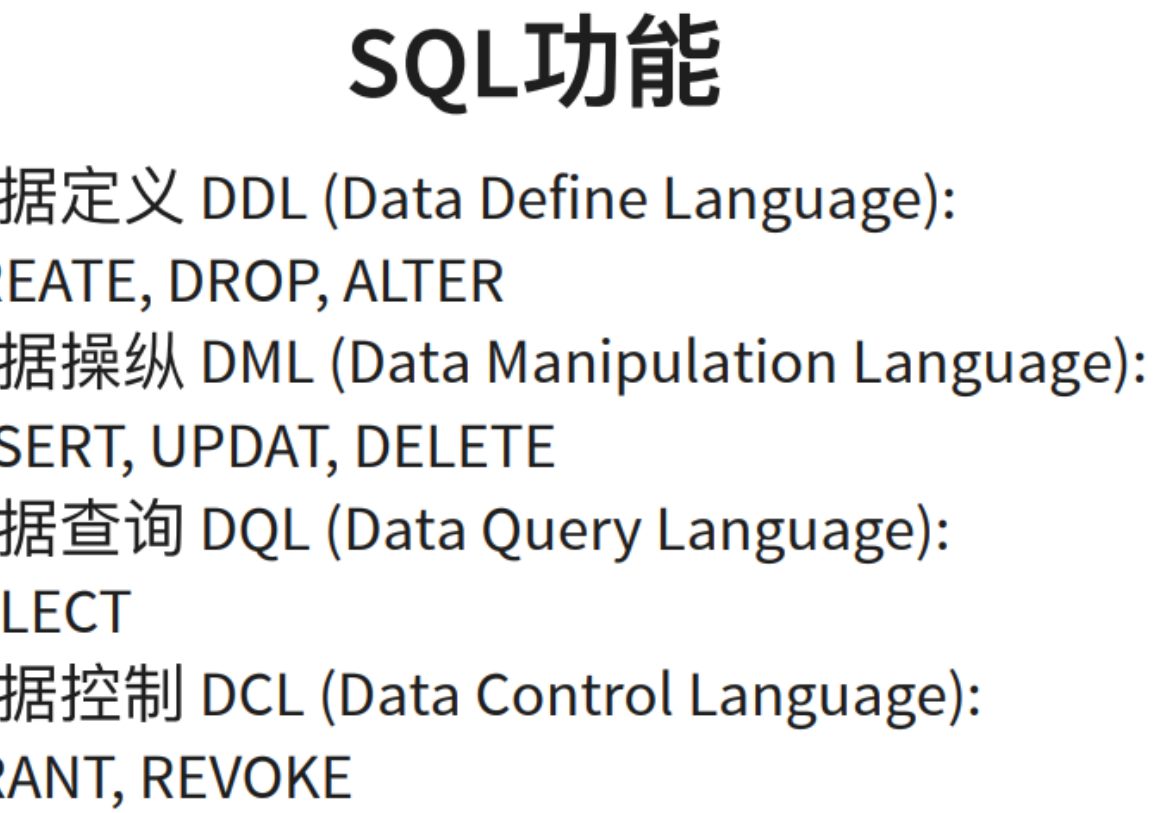
# 基本概念

- 基本功能




- 格式约定

# SQL格式约定

用扩展的巴克斯范式(Backus Naur Form, BNF)定义SQL语句

- 大写字母用于表示保留字
- 小写字母用于表示用户自定义字
- 竖线|用于表示从选项中进行选择
- 大括号{}用于表示所需元素
- 中括号[]用于表示可选择元素
- 省略号...用于表示选择项可重复0到多次

# 访问控制

- 创建用户
  **先到root权限下，然后用 CREATE USER '用户名称'@'host的ip地址' IDENTIFIED BY '端口'**

## MYSQL CREATE USER

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

```
CREATE USER 'u1'@'172.16.11.99' IDENTIFIED BY '123';
CREATE USER 'u1'@'172.16.12.24' IDENTIFIED BY '123';
CREATE USER 'u2'@'%' IDENTIFIED BY '123';
```

- 删除用户

# MYSQL DELETE USER

```
USE mysql;
DELETE FROM user
WHERE User='u1' AND Host='172.16.11.99';
FLUSH PRIVILEGES;
```

- 重命名用户

  **RENAME USER old_user TO new_user** 这个user要是'用户名称'@'host的ip地址'

  # MYSQL RENAME USER

  ## 语法

  ```
  RENAME USER old_user TO new_user
   [, old_user TO new_user] ...;
  ```

  ## 举例

  ```
  RENAME USER 'u1'@'172.16.12.24' TO
   'win.user'@'172.16.12.24';
  RENAME USER 'u2'@'%' TO 'back'@'127.0.0.1';
  ```

- 设置密码

# MYSQL CHANGE USER PASSWORD

```
ALTER USER 'userName'@'host'
IDENTIFIED BY 'auth_string';
```

```
SET PASSWORD [FOR user] auth_option
 [REPLACE 'current_auth_string']
 [RETAIN CURRENT PASSWORD];
auth_option: {
 = 'auth_string'
 | TO RANDOM
}
```

- 默认为当前用户设定口令

- 忘记密码的做法

# FORGET MYSQL PASSWORD

a. 先关闭mysqld
b. 修改/etc/my.cnf文件，注释掉口令策略参数
c. 安全启动mysqld

```
mysqld_safe --skip-grant-tables &
```

d. 无口令登录

```
mysql -u root
```

e. FLUSH PRIVILEGES
f. 修改口令，重启数据库

- MYSQL ROLE不知道有什么用

# MYSQL ROLE

```
CREATE ROLE [IF NOT EXISTS] role[, role] ...

CREATE ROLE 'admin', 'developer';
CREATE ROLE 'webapp'@'localhost';
```

```
DROP ROLE [IF EXISTS] role[, role] ...

DROP ROLE 'admin', 'developer';
DROP ROLE 'webapp'@'localhost';
```

- MYSQL GRANT
  有点像chmod来赋予不同的权限 **GRANT 权限 ON TABLE table的具体位置 TO 用户**
  用户要是'用户名称'@'host的ip地址'

# MYSQL授权

## 语法

```
GRANT privType [(colList)][, privType [(colList)]] ...
ON [objectType] privLevel
TO user|role[, user|role] ...
[WITH GRANT OPTION]
[AS user [WITH ROLE DEFAULT|NONE|ALL
          |ALL EXCEPT role[, role] ...
          |role[, role] ...]
];
```

```
GRANT PROXY ON user|role
TO user|role[, user|role] ...
[WITH GRANT OPTION];
```

- WITH GRANT OPTION, 允许授权转让
- 代理用户权限保存在mysql.proxies_priv表中

# MYSQL授权举例

```
GRANT SELECT ON TABLE *.* TO 'u1'@'172.16.12.24';
GRANT INSERT ON TABLE test.* TO 'u1'@'172.16.12.24';
GRANT ALL ON TABLE test.t1 TO 'u2'@'%';
GRANT SELECT (id) ON TABLE test.t1 TO 'u2'@'%';
```

```
CREATE VIEW viewABj AS
 SELECT * FROM tableA WHERE city = 'Beijing';
GRANT SELECT ON viewABj to PUBLIC;
```

- MYSQL ROVOKE
  和GRANT语法类似，作用不同，撤销某些权力
  **GRANT ON ... TO ...,REVOKE ON ... FROM ...**

# MYSQL REVOKE

## 语法

```
REVOKE [IF EXISTS] privType [(colList)][, privType [(colList)]] ...
ON [objectType] privLevel
FROM user|role[, user|role] ...
[IGNORE UNKNOWN USER];
```

```
REVOKE [IF EXISTS] ALL [PRIVILEGES], GRANT OPTION
FROM user|role[, user|role] ...
[IGNORE UNKNOWN USER];
```

```
REVOKE [IF EXISTS] PROXY ON user|role
FROM user|role[, user|role] ...
[IGNORE UNKNOWN USER];
```

```
REVOKE [IF EXISTS] role[, role ] ...
FROM user|role[, user|role] ...
[IGNORE UNKNOWN USER];
```

# MYSQL REVOKE

举例

```
REVOKE INSERT ON *.* FROM 'jeffrey'@'localhost';
REVOKE SELECT ON world.* FROM 'role3';
REVOKE 'role1', 'role2'
 FROM 'user1'@'localhost', 'user2'@'localhost';
```