메일
- 단 번호는 1부터 순차적으로 증가한다.

- 5개의 서로다른 데이터를 만들어 입력하시오.

ex) INSERT student VALUES ('22-02-01', '영문과', 1, '홍길동', '서울 관악구', 'aa@naver.com');

데이터의 변경

- UPDATE 문

- 데이터를 수정하기 위한 명령어

- UPDATE 테이블명 SET 열1=값1, 열2=값2 WHERE 조건;

UPDATE member SET addr='서울 관악구' WHERE name='정성훈';

데이터의 변경

- 데이터 변경 실습
 - student테이블에서 이름이 홍길동인 사람의 주소를 " 경기도 시흥시 "로 변경하시오.

데이터의 삭제

- DELETE 문
 - 데이터를 삭제하기 위한 명령어
 - DELETE FROM 테이블명 WHERE 조건 (UPDATE 와 유사)

```
DELETE FROM member WHERE id='3';
```

데이터의 조회

- SELECT 문
 - 원하는 데이터를 조회하는 명령어
 - 가장 많이 사용하며 복잡한 구조를 가짐

- 가장 간단한 형식

SELECT 열이름
FROM 테이블이름
WHERE 조건

```
SELECT
  [ALL | DISTINCT | DISTINCTROW ]
  [HIGH_PRIORITY]
  [STRAIGHT_JOIN]
  [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
  [SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
  select_expr [, select_expr] ...
  [into_option]
  [FROM table_references
    [PARTITION partition_list]]
  [WHERE where_condition]
  [GROUP BY {col_name | expr | position}, ... [WITH ROLLUP]]
  [HAVING where_condition]
  [WINDOW window_name AS (window_spec)
    [, window_name AS (window_spec)] ...]
  [ORDER BY {col_name | expr | position}
    [ASC | DESC], ... [WITH ROLLUP]]
  [LIMIT {[offset,] row_count | row_count OFFSET offset}]
  [into_option]
  [FOR {UPDATE | SHARE}
    [OF tbl_name [, tbl_name] ...]
    [NOWAIT | SKIP LOCKED]
    | LOCK IN SHARE MODE]
  [into_option]

into_option: {
  INTO OUTFILE 'file_name'
    [CHARACTER SET charset_name]
    export_options
  | INTO DUMPFILE 'file_name'
  | INTO var_name [, var_name] ...
}
```

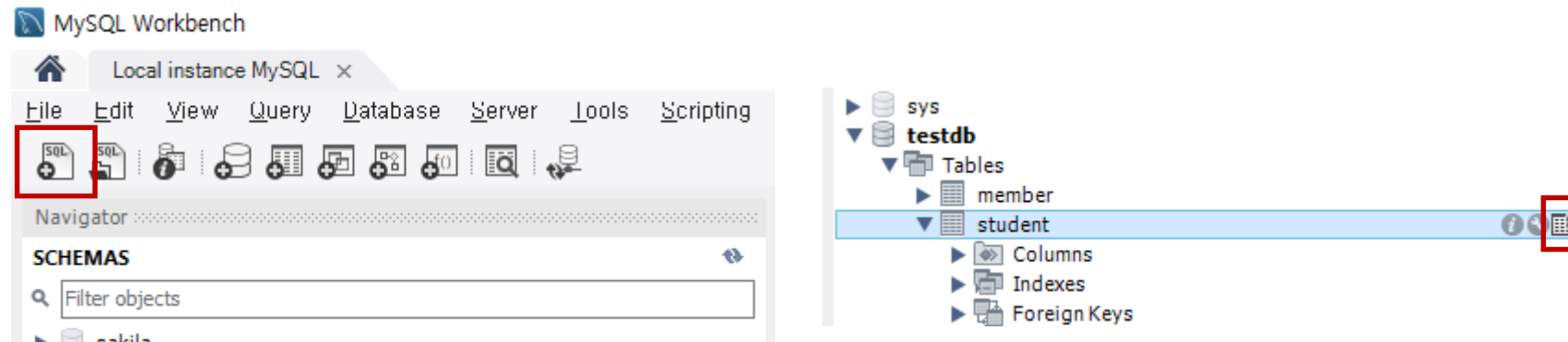
데이터의 조회

- USE 문
 - database를 지정하는 명령어
 - USE 데이터베이스이름

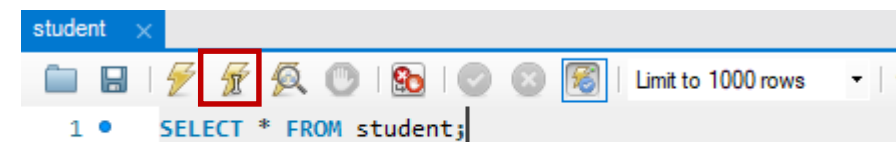
USE testdb;

데이터의 조회

- 기본적인 SELECT 문 사용법
 - SQL 실행탭 열기 또는 실행탭 short-cut 클릭



- 출력된 SQL 실행탭에서 아래와 같이 입력 후 실행 아이콘 클릭
- `SELECT * FROM student;`
- 블록 지정후 Ctrl + Enter로 실행가능



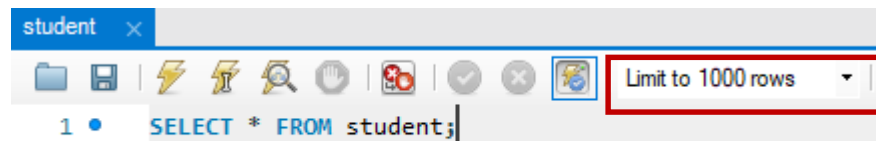
데이터의 조회

- 기본적인 SELECT 문 사용법
 - SELECT문의 실행결과

#	Time	Action	Message	Duration / Fetch
✓ 1	23:35:35	SELECT * FROM student LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

- TIME : 명령을 실행한 시간
- Action : 실행된 명령
- Message : SELECT 문이 조회된 행의 개수
- Duration/... : Duration : SQL문이 실행된 시간
Fetch : 데이터를 가져오는데 걸린 시간

** Action에 LIMIT가 걸린것은 과도한 row select를 방지하기 위함이다. 이는 변경 가능하다.



데이터의 조회

- 컬럼의 선택

- id, name 컬럼을 조회

SELECT id, name FROM student;

- student테이블에서 id, name을 조회

- student테이블에서 id, name을 가지고 와서 새로운 결과 테이블을 만든다는 의미로도 해석

SELECT id, name AS user_name FROM student;

SELECT id, name AS 'user name' FROM student; -- 공백이 있는 경우 "로 처리

SELECT id, name 'user name' FROM student; -- AS는 생략가능

데이터의 조회

- WHERE 절
 - SELECT등의 명령어에 대한 조건을 정의한 식
 - name이 홍길동인 row를 조회
SELECT * from student name='홍길동';

데이터의 조회

- 관계연산자를 이용한 데이터 조회

- 실습환경 이용

- 1970년 이후에 출생하고 신장이 182 인 회원을 조회

- SELECT * FROM usertbl WHERE birthYear >= 1970 AND height >= 182;

- 키가 180~183인 회원을 조회

- SELECT * FROM usertbl WHERE height >= 180 AND height <= 183;

- SELECT * FROM usertbl WHERE height BETWEEN 180 AND 183;

데이터의 조회

- 이산값 조회

- 경남, 전남 경북이 주소인 회원조회

```
SELECT * FROM usertbl WHERE addr IN ('경남', '전남', '경북');
```

- 문자열 검색 (LIKE검색)

- 김씨 성을 가진 회원 조회

```
SELECT * FROM usertbl WHERE name LIKE '김%';
```

실습

- 011전화번호를 가진 회원의 아이디와 이름 조회
- 1970년 이후에 출생했거나 신장이 182 인 회원의 아이디와 이름 조회
- 서울출신 이고 010 또는 011전화번호를 사용하는 회원 조회
- price가 100보다 작은 제품의 제품명 조회
- userID의 가운데 글자가 B인 회원을 조회