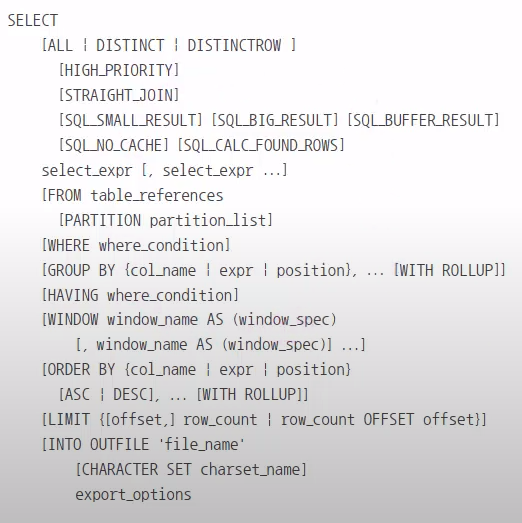
**6강-01교시**

6.1 SELECT문

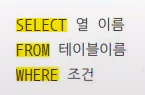
데이터베이스에서 내가 원하는 내용을 뽑는다.

6.1.1 원하는 데이터를 가져다 주는 기본적인(SELECT...FROM)

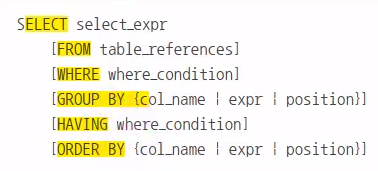
MySQL 8.0의 도움말에 나오는 SELECT의 구문 형식



기본구문



추가구문



USE 구문 - 사용할 데이터베이스 설정

USE 데이터베이스\_이름;

USE employees; (퀴리문)

=> 지금부터 employees를 사용하겠으니, 모든 퀴리는 employees에서 수행하라

6강 코드

USE employees;

많이 하는 실수!

USE mysql;

SELECT \* FROM employees;

SELECT \* FROM titles ;

(USE employees;를 안써도 사용 가능한 코드)

SELECT \* FROM employees.titles;

같은 코드(USE employees; 사용시)

SELECT \* FROM employees.titles;

SELECT \* FROM titles;

원하는 열 출력

SELECT first\_name FROM employees;

원하는 여러 열 출력

SELECT first\_name, last\_name, gender FROM employees;

주석 예시

-- 한줄 주석 연습

SELECT first\_name, last\_name, gender -- 이름과 성별 열을 가져옴

FROM employees;

/\* 블록 주석 연습

SELECT first\_name, last\_name, gender

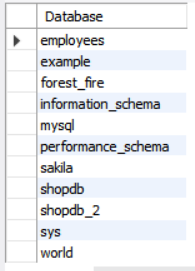
FROM employees;

\*/

-- <실습 1> --

데이터베이스 목록 확인

SHOW DATABASES;

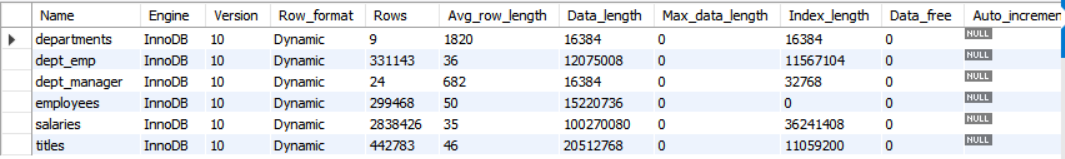


employees 데이터베이스 설정

USE employees;

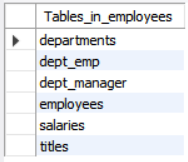
테이블 정보 확인

SHOW TABLE STATUS;



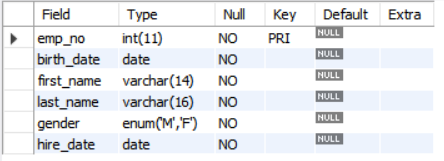
테이블 목록 보기

SHOW TABLES;



테이블 변수 정보 확인

DESCRIBE employees; -- 또는 DESC employees;



테이블의 원하는 열정보 확인

SELECT first\_name, gender FROM employees;



인덱스를 내가 원하는 인덱스로 바꾸기

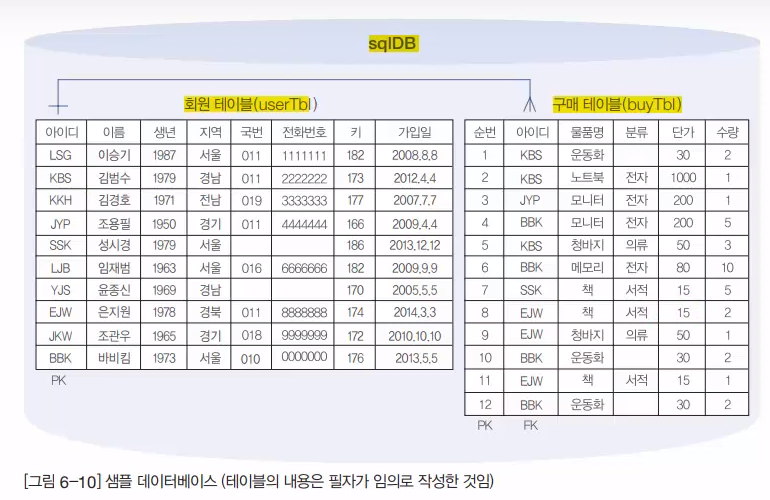
SELECT first\_name AS 이름 , gender 성별, hire\_date '회사 입사일'

FROM employees;



**6강-02교시**

6강의 DB



CHAR()과 VARCHAR()의 차이점

CHAR(X): X칸 꽉 차게 입력한 데이터가 저장됨 ex) CHAR(4) / 12 => 0012

VARCHAR(X): 입력한 데이터만큼만 저장됨 ex) VARCHAR(4) / 12 => 12

##### **SMALLINT[(M)] [UNSIGNED] [ZEROFILL]**

- 자료형의 크기 : 2바이트(2Byte, 2^16 = 16bit)

- 범위 : -32768 에서 32767 (UNSIGNED 일 경우 0 에서 65535)

##### **INT[(M)] [UNSIGNED] [ZEROFILL]**

- 자료형의 크기 : 4바이트(4Byte, 2^32 = 32bit)

- 범위 : -2147483648 에서 2147483647 (UNSIGNED 일 경우 0 에서 4294967295)

- 주의 : 모든 연산은 “signed BIGINT” 로 이루어진다. 따라서 더하기 곱하기 등으로 “signed BIGINT” 의 범위를 벗어난다면 값이 틀려질 것이다.

##### **CHAR(M) [BINARY]**

- 자료형의 크기 : char 필드의 CHARACTER SET 에 따라 가변적임. 예를들어 char(10) 이면서 utf4mb8 일 경우 고정 40Byte 사용.

- 범위 : 1 에서 255 글자

- 주의 : 고정길이로 길이가 작은 값이 입력되면 오른쪽에 스페이스로 채워져서 입력된다. BINARY 키워드가 주어지지 않을 경우 대소문자 구분없이 검색한다.

##### **VARCHAR(M) [BINARY]**

- 자료형의 크기 : varchar 필드의 CHARACTER SET 에 따라 가변적임. 예를들어 varchar(10) 이면서 utf4mb8 일 경우 최대(크기인덱스 1Byte + 데이터 40Byte 사용).

- 범위 : 1 에서 255 글자 (현재는 1에서 65535 글자 까지. 그러나 varchar(255) 이상을 사용하고 싶은 경우 text 자료형을 사용하여라)

- 주의 : BINARY 키워드가 주어지지 않을 경우 대소문자 구분없이 검색한다.

- TIP1 : 모든 데이터 값의 길이가 고정일 경우 char 가 varchar 보다 저장공간도 적게 차지하고 속도도 아주 약간 더 빠르다.

- TIP2 : 대부분의 상황에서 varchar 와 char 는 IN/OUT 이 동일하게 처리된다. 아무거나 써도 되지만 varchar 를 선택하기를 권장한다.

추가 변수 타입 관련: <https://blog.lael.be/post/115>

**기존의 sqldb제거 및 sqldb생성**

DROP DATABASE IF EXISTS sqldb; -- 만약 sqldb가 존재하면 우선 삭제한다.

CREATE DATABASE sqldb;

**sqldb지정 후에 table 생성**

USE sqldb;

CREATE TABLE usertbl -- 회원 테이블

( userID CHAR(8) NOT NULL PRIMARY KEY, -- 사용자 아이디(PK)

name VARCHAR(10) 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)

**⛦AUTO\_INCREMENT 번호를 따로 입력 안 할거기 때문에 자동으로 번호를 매기는 코드**

( num INT AUTO\_INCREMENT NOT NULL PRIMARY KEY, -- 순번(PK)

userID CHAR(8) NOT NULL, -- 아이디(FK)

prodName CHAR(6) NOT NULL, -- 물품명

groupName CHAR(4) , -- 분류

price INT NOT NULL, -- 단가

amount SMALLINT NOT NULL, -- 수량

FOREIGN KEY (userID) REFERENCES usertbl(userID)

);

**데이터 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');

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);

**테이블 확인**

SELECT \* FROM usertbl;

SELECT \* FROM buytbl;