
MARIA DATABASE

설치 및 활용

MARIADB 설치

<https://downloads.mariadb.org/>

MariaDB 10.1 Series

MariaDB 10.1 is a **stable** (GA) release of MariaDB. It is built on [MariaDB 10.0](#) with features from MySQL 5.6 & 5.7, and entirely new features not found anywhere else.

See "[What is MariaDB 10.1?](#)" for an overview.

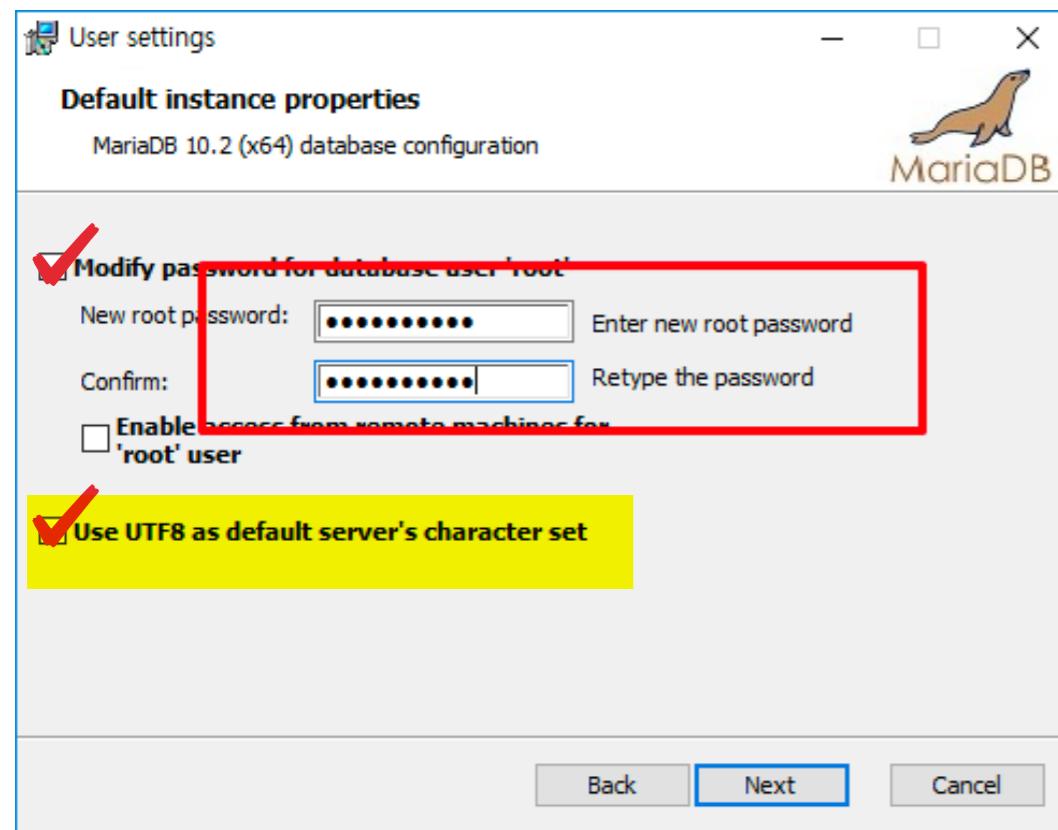
[Download 10.1.44 Stable Now!](#)

[Release Notes](#) [Changelog](#)

[View All MariaDB Releases](#)

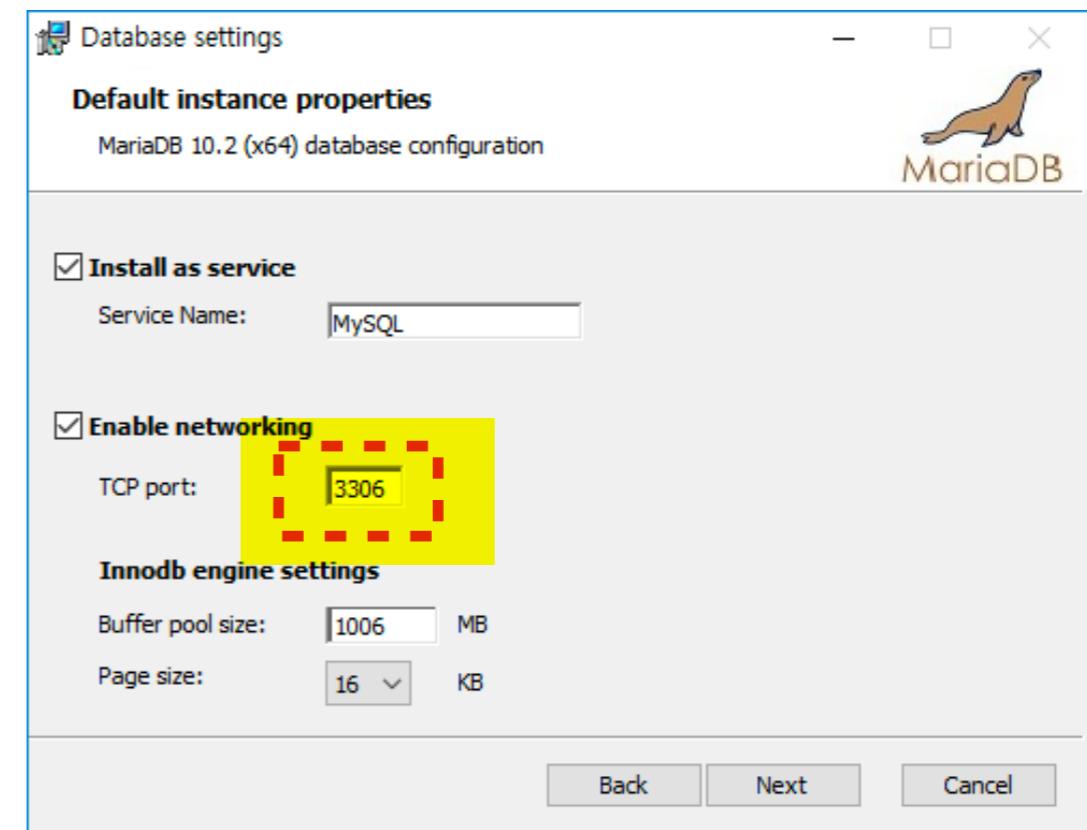
File Name	Package Type	OS / CPU	Size	Meta
Galera 25.3.28 source and packages		Source		
For best results with RPM and DEB packages, use the Repository Configuration Tool .				
mariadb-10.1.44.tar.gz	source tar.gz file	Source	63.7 MB	Checksum Instructions
mariadb-10.1.44-winx64.zip	ZIP file	Windows x86_64	181.5 MB	Checksum Instructions
mariadb-10.1.44-winx64.msi	MSI Package	Windows x86_64	57.0 MB	Checksum Instructions
mariadb-10.1.44-win32.msi	MSI Package	Windows x86	55.3 MB	Checksum Instructions

MARIADB 설치시 체크포인트



root password : root1234 로 통일

checkbox 둘다 체크하기



TCP port는 3306 만약, 충돌나면 3307

이후 부터는 Next 버튼 클릭하여 설치

데이터 베이스, 계정, 권한 생성

```
>mysql -u root -p mysql
```

사용자

데이터베이스

기본적으로 제공하는 데이터 말고 우리끼 데이터베이스를 만들고 사용자를 따로 만드는 과정
1.root 유저로 접속
root의 비밀번호 : kpc1212로 지정해놨음.
2.db 생성
3.user 생성
4.권한 부여
5.flush privileges
6.user 접속 /비밀번호 입력

```
>root 비밀번호입력
```

```
MariaDB [mysql]>create database kpc;
```

데이터베이스

```
MariaDB [mysql]> create user 'kpc12'@'localhost' identified by 'kpc  
1212';
```

사용자

접속위치

비밀번호

```
MariaDB [mysql]> grant all privileges on kpc.* to 'kpc12'@'localhost';
```

데이터베이스

사용자

접속위치

```
MariaDB [mysql]>flush privileges;
```

권한이 부여되면 mariadb 서버를 리로딩해야 함

exit로 빠져나간 후 다시 mysql에 kpc12 유
저로 실행을 시작한다.

```
>mysql -u kpc12 -p kpc
```



kpc12로 로그인 한 후 첨부한 sql 파일을 kpc db를 기준으로 하
여 실행시킨 후 확인해본다.

```
>kpc1212 비밀번호입력
```

CREATE TABLE 테이블명 (칼럼명 TYPE 제약조건, ...);

테이블명

```
MariaDB [kic]> create table member(  
    -> num int primary key,  
    -> name varchar(50),  
    -> addr varchar(50)  
    -> );  
Query OK, 0 rows affected (0.012 sec)
```

칼럼명

50 글자가 X
50 Byte O

속도 : char > varchar
--> 절대 바꿔지 않는 것들은
char로 정의 : 주민번호

CREATE TABLE 테이블명 (칼럼명 TYPE 제약조건, ...);

테이블 만들때 사용하는 예약어

```
MariaDB [kic]> create table member(
    -> num int primary key,
    -> name varchar(50),
    -> addr varchar(50)
    -> );
```

Query OK, 0 rows affected (0.012 sec)

칼럼의 데이터 TYPE

NUMBER => 숫자 Type

VARCHAR(50) => 가변 문자열 최대 영문자 50 (한글:25) 글자

CREATE TABLE 테이블명 (칼럼명 TYPE 제약조건, ...);

```
MariaDB [kic]> create table member(
-> num int primary key,
-> name varchar(50),
-> addr varchar(50)
-> );
```

Query OK, 0 rows affected (0.012 sec)

NOT NULL + UNIQUE 조건

1. 반드시 값을 넣어주어야한다.
2. 중복된 값을 허용하지 않는다.

DESC 테이블명

생성한 테이블의 구조 보기

```
[MariaDB [kic]]> desc member;
+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| num   | int(11)       | NO   | PRI | NULL    |       |
| name  | varchar(50)  | YES  |     | NULL    |       |
| addr  | varchar(50)  | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+
3 rows in set (0.004 sec)
```

INSERT INTO 테이블명 (칼럼명, ...) VALUES (값1, ...);

테이블에 데이터 저장하기



[MariaDB [kic]]> insert into member(num, name, addr) values(1, '성 영 한 ', '서 울 ');
Query OK, 1 row affected (0.001 sec)

특정 칼럼에만 값을 넣을 수도 있다.

[MariaDB [kic]]> insert into member(num, name) values(2, '손 정 의 ');\n**Query OK, 1 row affected (0.006 sec)**

addr : null로 insert 된다.

[MariaDB [kic]]> insert into member values(3, '스 티 브 잡 스 ', '미 국 ');\n**Query OK, 1 row affected (0.002 sec)**

모든 칼럼의 값을 순서대로 넣어줄 때는
칼럼명 생략 가능하다.

위의 방식은 추천하지 X
WHY? 테이블은 지속적으로 업데이트 되면서
칼럼의 개수가 바뀐다. (대개 증가한다.)
그럴 때, 위의 방식은 호환되지 않고,
에러를 발생한다.

실무에선 : 칼럼 수가 100개~200개 ..

SELECT 칼럼명1,.. FROM 테이블명

테이블에 저장된 데이터 출력해보기

보고 싶은 칼럼명을 나열한다.

```
[MariaDB [kic]]> select num, name, addr from member;
```

num	name	addr
1	성영한	서울
2	손정의	NULL
3	스티브잡스	미국

```
3 rows in set (0.001 sec)
```

SELECT 칼럼명1,.. FROM 테이블명

테이블에 저장된 데이터 출력해보기

보고 싶은 칼럼명을 나열한다.

```
MariaDB [kic]> select num, name from member;
+----+-----+
| num | name |
+----+-----+
| 1 | 성영한 |
| 2 | 손정의 |
| 3 | 스티브잡스 |
+----+-----+
3 rows in set (0.000 sec)
```

SELECT 칼럼명1,.. FROM 테이블명

테이블에 저장된 데이터 출력해보기

모든 칼럼의 내용을 다 보고 싶을때는 * 로 대치

```
MariaDB [kic]> select * from member;
+----+-----+-----+
| num | name          | addr   |
+----+-----+-----+
| 1  | 성영한         | 서울    |
| 2  | 손정의         | NULL    |
| 3  | 스티브잡스     | 미국    |
+----+-----+-----+
3 rows in set (0.000 sec)
```

* 쓰지 마세요.
노트 참고
WHY? : 쓸 데 없는 거까지 가져와서 성능저하.
Ex) user(web browser 단) :
ID와 PW를 통해서 로그인 시도
==> server(Web Server단) :
ID /PW 가 맞는지 확인해야 함
SQL문을 JDBC를 통해 DB에 전달
==> DB(In Disk)
ID/PW 가 맞는지 table을 통해 확인.
==> Table
SQL문 확인 후 정보 반환
(이 때 필요한 정보 : ONLY ID & PW인데
*을 하면 끌데 없는 정보까지 모두 반환
==> 성능저하)

DELETE FROM 테이블명 WHERE 조건절

테이블에서 특정 ROW를 삭제

```
[MariaDB [kic]]> delete from member where num = 2;  
Query OK, 1 row affected (0.002 sec)
```

```
[MariaDB [kic]]> select * from member;  
+----+-----+-----+  
| num | name          | addr      |  
+----+-----+-----+  
| 1   | 성영한         | 서울      |  
| 3   | 스티브잡스     | 미국      |  
+----+-----+-----+  
2 rows in set (0.000 sec)
```

PRIMARY KEY 값으로 지정된
칼럼을 조건절에서 이용한다.

UPDATE 테이블명 SET 칼럼명1=값1,.. WHERE 조건절

특정 ROW 의 데이터 수정하기

```
[MariaDB [kic]]> update member set addr='캐나다' where num = 3;  
Query OK, 1 row affected (0.001 sec)  
Rows matched: 1  Changed: 1  Warnings: 0
```

```
[MariaDB [kic]]> select * from member;  
+----+-----+-----+  
| num | name          | addr      |  
+----+-----+-----+  
| 1  | 성영한        | 서울      |  
| 3  | 스티브잡스    | 캐나다    |  
+----+-----+-----+  
2 rows in set (0.000 sec)
```

PRIMARY KEY 값으로 지정된
칼럼을 조건절에서 이용한다.

SCOTT.SQL 파일 MARIADB IMPORT 하기

본인계정으로 접속

```
youngui-MacBook-Pro:~ syh$ mysql -u kic12 -p kic  
Enter password:  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A
```

Welcome to the MariaDB monitor. Commands end with ; or \g.

Your MariaDB connection id is 31

Server version: 10.3.12-MariaDB Homebrew

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
MariaDB [kic]> source scott.sql
```

Mac에서 사용 가능.

이때 SCOTT.SQL 파일의 위치는
본인 계정으로 접속한 곳에 있어야 함

/Users/syh/scott.sql

1. DQL (DATA QUERY LANGUAGE)

[SELECT 기본형식]

SELECT 칼럼명1, 칼럼명2, ...

<< 실행순서 >> .5

FROM 테이블명

.1

WHERE 조건절

.2

GROUP BY 칼럼명

.3

HAVING 조건절

.4

ORDER BY 칼럼명 [ASC|DESC] => 오름 차순 혹은 내림차순. .6

LIMIT 시작인덱스(0부터), 길이

.7

SELECT문 사용하기

emp 테이블에서 사원번호, 사원이름, 직업을 출력.

MariaDB [kic]>

emp 테이블에서 사원번호, 급여, 부서번호를 출력하세요. 단, 급여가 많은 순서대로 출력

MariaDB [kic]>

emp 테이블에서 사원번호, 급여, 입사일을 출력. 단, 급여가 적은 순서대로.

MariaDB [kic]>

emp 테이블에서 직업, 급여를 출력. 단, 직업명으로 오름차순, 급여로 내림차순 정렬해서

MariaDB [kic]>

ALIAS 사용하기 (칼럼에 별칭 붙이기)

MariaDB [kic]> SELECT empno AS "사원번호" , ename AS "사원이름" FROM emp ;

-AS 와 큰따옴표 " 는 생략가능 하다(공백문자가 있으면 안됨 => 사원 번호)



공백문자 있으면
" " 안에 넣어줘
야 함.

MariaDB [kic]> SELECT empno 사원번호 , ename 사원이름 FROM emp ;

연산자-산술 연산자 (+ , - , * , /)

emp 테이블에서 부서번호가 10 번인 사원들의 급여와 10% 인상된 금액 같이 출력

MariaDB [kic]>

연산자-비교 연산자 (=, !=, >, <, >=, <=)

emp 테이블에서 급여가 3000 이상인 사원들의 모든 정보를 출력.

MariaDB [kic]>

emp 테이블에서 부서번호가 30번이 아닌 사람들의 이름과 부서번호를 출력.

MariaDB [kic]>

연산자-논리 연산자 (AND , OR , NOT)

emp 테이블에서 부서번호가 10번이고 급여가 3000 이상인 사원들의 이름과 급여를 출력.

MariaDB [kic]>

emp 테이블에서 직업이 SALESMAN 이거나 MANAGER 인 사원의 사원번호와 부서번호를 출력.

MariaDB [kic]>

연산자-SQL 연산자 (IN , BETWEEN , LIKE , IS NULL , IS NOT NULL)

1> IN 연산자 (OR 연산자와 비슷한 역할)

부서번호가 10번이거나 20번인 사원의 사원번호와 이름, 부서번호 출력.

MariaDB [kic]> SELECT empno, ename, deptno FROM emp

WHERE deptno=10 OR deptno=20 ;

IN 연산자를 사용한다면

MariaDB [kic]> SELECT empno, ename, deptno FROM emp WHERE deptno IN(10,20) ;

연산자-SQL 연산자 (IN , BETWEEN , LIKE , IS NULL , IS NOT NULL)

2> BETWEEN A AND B (A와 B 사이의 포함한 데이터를 얻어온다)

급여가 1000 과 2000 사이인 사원들의 사원번호,이름,급여를 출력.

MariaDB [kic]> SELECT empno, ename, sal FROM emp

WHERE sal BETWEEN 1000 AND 2000 ;

사원이름이 'FORD' 와 'SCOTT' 사이의 사원들의 사원번호,이름을 출력.

MariaDB [kic]> SELECT empno, ename FROM emp

WHERE ename BETWEEN 'FORD' AND 'SCOTT' ;

연산자-SQL 연산자 (IN , BETWEEN , LIKE , IS NULL , IS NOT NULL)

3>LIKE 연산자 (문자열 비교) 중요 !! *****

사원이름이 'J' 로 시작하는 사원의 사원이름과 부서번호를 출력.

MariaDB [kic]>

사원이름이 'J' 를 포함하는 사원의 사원이름과 부서번호를 출력.

MariaDB [kic]>

사원이름의 두번째 글자가 'A' 인 사원의 이름,급여,입사일을 출력.

MariaDB [kic]>

사원 이름이 'ES' 로 끝나는 사원의 이름,급여,입사일을 출력.

MariaDB [kic]>

-입사년도가 81년 인 사원들의 입사일과 사원번호를 출력.

MariaDB [kic]>

연산자-SQL 연산자 (IN , BETWEEN , LIKE , IS NULL , IS NOT NULL)

4> IS NULL (NULL 인경우 TRUE) , IS NOT NULL (NULL 이 아닌경우 TRUE)

커미션이 NULL 인 사원의 사원이름과 커미션을 출력.

MariaDB [kic]>

커미션이 NULL 이 아닌 사원의 사원이름과 커미션을 출력.

MariaDB [kic]>

5> 연결 연산자 (concat()) => 단순히 문자열을 연결해서 하나의 문자열로 리턴한다.

MariaDB [kic]> SELECT concat(ename,'의 직업은 ' ,job,'입니다') FROM emp

2. 함수 (FUNCTION)

- 어떠한 일을 수행하는 기능으로써 주어진 인수를 재료로 처리를 하여 그 결과를 반환하는 일을 수행한다.

함수의 종류

1) 단일행 함수

- 하나의 row 당 하나의 결과값을 반환하는 함수.

2) 복수행 함수

- 여러개의 row 당 하나의 결과값을 반환하는 함수.

단일행 함수 => 문자함수

1> CHR(아스키 코드)

MariaDB [kic]> SELECT CHR(65) ;

2>CONCAT(칼럼명, '붙일문자') =>문자열 연결함수

MariaDB [kic]> SELECT CONCAT(ename, ' 님') name FROM emp ;

3>LOWER('문자열') =>문자열을 소문자로 바꿔준다.

MariaDB [kic]> SELECT LOWER('HELLO!');

4>UPPER('문자열') =>문자열을 대문자로 바꿔준다.

MariaDB [kic]> SELECT UPPER('hello!');

5>LPAD('문자열' , 전체 자리수 , '남는자리를 채울 문자') =>왼쪽에 체운다.

MariaDB [kic]> SELECT LPAD('HI', 10 , '*');

단일행 함수 => 문자함수

6> LENGTH(), CHAR_LENGTH()

MariaDB [kic]> SELECT length('국어');

7>LEFT() : 왼쪽에서 문자열 자르기 :=>왼쪽부터 3자리까지 선택

MariaDB [kic]> SELECT LEFT('abcde',3);

8>RIGHT() : 오른쪽에서 시작 문자열 자르기 : 오른쪽부터 3자리까지 선택

MariaDB [kic]> SELECT RIGHT('abcde', 3);

9>SUBSTRING() : 문자열 중간에서 자르기 : 2번째부터 2자리까지 선택

MariaDB [kic]> SELECT SUBSTRING('abcde', 2, 2);

단일행 함수 => 문자함수

10>FORMAT("처리할 숫자", 자리수) : '처리할 숫자'를 천(1000)단위로 콤파(,)로 표시하고 반올림하여 소수점을 '자리수'까지 표현

MariaDB [kic]> SELECT FORMAT(12345.23456789, 1);

11> CONCAT() : 문자열 합치기

MariaDB [kic]> SELECT CONCAT ('ab', 'cd', 'ef');

12> NVL(칼럼명 , 값) : 해당 칼럼이 NULL 인경우 정해진 값을 반환한다.

MariaDB [kic]> SELECT ename,NVL(comm, 0) FROM emp ;

단일행 함수 => 숫자함수

1>ABS(숫자) => 숫자의 절대값을 반환한다.

MariaDB [kic]> SELECT ABS(-10);

2> CEIL(소수점이 있는 수) : 파라미터 값보다 같거나 가장 큰 정수를 반환(올림)

MariaDB [kic]> SELECT CEIL(3.1234) ;

3> FLOOR(소수점이 있는 수) : 파라미터 값보다 같거나 가장 작은 정수반환(내림)

MariaDB [kic]> SELECT FLOOR(3.2241) ;

4> ROUND(숫자,자리수) : 숫자를 자리수+1 번째 위치에서 반올림한다.

MariaDB [kic]> SELECT ROUND(3.22645, 2) ;

단일행 함수 => 숫자함수

5>MOD(숫자1 , 숫자2) =>숫자1을 숫자2로 나눈 나머지를 리턴한다.

MariaDB [kic]> SELECT MOD(10,3) ;

6>TRUNCATE() : 숫자를 지정한 소숫점 자리수 만큼만 잘라서 보여줌.

MariaDB [kic]> SELECT TRUNCATE(11111.23456789, 3);

단일행 함수 => 날짜함수

1> CURDATE(), CURTIME(), NOW()

MariaDB [kic]> SELECT NOW();

2> DATE_ADD(date, INTERVAL expr TYPE) , ADDDATE(date, INTERVAL expr TYPE)

MariaDB [kic]> SELECT ADDDATE(NOW(),INTERVAL 2 DAY);

3> DATE_SUB(date, INTERVAL expr TYPE) , SUBDATE(date, INTERVAL expr TYPE)

MariaDB [kic]> SELECT SUBDATE(NOW(),INTERVAL 2 HOUR);

4> DATEDIFF(날짜1, 날짜2); , TIMESTAMPDIFF(단위, 날짜1, 날짜2);

MariaDB [kic]> SELECT DATEDIFF('2018-04-02', '2018-03-01');

MariaDB [kic]> SELECT TIMESTAMPDIFF(HOUR, '2018-03-01', '2018-03-28');

단일행 함수 => 날짜를 문자열로 변환함수

1> DATE_FORMAT(date, format)

이 함수를 가장
많이 사용.

MariaDB [kic]> SELECT DATE_FORMAT(now(),'%Y-%m-%d %p %h:%i:%s %W');

금지 X

년도	Y : 2019(네자리) , y : 19(두자리)
월	m : 08(두자리) , c : 8(한자리) , b : Aug(영문)
일	d : 04(두자리) , e : 4(한자리)
시	H: 14(00~23) , h : 04(01 ~ 12)
분	i: 14(00~59)
초	s: 14(00~59)
오전, 오후	p : AM, PM
요일	W : (sunday,..saturday), w: (0~6)

단일행 함수 => 숫자변화함수, 문자열을 날짜로 변환함수

1> CONVERT(COL_1, UNSIGNED)

MariaDB [kic]> SELECT CONVERT('1234', UNSIGNED) + 1234;

2> STR_TO_DATE('날짜에 대응되는 문자','날짜 포맷')

MariaDB [kic]> SELECT STR_TO_DATE('2000-01-31', '%Y-%m-%d');

복수행(그룹) 함수

1> **COUNT(칼럼명)** => 해당 칼럼이 존재하는 ROW 의 갯수를 반환한다. 단, 저장된 데이터가 NULL 인 칼럼은 세지 않는다.

MariaDB [kic]> SELECT COUNT(ename) FROM emp;

MariaDB [kic]> SELECT COUNT(comm) FROM emp;

MariaDB [kic]> SELECT COUNT(*) FROM emp ; => 모든 행(row)의 갯수를 얻어온다.

2> **SUM(칼럼명)** => 해당 칼럼의 값을 모두 더한 값을 리턴한다.

MariaDB [kic]> SELECT SUM(sal) FROM emp;

복수행(그룹) 함수

3> AVG(칼럼명) => 해당 칼럼의 값을 모든값을 더한후 ROW 의 갯수로 나눈 평균값을 리턴한다.

단 NULL 칼럼은 제외된다.

MariaDB [kic]>SELECT AVG(sal) FROM emp;

MariaDB [kic]>SELECT AVG(comm) FROM emp;

ex) comm 이 NULL 인 사원도 평균에 포함 시켜서 출력을 하려면?

hint : NVL() 함수를 이용한다.

MariaDB [kic]>SELECT AVG(NVL(comm , 0)) FROM emp;

복수행(그룹) 함수

4> MAX(칼럼명) => 최대값을 리턴한다.

MariaDB [kic]>SELECT MAX(sal) FROM emp;

5> MIN(칼럼명) =>최소값을 리턴한다.

MariaDB [kic]>SELECT MIN(sal) FROM emp;

GROUP BY 절-그룹으로 묶을 때 사용

emp 테이블에서 부서번호와 부서별 급여의 총합을 출력

MariaDB [kic]>

emp 테이블에서 부서번호와 부서별 평균 급여를 출력.

MariaDB [kic]>

emp 테이블에서 부서번호와 부서별 평균 급여를 출력. (반올림해서 소수 첫째 자리 까지만)

MariaDB [kic]>

emp 테이블에서 직업과 직업별 최대 급여를 출력.

MariaDB [kic]>

GROUP BY 절-그룹으로 묶을 때 사용

emp 테이블에서 부서번호와 급여가 1000 이상인 사원들의 부서별 평균 급여의 반올림 값을 부서번호로 내림차순 정렬해서 출력.

MariaDB [kic]>

emp 테이블에서 부서번호와 급여가 2000 이상인 사원들의 부서별 평균 급여의 반올림 값을 평균 급여의 반올림 값으로 오름차순 정렬해서 출력

MariaDB [kic]>

GROUP BY 절-그룹으로 묶을 때 사용

emp 테이블에서 각 부서별 같은 업무(job)를 하는 사람의 인원수를 구해서 부서번호, 업무(job), 인원수를 부서번호에 대해서 오름차순 정렬해서 출력

MariaDB [kic]>

HAVING BY 절-GROUP BY로 집계된 값에 조건을 사용할 경우

emp 테이블에서 부서번호와 급여가 1000 이상인 사원들의 부서별 평균 급여를 출력. 단, 부서별 평균 급여가 2000 이상인 부서만 출력하세요.

MariaDB [kic]>SELECT deptno, AVG(sal) FROM emp WHERE sal >= 1000

GROUP BY deptno HAVING AVG(sal) >= 2000 ;

3. 조인 (JOIN)

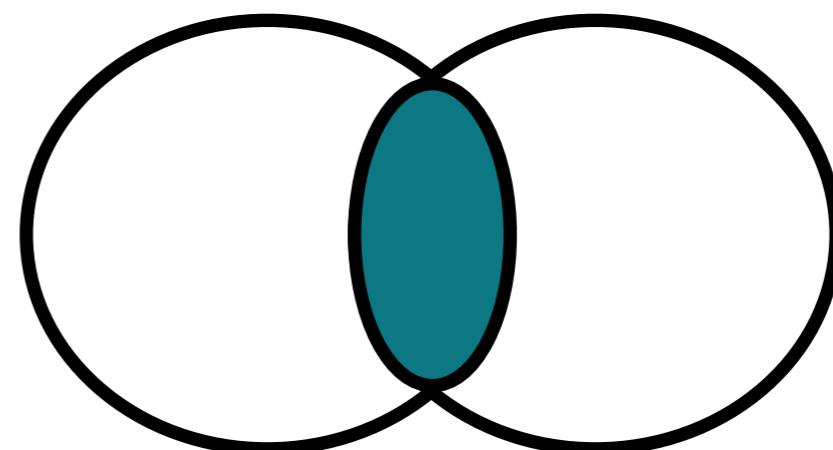
- 하나의 테이블로 원하는 칼럼정보를 참조 할 수 없는 경우 관련된 테이블을 논리적으로 결합하여 원하는 칼럼 정보를 참조하는 방법을 **JOIN** 이라고 한다.

[형식]

SELECT 칼럼명1, 칼럼명2...

FROM 테이블명1, 테이블명2...

WHERE JOIN 조건 AND 다른 조건 ...



JOIN -INNER 조인

emp 테이블의 모든 사원들의 이름, 부서번호, 부서명을 출력.

MariaDB [kic]>SELECT ename, emp.deptno, dname FROM emp, dept

WHERE emp.deptno = dept.deptno ;

* 테이블에 별칭(alias)를 붙인다면

MariaDB [kic]>SELECT ename, e.deptno, dname FROM emp (as) e, dept (as) d

WHERE e.deptno = d.deptno ;

* join ~ on 표현식을 사용한다면

MariaDB [kic]>SELECT ename, e.deptno, dname FROM emp e join dept d

on e.deptno = d.deptno ;

JOIN -**INNER** 조인

emp 테이블의 모든 사원들의 이름, 부서번호, 부서명을 출력.

* ANSI JOIN 표현식을 사용한다면

```
MariaDB [kic]>SELECT ename, e.deptno, dname FROM emp e join dept d  
using(deptno) ;
```

JOIN -**INNER** 조인

급여가 3000에서 5000 사이의 사원이름과 부서명을 출력.

MariaDB [kic]>

부서명이 'ACCOUNTNG'인 사원의 이름, 입사일, 부서번호, 부서명을 출력.

MariaDB [kic]>

JOIN -**INNER** 조인

- 커미션이 null 이 아닌 사원의 이름, 입사일, 부서명을 출력.

MariaDB [kic]>

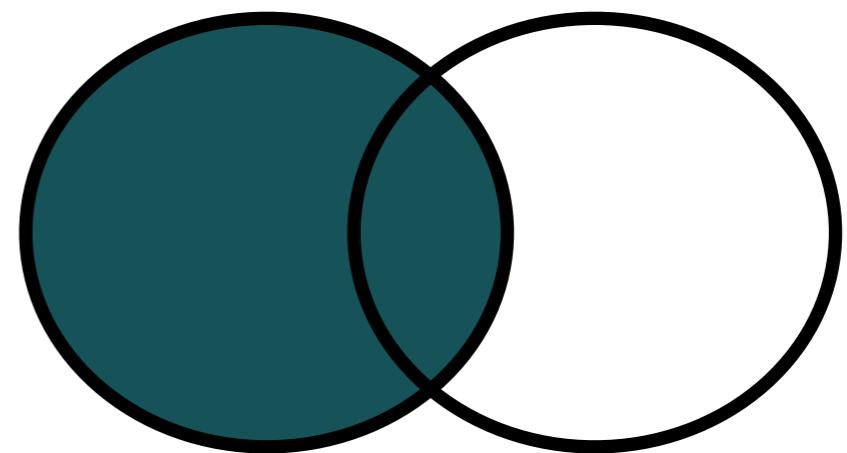
부서명이 'ACCOUNTNG' 인 사원의 이름, 입사일, 부서번호, 부서명을 출력.

MariaDB [kic]>

JOIN -LEFT 조인

- 왼쪽 테이블에는 해당하는 데이터가 존재하는데 오른쪽 테이블에는 데이터가 존재하지 않을때에도 모든 데이터를 추출하도록 하는 JOIN 방법
- 사원번호,부서번호,부서명을 출력하세요 단, 사원이 근무하지 않는 부서명도 같이 출력해보세요.

```
MariaDB [kic]>SELECT e.empno, d.deptno, d.dname FROM emp e LEFT JOIN dept d  
ON e.deptno = d.deptno ;
```



QUIZ

1. emp 테이블과 dept 테이블을 조인하여 부서번호, 부서명, 이름, 급여를 출력.

MariaDB [kic]>

2. 사원의 이름이 'ALLEN' 인 사원의 이름과 부서명을 출력.

MariaDB [kic]>

3. 모든 사원의 이름, 부서번호, 부서명, 급여를 출력. 단, emp 테이블에 없는 부서도 출력해보세요.

MariaDB [kic]>

QUIZ

5. 사원의 이름과 급여, 급여의 등급을 출력.

MariaDB [kic]>

6.사원의 이름과, 부서명, 급여의 등급을 출력.

MariaDB [kic]>

4. SUBQUERY

- 하나의 SQL 문장절에 포함된 또다른 SELECT 문장, 따라서 두번 질의를 해야 얻을수 있는 결과를 한번의 질의로 해결이 가능하게 하는 쿼리

[종류]

<1> 단일행 서브쿼리

- 서브쿼리의 실행결과가 하나의 칼럼과 하나의 행만을 리턴해주는 쿼리
(하나의 데이터만 리턴해주는 쿼리)

<2> 복수행 서브쿼리

- 서브쿼리의 실행결과가 하나의 칼럼과 여러개의 행을 리턴해주는 쿼리
(여러개의 데이터만 리턴해주는 쿼리)

4. SUBQUERY

*** 단일행 서브 쿼리 테스트 ***

1. 'SMITH' 가 근무하는 부서명을 서브쿼리를 이용해서 출력.

MariaDB [kic]> SELECT dname FROM dept

WHERE deptno = (SELECT deptno FROM emp WHERE ename='SMITH');

2. 'ALLEN' 과 같은 부서에서 근무하는 사원의 이름과 부서의 번호를 출력.

MariaDB [kic]>

3. 'ALLEN' 과 동일한 직책(job) 을 가진 사원의 사번과 이름, 직책을 출력.

MariaDB [kic]>

4. SUBQUERY

4. 'ALLEN' 의 급여와 동일하거나 더 많이 받는 사원의 이름과 급여를 출력.

MariaDB [kic]>

5. 'DALLAS' 에서 근무하는 사원의 이름, 부서번호를 출력.

MariaDB [kic]>

6. 'SALES' 부서에서 근무하는 모든 사원의 이름과 급여를 출력.

MariaDB [kic]>

4. SUBQUERY

7. 자신의 직속 상관이 'KING' 인 사원의 이름과 급여를 출력

MariaDB [kic]>

4. SUBQUERY

*** 다중행 서브 쿼리 테스트 ***

- 다중행 서브쿼리의 결과값을 조건절에서 사용할때는 반드시 **다중행 연산자와 함께 사용해야한다.** (IN, ALL, ANY, EXIST)

1. 급여를 3000 이상받는 사원이 소속된 부서와 동일한 부서에서 근무하는 사원들의 이름과 급여, 부서번호를 출력.

MariaDB [kic]> SELECT ename,sal,deptno FROM emp

WHERE deptno IN(SELECT deptno FROM emp WHERE sal>=3000);

4. SUBQUERY

*** 다중행 서브 쿼리 테스트 ***

2. IN 연산자를 이용하여 부서별로 가장 급여를 많이 받는 사원의 사원번호, 급여, 부서번호를 출력

MariaDB [kic]>

3. 직책이 MANAGER 인 사원이 속한 부서의 부서번호와 부서명과 부서의 위치를 출력

MariaDB [kic]>

4. 직책이 'SALESMAN' 보다 급여를 많이 받는 사원들의 이름과 급여를 출력 (ANY 연산자 이용)

MariaDB [kic]>

5. DML (DATA MULITIPULATION LANGUAGE)

-테이블 내의 데이터를 입력, 수정, 삭제

1) INSERT : 테이블에 데이터를 저장할때 사용

[형식]

INSERT INTO 테이블명(칼럼명1, 칼럼명2,...) VALUES (값1, 값2,...)

[입력시 제약 사항]

-한번에 하나의 행만 입력할수 있다

-INSERT 절에 명시되는 칼럼의 갯수와 VALUES 절의 갯수는 일치해야한다.

- 모든 칼럼의 내용을 다 저장할때는 칼럼명은 생략 가능하다.

5. DML (DATA MULITIPULATION LANGUAGE)

예) EMP 테이블에 아래와 같은 사원을 추가 해보세요.

EMPNO : 8000 , ENAME : 최수만, JOB : 방장, MGR : 7900

HIREDATE : 오늘 , SAL : 2000, COMM : 100 , DEPTNO : 40

MariaDB [kic]> INSERT INTO EMP(EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO) VALUES (8000, '최수만', '방장', 7900, SYSDATE, 18, 100, 40);

-연습용 테이블 만들기

MariaDB [kic]> CREATE TABLE member(num int PRIMARY KEY,
name VARCHAR(30),
addr VARCHAR(50)) ;

5. DML (DATA MULITIPULATION LANGUAGE)

2) UPDATE 문

- 데이터를 수정할때 사용하는 문장

[형식]

UPDATE 테이블명 SET 칼럼명1 = 수정값 , 칼럼명2 = 수정값 WHERE 조건절

ex) member 테이블의 내용중 num 칼럼이 3 인 회원의 주소를 한국 으로 수정.

MariaDB [kic]>

ex) member 테이블 내용중 num 칼럼이 2인 회원의 이름과 주소를 'jobs' , 'us' 으로 바꾸기.

MariaDB [kic]>

5. DML (DATA MULITIPULATION LANGUAGE)

2) DELETE 문

- DATA 를 삭제할때 사용하는 문장

[형식]

DELETE FROM 테이블명 WHERE 조건절

ex) member 테이블에서 주소가 '서울' 인 회원의 정보를 삭제

MariaDB [kic]>

6. TCL (TRANSACTION CONTROL LANGUAGE)

- DML 문이 실행되어 DBMS 에 저장되거나 되돌리기 위해 실행되는 SQL 문

1) 트랜잭션

- 분리되어서는 안되는 논리적 작업단위

ex) 상품을 구매한다고 가정을 한다면

<-- 트랜잭션의 시작 -->

- 결제 테이블 추가 , 주문테이블 추가

- 배송 테이블 추가 , 재고 테이블 수정

<-- 트랜잭션의 끝 -->

6. TCL (TRANSACTION CONTROL LANGUAGE)

(1) 트랜잭션의 시작

- START TRANSACTION 명령어로 시작

(2) 트랜잭션의 끝

- COMMIT 이나 ROLLBACK 이 실행되는 순간

(3) TCL 의 종류

- COMMIT : SQL 문의 결과를 영구적으로 DB 에 반영

- ROLLBACK : SQL 문의 실행결과를 취소 할때

- SAVEPOINT : 트랜잭션의 한지점에 표시하는 임시 저장점

6. TCL (TRANSACTION CONTROL LANGUAGE)

예)

MariaDB [kic]>START TRANSACTION;

MariaDB [kic]> INSERT INTO member VALUES(4, 'AAA','BBB');

MariaDB [kic]> SAVEPOINT myPoint;

MariaDB [kic]> INSERT INTO member VALUES(5, 'bbb','BBB');

MariaDB [kic]> INSERT INTO member VALUES(6, 'ccc','BBB');

MariaDB [kic]> ROLLBACK TO myPoint;

MariaDB [kic]> COMMIT ;

6. TCL (TRANSACTION CONTROL LANGUAGE)

트랜잭션은 아래 문장에서는 사용할 수 없음.

* DROP DATABASE

* DROP TABLE

* DROP

* ALTER TABLE

mariadb는 트랜잭션 기본값이 autocommit으로 되어 있기 insert, update, delete문을 수행할 경우 자동으로 commit 됨

SET AUTOCOMMIT=0; 으로 설정하면 autocommit을 지원하지 않음

SELECT @@AUTOCOMMIT; 으로 확인 할 수 있음.

7. DDL (DATA DEFINITION LANGUAGE)

-데이터 베이스 내의 객체(테이블, 뷰, ...) 등을
생성하고 변경하고 삭제하기 위해서 사용되는
SQL 문

DDL 의 종류

(1) CREATE : 객체를 생성할때

ex) 일반적인 테이블 생성

MariaDB [kic]>CREATE TABLE test(num NUMBER, name VARCHAR2(20)); #테이블 생성

-참고-

MariaDB [kic]>DESC test ; #테이블 구조 보기

7. DDL (DATA DEFINITION LANGUAGE)

(2)ALTER : 객체를 변경할때

ex)

MariaDB [kic]>alter table member rename member1; #테이블 이름 변경

(3)DROP : 객체를 삭제 할때

MariaDB [kic]>DROP TABLE dept2;

MariaDB [kic]>DROP TABLE dept3;

MariaDB [kic]>DROP TABLE dept4;

MariaDB [kic]>DROP TABLE simple;

8. 제약조건 (CONSTRAINT)

- 테이블의 해당 칼럼에 원하지 않는 데이터를 입력/수정/삭제 되는 것을 방지 하기 위해 테이블 생성 또는 변경시 설정하는 조건이다.(저장된 데이터의 신뢰성을 높이기 위해)

1) 종류

- (1) **NOT NULL** : NULL로 입력이 되어서는 안되는 칼럼에 부여하는 조건으로 칼럼 레벨에서만 부여할 수 있는 제약조건이다.
- (2) **UNIQUE KEY** (유일키) : 저장된 값이 중복되지 않고 오직 유일하게 유지되어야 할 때 사용하는 제약조건이다. (NULL은 허용된다)
- (3) **PRIMARY KEY** (대표키) : NOT NULL 조건과 UNIQUE KEY를 합친 조건이다.
- (4) **CHECK** : 조건에 맞는 데이터만 입력되도록 조건을 부여하는 제약 조건
- (5) **FOREIGN KEY** (외래키) : 부모 테이블의 PRIMARY KEY를 참조하는 칼럼에 붙이는 제약조건이다(예 emp 테이블의 deptno 칼럼)

8. 제약조건 (CONSTRAINT)

2) 제약조건 (예제)

MariaDB [kic]>CREATE TABLE dept2

(deptno int PRIMARY KEY,

dname VARCHAR(15) UNIQUE DEFAULT '영업부',

loc CHAR(1) NOT NULL CHECK(loc IN('1' , '2')) ;

8. 제약조건 (CONSTRAINT)

*** 외래키를 만들기 위해서는 부모 테이블을 먼저 만들어야한다.

MariaDB [kic]> CREATE TABLE dept2(

deptno TINYINT PRIMARY KEY,

dname VARCHAR(15) NOT NULL);

MariaDB [kic]> CREATE TABLE emp2(

empno SMALLINT PRIMARY KEY,

ename VARCHAR(15) NOT NULL,

deptno TINYINT,

FOREIGN KEY(deptno) REFERENCES dept2(deptno));

8. 제약조건 (CONSTRAINT)

4) 제약조건 알아보기

-제약조건 이름 검색하기

MariaDB [kic]> select * from information_schema.table_constraints;

-제약조건은 수정은 불가능하고 삭제만 가능하다

MariaDB [kic]> ALTER TABLE emp2 DROP FOREIGN KEY emp2_ibfk_1;

MariaDB [kic]> DROP INDEX dname ON dept3;

-제약조건 추가하기

MariaDB [kic]> ALTER TABLE emp2 ADD(CONSTRAINT emp2_fk FOREIGN KEY(deptno)
REFERENCES dept2(deptno));

MariaDB [kic]> ALTER TABLE dept3 ADD(CONSTRAINT dept_pk PRIMARY KEY(deptno));

MariaDB [kic]> ALTER TABLE dept3 MODIFY dname VARCHAR(15) NOT NULL;

9. 기타(컬럼, 복사,자료형)

- 테이블 만들기

MariaDB [kic]> CREATE TABLE test(num int);

- 칼럼 추가

MariaDB [kic]> ALTER TABLE test ADD(name VARCHAR(10));

- 칼럼 수정

MariaDB [kic]> ALTER TABLE test MODIFY name VARCHAR(20);

- 칼럼의 이름 바꾸기

MariaDB [kic]> ALTER TABLE test CHANGE name myname VARCHAR(20);

- 칼럼 삭제

MariaDB [kic]> ALTER TABLE test DROP myname;

9. 기타(컬럼, 복사,자료형)

테이블 복사하기

```
MariaDB [kic]> CREATE TABLE dept4 (deptno tinyint,  
dname VARCHAR(14), loc VARCHAR(13));
```

```
MariaDB [kic]> INSERT INTO dept4 SELECT * FROM dept;
```

ex) 테이블 복사2(CTAS 기법) =>제약조건은 복사가 안된다.

```
MariaDB [kic]> CREATE TABLE dept5 AS SELECT * FROM dept ;
```

ex) 테이블의 구조만 복사하려면 -조건이 항상 거짓이 되는 편법사용

```
MariaDB [kic]> CREATE TABLE dept6 AS SELECT * FROM dept WHERE 1=2;
```

9. 기타(컬럼, 복사, 자료형)

[숫자형 데이터 타입]

데이터 타입	의미	크기	설명
TINYINT	매우 작은 정수	1 byte	-128 ~ 127 (부호없이 0 ~ 255)
SMALLINT	작은 정수	2 byte	-32768 ~ 32767
MEDIUMINT	중간 크기의 정수	3 byte	-(-8388608) ~ -1(8388607)
INT	표준 정수	4 byte	~ -1
BIGINT	큰 정수	8 byte	- ~ -1
FLOAT	단정도 부동 소수	4 byte	-3.40E+45 ~ 3.40E+45 (no unsigned)
DOUBLE	배정도 부동 소수	8 byte	-1.7976E+320 ~ 1.7976E+320 (no unsigned)
DECIMAL(m,n)	고정 소수	m과 n에 따라 다르다	숫자 데이터지만 내부적으로 String형태로 저장됨. 최대 65자.
BIT(n)	비트 필드	m에 따라 다르다	

[날짜형 데이터 타입]

데이터 타입	의미	크기	설명
DATE	CCYY-MM-DD	3 byte	1000-01-01 ~ 9999-12-31
DATETIME	CCYY-MM-DD hh:m m:ss	8 byte	1000-01-01 00:00:00 ~ 9999-12-31 23: 59:59
TIMESTAMP	CCYY-MM-DD hh:m m:ss	4 byte	1970-01-01 00:00:00 ~ 2037
TIME	hh:mm:ss	3 byte	-839:59:59 ~ 839:59:59
YEAR	CCYY 또는 YY	1 byte	1901 ~ 2155

9. 기타(컬럼, 복사, 자료형)

[문자형 데이터 타입]

데이터 타입	의미	크기	설명
CHAR(n)	고정길이 비이진(문자) 문자열	n byte	255
VARCHAR(n)	가변 길이 비이진 문자열	Length + 1 byte	16383
BINARY(n)	고정길이 이진 문자열	n byte	
VARBINARY(n)	가변 길이 이진 문자열	Length + 1 byte or 2 byte	
TINYBLOB	매우작은 BLOB(Binary Large Object)	Length + 1 byte	
BLOB	작은 BLOB	Length + 2 byte	최대크기 64KB
MEDIUMBLOB,	중간 크기 BLOB	Length + 3 byte	최대크기 16MB
LONGBLOB	큰 BLOB	Length + 4 byte	최대크기 4GB
TINYTEXT	매우 작은 비이진 문자열	Length + 1 byte	255
TEXT	작은 비이진 문자열	Length + 2 byte	최대크기 64KB
MEDIUMTEXT	중간 크기 비이진 문자열	Length + 3 byte	최대크기 16MB
LONGTEXT	큰 비이진 문자열	Length + 4 byte	최대크기 4GB

version마다 다름

THANK YOU