第一章 脚本编程与 Linux 命令

Shell 脚本基础知识

- **$1 入参，空值时默认赋值技巧**

```
someval=$1
[[ -z $someval ]] && someval="some default val"
```

$* 和 $@ 区别

- $* 和 $@ 都表示传递给函数或脚本的所有参数，不被双引号(" ")包含时，都以"$1" "$2" … "$n" 的形式输出所有参数。
- 但是当它们被双引号(" ")包含时，"$*" 会将所有的参数作为一个整体，以"$1 $2 … $n"的形式输出所有参数；"
- $@" 会将各个参数分开，以"$1" "$2" … "$n" 的形式输出所有参数。

```
for var in "$*";do
  echo "$var"
done
for var in "$@";do
    echo "$var"
done
区别在于，分别输出 1 2 3  和
1
2
3
```

几个 Shell 里的内置变量

```
$#  入参个数
$$   PID
$0   文件名
$?    上个命令执行结果
```

awk 教程 https://coolshell.cn/articles/9070.html

sed 教程 https://coolshell.cn/articles/9104.html

Shell script 基础问答 https://linux.cn/article-5607-1.html

Shell 四则运算方法

```
let C=A+B
let A++              #let 时 变量前不需要再加$
echo $(($A+$B))
echo $[$A+$B]
expr $A + $B
echo $A+$B | bc   #使用 bc 可以做比较复杂的运算
```

函数定义与调用

```
function myfun(){    #function 可省略
    echo $1      echo "-----"
    echo $2      echo "-----"
    echo $3
 }
 myfun "tom" "lucy" "jack"
```

判断

```
if [[ biaodashi ]]
判断文件是否存在 -f 存在真
判断目录是否存在 -d 存在真
判断字符串是否空 -z  字符串长度为 0 真
判断文件大小  -s  文件大小非 0 真
if [[ 条件 1  -a 条件 2 ]]   # -a 表示与， -o 表示或
```

字符串与序列、随机数

name=John && echo 'My name is $name' 输出'My name is $name ，因为单引号里面的$只会当作字面值

字符串截取 a=123456789 ； echo ${a:0:3} 输出 123

echo 替换字符

```
var=tom_tom_lucy_jack_tomomttom
echo ${var/tom/mary} mary_tom_lucy_jack_tomomttom
echo ${var//tom/mary} mary_mary_lucy_jack_maryomtmary
```

序列

```
for i in {1..100}          #1,2,3,4...100  这种方法好些，还可以输出间隔系列
for i in {1..100..7}       #1,8,15....99
for i in `seq 100 `        #1,2,3,4...100
for i in `seq 3 5 `        #3,4,5
```

```
echo $RANDOM
head /dev/urandom |md5sum|cksum|cut -c 1-9
```

使用 cksum 从管道灌进去一些随机文本可生成一些随机数字，需要使用 cut 切割

字符串截取

```
a=123A123B456456C
echo ${a##*3}        从最左边找到 3 删除，贪婪 B456456C
echo ${a%%6*}      从最右边找到 6 删除，贪婪 123A123B45
echo ${a%?}          删除最右边一个字符 123A123B456456
```

提取最后一位

```
str="12345678"
i=$((${#str}-1))    #字符串长度减去 1，故 i=7
echo ${b:$i:1}      #输出 8
```

日期

```
date +%F" "%R    2018-10-14 00:09
```

删除空行

```
cat 1.txt|tr -s "\n"
sed -i '/^$/d' 1.txt
grep -v "^$" 1.txt
```

Shell script 语法在线检查与建议

https://www.shellcheck.net

考察 if 、head 、tail、sed

https://leetcode.com/problems/tenth-line

考察 ls 按时间、体积大小排序

ls -lt 最新在前 ls -ltr 最旧在前
ls -lS 最大在前 ls -lSr 最小在前

考察 comm 的使用

http://man.linuxde.net/comm
前提是，文件要排序过。结果的第一列是仅仅在文件 1 出现的，第二列是仅仅在文件 2 出现的，第三列是共同出现的。
-1 表示不显示第一列，-2 表示不显示第二列，-3 表示不显示第三列

考察 grep 与正则、子模式

https://leetcode.com/problems/valid-phone-numbers
https://www.interviewbit.com/problems/valid-phone-number/
答案 grep -P "(\d{3}-|\(\d{3}\)\s{1})\d{3}-\d{4}" input

另外，grep -c 表示匹配次数
-P 表示使用正则 -E 其实是扩展模式不是正则

考察容错处理、if、大于小于、正则、排序、去重统计、awk

https://leetcode.com/problems/word-frequency
#grep 使用 -o 输出、使用-P 表示正则、使用 awk 交换列
答案 grep -o -P "\w{1,}" words.txt |sort |uniq -c|sort -nr|awk '{print $2,$1}'

考察 awk 编程

- 给定一个文本文件，内有 M 行 N 列数字，求数字求和

- 给定 id 姓名 工资文本，计算工资和

```
1 tom 2500
2 mary 3200
3 jack 4700
4 who 6900
5 lee 2600
```

```
awk 'BEGIN{sum=0}{sum+=$3}END{print sum}' 1.txt
#注意 BEGIN、END 的位置；注意 awk 里定义的变量不用$号
```

考察 awk 编程、单引号里如何传递变量

https://www.interviewbit.com/problems/lines-in-a-given-range/

```
awk   'NR>='$L' && NR<='$R' {print $0}' input   #在单引号字句里 ，使用'$A' 以传递变量 A
```

使用 awk 一行检测磁盘分区> 90%

```
df -Ph | awk ' NR != 1 && $5 >= 90 '  #为什么单引号里加{}不行呢
```

考察 tr 替换、删除的使用、去除空格

https://www.interviewbit.com/problems/remove-punctuations/

```
cat input|tr -d  -c "a-zA-Z0-9[:space:]"  #注意 tr 里什么代表字母、什么代表数字集
```

检测主机是否存活的脚本 三次 ping 都失败

有个陷阱，ubuntu 16 里使用 sh 执行时，function xxx() 会报错。因此命令行用 bash 执行脚本

```
#!/bin/bash
iplist="123.125.115.110 123.125.115.1"

function check_ip() { #函数定义 function 可省略
    for ip in $iplist;do
        fail_count=0
        for (( i=0;i<3;i++ ));do  #使用 (( 表达式 )) 来做 for 循环 ping 3 次
            if ping  -W 1 -c 1 "$ip" > /dev/null;then  #使用-W 1 避免很久超时，单位只能整数秒有效
                break
            fi
```

```
            (( fail_count=fail_count+1 )) #使用(( 表达式 )) 比 let 更加高效
        done
        if [ $fail_count -eq 3 ];then
            echo "$ip is failed"
        else
            echo "$ip is ok"
        fi
    done
}
```

批量并发检测存活主机

```
#!/bin/bash
for ip in 119.29.192.{1..255}; #批量 IP 的技巧 ; 仅仅在 bash 生效, sh 不行
do
  (
   ping -c3 -W1 $ip >/dev/null ;
   if [ $? -eq 0 ];then
       echo "$ip alive"
   fi
   )& #使用( )& 挂在后台 并发, 更快
 done
 wait #等待所有子进程结束
```

检测站点 URL 是否存活 考察 curl 的使用、文件入参、待优化多次检测

```
#!/bin/bash
# this script read urls from url.txt,then check whether the site is available
INPUT_FILE="url.txt"
while read u;do
    curl -s --connect-timeout 3 -o /dev/null $u #注意几个参数的使用 --silent 可以用-s
    if [ $? -ne 0 ];then
        echo "$u"" failed."
    else
        echo "$u"" success"
    fi

done<$INPUT_FILE
```

让进程在后台可靠运行的方法（意思是不会随着子 shell 的退出而挂）

- nohup command > /dev/null 2>&1 &
- screen # 用快捷键 CTRL-a d 来暂时断开当前会话
- (command &) #注意，这样标准输出可能会在屏幕里翻滚

第二章 接入层与网络基础

网络基础知识

- MSS 与 MTU 的区别，默认大小各是多少？
  - 标准以太网接口缺省的 MTU（最大传输单元）为 1500 字节，是最大帧 1518 减去源宿的 MAC、FCS 后最大的 IP packet 大小；
  - MTU 减去 20 字节 IP 包头减去 20 字节 TCP 包头 ，即是 MSS，1460 字节
  - 一台交换机要保证接口 MTU 的一致性。如果在一个 VLAN 上、或整个交换机都采用同样的 MTU，避免一些奇怪的问题

    https://www.zhihu.com/question/21524257
- TIME_WAIT 与 CLOSE_WAIT 的区别
- netstat 或者 ss 怎么查看是否有 UDP 丢包
- 如何查看机器上所有的 tcp 连接？

  netstat -antp
- 如何统计 time_wait 状态的连接？

  netstat -antp|grep TIME_WAIT|wc -l
- IP 包头大小？

  20 字节
- IDC 内部网络耗时

  深圳-上海 33ms

  深圳-天津 38ms

  上海-天津 30ms

  深圳同城跨机房 1.5~2 ms

  深圳同机房 0.08~0.1 ms

url.cn. 是几级域名？ www.qq.com. 是几级域名？

二级 三级

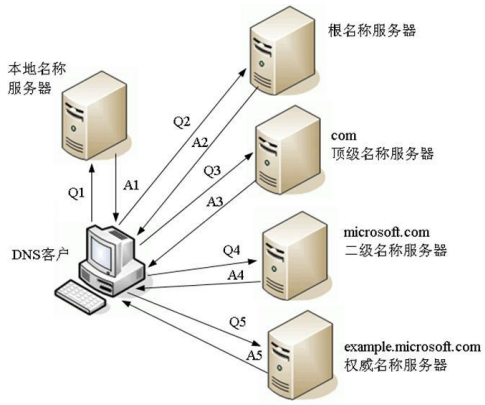
url.cn 可以 CNAME 到 sparta.qb.mig.tencent-cloud.net 吗？

不建议这么操作，实际上很多场合是不可以这么操作的。

https://serverfault.com/questions/613829/why-cant-a-cname-record-be-used-at-

the-apex-aka-root-of-a-domain

递归解析与迭代解析的区别?



- 先理解 13 组根 DNS 服务器、顶级域 DNS 服务器（各解析.com .net .gov 等）、权威 DNS 服务器、本地 DNS 服务器。
- 考虑浏览器客户端访问 www.qq.com ，会向小区宽带的本地 DNS 查询域名解析 IP ，而本地 DNS 会向根 DNS、顶级域 DNS、权威 DNS 逐个查询。
- 从客户端到本地 DNS 的查询是递归的；而其余的查询是迭代的。（参考自顶向下理解计算机网络 89 页）

x-forwarded-for 与 remote_addr 的区别?

- X-Forwarded-For: client1, proxy1, proxy2

<proxy1>, <proxy2>如果一个请求经过了多个代理服务器，那么每一个代理服务器的 IP 地址都会被依次记录在内。也就是说，最右端的 IP 地址表示最近通过的代理服务器，而最左端的 IP 地址表示最初发起请求的客户端的 IP 地址。

https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Headers/X-Forwarded-For

鉴于伪造这一字段非常容易，应该谨慎使用 X-Forwarded-For 字段。正常情况下 XFF 中最后一个 IP 地址是最后一个代理服务器的 IP 地址

- remote_addr 是通讯客户端与服务器实际进行 TCP 通信的 IP

IPv4 地址分类 （IPv4 地址 32 位）

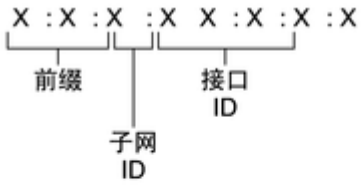
A 类 0.0.0.0/8 127.255.255.255 结束
B 类 128.0.0.0/16 191.255.255.255 结束
C 类 192.0.0.0/24
D 类 224.0.0.0~239.255.255.255

192.168.2.47/26 这个子网的开始 IP 和结束 IP 是?

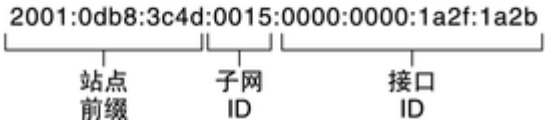
（32-26）=6 ， 2^6 = 64 ，故每 64 个 IP 是一个子网。
而 47 落在 0-63 这个子网。

故 192.168.2.0 是网络号；可用 IP 是 192.168.2.1~192.168.2.62 ；192.168.2.63 是广播 IP 地址。

IPv6 地址的格式与缩写规则，怎么判断哪些是合法的 IPv6 地址？



示例：



（IPv4 地址是 32 位）

https://www.ibm.com/support/knowledgecenter/zh/ssw_ibm_i_72/rzai2/rzai2ipv6addrformat.htm

- IPv6 地址大小为 128 位 完整写法是 8 段 ，0000:0000:0000:0000:0000:0000:0000:0000 至 ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
- **省略前导零**通过省略前导零指定 IPv6 地址。例如，IPv6 地址 1050:0000:0000:0000:0005:0600:300c:326b 可写作 1050:0:0:0:5:600:300c:326b。
- **双冒号**通过使用双冒号（::）替换一系列零来指定 IPv6 地址。例如，IPv6 地址 ff06:0:0:0:0:0:0:c3 可写作 ff06::c3。一个 IP 地址中只可使用一次双冒号。
- IPv4 地址如何映射到 IPv6 例如 0:0:0:0:0:ffff:192.1.56.10 和 ::ffff:192.1.56.10/96（短格式）
- Ipv6 保留地址 https://zh.wikipedia.org/wiki/%E4%BF%9D%E7%95%99IP%E5%9C%B0%E5%9D%80

LVS 三种工作模式原理、以及优缺点比较

NAT，IP tunneling， Direct Route

TGW 使用的是哪种模式？

- 基于 IP 隧道模式
- TGW 把外网不同运营商的请求，通过内网隧道转发给 server，server 返回数据时，再把数据通过内网隧道返回给 TGW，再由 TGW 发送给不同的运营

商。与传统的反向代理服务器最大的不同是，由于 TGW 使用了隧道技术，服务器看到的是用户的真实 IP 地址，而不是 TGW 的地址。

tcpdump 抓包的几个参数： 限制来源 IP 目的 IP 包个数 协议类型 写成 pcap 文件

与或条件 显示详细信息 显示原始端口而不是端口服务

src host
dst host
-c 包个数
-w xxx.pcap
port 端口
not
portrange 1-1024
-n 不将 IP 显示为域名
-nn 显示原始端口号和协议
http://packetlife.net/media/library/12/tcpdump.pdf

iptables 实例

iptables -t 表名 <-A/I/D/R> 规则链名 [规则号] <-i/o 网卡名> -p 协议名 <-s 源 IP/源子网> --sport 源端口 <-d 目标 IP/目标子网> --dport 目标端口 -j 动作

三表五链

1. filter 定义允许或者不允许的，只能做在 3 个链上：INPUT ，FORWARD ，OUTPUT
2. nat 定义地址转换的，也只能做在 3 个链上：PREROUTING ，OUTPUT ，POSTROUTING
3. mangle 功能:修改报文原数据，是 5 个链都可以做：PREROUTING，INPUT，FORWARD，OUTPUT，POSTROUTING

持久化

cp /etc/sysconfig/iptables /etc/sysconfig/iptables.bak # 任何改动之前先备份
iptables-save > /etc/sysconfig/iptables
https://wangchujiang.com/linux-command/c/iptables.html

iptables 如何设置只允许内网 10.142.31.1 来访问本机的 7001 7002 7003 端口

iptables -I INPUT -p tcp port 7001:7003 -s 10.142.31.1 -j ACCEPT
iptables -I INPUT -p tcp port 7001:7003 -j DROP

Iptables 设置允许 8080 80 443 以及主动出去的，其他拒绝

iptables -I INPUT -p tcp --dport 443 -i eth0 -j ACCEPT

```
iptables -I INPUT -p tcp --dport 8080 -i eth0 -j ACCEPT
iptables -I INPUT -p tcp --dport 80 -i eth0 -j ACCEPT
iptables -I INPUT -i eth0 -m state --state ESTABLISHED -j ACCEPT
iptables -A INPUT -i eth0  -j ACCEPT
```

常见加密算法

- 对称 DES、3DES、Blowfish、IDEA、RC4、RC5、RC6 和 AES
- 非对称 RSA、ECC（移动设备用）、Diffie-Hellman、El Gamal、DSA（数字签名用）

nginx 的负载种类

轮询（默认）、权重 weight、IP hash、URL hash

nginx 日志

- nginx 日志统计分析，得到访问 ip 最多的前 10 个
  (nginx 日志路径：/home/logs/nginx/default/access.log)

```
cd /home/logs/nginx/default && awk '{print $1}' access.log |sort|uniq -c|sort -nr|head
```

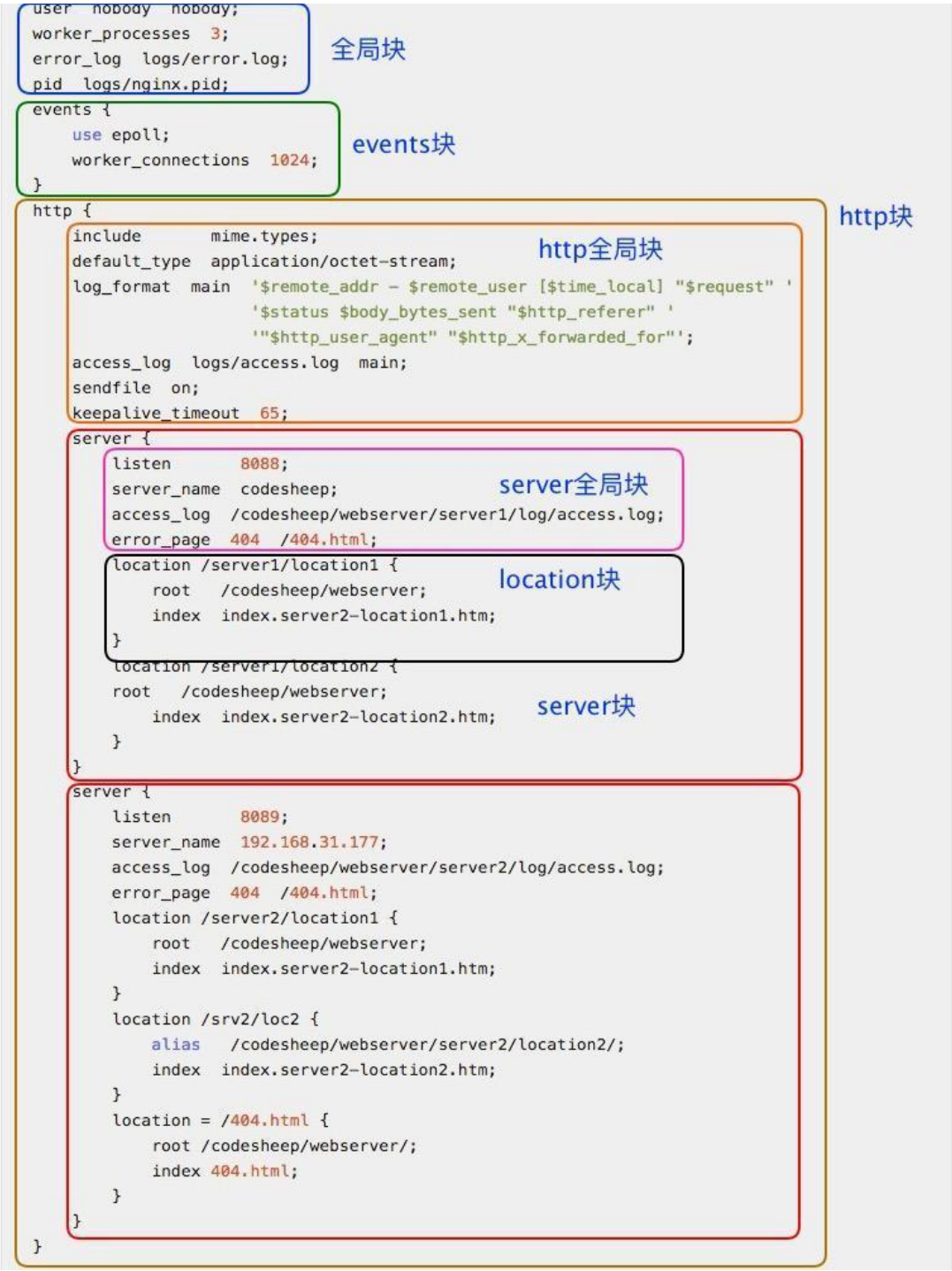
- 上面方法有个 bug ，显示的是斯巴达 proxy 内网的 IP，怎么正确显示为用户的真实 IP?

nginx location 配置

- = 用于标准 uri 前，要求请求字符串与 uri 严格匹配，一旦匹配成功则停止
- ~ 用于正则 uri 前，并且区分大小写
- ~* 用于正则 uri 前，但不区分大小写
- ^~ 用于标准 uri 前，要求 Nginx 找到标识 uri 和请求字符串匹配度最高的 location 后，立即使用此 location 处理请求，（匹配符合以后，停止往下搜索正则，采用这一条）而不再使用 location 块中的正则 uri 和请求字符串做匹配
  http://seanlook.com/2015/05/17/nginx-location-rewrite/

nginx 配置文件结构

https://segmentfault.com/a/1190000015646701




```
user  nobody  nobody;
worker_processes  3;
error_log  logs/error.log;
pid  logs/nginx.pid;
events {
    use epoll;
    worker_connections  1024;
}
http {
    include       mime.types;
    default_type  application/octet-stream;
    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';
    access_log  logs/access.log  main;
    sendfile  on;
    keepalive_timeout  65;
    server {
        listen       8088;
        server_name  codesheep;
        access_log  /codesheep/webserver/server1/log/access.log;
        error_page  404  /404.html;
        location /server1/location1 {
            root   /codesheep/webserver;
            index  index.server2-location1.htm;
        }
        location /server1/location2 {
        root   /codesheep/webserver;
            index  index.server2-location2.htm;
        }
    }
    server {
        listen       8089;
        server_name  192.168.31.177;
        access_log  /codesheep/webserver/server2/log/access.log;
        error_page  404  /404.html;
        location /server2/location1 {
            root   /codesheep/webserver;
            index  index.server2-location1.htm;
        }
        location /srv2/loc2 {
            alias   /codesheep/webserver/server2/location2/;
            index  index.server2-location2.htm;
        }
        location = /404.html {
            root /codesheep/webserver/;
            index 404.html;
        }
    }
}
```

第三章 MySQL 与 SQL 优化

MyISAM 与 InnoDB 的区别？

InnoDB 支持事务，MyISAM 不支持

InnoDB 支持行级锁，MyISAM 只能表级锁
InnoDB 的数据和索引在同一个表空间，MyISAM 分开存储
InnoDB 的磁盘空间使用比 MyISAM 更大。
MyISAM 支持全文索引、merge 表

一般选择那个引擎会更优些？

90%的场景推荐使用 InnoDB，MDB 系统只支持 InnoDB。InnoDB 的高并发更好
MyISAM 在少量需要 like 的全文索引、高速 insert 和 load data 会更好些。

单表体积要求

大小控制在 1-2GB 内，1000 万条记录以内

数据类型选型推荐：性别或状态用什么类型？ IP 或 Unix 时间戳用什么类型？datetime 与 timestamp 区别

性别或状态用 TINYINT
IP 或 Unix 时间戳用 INT UNSIGNED ，搭配内置函数 select INET_ATON("192.168.1.1")
datetime 5 字节支持时间更长，从 1000 年到 9999 年；timestamp 4 字节 最多只能到 2028 年

表名字段名的规范，是否可以用 t_MyTable 这样的表名

不要使用大小写混合，统一用小写，以免自己挖坑。MySQL 对库表名大小写敏感，字段名不敏感。
不要使用空格、特殊字符、中文；不要使用系统保留子 select update 等

索引名规范？

非唯一索引通常以 idx_开头，唯一索引以 uniq_开头

如何创建、查看索引

```
ALTER TABLE table_name ADD INDEX index_name (column_list)
ALTER TABLE table_name ADD UNIQUE (column_list)
ALTER TABLE table_name ADD PRIMARY KEY (column_list)
CREATE INDEX idx_name  ON TABLE table_name (column_list)
SHOW INDEX FROM table_name
```

某个 UGC 内容表，在显示 emoji 表情时乱码，应该使用什么字符集修复？

utf8mb4

事务的四个特性，ACID 分别是什么，怎么理解

- **Atomicity（原子性）**：一个事务（transaction）中的所有操作，要么全部完成，要么全部不完成，不会结束在中间某个环节。事务在执行过程中发生错误，会被恢复（Rollback）到事务开始前的状态，就像这个事务从来没有执行过一样。
- **Consistency（一致性）**：在事务开始之前和事务结束以后，数据库的完整性没有被破坏。这表示写入的资料必须完全符合所有的预设规则，这包含资料的精确度、串联性以及后续数据库可以自发性地完成预定的工作。
- **Isolation（隔离性）**：数据库允许多个并发事务同时对其数据进行读写和修改的能力，隔离性可以防止多个事务并发执行时由于交叉执行而导致数据的不一致。事务隔离分为不同级别，包括读未提交（Read uncommitted）、读提交（read committed）、可重复读（repeatable read）和串行化（Serializable）。
- **Durability（持久性）**：事务处理结束后，对数据的修改就是永久的，即便系统故障也不会丢失。

阿里 SQL 优化比赛题解

https://yq.aliyun.com/articles/183749

调优方法

explain
explain extends

简明优化规则

- 避免 select * ，而是指定列名
- 避免使用 null 和 is null 判断，而是给默认值 0 `age` int not null default 0
- 避免在 where 子句使用<> 或!=
- or 字句用 in 代替更好， o(n)与 o(lgn) 区别
- 若 where 子句中使用 or，可用 union all 代替 or
- exists 代替 in
  select num from a where num in(select num from b)
  用下面的语句替换：

select num from a where exists(select 1 from b where num=a.num)

- 在 where 子句 where num/2=100 会效率很低，左等号前不能是表达式、函数、运算
- 使用 between 代替 in ，between 实际是大于 and 小于的语法糖
- 避免%abc% 的 like 判断，最好能去掉前面的百分号
- 无须给性别这种重复值很多的字段加索引
- 避免隐式转换，例如字段定义是 varchar ，查询是传数值进入
- 大表避免 limit 10000,10 ；可通过程序逻辑，将上一页的最大值传入，通过 where 子句查询限制

8 种常被忽视的 SQL 错误用法（优化集锦）

https://mp.weixin.qq.com/s/1WpspGr7R-EjXfhWzlsZvQ