MariaDB

■ mariadb
[user@localhost ~]$ su root

[설치]
[root@localhost user]#yum install mariadb mariadb-server

[실행/설치/동작 확인]
[root@localhost user]#systemctl restart mariadb
or
[root@localhost user]#systemctl enable mariadb
[root@localhost user]# systemctl status mariadb | grep Active
Active: active (running) since 월 2022-04-18 09:12:56 KST; 53s ago
[root@localhost user]# rpm -qa | grep -i mariadb
MariaDB-client-10.4.22-1.el7.centos.x86_64
MariaDB-compat-10.4.22-1.el7.centos.x86_64
MariaDB-common-10.4.22-1.el7.centos.x86_64
MariaDB-server-10.4.22-1.el7.centos.x86_64

[방화벽 열기]
[root@localhost user]# firewall-config
방화벽 설정 → 런타임에서 영구적으로 변경 → mariadb 또는 mysql 찾아서 체크 되어 있는지 확인 후 옵션에서 firewalld 다시 불러오기 눌러주기(GUI)

※ CUI(미니멀 모드같은 텍스트환경)
[root@localhost user]#firewall-cmd —list-all(현재 설정 표시)
[root@localhost user]#firewall-cmd —permanent —add-service=mysql
[root@localhost user]#firewall-cmd —reload

■ employees DB 가져오기
yum –y install git
[root@localhost user]# git clone https://github.com/datacharmer/test_db.git
cd test_db
mysql –u root –p
password : 1234
MariaDB [(none)]> source employees.sql;
MariaDB [(none)]> flush privileges;
→ workbench에서 접속이 안될 때 사용

그래도 접속 안 될때는 [root@localhost user]# systemctl restart mariadb
그래도 또 안되면 MariaDB [(none)]> grant all privileges on *.* to root@'%' identified by '1234';
MariaDB [(none)]> flush privileges; → Privilege 즉시 적용됨
접속이 안 되면 전체 다 끄고 컴퓨터 재시작후 다시 시도 해 볼 것

사용자 추가/권한부여
use mysql;
select host, user, password from user;

// 사용자 추가
create user 사용자ID;

// 사용자(user)를 추가하면서 패스워드까지 설정
create user userid@localhost identified by '비밀번호';

// '%' 의 의미는 외부에서의 접근을 허용
create user 'userid'@'%' identified by '비밀번호';

drop user '사용자ID'@localhost; // 사용자 삭제

권한설정

// 계정이 이미 존재 하는데
// identified by '비밀번호' 부분을 추가하면 비밀번호가 변경된다
GRANT ALL PRIVILEGES ON DB명.테이블 TO 계정아이디@host IDENTIFIED BY '비밀번호';
GRANT ALL privileges ON DB명.* TO 계정아이디@locahost IDENTIFIED BY '비밀번호';
GRANT ALL privileges ON DB명.* TO 계정아이디@'%' IDENTIFIED BY '비밀번호';

//모든 원격지에서 접속 권한 추가
grant all privileges on DB명.* to userid@'%' identified by '비밀번호' ;

// user 에게 test 데이터베이스 모든 테이블에 select, insert, update 권한 부여
grant select, insert, update on test.* to userid@localhost identified by '비밀번호';

-- 패스워드는 변경없이 권한만 부여하는 경우
-- user 에게 test 데이터베이스 모든 테이블에 select, insert, update 권한 부여
grant select, insert, update on test.* to user@localhost;

권한 확인

```
-- userid 와 host명까지 붙여서 검색해야 함
SHOW GRANTS FOR test@localhost;
SHOW GRANTS FOR test@'%';
SHOW GRANTS FOR test@'200.100.100.50';

# 권한 제거
-- 모든 권한을 삭제
revoke all on DB명.테이블명 from 사용자ID;

# 사용자 계정 삭제
drop user userid@'%';
 drop user userid@localhost;

# 데이터베이스별 용량 확인
# du -h /var/lib/mysql
SELECT table_schema "Database",
ROUND(SUM(data_length+index_length)/1024/1024,1) "MB"
FROM information_schema.TABLES GROUP BY 1;

# 전체 용량 확인
# du -sh /var/lib/mysql
SELECT SUM(data_length+index_length)/1024/1024 used_MB,
SUM(data_free)/1024/1024 free_MB
FROM information_schema.tables;

## 로그 분석
general log 란 MySQL 에서 실행되는 전체 쿼리에 대한 로그이며,
general log 를 활성화하면 MySQL 이 쿼리 요청을 받을 때
즉시 general log 에 기록합니다.

# log 입력방식 확인
select @@GLOBAL.log_output;
show variables like 'general%';

# 로그 사용여부 확인
select @@GLOBAL.general_log;
set global general_log=on;

# log 입력방식 변경
SET GLOBAL log_output = 'table';
SET GLOBAL log_output = 'file';
```

-- 로그 조회 (table 입력방식 일때)
SELECT * FROM mysql.general_log;

패스워드 복잡도 설정

패스워드 복잡도 상태 확인
show variables like 'validate_password%';

| Variable_name | Value |
+---+---+
| validate_password_check_user_name | ON | 암호에 해당 계정의 이름이 들어갔는지 체크
| validate_password_dictionary_file | |
| validate_password_length | 8 | 8글자 이상
| validate_password_mixed_case_count | 1 | 최소한 1개 이상의 대문자/소문자
| validate_password_number_count | 1 | 최소한 1개 이상의 숫자
| validate_password_policy | MEDIUM |
| validate_password_special_char_count | 1 | 최소한 1개 이상의 특수문자

validate_password_policy
: LOW - 비밀번호 길이만 테스트
MEDIUM - 암호에 적어도 1개의 숫자,
1개의 소문자,
1개의 대문자,
1개의 영숫자가 아닌 문자를 포함

플러그인 정보 확인
show global variables like '%plu%';

패스워드 복잡도 활성화 확인
select plugin_name, plugin_status
from information_schema.plugins where plugin_name like 'validate%';

windows 패스워드 복잡도 플러그인 설치
Linux는 validate_password.so
install plugin validate_password soname 'validate_password.dll';

```
-- 패스워드 복잡도 정책 변경
-- SET GLOBAL validate_password_length = <원하는 길이>;
SET GLOBAL validate_password_policy = 'LOW';

SET GLOBAL validate_password_length = 4;



# 플러그인 삭제
uninstall plugin validate_password;
```

데이터베이스 백업

```
# DB백업
mysqldump -u [사용자 계정] -p [패스워드] [원본 데이터베이스명] > [생성할 백업 DB명].sql
mysqldump -u test_user -p test_db > backup_test_db.sql

# DB 복원
mysql -u [사용자 계정] -p [패스워드] [복원할 DB] < [백업된 DB].sql
mysql -u test_user -p test_db < backup_test_db.sql

# 모든 DB 백업
mysqldump --all-databases -u [사용자 계정] -p --default-character-set=euckr < [백업된 DB].sql
mysqldump --all-databases -uroot -p --default-character-set=euckr > all.sql

# 모든 DB 복원
mysql --all-databases -u [사용자 계정] -p < [백업된 DB].sql
mysql -uroot -p < all.sql
```

데이터베이스 백업 스크립트

```
#!/bin/bash

DB_BACKUP="/home/dbbackup/"
DB_USER="username"
DB_PASSWD="passwd"
db="dbname"
table="tablename"

# Remove backups older than 3 days
find $DB_BACKUP -ctime +3 -exec rm -f {} ₩;
```

```
# 데이터베이스를 모두 백업할경우
mysqldump --user=$DB_USER --password=$DB_PASSWD -A | gzip > "$DB_BACKUP/mysqldump-$db-$(date +%Y-%m-%d).gz";

# 데이터베이스를 백업할경우

mysqldump --user=$DB_USER --password=$DB_PASSWD $db | gzip > "$DB_BACKUP/mysqldump-$db-$(date +%Y-%m-%d).gz";

# 데이터베이스의 특정 테이블을 백업할경우
mysqldump --user=$DB_USER --password=$DB_PASSWD $db $table | gzip > "$DB_BACKUP/mysqldump-$db-$table-$(date +%Y-%m-%d).gz";
```