데이터베이스

관계형 데이터베이스

(mariadb)

데이터베이스 레퍼런스 문서

https://mariadb.org/documentation/

mariadb programming connetors driver

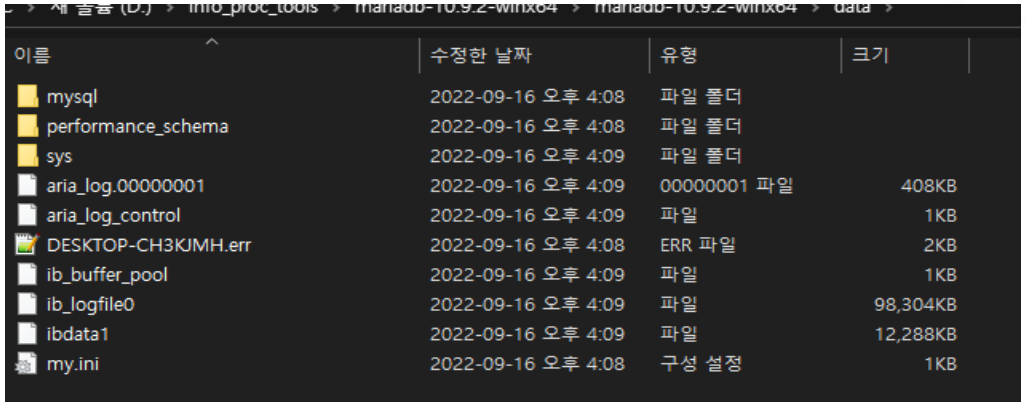
https://mariadb.org/download/?t=connector&p=connector-c&r=3.2.7&os=source&cpu=Not+Set

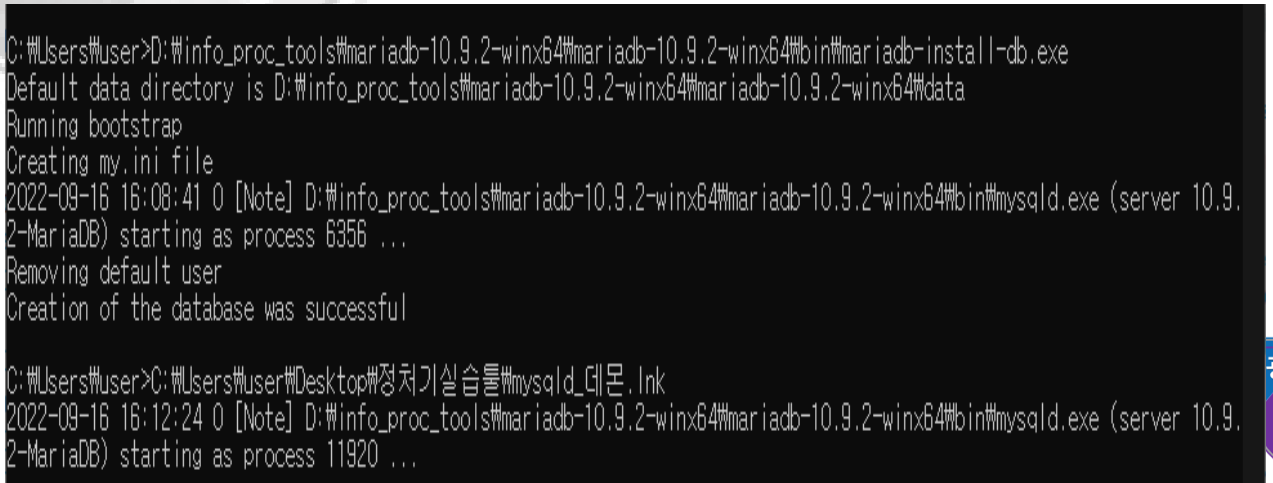
mariadb programming java connetors driver

https://mariadb.org/connector-java/all-releases/

데이터베이스 설치

1. 압축 풀기 및 설치
2. **Running MariaDB from the Build Directory 레퍼런스 참조**
   mariadb-install-db.exe
3. 빌드 파일 확인(위 명령으로 자동생성된 빌드 파일)





4. 서버데몬 실행
   ₩maridb₩bin₩mysqld.exe

5. 클라이언트 접속

mariadb₩bin₩mysql.exe –uroot –p

password enter

```
C:₩Users₩user>                              ₩mysql_클라이언트.lnk -uroot -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or ₩g.
Your MariaDB connection id is 7
Server version: 10.9.2-MariaDB mariadb.org binary distribution

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 [(none)]>
```

show datases;

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.002 sec)

MariaDB [(none)]>
```

사용자생성

```
MariaDB [(none)]> CREATE USER 'test'@'%';
Query OK, 0 rows affected (0.587 sec)
```

사용자권한부여 및 패스워드

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'test'@'%' IDENTIFIED BY '!c 패스워드 5' WITH GRANT OPTION;
Query OK, 0 rows affected (0.577 sec)

MariaDB [(none)]>
```

관리자용 DB확인

```
MariaDB [mysql]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.001 sec)
```

SET PASSWORD [FOR user] = { PASSWORD('some password') | OLD_PASSWORD('some password') | 'encrypted password' }

```
MariaDB [(none)]> SET PASSWORD = password('!        5');
Query OK, 0 rows affected (0.577 sec)

MariaDB [(none)]>
```

사용할 DB 선택 및 테이블 확인

```
MariaDB [mysql]> use mysql
Database changed
MariaDB [mysql]> show tables;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| column_stats              |
| columns_priv              |
| db                        |
| event                     |
| func                      |
```

검색대상 테이블 스키마 확인

```
MariaDB [mysql]> desc user;
+------------------------+---------------------+------+-----+----------+-------+
| Field                  | Type                | Null | Key | Default  | Extra |
+------------------------+---------------------+------+-----+----------+-------+
| Host                   | char(255)           | NO   |     |          |       |
| User                   | char(128)           | NO   |     |          |       |
| Password               | longtext            | YES  |     | NULL     |       |
| Select_priv            | varchar(1)          | YES  |     | NULL     |       |
| Insert_priv            | varchar(1)          | YES  |     | NULL     |       |
| Update_priv            | varchar(1)          | YES  |     | NULL     |       |
```

추가된 사용자 권한 및 원격지 확인

```
MariaDB [mysql]> select host,user,password,select_priv,insert_priv,update_priv
    -> ,delete_priv FROM user WHERE user='test';
+------+------+-------------------------------------------+-------------+-------------+-------------+-------------+
| Host | User | Password                                  | Select_priv | Insert_priv | Update_priv | Delete_priv |
+------+------+-------------------------------------------+-------------+-------------+-------------+-------------+
| %    | test | *6EDC27DC2DC7AA4B90824ED3E314E28E8B593464 | Y           | Y           | Y           | Y           |
+------+------+-------------------------------------------+-------------+-------------+-------------+-------------+
1 row in set (0.001 sec)

MariaDB [mysql]>
```

접속종료

```
MariaDB [mysql]> quit
Bye

C:₩Users₩user>
```

서버데몬 프로세스 종료

```
C:₩Users₩user>taskkill /f /im mysqld.exe
성공: 프로세스 "mysqld.exe"(PID 11920)이(가) 종료되었습니다.

C:₩Users₩user>
```

데몬 실행 및 사용자 접속

```
C:₩Users₩user>C:₩                                    ql_클라이언트.lnk -utest -p!      패스워드      5 -h localhost
Welcome to the MariaDB monitor.  Commands end with ; or ₩g.
Your MariaDB connection id is 5
Server version: 10.9.2-MariaDB mariadb.org binary distribution

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 [(none)]>
```

데이터베이스 생성

```
MariaDB [(none)]> create database info_test;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]>
```

생성데이터베이스 정보확인

```
MariaDB [(none)]> use information_schema
Database changed
MariaDB [information_schema]> show tables;
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
```

```
MariaDB [information_schema]> desc schemata;
+----------------------------+---------------+------+-----+---------+-------+
| Field                      | Type          | Null | Key | Default | Extra |
+----------------------------+---------------+------+-----+---------+-------+
| CATALOG_NAME               | varchar(512)  | NO   |     | NULL    |       |
| SCHEMA_NAME                | varchar(64)   | NO   |     | NULL    |       |
| DEFAULT_CHARACTER_SET_NAME | varchar(32)   | NO   |     | NULL    |       |
| DEFAULT_COLLATION_NAME     | varchar(32)   | NO   |     | NULL    |       |
| SQL_PATH                   | varchar(512)  | YES  |     | NULL    |       |
| SCHEMA_COMMENT             | varchar(1024) | NO   |     | NULL    |       |
+----------------------------+---------------+------+-----+---------+-------+
6 rows in set (0.015 sec)

MariaDB [information_schema]>
```

```
MariaDB [information_schema]> SELECT schema_name,default_character_set_name,
    -> default_collation_name FROM schemata WHERE schema_name='info_test';
+-------------+----------------------------+------------------------+
| schema_name | default_character_set_name | default_collation_name |
+-------------+----------------------------+------------------------+
| info_test   | latin1                     | latin1_swedish_ci      |
+-------------+----------------------------+------------------------+
1 row in set (0.001 sec)

MariaDB [information_schema]>
```

utf8 한글설정 https://mariadb.com/kb/en/character-sets/

```
MariaDB [information_schema]> show variables like 'c%';
+----------------------------------+-----------------------------------------------------------------------------+
| Variable_name                    | Value                                                                       |
+----------------------------------+-----------------------------------------------------------------------------+
| character_set_client             | utf8mb4                                                                     |
| character_set_connection         | utf8mb4                                                                     |
| character_set_database           | utf8mb3                                                                     |
| character_set_filesystem         | binary                                                                      |
| character_set_results            | utf8mb4                                                                     |
| character_set_server             | latin1                                                                      |
| character_set_system             | utf8mb3                                                                     |
| character_sets_dir               | D:₩info_proc_tools₩mariadb-10.9.2-winx64₩mariadb-10.9.2-winx64₩share₩charsets₩ |
| check_constraint_checks          | ON                                                                          |
| collation_connection             | utf8mb4_general_ci                                                          |
| collation_database               | utf8mb3_general_ci                                                          |
| collation_server                 | latin1_swedish_ci                                                           |
| column_compression_threshold     | 100                                                                         |
| column_compression_zlib_level    | 6                                                                           |
| column_compression_zlib_strategy | DEFAULT_STRATEGY                                                            |
| column_compression_zlib_wrap     | OFF                                                                         |
| completion_type                  | NO_CHAIN                                                                    |
| concurrent_insert                | AUTO                                                                        |
| connect_timeout                  | 10                                                                          |
| core_file                        | ON                                                                          |
+----------------------------------+-----------------------------------------------------------------------------+
20 rows in set (0.001 sec)

MariaDB [information_schema]>
```

my.ini || my.cfg 한글설정 설정추가

```
[mysqld]
character-set-client-handshake = FALSE
character-set-server = utf8mb4
collation-server = utf8mb4_unicode_ci
[client]
default-character-set = utf8mb4
[mysql]
default-character-set=utf8mb4
```

기존 DB / 테이블 변경시

```
ALTER DATABASE dbname CHARACTER SET = utf8mb4
                                    COLLATE = utf8mb4_unicode_ci;

ALTER TABLE tbname CONVERT TO CHARACTER SET utf8mb4
                                    COLLATE utf8mb4_unicode_ci;
```

데몬 재가동후 인코딩 확인

```
MariaDB [(none)]> show variables like 'c%';
+----------------------------------+-----------------------------------------------------------------------------+
| Variable_name                    | Value                                                                       |
+----------------------------------+-----------------------------------------------------------------------------+
| character_set_client             | utf8mb4                                                                     |
| character_set_connection         | utf8mb4                                                                     |
| character_set_database           | utf8mb4                                                                     |
| character_set_filesystem         | binary                                                                      |
| character_set_results            | utf8mb4                                                                     |
| character_set_server             | utf8mb4                                                                     |
| character_set_system             | utf8mb3                                                                     |
| character_sets_dir               | D:₩info_proc_tools₩mariadb-10.9.2-winx64₩mariadb-10.9.2-winx64₩share₩charsets₩ |
| check_constraint_checks          | ON                                                                          |
| collation_connection             | utf8mb4_unicode_ci                                                          |
| collation_database               | utf8mb4_unicode_ci                                                          |
| collation_server                 | utf8mb4_unicode_ci                                                          |
| column_compression_threshold     | 100                                                                         |
| column_compression_zlib_level    | 6                                                                           |
| column_compression_zlib_strategy | DEFAULT_STRATEGY                                                            |
| column_compression_zlib_wrap     | OFF                                                                         |
| completion_type                  | NO_CHAIN                                                                    |
| concurrent_insert                | AUTO                                                                        |
| connect_timeout                  | 10                                                                          |
| core_file                        | ON                                                                          |
+----------------------------------+-----------------------------------------------------------------------------+
20 rows in set (0.001 sec)

MariaDB [(none)]>
```

생성된 기존 DB 인코딩 확인 및 변경

```
MariaDB [(none)]> use info_test;
Database changed
MariaDB [info_test]> show variables like 'c%';
+----------------------------------+----------------------------------------------------------------------------+
| Variable_name                    | Value                                                                      |
+----------------------------------+----------------------------------------------------------------------------+
| character_set_client             | utf8mb4                                                                    |
| character_set_connection         | utf8mb4                                                                    |
| character_set_database           | latin1                                                                     |
| character_set_filesystem         | binary                                                                     |
| character_set_results            | utf8mb4                                                                    |
| character_set_server             | utf8mb4                                                                    |
| character_set_system             | utf8mb3                                                                    |
| character_sets_dir               | D:₩info_proc_tools₩mariadb-10.9.2-winx64₩mariadb-10.9.2-winx64₩share₩charsets₩ |
| check_constraint_checks          | ON                                                                         |
| collation_connection             | utf8mb4_unicode_ci                                                         |
| collation_database               | latin1_swedish_ci                                                          |
| collation_server                 | utf8mb4_unicode_ci                                                         |
| column_compression_threshold     | 100                                                                        |
| column_compression_zlib_level    | 6                                                                          |
| column_compression_zlib_strategy | DEFAULT_STRATEGY                                                           |
| column_compression_zlib_wrap     | OFF                                                                        |
| completion_type                  | NO_CHAIN                                                                   |
| concurrent_insert                | AUTO                                                                       |
| connect_timeout                  | 10                                                                         |
| core_file                        | ON                                                                         |
+----------------------------------+----------------------------------------------------------------------------+
20 rows in set (0.001 sec)

MariaDB [info_test]>
```

기존 테이블 인코딩 변경후 확인

```
MariaDB [(none)]> ALTER DATABASE info_test CHARACTER SET = utf8mb4 COLLATE = utf8mb4_unicode_ci;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> use info_test
Database changed
MariaDB [info_test]> show variables like 'c%';
+----------------------------------+-----------------------------------------------------------------------+
| Variable_name                    | Value                                                                 |
+----------------------------------+-----------------------------------------------------------------------+
| character_set_client             | utf8mb4                                                               |
| character_set_connection         | utf8mb4                                                               |
| character_set_database           | utf8mb4                                                               |
| character_set_filesystem         | binary                                                                |
| character_set_results            | utf8mb4                                                               |
| character_set_server             | utf8mb4                                                               |
| character_set_system             | utf8mb3                                                               |
| character_sets_dir               | D:₩info_proc_tools₩mariadb-10.9.2-winx64₩mariadb-10.9.2-winx64₩share₩charsets₩ |
| check_constraint_checks          | ON                                                                    |
| collation_connection             | utf8mb4_unicode_ci                                                    |
| collation_database               | utf8mb4_unicode_ci                                                    |
| collation_server                 | utf8mb4_unicode_ci                                                    |
| column_compression_threshold     | 100                                                                   |
| column_compression_zlib_level    | 6                                                                     |
| column_compression_zlib_strategy | DEFAULT_STRATEGY                                                      |
| column_compression_zlib_wrap     | OFF                                                                   |
| completion_type                  | NO_CHAIN                                                              |
| concurrent_insert                | AUTO                                                                  |
| connect_timeout                  | 10                                                                    |
| core_file                        | ON                                                                    |
+----------------------------------+-----------------------------------------------------------------------+
20 rows in set (0.001 sec)

MariaDB [info_test]>
```

6. STUDENT 테이블에서 컴퓨터과 학생 50명, 인터넷과 학생 100명, 사무자동화과 학생 50명의 정보가 저장되어 있을 때, 다음 SQL문의 실행 결과에 따른 튜플의 수는? (단, DEPT 칼럼은 학과명이다.)

1) SELECT DEPT FROM STUDENT;
2) SELECT DISTINCT DEPT FROM STUDENT;
3) SELECT COUNT(DISTINCT DEPT) FROM STUDENT
WHERE DEPT = '컴퓨터과';

닫기1. 200
2. 3
3. 1

2020년 2회

닫기

SELECT 학번, 이름 FROM 학생 WHERE 학번 IN (3,4);

6. 학생 테이블은 학번, 이름, 학년, 수강과목, 점수, 연락처를 속성으로 가진다. 아래 조건을 만족하는 SQL문을 작성하시오.

1) 학생 테이블에서 3,4학년인 학번, 이름을 조회한다.
2) IN 연산자 사용해야 한다.

[학생]

학번	이름	학년	수강과목	점수	연락처
1000	김이름	1	수학	90	010-1111-2222
2000	장이름	2	과학	95	010-2222-2222
3000	허이름	3	미술	90	010-3333-3333
4000	조이름	4	음악	95	010-4444-4444

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

CREATE INDEX IDX_NAME ON 학생(NAME)

12. 학생 테이블의 name속성에 IDX_NAME 이름으로 인덱스 생성하는 SQL문을 작성하시오.

STID	NAME	SCORE	DEPTID
1000	김이름	90	1
2000	허이름	95	2
3000	조이름	90	3
4000	장이름	95	4

CREATE INDEX *index_name*
ON *table_name* (*column1*, *column2*, ...);

2020년 3회

8. 다음 조건을 만족하면서, 과목별 점수의 평균이 90이 상인 과목이름, 최소점수, 최대점수를 구하는 SQL문을 작성하시오.
- 대소문자를 구분하지 않는다.
- WHERE 구분을 사용하지 않는다.
- GROUP BY, HAVING구문을 반드시 사용한다.
- 세미콜론(;)은 생략 가능하다.
- 별칭(AS)을 사용해야 한다.

닫기

```
SELECT 과목이름, MIN(점수) AS 최소점수, MAX(점수) AS 최대점수 FROM 성적
GROUP BY 과목이름 HAVING AVG(점수) >= 90;
```

[성적]

과목코드	과목이름	학점	점수
1000	컴퓨터과학	A+	95
2000	운영체제	B+	85
1000	컴퓨터과학	B+	85
2000	운영체제	B	80

[결과]

과목이름	최소점수	최대점수
컴퓨터과학	85	95

2020년 3회

9. 학생 테이블에서 이름이 민수인 튜플을 삭제하는 SQL문을 작성하시오

[학생]

학번	이름	점수	과목이름
1000	김정미	90	알고리즘
2000	강은미	95	데이터베이스
3000	홍길동	90	전산수학
4000	민수	95	운영체제

닫기

```
DELETE FROM 학생 WHERE 이름 = '민수';
```

2020년 3회

닫기
1.ALTER
2.ADD

20. 학생 테이블에 주소 속성을 추가하는 SQL문을 작성하시오.

(1.) TABLE 학생 (2.) 주소 VARCHAR(20);

2020년 4회

16. 다음 조건을 만족하면서 학과별로 튜플 수가 얼마인지 구하는 SQL문을 작성하시오.
- 대소문자를 구분하지 않는다.
- WHERE 구문을 사용하지 않는다.
- GROUP BY 를 사용한다.
- 세미콜론(;)은 생략 가능하다.
- 별칭(AS)을 사용해야 한다. (별칭 사용 시 별칭은 작은 따옴표를 써야 함)
- 집계 함수를 사용해야 한다.

닫기

SELECT 학과, COUNT(학과) AS 학과별튜플수 FROM 학생 GROUP BY 학과;

[학생]

학과	학생
전기	이순신
컴퓨터	안중근
컴퓨터	윤봉길
전자	이봉창
전자	강우규

[결과]

학과	학과별튜플수
전기	1
컴퓨터	2
전자	2

6. 다음 SQL 실행 결과를 숫자만 쓰시오.

EMPNO	SAL
100	1000
200	3000
300	1500

```
SELECT COUNT(*) FROM 급여
        WHERE EMPNO > 100 AND SAL >= 3000 OR EMPNO = 200;
```

14. 주어진 테이블의 Cardinality / Degree를 구하시오.

닫기
Cardinality : 5
Degree : 4

학번	이름	학년	학과
20202020	김제원	3	무역과
20202021	김한국	1	통신과
20202022	허달력	4	영어과
20202023	이소파	2	영어과
20202024	장가위	3	중국어과

2021년 2회

5. 다음은 테이블을 수정할때의 상황입니다. SQL 보기에서 괄호안에 알맞는 문장을 작성하시오.
(　1　) 테이블명 (　2　) 컬럼 = 값 WHRE 점수 >= 90;

닫기
1. UPDATE
2. SET

2021년 2회

6. 다음 SQL 보기에서 JOIN할 경우 괄호안에 알맞는 문장을 작성하시오.
SELECT .... FROM 학생정보 a JOIN 학과정보 b (1) a.학과 = b.(2)

닫기
1. ON
2. 학과

2021년 2회

10. SQL문에서 괄호안에 알맞은 답안을 작성하시오. (실제 시험에는 결과 이미지가 있습니다.)
('이름'이란 컬럼에 '이'로 시작하는 문자열을 '내림차순'하는 쿼리 결과 내용입니다.)

SELECT .... FROM ... WHERE 이름 LIKE (1) ORDER BY (2)

닫기
1. '이%'
2. DESC

13. 다음은, 테이블에서 조건값을 실행한 화면이다. 이에 대한 알맞는 결과값을 작성하시오.

T1

CODE	NAME
3258	smith
4324	allen
5432	scott

T2

NO	RULE
12	s%
32	%t%

실행값
SELECT COUNT(*) CNT FROM T1 A CROSS JOIN T2 B WHERE A.NAME LIKE B.RULE;

결과

닫기

4

닫기
1. ORDER
2. score
3. DESC

4. 다음 SQL 결과에 알맞는 쿼리을 작성하시오.

SELECT name, score FROM 성적 (1) BY (2) (3)

성적 테이블

Index	name	score
1	Kim	95
2	Gun	90
3	Son	80
4	Jung	60

2022년 2회

3. H회사의 전체 제품 단가 보다 큰 제품 출력을 하고자 한다. 괄호안에 들어갈 알맞는 용어를 작성하시오.

[제품테이블]

제조사	제품명	단가
A	과자	1,000
B	초콜릿	6,000
H	사탕	2,000
C	아이스크림	5,000
H	사탕	3,000

닫기
ALL

TIP
ALL – 서브쿼리결과를 and 조건
서브쿼리의 모든 결과에 만족해야 출력

ANY – 서브쿼리결과를 or 조건
서브쿼리의 결과중 하나만 만족하면 출력

SELECT 제조사, 제품명, 단가 FROM 제품
WHERE 단가 > ( ) (SELECT 단가 FROM 제품 WHERE 제조사='H')

서브쿼리 사용가능 키워드

```
SELECT * FROM AMT_MST_TEST T1
   WHERE T1.SALARY IN (
    SELECT S1.SALARY FROM AMT_MST_TEST S1 LEFT JOIN DEPT_MST_TEST S2
       ON (S2.DEPT_CD = S1.DEPT_CD) WHERE S2.DEPT_NM_KOR = '마케팅부서')
```

서브쿼리 결과에 들어 있는 값만 조회시 IN, 부정은 NOT IN


```
SELECT * FROM AMT_MST_TEST T1
   WHERE EXISTS (
      SELECT S1.SALARY FROM AMT_MST_TEST S1 LEFT JOIN DEPT_MST_TEST S2
         ON (S2.DEPT_CD = S1.DEPT_CD) WHERE S2.DEPT_NM_KOR = '마케팅부서')
```

EXISTS 서브쿼리 결과가 존재한다면 출력, 부정은 NOT EXISTS

닫기 4

4. 다음 SQL 결과에 알맞는 답을 작성하시오.

[TABLE]

Index	col1	Col2
1	2	Null
2	3	6
3	5	5
4	6	3
5	Null	3

Tip
count is NULL values are not counted.

SELECT count(col2) FROM TABLE WHERE col1 in(2,3) or col2 in(3,5);

데이터베이스 주요 키워드 요약

- SELECT - extracts data from a database
- UPDATE - updates data in a database
- DELETE - deletes data from a database
- INSERT INTO - inserts new data into a database
- CREATE DATABASE - creates a new database
- ALTER DATABASE - modifies a database
- CREATE TABLE - creates a new table
- ALTER TABLE - modifies a table
- DROP TABLE - deletes a table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

WHERE -> GROUP -> HAVING
group 전 where 구문에서 집계함수 사용시 에러유발
그룹핑전 집계하였기에 1개의 로우로 그룹할 수 없음

Query 우선 순위
FROM 테이블명 ---------- (1)
WHERE 테이블 조건 ---------- (2)
GROUP BY 컬럼명 ---------- (3)
HAVING 그룹 조건 ---------- (4)
SELECT 컬럼명 ---------- (5)
ORDER BY 컬럼명 ---------- (6)

FROM 테이블에서 WHERE 조건에 따른 데이터 필터링후 가상테이블 반환

GROUP BY 그룹핑후 HAVING 조건에 따라 필터링후 가상테이블 반환

SELECT 해당열을 추출하여 가상테이블 반환

ORDER BY 순으로 정렬하여 최종 가상테이블 반환

SELECT

```
SELECT column1, column2, ...
  FROM table_name;
```

```
SELECT * FROM table_name;
```

```
SELECT CustomerName, City FROM Customers;
```

SELECT DISTINCT Syntax

```
SELECT DISTINCT column1, column2, ...
  FROM table_name;
```

```
SELECT COUNT(DISTINCT Country) FROM Customers;
```

WHERE Syntax

```
SELECT column1, column2, ...
  FROM table_name
    WHERE condition;
```

```
SELECT * FROM Customers
  WHERE Country='Mexico';
```

Operators in The WHERE Clause

Operator	Description
=	Equal
<	Less than
>=	Greater than or equal
<=	Less than or equal
<>	Not equal. **Note:** In some versions of SQL this operator may be written as !=
BETWEEN	Between a certain range
LIKE	Search for a pattern
IN	To specify multiple possible values for a column

> Greater than

The SQL AND, OR and NOT Operators

```
SELECT column1, column2, ...
  FROM table_name
    WHERE condition1 AND condition2 AND condition3 ...;
```

```
SELECT * FROM Customers
WHERE Country='Germany' AND City='Berlin';
```

```
SELECT column1, column2, ...
  FROM table_name
    WHERE condition1 OR condition2 OR condition3 ...;
```

```
SELECT * FROM Customers
WHERE City='Berlin' OR City='München';
```

```
SELECT column1, column2, ...
  FROM table_name
    WHERE NOT condition;
```

```
SELECT * FROM Customers
WHERE NOT Country='Germany';
```

ORDER BY Syntax

```
SELECT column1, column2, ...
  FROM table_name
    ORDER BY column1, column2, ... ASC|DESC;
```

```
SELECT * FROM Customers
ORDER BY Country DESC;
```

INSERT INTO Syntax

```
INSERT INTO table_name (column1, column2, column3, ...)
  VALUES (value1, value2, value3, ...);
```

```
INSERT INTO table_name
  VALUES (value1, value2, value3, ...);
```

```
INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
    VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

IS NULL Syntax

```
SELECT column_names
  FROM table_name
    WHERE column_name IS NULL;
```

```
SELECT column_names
  FROM table_name
    WHERE column_name IS NOT NULL;
```

```
SELECT CustomerName, ContactName, Address
  FROM Customers
    WHERE Address IS NULL;
```

UPDATE Syntax

```
UPDATE table_name
  SET column1 = value1, column2 = value2, ...
    WHERE condition;
```

```
UPDATE Customers
  SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
    WHERE CustomerID = 1;
```

DELETE Syntax

```
DELETE FROM table_name WHERE condition;
```

```
DELETE FROM Customers WHERE
  CustomerName='Alfreds Futterkiste';
```

MIN() Syntax

```
SELECT MIN(column_name)
  FROM table_name
    WHERE condition;
```

MAX() Syntax

```
SELECT MAX(column_name)
  FROM table_name
    WHERE condition;
```


집계함수 – 집단적인 가상테이블을 만들어 리턴한다.

sum, avg, count, max, min, stdev, var

COUNT() Syntax

```
SELECT COUNT(column_name)
  FROM table_name
    WHERE condition;
```

AVG() Syntax

```
SELECT AVG(column_name)
  FROM table_name
    WHERE condition;
```

SUM() Syntax

```
SELECT SUM(column_name)
  FROM table_name
    WHERE condition;
```

LIKE Syntax

```
SELECT column1, column2, ...
  FROM table_name
    WHERE columnN LIKE pattern;
```

LIKE Operator	Description
WHERE CustomerName LIKE 'a%'	Finds any values that start with "a"
WHERE CustomerName LIKE '%a'	Finds any values that end with "a"
WHERE CustomerName LIKE '%or%'	Finds any values that have "or" in any position
WHERE CustomerName LIKE '_r%'	Finds any values that have "r" in the second position
WHERE CustomerName LIKE 'a_%'	Finds any values that start with "a" and are at least 2 characters in length
WHERE CustomerName LIKE 'a__%'	Finds any values that start with "a" and are at least 3 characters in length
WHERE ContactName LIKE 'a%o'	Finds any values that start with "a" and ends with "o"

% 모든문자
_ 한문자
[abc] abc 중 한문자

IN Syntax

```
SELECT column_name(s)
  FROM table_name
   WHERE column_name IN (value1, value2, ...);
```

```
SELECT * FROM Customers
  WHERE Country NOT IN ('Germany', 'France', 'UK');
```

```
SELECT column_name(s)
  FROM table_name
   WHERE column_name IN (SELECT STATEMENT);
```

```
SELECT * FROM Customers
  WHERE Country IN (SELECT Country FROM Suppliers);
```

BETWEEN Syntax

```
SELECT column_name(s)
  FROM table_name
   WHERE column_name BETWEEN value1 AND value2;
```

value1 과 value2 포함

```
SELECT * FROM Products
  WHERE Price NOT BETWEEN 10 AND 20;
```

Alias Column Syntax

```
SELECT column_name AS alias_name
  FROM table_name;
```

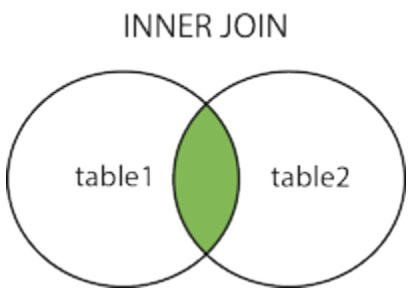
Alias Table Syntax

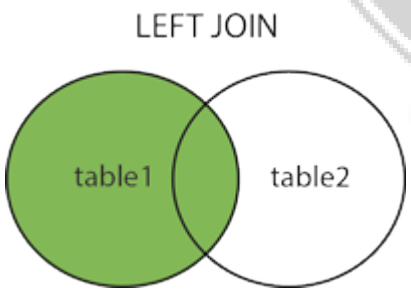
```
SELECT column_name(s)
  FROM table_name AS alias_name;
```

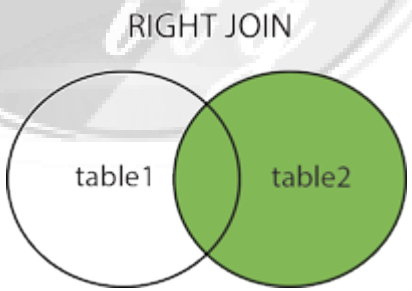
SQL JOIN

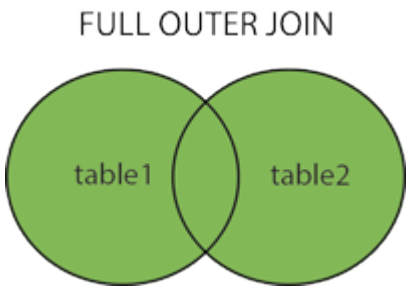
```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate FROM Orders
  INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;
```

- (INNER) JOIN: Returns records that have matching values in both tables
- LEFT (OUTER) JOIN: Returns all records from the left table, and the matched records from the right table
- RIGHT (OUTER) JOIN: Returns all records from the right table, and the matched records from the left table
- FULL (OUTER) JOIN: Returns all records when there is a match in either left or right table



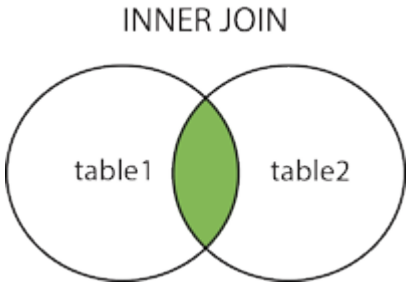






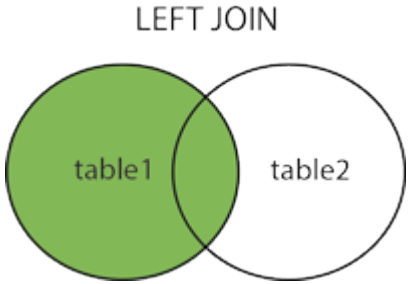
INNER JOIN Syntax

```
SELECT column_name(s) FROM table1
  INNER JOIN table2 ON table1.column_name = table2.column_name;
```



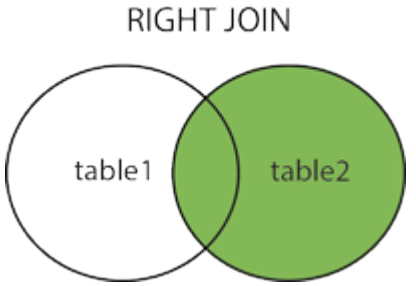
LEFT JOIN Syntax

```
SELECT column_name(s) FROM table1
  LEFT JOIN table2 ON table1.column_name = table2.column_name;
```



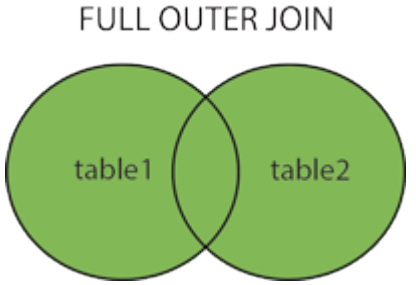
RIGHT JOIN Syntax

```
SELECT column_name(s) FROM table1
  RIGHT JOIN table2 ON table1.column_name = table2.column_name;
```



FULL OUTER JOIN Syntax

```
SELECT column_name(s) FROM table1
  FULL OUTER JOIN table2 ON table1.column_name = table2.column_name
  WHERE condition;
```

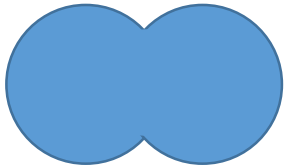


Self Join Syntax

```
SELECT column_name(s)
  FROM table1 T1, table1 T2
    WHERE condition;
```

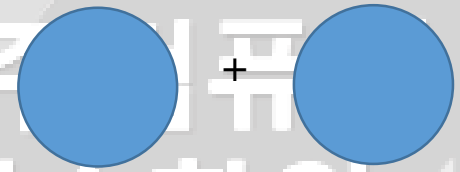
UNION Syntax

```
SELECT column_name(s) FROM table1
  UNION
    SELECT column_name(s) FROM table2;
```



UNION ALL Syntax

```
SELECT column_name(s) FROM table1
  UNION ALL
    SELECT column_name(s) FROM table2;
```



GROUP BY Syntax

```
SELECT column_name(s)
  FROM table_name
   WHERE condition
    GROUP BY column_name(s)
     ORDER BY column_name(s);
```

```
SELECT COUNT(CustomerID), Country
  FROM Customers
   GROUP BY Country
    ORDER BY COUNT(CustomerID) DESC;
```

HAVING Syntax

```
SELECT column_name(s) FROM table_name
  WHERE condition GROUP BY column_name(s)
    HAVING condition
     ORDER BY column_name(s);
```

```
SELECT COUNT(CustomerID), Country FROM Customers
  GROUP BY Country HAVING COUNT(CustomerID) > 5
    ORDER BY COUNT(CustomerID) DESC;
```

EXISTS Syntax

```
SELECT column_name(s) FROM table_name
  WHERE EXISTS
    (SELECT column_name FROM table_name WHERE condition);
```

존재시 출력, 존재하지 않으면 출력안함

ANY Syntax

```
SELECT column_name(s) FROM table_name
  WHERE column_name operator ANY
    (SELECT column_name  FROM table_name  WHERE condition);
```

서브쿼리결과 or

ALL Syntax With WHERE or HAVING

```
 SELECT column_name(s) FROM table_name
 WHERE column_name operator ALL
   (SELECT column_name  FROM table_name  WHERE condition);
```

서브쿼리결과 and

SELECT INTO Syntax Copy all columns into a new table: new column names using the AS

```
SELECT *[col1,col2,...] INTO newtable [IN externaldb]
  FROM oldtable  WHERE condition;
```

```
SELECT * INTO CustomersBackup2017 IN 'Backup.mdb'
    FROM Customers;
```

INSERT INTO SELECT Syntax

```
INSERT INTO table2
  SELECT *[col1,col2, ...] FROM table1 WHERE condition;
```

CASE Syntax

```
CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;
```

```
SELECT OrderID, Quantity,
CASE
    WHEN Quantity > 30 THEN 'The quantity is greater than 30'
    WHEN Quantity = 30 THEN 'The quantity is 30'
    ELSE 'The quantity is under 30'
END AS QuantityText
FROM OrderDetails;
```

SQL Operators

+	Add
-	Subtract
*	Multiply
/	Divide
%	Modulo
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
=	Equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
<>	Not equal to

SQL Logical Operators

ALL	TRUE if all of the subquery values meet the condition 서브쿼리의 결과에 모두 매칭 되어야함
AND	TRUE if all the conditions separated by AND is TRUE
ANY	TRUE if any of the subquery values meet the condition 서브쿼리의 결과에 어떤 값이든 매칭되어도됨
BETWEEN	TRUE if the operand is within the range of comparisons
EXISTS	TRUE if the subquery returns one or more records
IN	TRUE if the operand is equal to one of a list of expressions
LIKE	TRUE if the operand matches a pattern
NOT	Displays a record if the condition(s) is NOT TRUE
OR	TRUE if any of the conditions separated by OR is TRUE
SOME	TRUE if any of the subquery values meet the condition

```
CREATE DATABASE databasename;

DROP DATABASE databasename;

BACKUP DATABASE databasename TO DISK = 'filepath';

BACKUP DATABASE databasename TO DISK = 'filepath'
  WITH DIFFERENTIAL;

CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....);

CREATE TABLE new_table_name AS
    SELECT column1, column2,...
    FROM existing_table_name
    WHERE ....;

DROP TABLE table_name;
```

```
ALTER TABLE table_name
  ADD column_name datatype;

ALTER TABLE table_name
  MODIFY|ALTER COLUMN column_name
datatype;
```

```
CREATE TABLE table_name (
    column1 datatype constraint,
    column2 datatype constraint,
    column3 datatype constraint,
    ....
);
```

- NOT NULL - Ensures that a column cannot have a NULL value ID int NOT NULL
- UNIQUE - Ensures that all values in a column are different ID int NOT NULL UNIQUE,
- PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table

  ID int NOT NULL PRIMARY KEY || PRIMARY KEY (ID) || CONSTRAINT PK_Person PRIMARY KEY (ID,LastName)
- FOREIGN KEY - Prevents actions that would destroy links between tables

  FOREIGN KEY (PersonID) REFERENCES Persons(PersonID) ||

  PersonID int FOREIGN KEY REFERENCES Persons(PersonID) ||

  CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
- CHECK - Ensures that the values in a column satisfies a specific condition CHECK (Age>=18) ||

  CONSTRAINT CHK_Person CHECK (Age>=18 AND City='Sandnes')
- DEFAULT - Sets a default value for a column if no value is specified

CREATE INDEX Syntax

```
CREATE INDEX index_name
  ON table_name (column1, column2, ...);
```

```
CREATE INDEX idx_lastname
  ON Persons (LastName);
```

AUTO INCREMENT

Syntax for MySQL

```
CREATE TABLE Persons (
    Personid int NOT NULL AUTO_INCREMENT,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,    PRIMARY KEY (Personid)
);
```

```
ALTER TABLE Persons AUTO_INCREMENT=100;
```

DATE - format YYYY-MM-DD

DATETIME - format: YYYY-MM-DD HH:MI:SS

TIMESTAMP - format: YYYY-MM-DD HH:MI:SS

```
SELECT * FROM Orders WHERE OrderDate='2008-11-11'
```

Syntax for Oracle

```
CREATE SEQUENCE seq_person
  MINVALUE 1
  START WITH 1
  INCREMENT BY 1;
```

CREATE VIEW Syntax

```
CREATE VIEW view_name AS
  SELECT column1, column2, ...
    FROM table_name
      WHERE condition;
```

```
CREATE VIEW [Brazil Customers] AS
  SELECT CustomerName, ContactName
    FROM Customers
      WHERE Country = 'Brazil';

SELECT * FROM [Brazil Customers];
```

SQL CREATE OR REPLACE VIEW Syntax

```
CREATE OR REPLACE VIEW view_name AS
  SELECT column1, column2, ...
    FROM table_name
      WHERE condition;

DROP VIEW view_name;
```


SQL Injection

```
SELECT * FROM Users WHERE UserId = 105 OR 1=1;
```

Partitioning Tables

LIST Partitioning Type

```
PARTITION BY LIST (partitioning_expression)
   (
      PARTITION partition_name VALUES IN (value_list),
      [ PARTITION partition_name VALUES IN (value_list), ... ] [ PARTITION partition_name DEFAULT ]
   )
```

RANGE Partitioning Type

```
PARTITION BY RANGE (partitioning_expression)
 (
    PARTITION partition_name VALUES LESS THAN (value),
  [ PARTITION partition_name VALUES LESS THAN (value), ... ]
 )
```



```
CREATE TABLE log
  ( id INT UNSIGNED NOT NULL AUTO_INCREMENT, timestamp DATETIME NOT NULL,
    user INT UNSIGNED, ip BINARY(16) NOT NULL, action VARCHAR(20) NOT NULL, PRIMARY KEY (id, timestamp) ) ENGINE = InnoDB

PARTITION BY RANGE (YEAR(timestamp))
(
  PARTITION p0 VALUES LESS THAN (2013),
  PARTITION p1 VALUES LESS THAN (2014),
  PARTITION p2 VALUES LESS THAN (2015),
  PARTITION p3 VALUES LESS THAN (2016)
);
```

HASH Partitioning Type

```
PARTITION BY HASH (partitioning_expression) [PARTITIONS(number_of_partitions)]
```

```
CREATE OR REPLACE TABLE t1 (c1
, c2 DATETIME) PARTITION BY HASH(TO_DAYS(c2)) PARTITIONS 5;
```

RANGE COLUMNS and LIST COLUMNS Partitioning Types

변수 선언
DECLARE
    변수이름1 데이터형식;
    변수이름2 데이터형식;

프로시저 & 함수

```
DELIMITER //
 CREATE PROCEDURE simpleproc (OUT param1 INT)
  BEGIN
     SELECT COUNT(*) INTO param1 FROM t;
  END;
 //
DELIMITER ;

CALL simpleproc(@a);
```

IN/OUT/INOUT
in – inparam
out outparam
inout – in and out param

프로시저
쿼리의 집합이며, 일련의 작업을 정리한 절차
반환값을 가질 수도 있고 가지지 않을 수도 있습니다.
SELECT,WHERE 문 등에서 사용 불가합니다.
매개변수를 입력,출력,입출력 형식으로 받을 수 있습니다.

```
DELIMITER $$
  CREATE FUNCTION add_func4(IN a INT, IN b INT, d INT) RETURNS INT
    BEGIN DECLARE c, res INT;
      SET res = add_func3(a, b, c) + d;
      if (c > 99) then
          return 3;
      else
          return res;
      end if;
    END;
$$ DELIMITER ;

SELECT add_func4(1,2,3);
```

함수
 간단한 계산, 수치 등을 나타낼 때 사용
매개변수를 입력 형식으로만 받을 수 있습니다.
반환값을 반드시 가져야 합니다.
SELECT,WHERE 문 등에서 사용이 가능합니다.

TRIGGER

Syntax

```
CREATE [OR REPLACE]
  [DEFINER = { user | CURRENT_USER | role | CURRENT_ROLE }]
  TRIGGER [IF NOT EXISTS] trigger_name trigger_time trigger_event
  ON tbl_name FOR EACH ROW
  [{ FOLLOWS | PRECEDES } other_trigger_name ]
   trigger_stmt;
```

```
CREATE DEFINER=`root`@`localhost` TRIGGER IF NOT EXISTS increment_animal
    AFTER INSERT ON animals FOR EACH ROW
      UPDATE animal_count SET animal_count.animals = animal_count.animals+1;
```

```
DELIMITER //
   CREATE TRIGGER the_mooses_are_loose
   AFTER INSERT ON animals FOR EACH ROW
   BEGIN
     IF NEW.name = 'Moose' THEN
         UPDATE animal_count SET animal_count.animals = animal_count.animals+100;
     ELSE
        UPDATE animal_count SET animal_count.animals = animal_count.animals+1;
    END IF;
  END;
 // DELIMITER ;
```

이벤트 AFTER (INSERT,UPDATE,DELETE)
이벤트 BEFORE (INSERT,UPDATE,DELETE)
키워드 OLD 이전값
키워드 NEW 바뀔값

Event Scheduler

```
CREATE EVENT myevent
  ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 HOUR
  DO
      UPDATE myschema.mytable SET mycol = mycol + 1;
```

```
CREATE EVENT example
  ON SCHEDULE AT CURRENT_TIMESTAMP + INTERVAL 1 DAY + INTERVAL 3 HOUR
  DO something;
```

그룹함수

ROLLUP - 중간집계

GROUP BY ROLLUP(dept,job) – dept 별 중간 집계

CUBE - 다차원집계

GROUP BY CUBE(dept,job) – dept 와 job 별 중간집계

GROUPING SETS - 개별집계

GROUP BY GROUPING SET(dept,job) – dept별 job별 집계

윈도우함수

RANK - 순위표

SELECT name, salary, RANK() OVER(ORDER BY salary DESC) A) FROM employee;

CREATE OR REPLACE USER foo2@test IDENTIFIED BY 'password';

CREATE USER foo2@test IDENTIFIED BY PASSWORD '*54958E764CE10E50764C2EECBB71D01F08549980';

```
SELECT PASSWORD('mariadb');
 +-------------------------------------------+
| PASSWORD('mariadb') |
 +-------------------------------------------+
| *54958E764CE10E50764C2EECBB71D01F08549980 |
 +-------------------------------------------+
 1 row in set (0.00 sec)
```

hash 로 일방향 암호화된 비밀번호

사용자 생성 및 권한 부여 예

```
CREATE USER 'joffrey'@'192.168.0.3';
CREATE USER 'joffrey'@'%';
GRANT SELECT ON test.t1 to 'joffrey'@'192.168.0.3';
GRANT SELECT ON test.t2 to 'joffrey'@'%';
```

현재 사용자 비밀번호 변경 예

ALTER USER CURRENT_USER() IDENTIFIED BY 'mariadb';

사용자 삭제 예

DROP USER bob;

dba user 에게 모든 권한 위임 예

```
GRANT ALL PRIVILEGES ON *.* TO 'dba'@'%' ;
```

alice user 에게 모든 권한 위임과 비밀번호 변경과 자신의 권한을 다른 사용자에게 위임할 수 있는 예

```
GRANT ALL PRIVILEGES ON *.* TO 'alice'@'localhost'
  IDENTIFIED BY PASSWORD '*2470C0C06DEE42FD1618BB99005ADCA2EC9D1E19'
      WITH GRANT OPTION
```

```
GRANT ALL PRIVILEGES ON *.* TO 'alice'@'localhost'
  IDENTIFIED BY 'mariadb'
      WITH GRANT OPTION
```

일부권한과 일부 테이블에 위임 예

```
GRANT SELECT, INSERT ON mydb.mytb TO 'alice'@'%';
```

권한 회수의 예

```
REVOKE ALL PRIVILEGES, GRANT OPTION FROM user [, user] ...
```

권한의 집합을 만들어 권한 집합의 이름으로 권한 부여 가능

CREATE ROLE
DROP ROLE

종속성

y = x + 3

x는 독립변수 y값을 결정짓는 결정자이고
y는 x 값에 값이 좌우되는 종속변수이다.
표기법 : x -> y

정규화
1NF : 원자값이여야 한다. > 완료시 모두 원자값

2NF : 부분함수종속제거로 완전종속 > 완료시 모두 완전종속

3NF : 이행함수 종속제거 > 완료시 이행함수 종속이 제거됨
(A->B, B->C, A->C 제거)

BCNF : 결정자이면서 후보키가 아닌것 제거

4NF : 다치종속제거

5NF : 조인종속 제거

데이터베이스 정규화 – 논리데이터모델링 중 수행

normalization form

개체 == 테이블 이란 개념에서 부터 접근

1NF - 속성은 원자값이여야한다.

- 한속성에 두개이상의 값이 있어서는 안된다.

이름	나이	좋아하는 음식
장영	25	떡볶이, 쌀국수
이진	25	돈까스
주혜	24	소고기
정희	24	치킨

이름	나이	좋아하는 음식
장영	25	떡볶이
장영	25	쌀국수
이진	25	돈까스
주혜	24	소고기
정희	24	치킨

2NF - 부분종속 제거 ABC → AB AC

- 완전함수종속(다른 필드와 관계없는 필드 분리)

분리된 B와 C가 관계 없는지 판단

관계가 없다면 부분종속

좋아하는 음식은 나이를 결정하지 않는다

이름	나이	좋아하는 음식
장영	25	떡볶이
장영	25	쌀국수
이진	25	돈까스
주혜	24	소고기
정희	24	치킨

나이는 이름에만 종속되고
좋아하는 음식과는 관계가 없다.

이름	좋아하는 음식
장영	떡볶이
장영	쌀국수
이진	돈까스
주혜	소고기
정희	치킨

이름	나이
장영	25
장영	25
이진	25
주혜	24
정희	24

* 분리된 나이와 좋아하는 음식은 관계가 없으므로 부분종속

3NF – 이행종속제거
- A -> B , B -> C , A -> C 참조시

도시는 zip code에 따라 결정되고 이름에 따라서도 결정된다.

기본키

이름 A	나이	도시 B	Zip code C
장영	25	제주	01234
장영	25	제주	01234
이진	25	서울	56789
주혜	24	부산	13579
정희	24	강릉	24681

이름은 도시를 결정하고
zip code 또한 도시를 결정한다.

ABC → AB BC

이름	나이	Zip code
장영	25	01234
장영	25	01234
이진	25	56789
주혜	24	13579
정희	24	24681

Zip code	도시
01234	제주
01234	제주
56789	서울
13579	부산
24681	강릉

* 분리된 이름과 도시는 관계가 있으므로 이행종속

BCNF(Boyce-Codd Normal Form)

- 모든 결정자가 후보키이다.

기본키

이름	과목	교수님	학점
장영	말과승마	Dr.김	A
장영	데이터베이스	Dr.윤	B
이진	운영체제	Dr.임	B+
주혜	모바일프로그래밍	Dr.최	A+
정희	사물인터넷	Dr.고	C+

이름은 과목을 결정하지 않지만
교수는 과목을 결정한다.

A1A2 B C A1A2BC → A1A2B A2C

C는 기본키가 될수 있는 후보키이다.

이름	과목	학점
장영	말과승마	A
장영	데이터베이스	B
이진	운영체제	B+
주혜	모바일프로그래밍	A+
정희	사물인터넷	C+

교수님	과목
Dr.김	말과승마
Dr.윤	데이터베이스
Dr.임	운영체제
Dr.최	모바일프로그래밍
Dr.고	사물인터넷

- 이름은 과목을 결정하지 않지만 교수는 과목을 결정하는 결정자 이다.
- 교수는 좌측테이블에 없어도 유일성확보에 문제가 없으므로 후보키가 될수 없다.

4차 정규화-다치종속 ABC → AB AC

분리된 B와 C 가 관계가 있는지 판단
- 관계가 있다면 다치종속

하나의 속성이 2개 이상의 속성에 대응

사원번호	기술코드	프로젝트코드
10	MODELING	SI
10	MODELING	OO
10	DBA	PA
20	DBA	PA
20	XML	PA

보유기술

사원번호	기술코드
10	MODELING
10	DBA
20	DBA
20	XML

지원프로젝트 ↓

사원번호	프로젝트코드
10	SI
10	OO
10	PA
20	PA

5차정규화
-조인종속제거

ABC → AB AC BC

SPC

SN	PN	CN
S1	P1	C2
S1	P2	C1
S2	P1	C1
S1	P1	C1

5차 정규화

SP

SN	PN
S1	P1
S1	P2
S2	P1

PC

PN	CN
P1	C2
P2	C1
P1	C1

CS

CN	SN
C2	S1
C1	S1
C1	S2