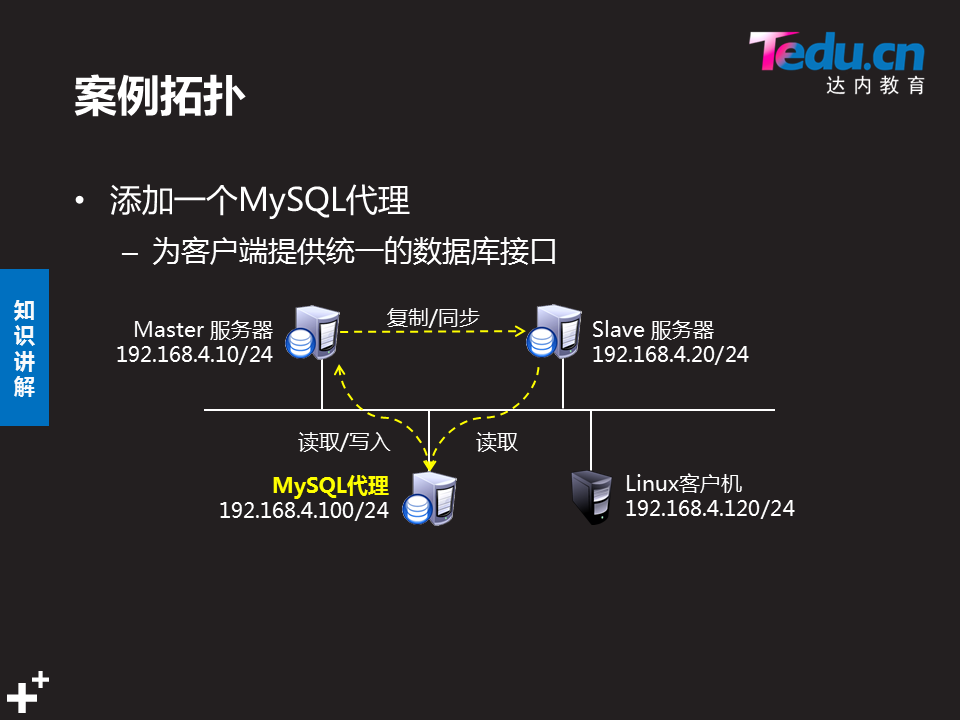
**Morning：**

1. 读写分离概述

**-What’s读写分离？**

把客户端访问数据的查询请求、写入请求，给不同的数据库服务器执行处理。

**-读写分离原理**



**-读写分离优、缺点**

优：减轻单台服务器的服务器，负载均衡。提升从服务器的硬件利用。

缺：单点故障；访问量大时，代理服务器卡顿。 解决方案：高可用集群。

1. 构建读写分离

**步骤1：部署mysql主从同步结构**

参考：阶段3 --> 2.DB2 --> DAY01 --> Page01

**步骤2：部署数据读写分离服务器**

Mysql中间件：mycat、maxscale、mysql-proxy

1）安装软件包maxscale

[root@maxsc ~]# rpm -ivh maxscale-2.1.2-1.rhel.7.x86\_64.rpm

1. 修改配置文件

[root@maxsc ~]# cp /etc/maxscale.cnf /root/

#备份源配置文件(防止出错)

[root@maxsc ~]# vim /etc/maxscale.cnf

###定义线程数###

[maxscale]

threads=auto //auto根据CPU核数创建线程数

###定义数据库服务器1信息###

[server1]

type=server

address=192.168.4.51

port=3306

protocol=MySQLBackend

###定义数据库服务器2信息###

[server2]

type=server

address=192.168.4.52

port=3306

protocol=MySQLBackend

###定义需监视的数据库服务器###

[MySQL Monitor]

type=monitor

module=mysqlmon

servers=server1, server2

user=scalemon //供读写分离服务器连接server1/2,完成**监视任务**

passwd=123456 //监视任务:判断运行状态,主从结构,谁是主库/从库

monitor\_interval=10000

###定义读写分离服务###

[Read-Write Service]

type=service

router=readwritesplit

servers=server1, server2 //读写分离在server1,server2中进行

user=gjqadmin //供读写分离服务器连接server1/2，验证**客户权限**

passwd=123456 //验证权限：以此用户名访问mysql.user表查看

max\_slave\_connections=100%

###定义管理服务(保持默认)###

[MaxAdmin Service]

type=service

router=cli //命令行

###定义读写分离程序使用的端口号###

[Read-Write Listener]

type=listener

service=Read-Write Service

protocol=MySQLClient

port=4006

###管理服务使用的端口号###

[MaxAdmin Listener]

type=listener

service=MaxAdmin Service

protocol=maxscaled

socket=default

port=4026

3）数据库服务器创建授权用户

mysql> grant replication slave, replication client on \*.\* to

> scalemon@’%’ identified by ‘123456’;

#创建监控用户，@’%’是因为分离服务器转发时，会加上一段ipv6地址

mysql> grant select on mysql.\* to

> gjqadmin@’%’ identified by ‘123456’;

#创建路由用户

**步骤3：启动读写分离服务**

[root@maxsc ~]# maxscale -f /etc/maxscale.cnf

#启动服务

[root@maxsc ~]# netstat -nptlu | grep maxscale

#查看端口4006，4026

[root@maxsc ~]# killall -9 maxscale

#停止服务，读写分离服务器不需启动mysqld服务

**步骤4：客户端测试**

1）客户机访问测试

[root@mysql50 ~]# mysql -uyaya100 -p123456 \

> -P4006 -h'192.168.4.57'

测试方法：

#从机插入一条记录，在客户端测试查询请求；

#客户端插入记录，查看主从库是否同步，测试写请求。

1. 分离服务器查看管理服务

[root@maxsc ~]# maxadmin -uadmin -pmariadb -P4026

MaxScale> list servers

**Afternoon：**

1、Mysql多实例

在一台物理主机上运行多个数据库服务

**步骤1：安装软件包(支持多实例)**

[root@mysql50 ~]# systemctl stop mysqld

[root@mysql50 ~]# systemctl disable mysqld #停止服务

[root@mysql50 ~]# tar -zvxf mysql-5.7.20-linux-glibc...tar.gz

[root@mysql50 ~]# mv mysql-5.7.20-linux-... /usr/local/mysql

[root@mysql50 ~]# vim /etc/profile

PATH=/usr/local/mysql/bin:$PATH #源码目录加入环境变量

[root@mysql50 ~]# source /etc/profile

**步骤2：修改主配置文件**

[root@mysql50 ~]# mv /etc/my.cnf /root/

[root@mysql50 ~]# vim /etc/my.cnf

[mysqld\_multi] #启用多实例

mysqld=/usr/local/mysql/bin/mysqld\_safe #指定进程文件路径

mysqladmin=/usr/local/mysql/bin/mysqladmin #指定管理命令路径

user=root #指定进程用户

[mysqld1] #实例进程名称

port=3307 #端口号

datadir=/data3307 #数据库目录(手动创建)

socket=/data3307/mysql.sock

#指定套接字文件路径(本机访问时通过sock传递请求给进程)

pid-file=/data3307/mysqld.pid #指定进程pid号文件

log-error=/data3307/mysqld.err #指定错误日志文件

[mysqld2]

port=3308

datadir=/data3308

socket=/data3308/mysql.sock

pid-file=/data3308/mysqld.pid

log-error=/data3308/mysqld.err

**步骤3：管理实例进程**

1）启动实例进程

[root@mysql50 ~]# mysqld\_multi start 1 #启动实例1

[root@mysql50 ~]# mysql -uroot -p'Xme\*e/+qp9>1' \

> -S /data3307/mysql.sock #本机使用初始密码登陆

mysql> alter user root@”localhost” identified by “123456”;

mysql> grant all on \*.\* to root@”%”

> identified by “123456”; #授权远程用户

[root@client ~]# mysql -uroot -p123456 -h’192.168.4.50’ -P 3307

#远程使用授权用户登陆

Tips：-如果数据库目录为空，启动多实例进程时会自动初始化

-初始化的过程：通过读取/etc/my.cnf生成服务文件到实例数据库目录

-多实例默认无密码策略

2）停止实例进程

[root@mysql50 ~]# mysqld\_multi --user=root --password=123456 \

> stop 1

1. Mysql性能调优(面试)



**优化1：查看硬件使用情况**

1. 使用monitor监控
2. 远程到服务器，使用命令行查看

[root@room12pc2 ~]# uptime 或 top #CPU使用情况

[root@room12pc2 ~]# free -h #内存使用情况

[root@room12pc2 ~]# iostat #IO读写速度

[root@room12pc2 ~]# ethtool eth0 #查看网卡最大带宽

Speed： 1000Mb/s

**优化2：并发及连接控制**

Tips：当/etc/my.cnf不存在，mysqld服务会按默认配置启动。

如果要调整运行参数，可以自定义/etc/my.cnf

|  |  |
| --- | --- |
| 变量 | 含义 |
| max\_used\_connections | 历史最大连接数 |
| max\_connections | 允许最大并发连接数 |
| connect\_timeout | 等待连接超时，默认为10s |
| wait\_timeout | 连接后不活动超时，默认28800s(8h) |

max\_used\_connections/max\_connections=**85%**（经验值）

mysql> show variables like “%变量名%” #查看变量值

mysql> set global 变量名=值 #设置变量值

mysql> flush status; #刷新状态信息

mysql> show processlist;

#显示当前服务器的程序列表(谁访问,执行了什么)

**优化3：缓存参数**

|  |  |
| --- | --- |
| 变量 | 含义 |
| key\_buffer\_size | 存放索引信息的缓存大小，可优化查询速度 |
| sort\_buffer\_size | 加快order by、group by排序查询速度 |
| read\_buffer\_size | 加快select \* 全表查询速度 |
| thread\_cache\_size | 允许在内存中被重用线程数量，默认9个。  线程处理请求后，不杀死继续重用，优化效率；  但会占用内存，按需调整数量。 |
| table\_open\_cache | 缓存打开表的数量(一般设置成经常被访问的表数量)  访问表：从硬盘打开 -> 调入内存 -> 写入硬盘并关闭 |

**优化4：查询缓存**

1）查询缓存全局变量

|  |  |
| --- | --- |
| 变量 | 含义 |
| query\_cache\_limit | 查询缓存限制(1M),当查询结果<=1M,可以缓存 |
| query\_cache\_size | 查询缓存大小 |
| query\_cache\_type | 0=OFF 1=查询结果自动缓存  2=程序员需在查询语句加上关键字,才可缓存 |
| query\_cache\_wlock\_invalidate | 查询缓存写锁无效，OFF即为有效  当被查询的表同时被写处理，数据库服务器不从查询缓存查找数据返回，而等待写操作完成后重新从数据库查询返回客户端。 |

Tips：生产环境下，**一般不开启查询缓存**，而使用**单独的缓存服务器**，把经常被访问的数据提前写入到缓存服务器，客户端先访问缓存服务器。

2）查询缓存状态变量

mysql> show global status like "Qcache%";

#查看当前的查询缓存状态变量

|  |  |
| --- | --- |
| 状态变量 | 含义 |
| Qcache\_hits | 当客户请求结果从查询缓存返回，hits值自加1 |
| Qcache\_inserts | 服务器每接收到查询请求，inserts值自加1 |
| Qcache\_lowmem\_prunes | 记录删除查询缓存的次数 |

Qcache\_hits/Qcache\_inserts：查询缓存命中率

3）缓存的释放

使用LUR策略（最新最少使用清除）：存满之后，再有查询请求时，会按LUR策略清除部分缓存，腾出足够的内存空间存放本次返回结果。

**优化5：Mysql日志**

1. 常用日志分类

|  |  |  |
| --- | --- | --- |
| 类型 | 用途 | 配置 |
| 错误日志 | 记录启动/运行/停止过程错误消息 | log-error[=name] |
| 查询日志 | 记录所有SQL操作 | general-log  general-log-file= |
| 慢查询日志 | 记录耗时较长或不使用索引的查询操作 | slow-query-log  slow-query-log-file=  long-query-time= |

1. 查询日志

[root@mysql54 ~]# vim /etc/my.cnf

general-log #默认命名：主机名.log

或general-log-file=文件名

1. 慢查询日志

Tips：通过慢查询日志查看比较慢的查询操作，通知优化程序员优化

[root@mysql54 ~]# vim /etc/my.cnf

slow-query-log #默认命名主机名-slow.log

或slow-query-log-file=文件名

long-query-time=20s #指定超长时间，不写默认为10s

log-queries-not-using-indexes #记录未使用索引的查询

[root@mysql54 mysql]# tailf ./mysql54-slow.log

mysql> select sleep(11);

#测试慢查询日志的记录

[root@mysql54 ~]# mysqldumpslow ./mysql54-slow.log

#统计慢查询日志记录的SQL命令

**优化6：调整网络架构**

例如：当并发访问量很大时，对Mysql服务器进行调度，实现负载均衡。