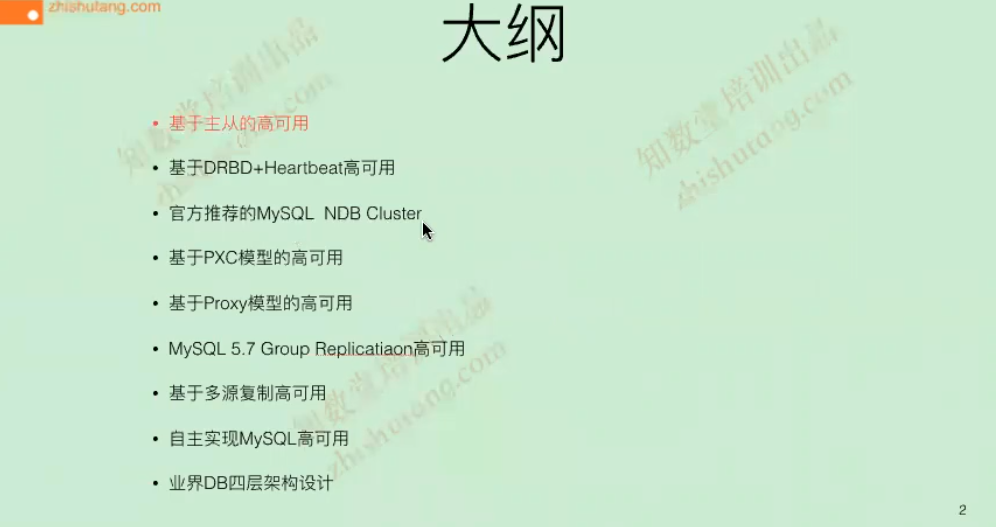
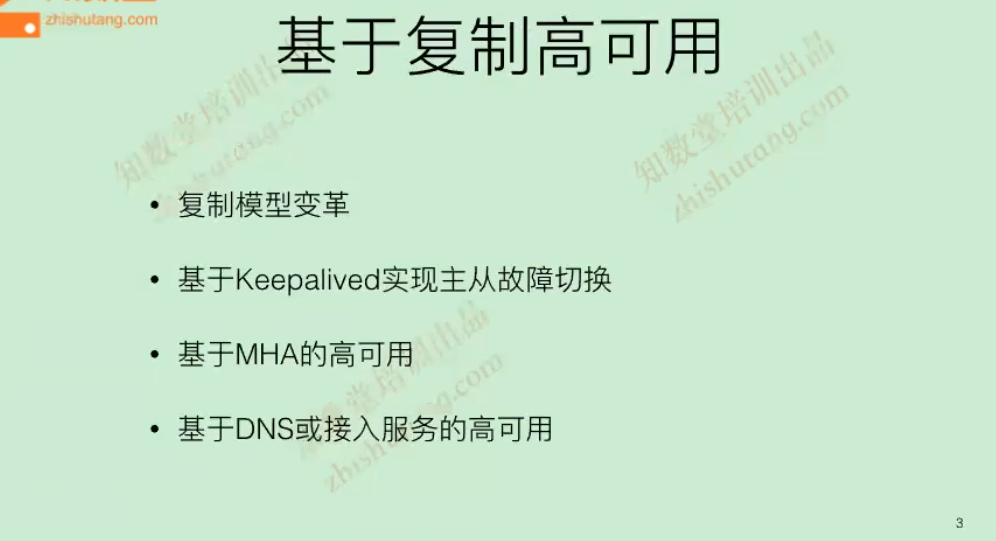


业界使用的高可用，及需要注意什么。

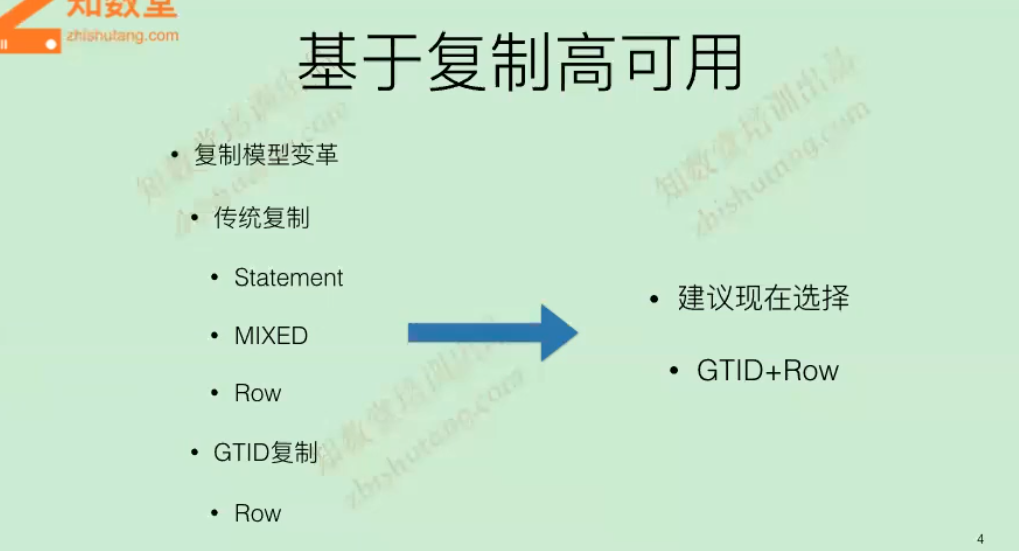


前两个就不要使用了。

目前来讲，基于复制的高可用是最成熟的高可用框架



复制的高可用有什么优势？



再生产上选择GTID+Row模式，不要自己去选择了。

在讲课中mysql5.7主要是为了binlog group commit，增加主库日志写的更快，提交主库写日志的性能。

在8.0上 增加了writeset ,这个主要是增加了sql\_thread的性能，让从库应用的更快。

如果使用gtid推荐使用row格式的，要是使用多源复制也要使用row格式的。使用gtid更加的好管理。



Keeplived模式的高可用可以实现切换功能，VIP M1挂了之后会自动切换到M2。

劣势：

1）健康检查脚本需要自己去写，keeplived可以切换，但是不会感知mysql出问题。

Keeplived切换脚本需要自己来编写。

2）脑裂，如果都认为自己是vip，就都会对外提供服务，就可能出现数据写乱的场景。解决办法就是在健康检查的脚本里面ping一下网关，如果通可以继续原来的逻辑，如果不同就什么都不要做了。

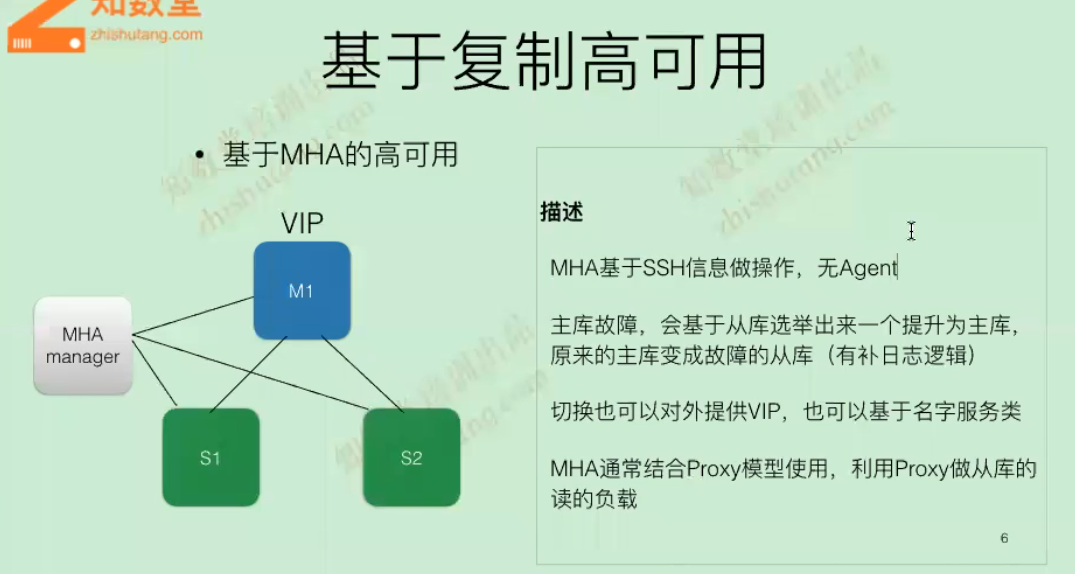
在故障节点处理完故障之后怎么加回原来的网络，如果是gtid，直接change就可以了。

阿里和腾讯云都使用的SDN网络，只开启必要的网络协议。

场景一：

M2现在是VIP，M1故障节点处理完之后加回来怎么做？

青云的xenon可以帮助我们借壳实现。



MHA的工作原理是什么？或者是MHA是怎么工作的？

MHA manager 节点安装manger和node，其他节点安装node就可以了。

Manager 监控整个主从结构是不是正常，如果整个系统正常，manager会把整个的拓扑结构吐出来。可以在拓扑图上看清整个网络节点的结构。

Manager 会一直监控M1是不是存活状态，如果M1没有存活：

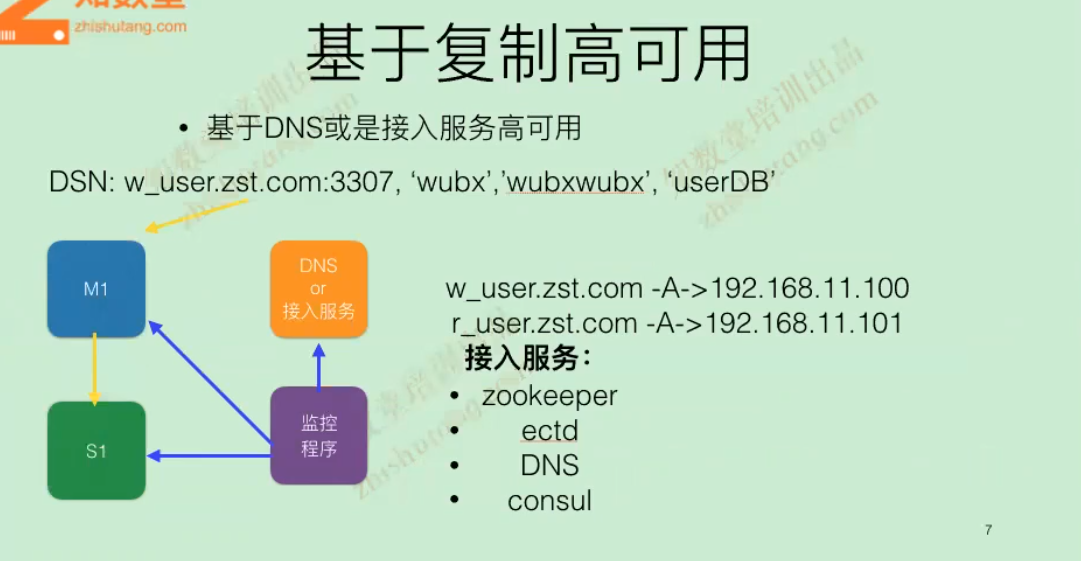
场景1：系统没有挂掉但是进程挂掉了，那么manager可以登陆M1把binglog给s1和s2补齐。

场景2：系统挂掉了，这个时候manager会对比s1和s2，查看那个binlog更加靠前，然后进行补全，利用的技术就是relay-log

另外一种方法，对比之后把binlog落后的节点change到靠前的节点，把靠前的节点的binlog给落后的节点。（具体怎么补全binlogh需要自己去实现）

MHA可以开GTID，建议使用0.57版本

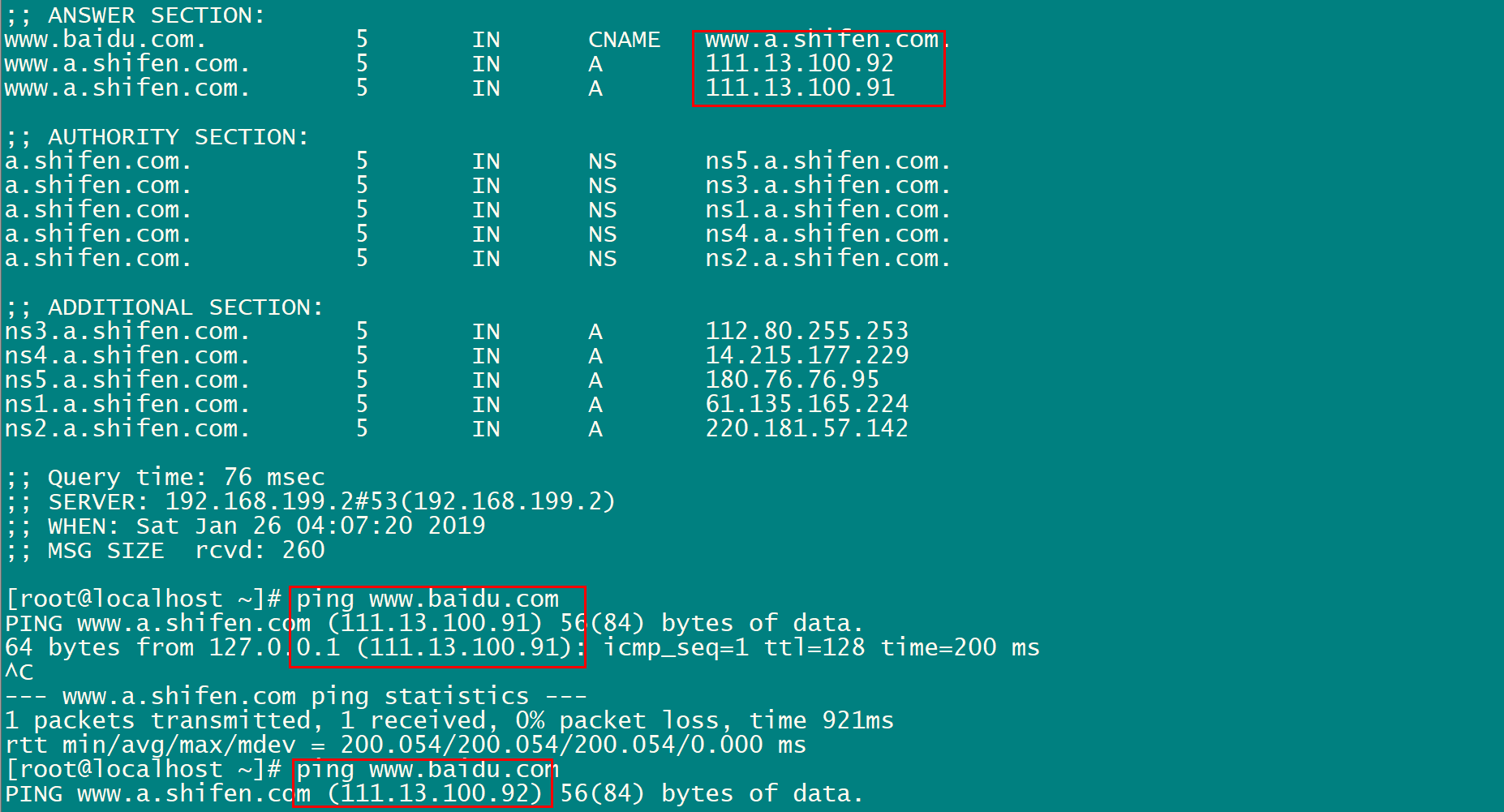
今天重点：



市面上可以抗住生产压力的dns 有bind-dlz，consul，但是consul要加一层缓存。

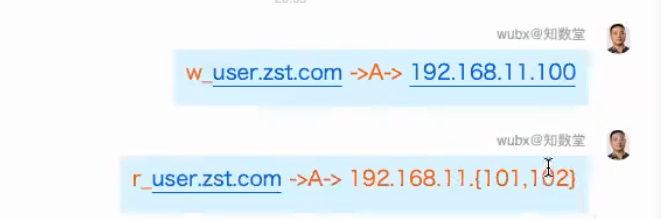
Bind-dlz 超出了太多，可以自己去尝试。

Consul 后期会讲。

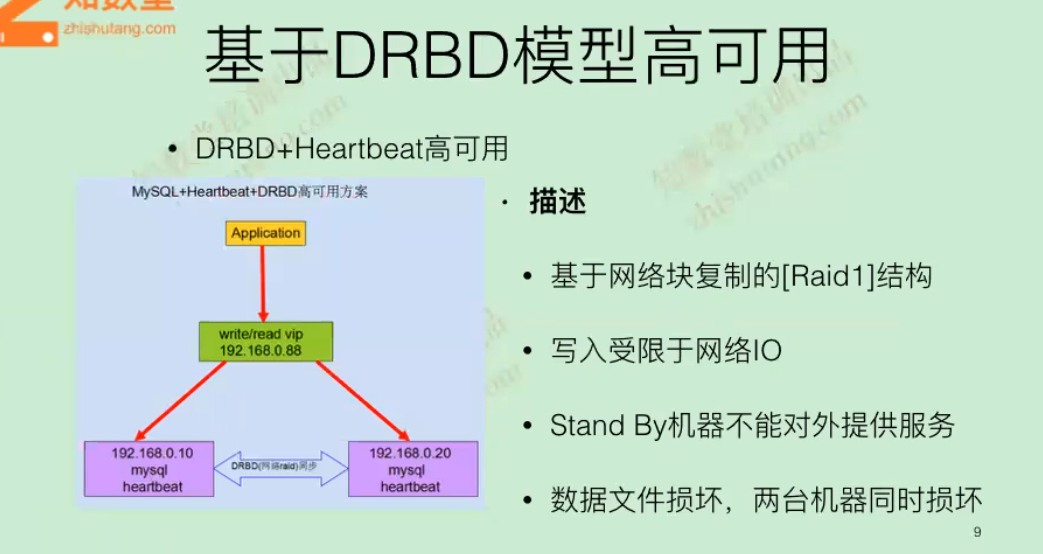


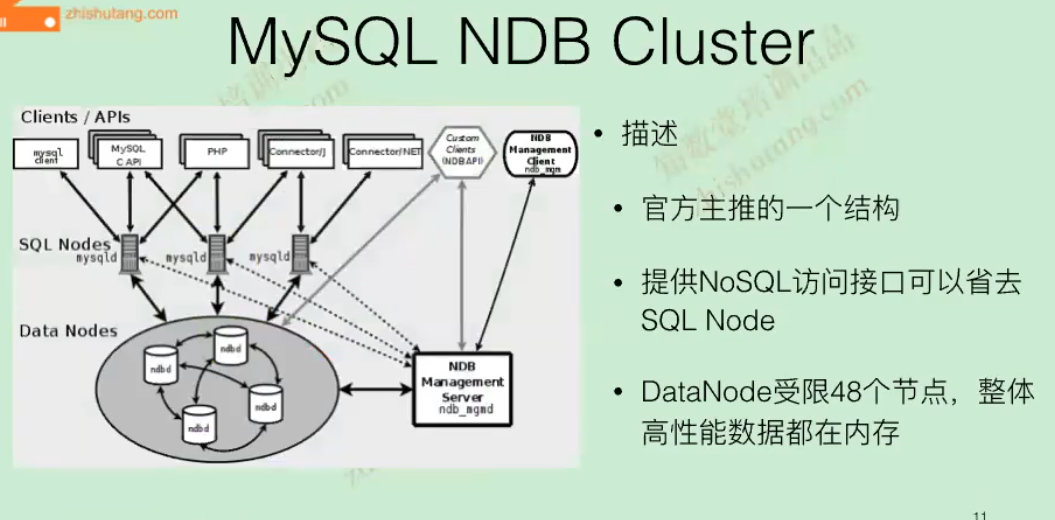
DNS：两次ping 百度，ping的ip都不一样，这就是DNS的解析，同时控制了入口，第一次ping 走的91，第二次走的92，但是无论走91还是92都在dns的控制范围内，这样就控制了入口。

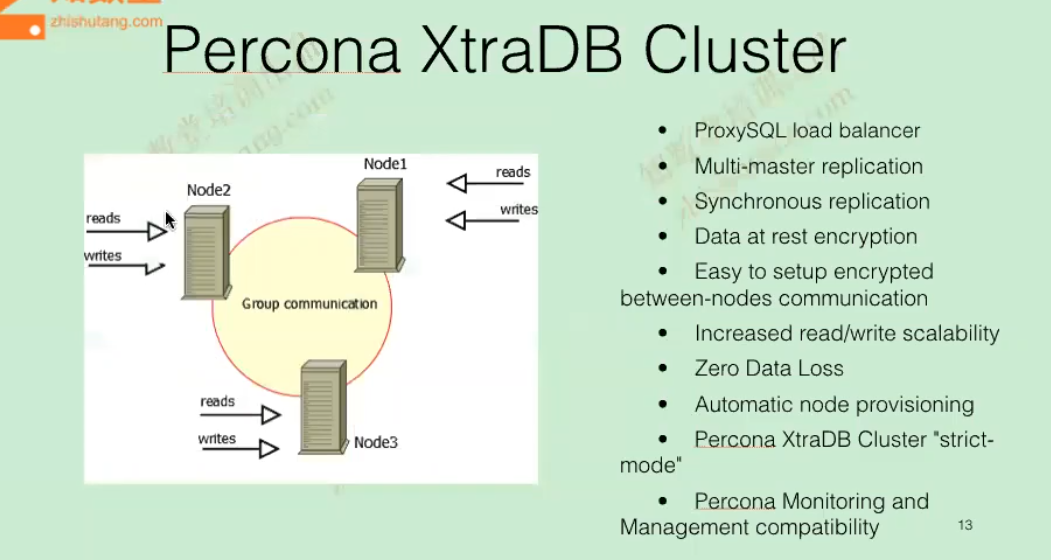
同样的道理：



当我去写数据的时候走的w\_user.zst.com，去读的时候走的r\_user.zst.com，两个网址都对应着ip，其中r库对应着两个ip，可以轮询。







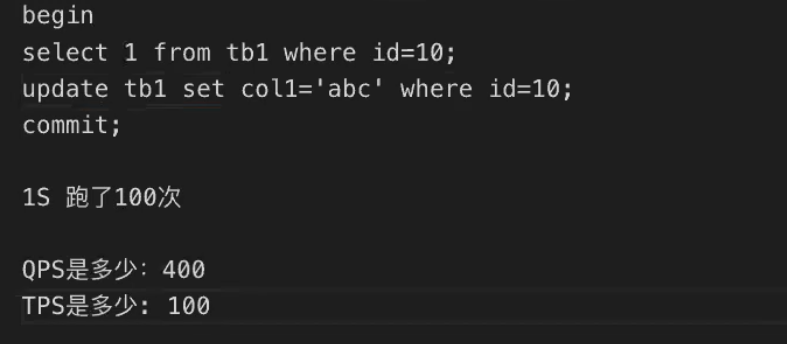
如果MGR和PXC选择的话，还是要选择MGR.

Mysql的qps，如果在理想的情况下可以到3w，但是要看自己的业务需要这么大么？如果在支付环境中qps能到5000，那么这个公司可以去美国上市了。

性能的提升必要牺牲一些其他的东西，比如安全比如硬件等等。

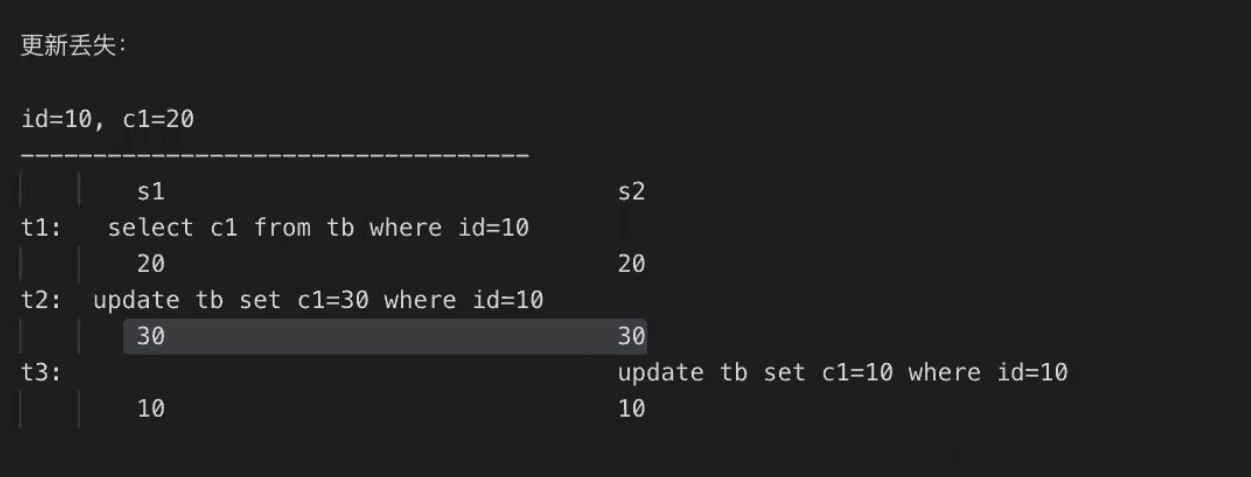
Tps和qps的概念。

TPS的T代表的是事务。Tps更多是业务层面的指标，



更新丢失：

Pxc的每个节点都可以写。



T1 时间做select查询，返回结果都是20

T2时间s1更新，那么s1和s2的结果都是30

T3 时间s2更新，结果s1 和s2 结果为10

那么T2更新的数据，就”不见了”。

解决办法就是多点写入的时候要单点写入。只要是多节点写入就会出现这个问题。



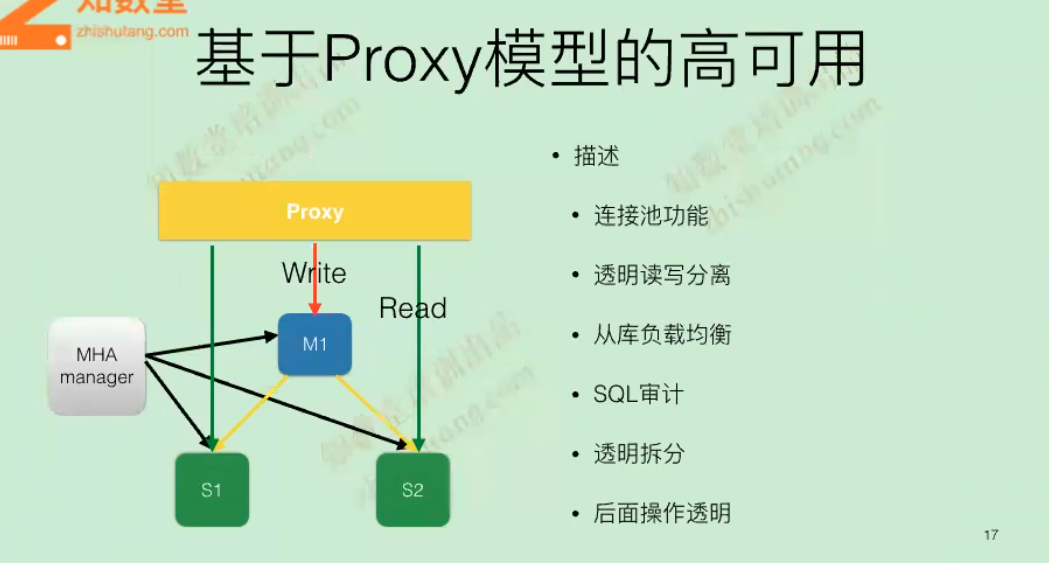
官方版本和percona。

如果要是mysql8.0 最好使用官方的，mysql5.7的可以随意选择。

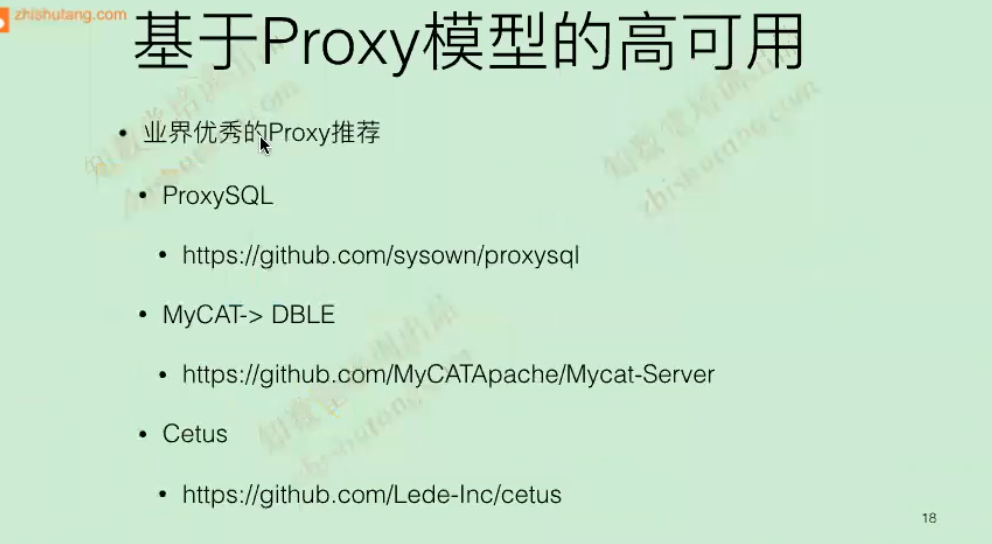
淘宝基于percona做二次开发。



Proxy模型

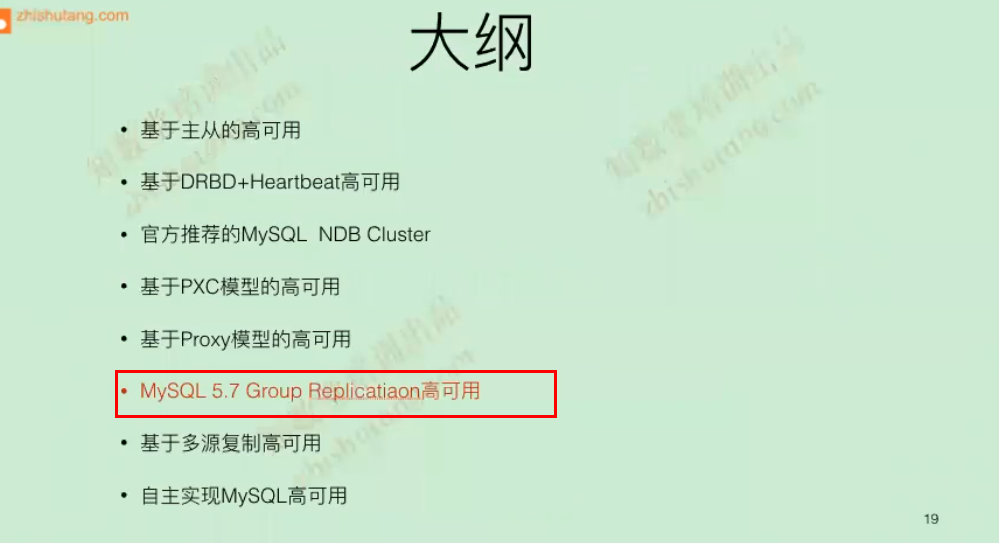


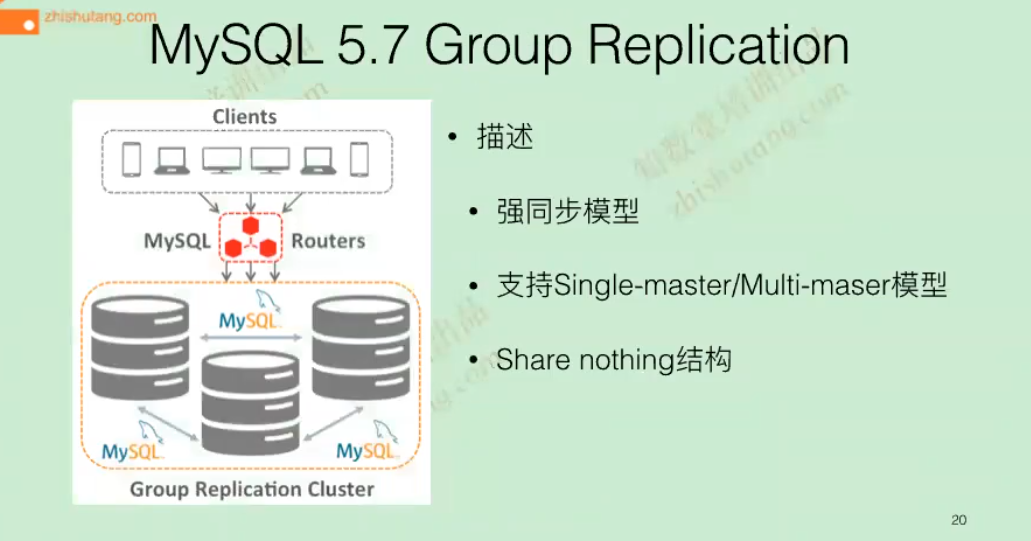
首推ProxySQL，也是知数堂讲解的内容



业内优秀的Proxy推荐

Mycat不推荐使用，作者不维护了。



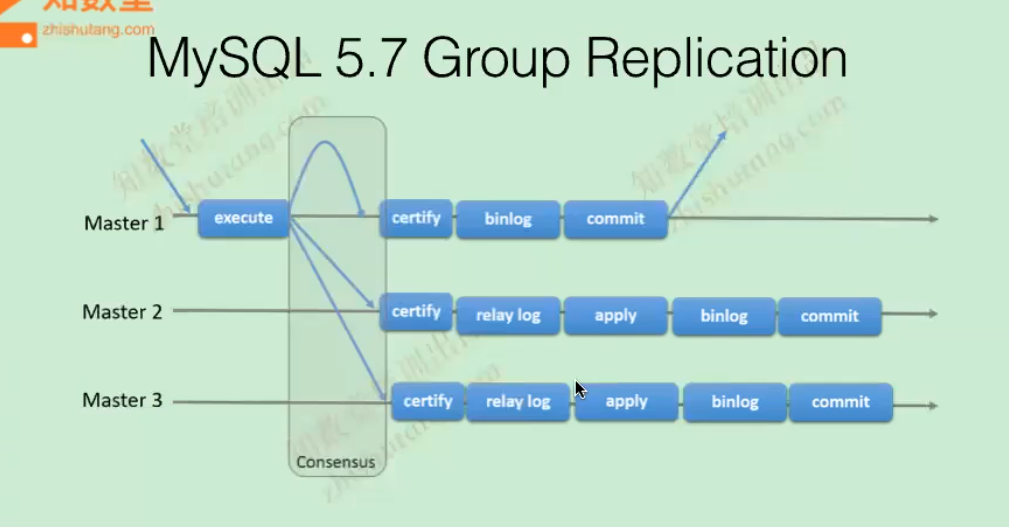


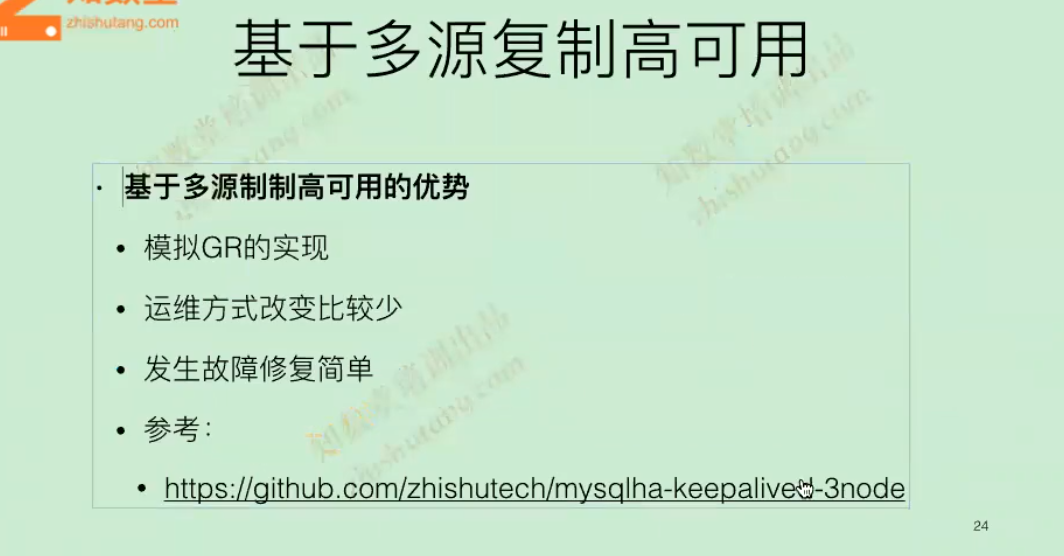
Group Replication 实际上就是我们说的MGR。官方主推的，如果在使用中发现MGR或者是MySql8.0的bug，官方是很重视的，如果不是MGR或者mysql8.0的bug，官方会把bug降级。

在使用MGR的时候要注意的：

1. 大事务
2. 在Multi-master模式的情况下，更新丢失，DDL会造成整个集群cresh。

在避免了以上两个问题的情况下MGR很稳定。





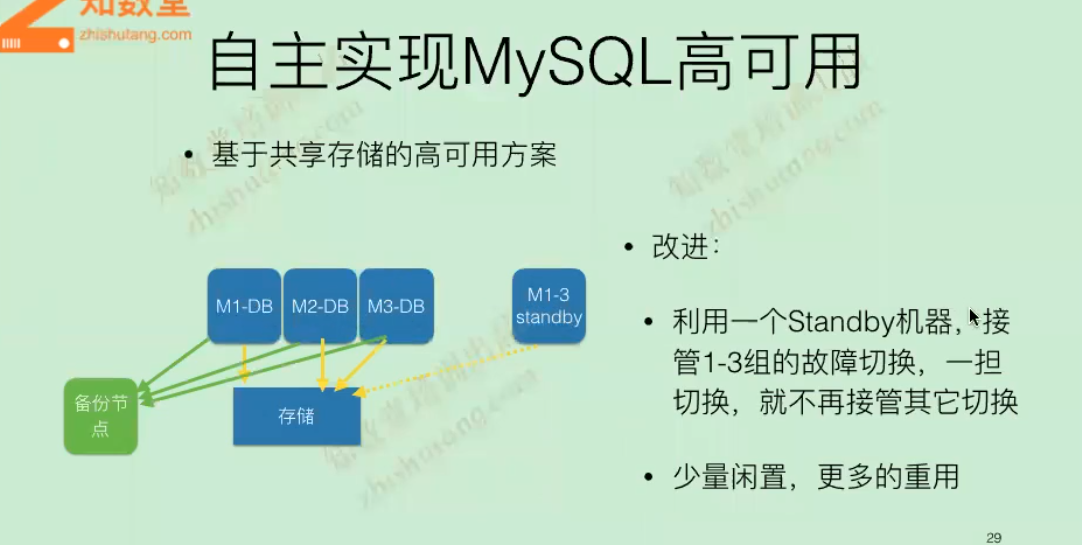
去网址去看一看即可。可以自己尝试去搭建一下。

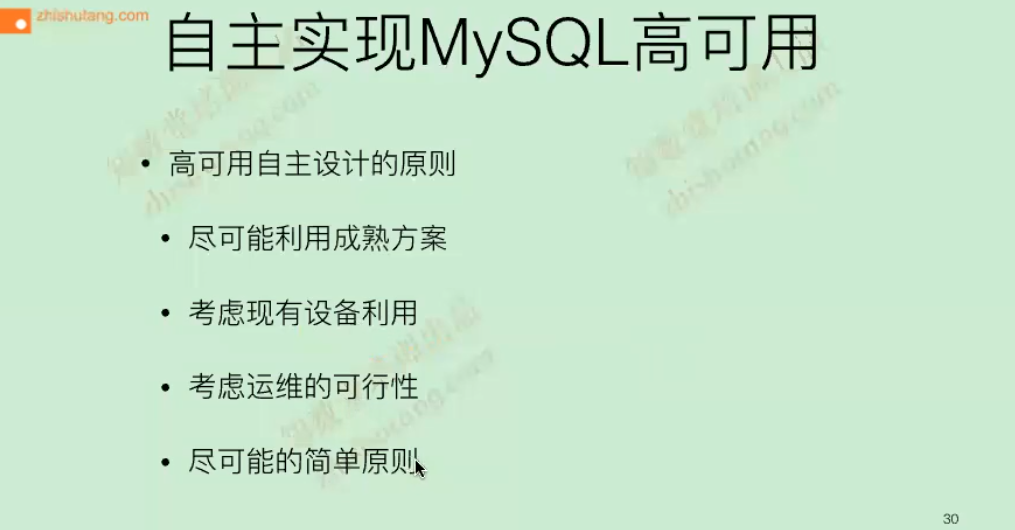


Golang 二次开发，跟里面的人很熟悉。

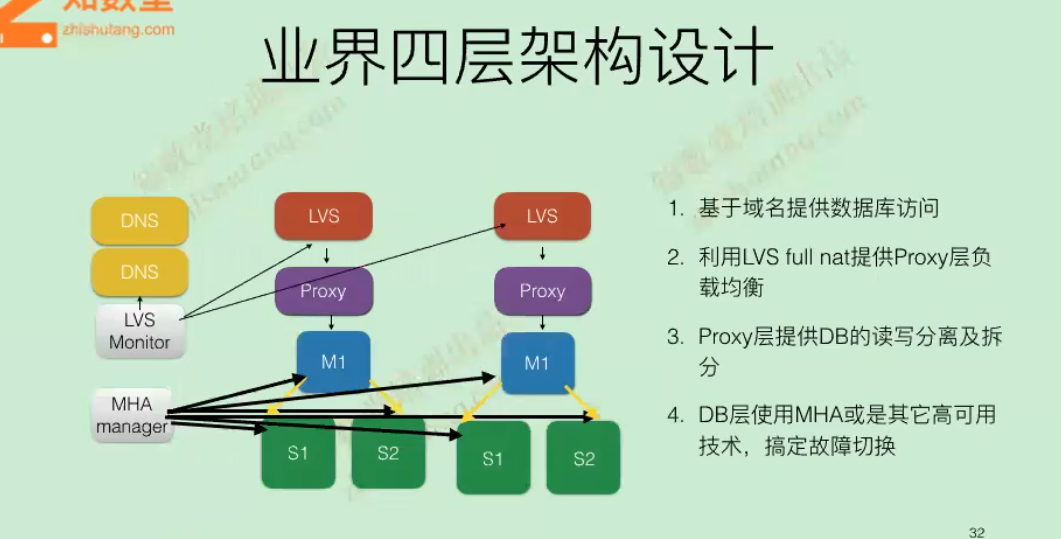


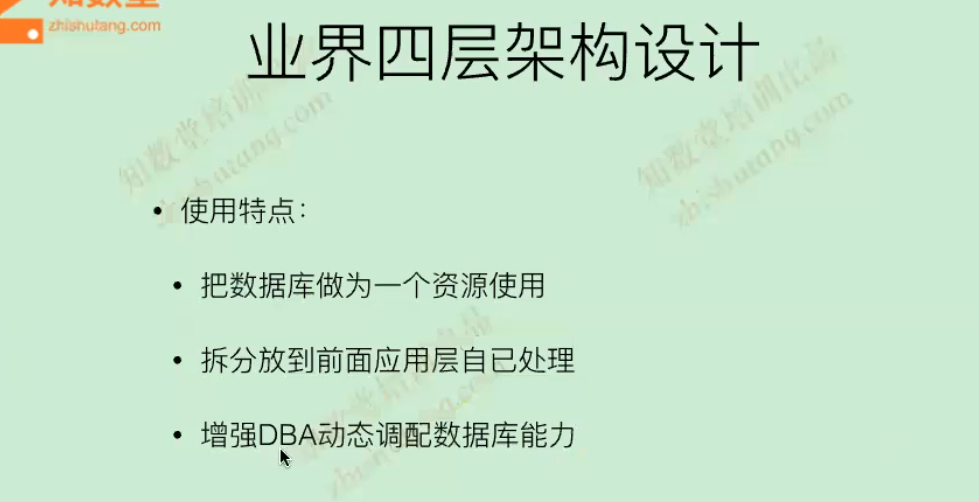
共享存储，如果某个库把存储的IO全部占用，其他的库可能就没有办法使用了。错峰





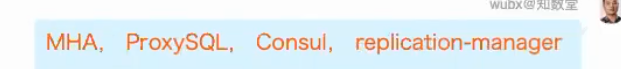
Baidu使用的解决方案：







Orchestrator - - 头条，JD都在使用

都会讲解。

为什么要用MHA，因为大公司最为成熟的技术就是MySql5.6