Chapter 05

SQL 고급

Contents

Contents

CHAPTER 05 SQL 고급

데이터 형식과 변수의 사용, 대용량 데이터의 저장 방식, MySQL 프로그래밍을 위한 내용을 살펴본다.

SECTION 01 MySQL의 데이터 형식

MySQL에서 지원하는 데이터 형식의 종류

- Data Type으로 표현
  - 데이터 형식, 데이터형, 자료형, 데이터 타입등 다양하게 불림
- 데이터 형식에 대한 이해가 필요한 이유
  - SELECT문 더욱 잘 활용
  - 테이블의 생성 효율적으로 하기 위해 필요
- MySQL에서 데이터 형식의 종류는 30개 정도
  - **중요하고 자주 쓰는 형식에 대해 중점 학습**

SECTION 01 MySQL의 데이터 형식

MySQL에서 지원하는 데이터 형식의 종류

◦ 숫자 데이터 형식

데이터 형식	바이트 수	숫자 범위	설명
BIT(N)	N/8		1~64bit를 표현. b'0000' 형식으로 표현
TINYINT	1	-128~127	정수
★SMALLINT	2	-32,768~32,767	정수
MEDIUMINT	3	-8,388,608~8,388,607	정수
★INT INTEGER	4	약 -21억 ~ +21억	정수
★BIGINT	8	약 -900경 ~ +900경	정수
★FLOAT	4	-3.40E+38~-1.17E-38	소수점 아래 7자리까지 표현
DOUBLE REAL	8	-1.22E-308~1.79E+308	소수점 아래 15자리까지 표현
★DECIMAL(m,[d]) NUMERIC(m,[d])	5~17	$-10^{38}+1 \sim +10^{38}-1$	전체 자릿수(m)와 소수점 이하 자릿수(d)를 가진 숫자형 예) decimal(5,2)는 전체 자릿수를 5자리로 하되, 그 중 소수점 이하를 2자리로 하겠다는 의미

[표 7-1] 숫자 데이터 형식

SECTION 01 MySQL의 데이터 형식

MySQL에서 지원하는 데이터 형식의 종류

◦ 문자 데이터 형식

데이터 형식		바이트 수	설명
★CHAR(n)		1~255	고정길이 문자형. n을 1부터 255까지 지정. character의 약자 그냥 CHAR만 쓰면 CHAR(1)과 동일
★VARCHAR(n)		1~65535	가변길이 문자형. n을 사용하면 1부터 65535까지 지정. Variable character의 약자
BINARY(n)		1~255	고정길이의 이진 데이터 값
VARBINARY(n)		1~255	가변길이의 이진 데이터 값
TEXT 형식	TINYTEXT	1~255	255 크기의 TEXT 데이터 값
	TEXT	1~65535	N 크기의 TEXT 데이터 값
	MEDIUMTEXT	1~16777215	16777215 크기의 TEXT 데이터 값
	★LONGTEXT	1~4294967295	최대 4GB 크기의 TEXT 데이터 값
BLOB 형식	TINYBLOB	1~255	255 크기의 BLOB 데이터 값
	BLOB	1~65535	N 크기의 BLOB 데이터 값
	MEDIUMBLOB	1~16777215	16777215 크기의 BLOB 데이터 값
	★LONGBLOB	1~4294967295	최대 4GB 크기의 BLOB 데이터 값
ENUM(값들…)		1 또는 2	최대 65535개의 열거형 데이터 값
SET(값들…)		1, 2, 3, 4, 8	최대 64개의 서로 다른 데이터 값

[표 7-2] 문자 데이터 형식

SECTION 01 MySQL의 데이터 형식

MySQL에서 지원하는 데이터 형식의 종류

◦ 날짜와 시간 데이터 형식

데이터 형식	바이트 수	설명
★DATE	3	날짜는 1001-01-01 ~ 9999-12-31까지 저장되며 날짜 형식만 사용 'YYYY-MM-DD' 형식으로 사용됨.
TIME	3	-838:59:59.000000 ~ 838:59:59.000000까지 저장되며. 'HH:MM:SS' 형식으로 사용
★DATETIME	8	날짜는 1001-01-01 00:00:00 ~ 9999-12-31 23:59:59까지 저장되며 형식은 'YYYY-MM-DD HH:MM:SS' 형식으로 사용
TIMESTAMP	4	날짜는 1001-01-01 00:00:00 ~ 9999-12-31 23:59:59까지 저장되며 형식은 'YYYY-MM-DD HH:MM:SS' 형식으로 사용. time_zone 시스템 변수와 관련이 있으며 UTC 시간대 변환하여 저장
YEAR	1	1901 ~ 2155까지 저장. 'YYYY' 형식으로 사용

[표 7-3] 날짜와 시간 데이터 형식

	DATE
▶	2020-10-19

	TIME
▶	12:35:29

	DATETIME
▶	2020-10-19 12:35:29

SECTION 01 MySQL의 데이터 형식

MySQL에서 지원하는 데이터 형식의 종류

◦ 기타 데이터 형식

데이터 형식	바이트 수	설명
★GEOMETRY	N/A	공간 데이터 형식으로 선, 점 및 다각형 같은 공간 데이터 개체를 저장하고 조작
★JSON	8	JSON(JavaScript Object Notation) 문서를 저장

[표 7-4] 기타 데이터 형식

◦ LONGTEXT, LONGBLOB

- LOB(Large Object, 대량의 데이터)을 저장하기 위해 LONGTEXT, LONGBLOB 데이터 형식 지원
- 지원되는 데이터 크기는 약 4GB의 파일을 하나의 데이터로 저장 가능
- LONGTEXT
  - ex) 장편소설과 같은 큰 텍스트 파일
- LONGBLOG
  - ex) 동영상 파일과 같은 큰 바이너리 파일

SECTION 01 MySQL의 데이터 형식

변수의 사용

- Workbench를 재시작할 때까지는 계속 유지, Workbench를 닫았다가 재시작하면 소멸
- 변수의 선언과 값의 대입 형식

```
SET @변수이름 = 변수의 값 ;        -- 변수의 선언 및 값 대입
SELECT @변수이름 ;                 -- 변수의 값 출력
```

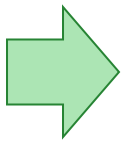
- 변수 사용 실습

```
USE sqldb;

SET @myVar1 = 5 ;
SET @myVar2 = 3 ;
SET @myVar3 = 4.25 ;
SET @myVar4 = '가수 이름==> ' ;

SELECT @myVar1 ;.
SELECT @myVar2 + @myVar3 ;

SELECT @myVar4 , Name FROM usertbl WHERE height > 180 ;
```



@myVar1
5

@myVar2 + @myVar3
7.250000000000000000000000000000

@myVar4	Name
가수 이름==>	임재범
가수 이름==>	이승기
가수 이름==>	성시경

SECTION 01 MySQL의 데이터 형식

데이터 형식과 형 변환

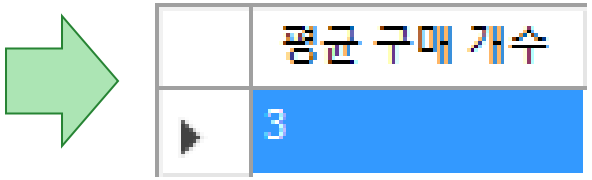
◦ 데이터 형식 변환 함수

- CAST( ), CONVERT( ) 함수를 가장 일반적으로 사용
- 데이터 형식 중에서 가능한 것은 BINARY, CHAR, DATE, DATETIME, DECIMAL, JSON, SIGNED INTEGER, TIME, UNSIGNED INTEGER
- 함수 사용법

```
형식:
CAST ( expression AS 데이터형식 [ (길이) ] )
CONVERT ( expression ,  데이터형식 [ (길이) ] )
```

• ex) sqlDB의 구매 테이블(buyTbl)에서 평균 구매 개수를 구하는 쿼리문

```
SELECT CAST(AVG(amount) AS SIGNED INTEGER) AS '평균 구매 개수'  FROM buytbl ;
또는
SELECT CONVERT(AVG(amount) , SIGNED INTEGER) AS '평균 구매 개수'  FROM buytbl ;
```



SECTION 01 MySQL의 데이터 형식

데이터 형식과 형 변환

◦ 암시적인 형 변환

- CAST( )나 CONVERT( ) 함수를 사용하지 않고 형이 변환되는 것

```
SELECT '100' + '200' ; -- 문자와 문자를 더함(정수로 변환되서 연산됨)
SELECT CONCAT('100', '200'); -- 문자와 문자를 연결(문자로 처리)
SELECT CONCAT(100, '200'); -- 정수와 문자를 연결(정수가 문자로 변환되서 처리)
SELECT 1 > '2mega'; -- 정수인 2로 변환되어서 비교
SELECT 3 > '2MEGA'; -- 정수인 2로 변환되어서 비교
SELECT 0 = 'mega2'; -- 문자는 0으로 변환됨
```

'100' + '200'	CONCAT('100', '200')	CONCAT(100, '200')	1 > '2mega'	3 > '2MEGA'	0 = 'mega2'
300	100200	100200	0	1	1

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 내장 함수
  - 흐름 함수, 문자열 함수, 수학 함수, 날짜/시간 함수, 전체 텍스트 검색 함수, 형 변환 함수, XML 함수, 비트 함수, 보안/압축 함 수, 정보 함수, 공간 분석 함수, 기타 함수 등
- 제어 흐름 함수
  - 프로그램의 흐름 제어
  - IF (수식, 참, 거짓)
    - 수식이 참 또는 거짓인지 결과에 따라서 2중 분기

```
SELECT IF (100>200, '참이다', '거짓이다');
```

  - IFNULL(수식1, 수식2)
    - 수식1이 NULL이 아니면 수식1이 반환되고 수식1이 NULL이면 수식2가 반환

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 제어 흐름 함수
  - NULLIF(수식1, 수식2)
    - 수식1과 수식2가 같으면 NULL을 반환, 다르면 수식1을 반환
  - CASE ~ WHEN ~ ELSE ~ END
    - CASE는 내장 함수는 아니며 연산자(Operator)로 분류
    - 다중 분기에 사용되므로 내장함수와 함께 알아두자

```
SELECT  CASE 10
              WHEN 1  THEN  '일'
              WHEN 5  THEN  '오'
              WHEN 10 THEN  '십'
              ELSE '모름'
        END AS 'CASE연습';
```

    - CASE 뒤의 값이 10이므로 세 번째 WHEN이 수행되어 '십' 반환
    - 만약, 해당하는 사항이 없다면 ELSE 부분이 반환

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - 문자열 조작, 활용도 높음
  - ASCII (아스키 코드),
    - 문자의 아스키 코드값 반환
  - CHAR(숫자)
    - 숫자의 아스키 코드값에 해당하는 문자 반환

```
SELECT ASCII('A'), CHAR(65);
```

  - BIT_LENGTH(문자열), CHAR_LENGTH(문자열), LENGTH(문자열)
    - 할당된 Bit 크기 또는 문자 크기 반환
    - CHAR_LENGTH( )는 문자의 개수 반환
    - LENGTH( )는 할당된 Byte 수 반환

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

◦ 문자열 함수

- CONCAT(문자열1, 문자열2,...), CONCAT_WS(구분자, 문자열1, 문자열2,...)
  - CONCAT( ) : 문자열을 이어줌
  - CONCAT_WS( ) : 구분자와 함께 문자열을 이어주는 역할

```
SELECT CONCAT_WS('/', '2025', '01', '01');
```

  - 2025/01/01 반환

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - ELT(위치, 문자열1, 문자열2, ...), FIELD(찾을 문자열, 문자열1, 문자열2, ...), FIND_IN_SET (찾을 문자열, 문자열 리스트), INSTR(기준 문자열, 부분 문자열), LOCATE(부분 문자열, 기준 문자열)
    - ELT( ) : 위치 번째에 해당하는 **문자열 반환**
    - FIELD( ) : 찾을 문자열의 **위치를 찾아 반환**, 없으면 0
    - FIND_IN_SET( ) : 찾을 문자열을 문자열 리스트에서 찾아 **위치 반환**
      - 문자열 리스트는 콤마(,)로 구분되어 있고 공백이 없어야 함
    - INSTR( )는 기준 문자열에서 부분 문자열 찾아 그 **시작 위치 반환**
    - LOCATE( )는 INSTR( )와 동일하지만 파라미터의 순서가 반대

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - FORMAT(숫자, 소수점 자릿수)
    - 숫자를 소수점 아래 자릿수까지 표현, 1,000단위마다 콤마 표시해 줌
  - BIN(숫자), HEX(숫자), OCT(숫자)
    - 2진수, 16진수, 8진수의 값을 반환
  - INSERT(기준 문자열, 위치, 길이, 삽입할 문자열)
    - 기준 문자열의 위치부터 길이만큼 지우고 삽입할 문자열 끼워 넣음

```
SELECT INSERT('abcdefghi', 3, 4, '@@@@'), INSERT('abcdefghi', 3, 2, '@@@@');
```

    - 'ab@@@@ghi'와 'ab@@@@efghi' 반환

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - LEFT(문자열, 길이), RIGHT(문자열, 길이)
    - 왼쪽 또는 오른쪽에서 문자열의 길이만큼 반환

```
SELECT LEFT('abcdefghi', 3), RIGHT('abcdefghi', 3);
```

    - 'abc'와 'ghi' 반환

  - UPPER(문자열), LOWER(문자열)
    - 소문자를 대문자로, 대문자를 소문자로 변경
  - LPAD(문자열, 길이, 채울 문자열), RPAD(문자열, 길이, 채울 문자열)
    - 문자열을 길이만큼 늘린 후에 빈 곳을 채울 문자열로 채움

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - LTRIM(문자열), RTRIM(문자열)
    - 문자열의 왼쪽/오른쪽 공백을 제거, 중간의 공백은 제거되지 않음
  - TRIM(문자열), TRIM(방향 자를_문자열 FROM 문자열)
    - TRIM(문자열)은 문자열의 앞뒤 공백을 모두 없앰
    - TRIM(방향 자를_문자열 FROM 문자열) 에서 방향은 LEADING(앞), BOTH(양쪽), TRAILING(뒤) 으로 표시

```
SELECT TRIM('    이것이    '), TRIM(BOTH 'ㅋ' FROM 'ㅋㅋㅋ재밌어요.ㅋㅋㅋ');
```

    - '이것이'와 '재밌어요.' 반환

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - REPEAT(문자열, 횟수)
    - 문자열을 횟수만큼 반복
  - REPLACE(문자열, 원래 문자열, 바꿀 문자열)
    - 문자열에서 원래 문자열을 찾아서 바꿀 문자열로 바꿈

```
SELECT REPLACE ('이것이 MySQL이다', '이것이' , 'This is');
```

    - 'This is MySQL이다' 반환

  - REVERSE(문자열)
    - 문자열의 순서를 거꾸로 바꿈

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 문자열 함수
  - SPACE(길이)
    - 길이만큼의 공백을 반환
  - SUBSTRING(문자열, 시작위치, 길이) 또는 SUBSTRING(문자열 FROM 시작위치 FOR 길이)
    - 시작위치부터 길이만큼 문자를 반환, 길이가 생략되면 문자열의 끝까지 반환

```
SELECT SUBSTRING('대한민국만세', 3, 2);
```

    - '민국' 반환

  - SUBSTRING_INDEX(문자열, 구분자, 횟수)
    - 문자열에서 구분자가 왼쪽부터 횟수 번째까지 나오면 그 이후의 오른쪽은 버림
    - 횟수가 음수면 오른쪽부터 세고 왼쪽을 버림

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 수학 함수
  - ABS(숫자)
    - 숫자의 절댓값 계산
  - ACOS(숫자), ASIN(숫자), ATAN(숫자), ATAN2(숫자1, 숫자2), SIN(숫자), COS(숫자), TAN(숫자)
    - 삼각 함수와 관련된 함수 제공
  - CEILING(숫자), FLOOR(숫자), ROUND(숫자)
    - 올림, 내림, 반올림 계산
  - CONV(숫자, 원래 진수, 변환할 진수)
    - 숫자를 원래 진수에서 변환할 진수로 계산
  - DEGREES(숫자), RADIANS(숫자), PI ( )
    - 라디안 값을 각도값으로, 각도값을 라디안 값으로 변환, PI( )는 3.141592 반환
  - EXP(X), LN(숫자), LOG(숫자), LOG(밑수, 숫자), LOG2(숫자), LOG10(숫자)
    - 지수, 로그와 관련된 함수 제공

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 수학 함수
  - MOD(숫자1, 숫자2) 또는 숫자1 % 숫자2 또는 숫자1 MOD 숫자2
    - 숫자1을 숫자2로 나눈 나머지 값을 구함
  - POW(숫자1, 숫자2), SQRT(숫자)
    - 거듭제곱값 및 제곱근을 구함
  - RAND( )
    - RAND( )는 0 이상 1 미만의 실수 구함
    - 'm<= 임의의 정수 < n'를 구하고 싶다면 FLOOR(m + (RAND( ) * (n-m)) 사용
  - SIGN(숫자)
    - 숫자가 양수, 0, 음수인지 판별, 결과는 1, 0, -1 셋 중에 하나 반환
  - TRUNCATE(숫자, 정수)
    - 숫자를 소수점을 기준으로 정수 위치까지 구하고 나머지는 버림

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 날짜 및 시간 함수
  - ADDDATE(날짜, 차이), SUBDATE(날짜, 차이)
    - 날짜를 기준으로 차이를 더하거나 뺀 날짜 구함
  - ADDTIME(날짜/시간, 시간), SUBTIME(날짜/시간, 시간)
    - 날짜/시간을 기준으로 시간을 더하거나 뺀 결과를 구함
  - CURDATE( ), CURTIME( ), NOW( ), SYSDATE( )
    - CURDATE( ) : 현재 연-월-일
    - CURTIME( ) : 현재 시 : 분 : 초
    - NOW( ), SYSDATE( ) : 현재 '연-월-일 시 : 분 : 초
  - YEAR(날짜), MONTH(날짜), DAY(날짜), HOUR(시간), MINUTE(시간), SECOND(시간), MICROSECOND(시간)
    - 날짜 또는 시간에서 연, 월, 일, 시, 분, 초, 밀리초 구함

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 날짜 및 시간 함수
  - DATE( ), TIME( )
    - DATETIME 형식에서 연-월-일 및 시 : 분 : 초만 추출
  - DATEDIFF(날짜1, 날짜2), TIMEDIFF(날짜1 또는 시간1, 날짜1 또는 시간2)
    - DATEDIFF( )는 날짜1-날짜2의 일수를 결과로 구함
  - DAYOFWEEK(날짜), MONTHNAME( ), DAYOFYEAR(날짜)
    - 요일(1:일, 2:월~7:토) 및 1년 중 몇 번째 날짜인지 구함
  - LAST_DAY(날짜)
    - 주어진 날짜의 마지막 날짜를 구함

SECTION 01 MySQL의 데이터 형식

MySQL 내장 함수

- 날짜 및 시간 함수
  - MAKEDATE(연도, 정수)
    - 연도에서 정수만큼 지난 날짜 구함
  - MAKETIME(시, 분, 초)
    - 시, 분, 초를 이용해서 '시 : 분 : 초'의 TIME 형식 만듦
  - PERIOD_ADD(연월, 개월수), PERIOD_DIFF(연월1, 연월2)
    - PERIOD_ADD( )는 연월에서 개월만큼의 개월이 지난 연월 구함
    - PERIOD_DIFF( )는 연월1-연월2의 개월수 구함
  - QUARTER(날짜)
    - 날짜가 4분기 중에서 몇 분기인지를 구함
  - TIME_TO_SEC(시간)
    - 시간을 초 단위로 구함

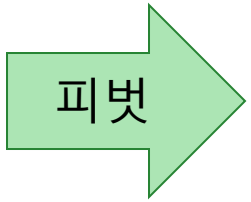
SECTION 01 MySQL의 데이터 형식

피벗의 구현

◦ 피벗(Pivot) 이란?

- 한 열에 포함된 여러 값 출력, 이를 여러 열로 변환하여 테이블 반환식 회전, 필요하면 집계까지 수행

uName	season	amount
김범수	겨울	10
윤종신	여름	15
김범수	가을	25
김범수	봄	3
김범수	봄	37
윤종신	겨울	40
김범수	여름	14
김범수	겨울	22
윤종신	여름	64



uName	봄	여름	가을	겨울	합계
김범수	40	14	25	32	111
윤종신	0	79	0	40	119

SECTION 01 MySQL의 데이터 형식

JSON 데이터

◦ JSON (JavaScript Object Notation)이란?

- 웹과 모바일 응용프로그램 등과 데이터 교환하기 위한 개방형 표준 포맷
- 속성(Key) 과 값(Value)으로 쌍을 이루며 구성
- JavaScript 언어에서 파생
- 특정한 프로그래밍 언어에 종속되어 있지 않은 독립적인 데이터 포맷
- 포맷이 단순하고 공개되어 있기에 거의 대부분의 프로그래밍 언어에서 쉽게 읽거나 쓸 수 있도록 코딩 가능
- MySQL 5.7.8부터 지원
- MySQL 8.0에서는 인라인 패스라 불리는 → 연산자 및 JSON_ARRAYAGG(), JSON_OBJECTAGG(), JSON_PRETTY(), JSON_STORAGE_SIZE(), JSON_STORAGE_FREE(), JSON_MERGE_PATCH()등의 함수가 추가됨

	JSON 값
▶	{"name": "임재범", "height": 182}
	{"name": "이승기", "height": 182}
	{"name": "성시경", "height": 186}

SECTION 02 조인

조인(Join)

- 조인
  - 두 개 이상의 테이블을 서로 묶어서 하나의 결과 집합으로 만들어 내는 것
  - 종류 : INNER JOIN, OUTER JOIN, CROSS JOIN, SELF JOIN

- 데이터베이스의 테이블
  - 중복과 공간 낭비를 피하고 데이터의 무결성을 위해서 여러 개의 테이블로 분리하여 저장
  - 분리된 테이블들은 서로 관계(Relation)를 가짐
  - 1대 다 관계 보편적

SECTION 02 조인

INNER JOIN(내부 조인)

◦ 조인 중에서 가장 많이 사용되는 조인

- 대개의 업무에서 조인은 INNER JOIN 사용
- 일반적으로 JOIN이라고 얘기하는 것이 이 INNER JOIN 지칭
- 사용 형식

```
SELECT <열 목록>
FROM <첫 번째 테이블>
        INNER JOIN <두 번째 테이블>
        ON <조인될 조건>
[WHERE 검색조건]
```

- JOIN만 써도 INNER JOIN으로 인식함

SECTION 02 조인

INNER JOIN(내부 조인)

◦ 조인 중에서 가장 많이 사용되는 조인

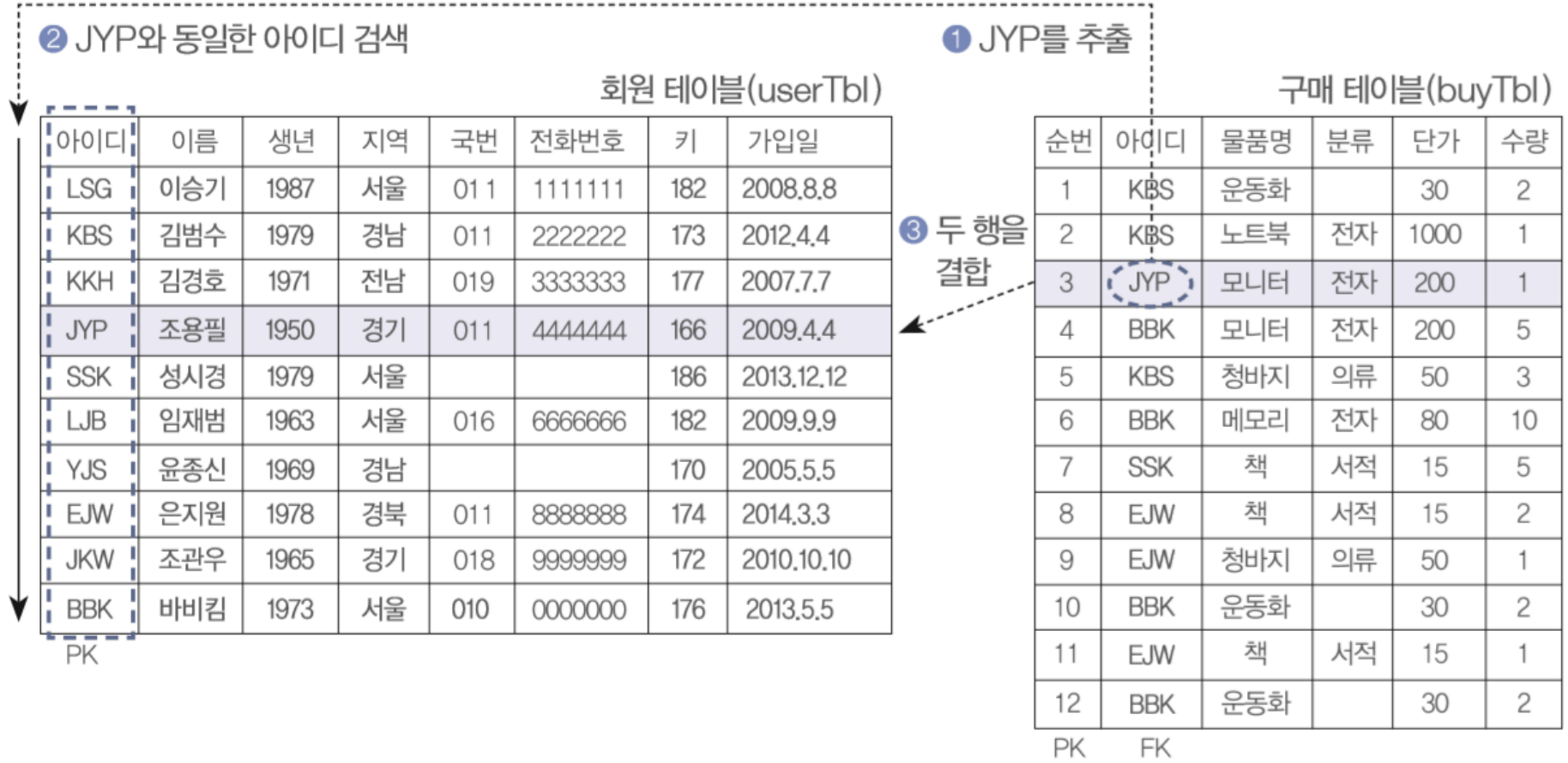
```
USE sqldb;
SELECT *
   FROM buytbl
     INNER JOIN usertbl
        ON buytbl.userID = usertbl.userID
   WHERE buytbl.userID = 'JYP';
```

num	userID	prodName	groupName	price	amount	userID	name	birthYear	addr	mobile1	mobile2	height	mDate
3	JYP	모니터	전자	400	1	JYP	조용필	1950	경기	011	4444444	166	2009-04-04

SECTION 02 조인

INNER JOIN(내부 조인)

◦ 조인 중에서 가장 많이 사용되는 조인



회원 테이블(userTbl)

아이디	이름	생년	지역	국번	전화번호	키	가입일
LSG	이승기	1987	서울	011	1111111	182	2008.8.8
KBS	김범수	1979	경남	011	2222222	173	2012.4.4
KKH	김경호	1971	전남	019	3333333	177	2007.7.7
JYP	조용필	1950	경기	011	4444444	166	2009.4.4
SSK	성시경	1979	서울			186	2013.12.12
LJB	임재범	1963	서울	016	6666666	182	2009.9.9
YJS	윤종신	1969	경남			170	2005.5.5
EJW	은지원	1978	경북	011	8888888	174	2014.3.3
JKW	조관우	1965	경기	018	9999999	172	2010.10.10
BBK	바비킴	1973	서울	010	0000000	176	2013.5.5

PK

구매 테이블(buyTbl)

순번	아이디	물품명	분류	단가	수량
1	KBS	운동화		30	2
2	KBS	노트북	전자	1000	1
3	JYP	모니터	전자	200	1
4	BBK	모니터	전자	200	5
5	KBS	청바지	의류	50	3
6	BBK	메모리	전자	80	10
7	SSK	책	서적	15	5
8	EJW	책	서적	15	2
9	EJW	청바지	의류	50	1
10	BBK	운동화		30	2
11	EJW	책	서적	15	1
12	BBK	운동화		30	2

PK FK

[그림 7-28] INNER JOIN의 작동

SECTION 02 조인

OUTER JOIN(외부 조인)

◦ 조인의 조건에 만족되지 않는 행까지도 포함시키는 것

```
SELECT <열 목록>
FROM <첫 번째 테이블(LEFT 테이블)>
 <LEFT ¦ RIGHT ¦ FULL> OUTER JOIN <두 번째 테이블(RIGHT 테이블)>
      ON <조인될 조건>
[WHERE 검색조건] ;
```

◦ LEFT OUTER JOIN

- 왼쪽 테이블의 것은 모두 출력되어야 한다로 이해
- 줄여서 LEFT JOIN으로 쓸수있음

◦ RIGHT OUTER JOIN

- 오른쪽 테이블의 것은 모두 출력되어야 한다로 이해

SECTION 02 조인

OUTER JOIN(외부 조인)

◦ LEFT OUTER JOIN

```
USE sqldb;
SELECT U.userID, U.name, B.prodName, U.addr, CONCAT(U.mobile1, U.mobile2)  AS '연락처'
   FROM usertbl U
      LEFT OUTER JOIN buytbl B
         ON U.userID = B.userID
   ORDER BY U.userID;
```

userID	name	prodName	addr	연락처
BBK	바비킴	모니터	서울	0100000000
BBK	바비킴	메모리	서울	0100000000
BBK	바비킴	운동화	서울	0100000000
BBK	바비킴	운동화	서울	0100000000
EJW	은지원	책	경북	0118888888
EJW	은지원	청바지	경북	0118888888
EJW	은지원	책	경북	0118888888
JKW	조관우	NULL	경기	0189999999
JYP	조용필	모니터	경기	0114444444
KBS	김범수	운동화	경남	0112222222
KBS	김범수	노트북	경남	0112222222
KBS	김범수	청바지	경남	0112222222
KKH	김경호	NULL	전남	0193333333
LJB	임재범	NULL	서울	0166666666
LSG	이승기	NULL	서울	0111111111
SSK	성시경	책	서울	NULL
YJS	윤종신	NULL	경남	NULL

SECTION 02 조인

CROSS JOIN(상호 조인)

- 한쪽 테이블의 모든 행들과 다른 쪽 테이블의 모든 행을 조인시키는 기능
- CROSS JOIN의 결과 개수 = **두 테이블 개수를 곱한 개수**

회원 테이블(userTbl)

아이디	이름	생년	지역	국번	전화번호	키	가입일
LSG	이승기	1987	서울	011	1111111	182	2008.8.8
KBS	김범수	1979	경남	011	2222222	173	2012.4.4
KKH	김경호	1971	전남	019	3333333	177	2007.7.7
JYP	조용필	1950	경기	011	4444444	166	2009.4.4
SSK	성시경	1979	서울			186	2013.12.12
LJB	임재범	1963	서울	016	6666666	182	2009.9.9
YJS	윤종신	1969	경남			170	2005.5.5
EJW	은지원	1978	경북	011	8888888	174	2014.3.3
JKW	조관우	1965	경기	018	9999999	172	2010.10.10
BBK	바비킴	1973	서울	010	0000000	176	2013.5.5

PK

구매 테이블(buyTbl)

순번	아이디	물품명	분류	단가	수량
1	KBS	운동화		30	2
2	KBS	노트북	전자	1000	1
3	JYP	모니터	전자	200	1
4	BBK	모니터	전자	200	5
5	KBS	청바지	의류	50	3
6	BBK	메모리	전자	80	10
7	SSK	책	서적	15	5
8	EJW	책	서적	15	2
9	EJW	청바지	의류	50	1
10	BBK	운동화		30	2
11	EJW	책	서적	15	1
12	BBK	운동화		30	2

PK FK

[그림 7-43] CROSS JOIN(상호 조인) 방식

SECTION 02 조인

CROSS JOIN(상호 조인)

- 테스트로 사용할 많은 용량의 데이터를 생성할 때 주로 사용
- ON 구문을 사용할 수 없음
- 대량의 데이터를 생성하면 시스템이 다운되거나 디스크 용량이 모두 찰 수 있어 COUNT(*) 함수로 개수만 카운트

```
USE employees;
SELECT  COUNT(*) AS '데이터개수'
   FROM employees
     CROSS JOIN titles;
```

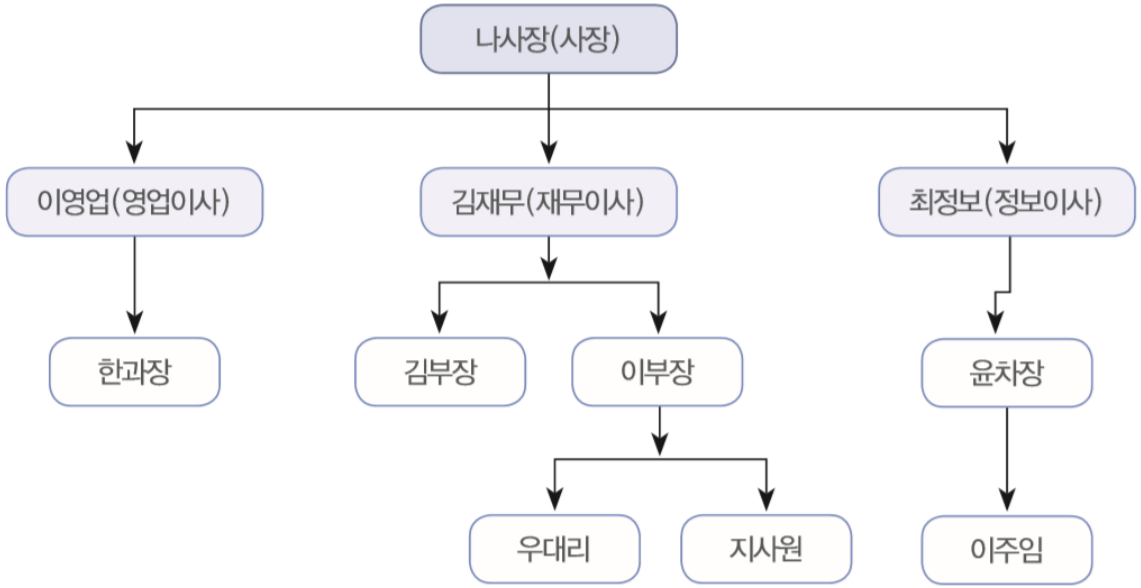
	데이터개수
▶	133003039392

SECTION 02 조인

SELF JOIN(자체 조인)

◦ 자기 자신과 자기 자신이 조인한다는 의미

- 대표적인 예
  - 조직도와 관련된 테이블



[그림 7-45] 간단한 조직도 예

직원 이름(EMP) - 기본 키	상관 이름(MANAGER)	구내 번호
나사장	없음 (NULL)	0000
김재무	나사장	2222
김부장	김재무	2222-1
이부장	김재무	2222-2
우대리	이부장	2222-2-1
지사원	이부장	2222-2-2
이영업	나사장	1111
한과장	이영업	1111-1
최정보	나사장	3333
윤차장	최정보	3333-1
이주임	윤차장	3333-1-1

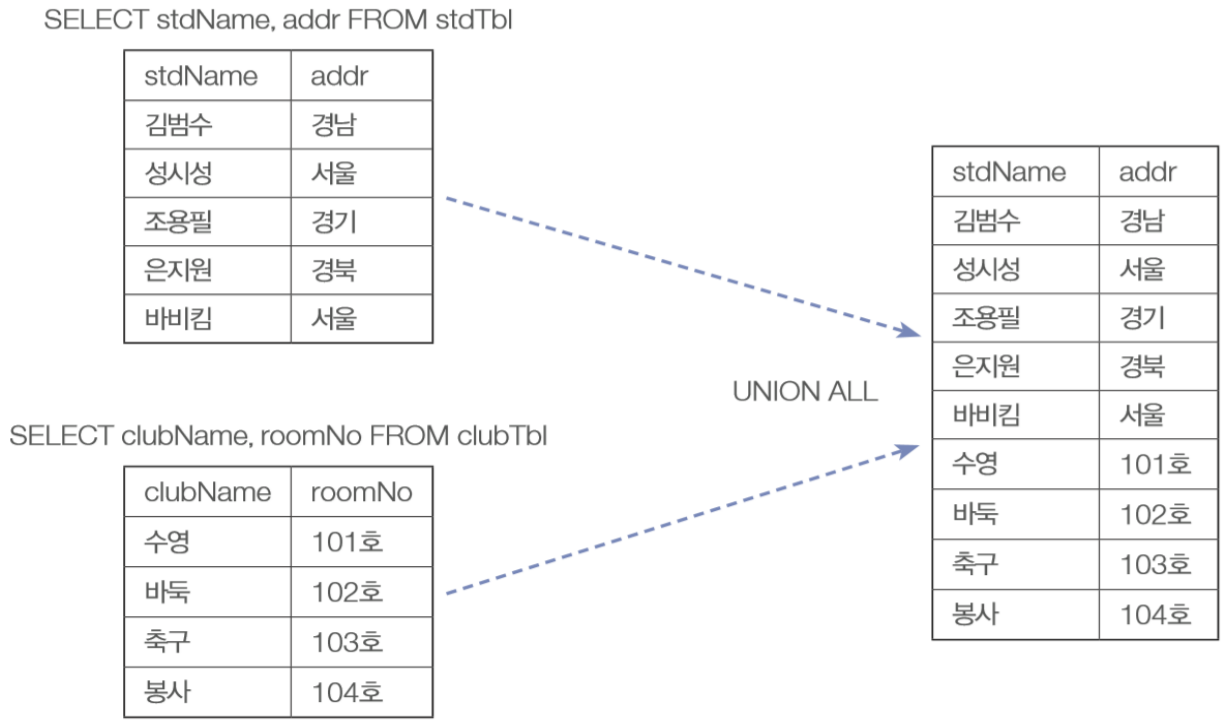
[표 7-5] 조직도 테이블

SECTION 02 조인

UNION / UNION ALL / NOT IN / IN

◦ 두 쿼리의 결과를 행으로 합치는 것

```
SELECT 문장1
   UNION [ALL]
SELECT 문장2
```



[그림 7-47] UNION의 결합과정

SECTION 03 SQL 프로그래밍

스토어드 프로시저를 이용한 프로그래밍

- 10장에서 심화 학습
- 형식

```
DELIMITER $$
CREATE PROCEDURE 스토어드 프로시저이름()
BEGIN

    이 부분에 SQL 프로그래밍 코딩..

END $$
DELIMITER ;
CALL 스토어드 프로시저이름();
```

IF...ELSE

- 조건에 따라 분기
  - 참 / 거짓 두가지만 있기에 **2중 분기**
- 한 문장 이상 처리되어야 할때 BEGIN.. END로 묶어주기
- 형식

```
형식:
IF <부울 표현식> THEN
        SQL문장들1..
ELSE
        SQL문장들2..
END IF;
```

  - 부울 표현식 부분이 참이면 SQL문장들1 수행 / 거짓이면 SQL문장들2 수행

CASE

- 조건에 따라 분기
  - **다중 분기**
  - 조건에 맞는 WHEN이 여러 개더라도 먼저 조건이 만족하는 WHEN 처리됨
  - SELECT문에서 많이 사용됨
  - 점수로 성적을 판단하는 경우처럼 여러 단계로 분기 될 때 사용

WHILE과 ITERATE/LEAVE

◦ WHILE문

- 다른 프로그래밍 언어의 WHILE과 동일한 개념
- 해당 부울식이 참인 동안에 계속 반복되는 반복문

```
형식:
WHILE <부울 식> DO
      SQL 명령문들…
END WHILE;
```

- ITERATE문을 만나면 WHILE문으로 이동해서 비교를 다시함
  - 다른 프로그래밍 언어의 Continue와 동일한 개념
- LEAVE문을 만나면 WHILE문을 빠져 나옴
  - 다른 프로그래밍 언어의 Break와 동일한 개념

오류 처리

◦ 형식

```
DECLARE 액션 HANDLER FOR 오류조건 처리할_문장;
```

- 액션
  - 오류 발생 시에 행동 정의
  - CONTINUE와 EXIT 둘 중 하나 사용, CONTINUE가 나오면 제일 뒤의 '처리할_문장' 부분이 처리
- 오류조건 : 어떤 오류를 처리할 것인지를 지정
  - MySQL의 오류 코드 숫자가 오거나 SQLSTATE'상태코드', SQLEXCEPTION, SQLWARNING, NOT FOUND등이 올 수 있음
- 처리할_문장
  - 처리할 문장이 여러 개일 경우에는 BEGIN...END로 묶어줌

SECTION 03 SQL 프로그래밍

오류 처리

```
DROP PROCEDURE IF EXISTS errorProc;
DELIMITER $$
CREATE PROCEDURE errorProc()
BEGIN
    DECLARE CONTINUE HANDLER FOR 1146 SELECT '테이블이 없어요ㅠㅠ' AS '메시지';
    SELECT * FROM noTable;  -- noTable은 없음.
END $$
DELIMITER ;
CALL errorProc();
```

```
결과 값:
테이블이 없어요ㅠㅠ
```

SECTION 03 SQL 프로그래밍

동적 SQL

- PREPARE문
  - SQL문을 실행하지는 않고 미리 준비만 해놓음

- EXECUTE문
  - 준비한 쿼리문 실행
  - 실행 후에는 DEALLOCATE PREFARE로 문장 해제