私塾在线《高级软件架构师实战培训阶段二》

——跟着cc学架构系列精品教程

MySQL的主从复制-1

- 概述

MySQL提供了复制的功能，就是把一个数据库的数据复制到其它数据库的机制，源数据库称为主服务器，目的数据库称为从服务器；是构建高性能应用的基础，比如：实现数据库的读写分离；实现数据库的高可用等。

- 基本实现的原理

1：主服务器将更新操作写入二进制日志文件（binary log ）

2：当从服务器连上主服务器的时候，它告诉主服务器它最后一次成功读取更新日志的位置，然后拷贝该位置后的二进制日志文件到它的中继日志(relay log)

4：从服务器会重做中继日志中的事件，更新到自己数据库中

- 支持的复制类型

1：基于语句的复制：在从服务器上执行与主服务器上一样的SQL语句

2：基于行的复制：把改变的内容复制过去，而不是把命令在从服务器上执行一

3：混合类型的复制：默认采用基于语句的复制，一旦发现基于语句的无法精确的复制时，就会采用基于行的复制。

MySQL的主从复制-2

- 复制的常用拓扑结构

1：一主多从

由一个master和多个slave组成，Slave之间并不相互通信，只能与master进行通信。如果写操作较少，而读操作很多时，可以采取这种结构。

注意当slave增加到一定数量时，slave对master的负载以及网络带宽都会成为一个严重的问题。

2：互为主从（主动双主Master-Master）

复制中的两台服务器，既是master，又是另一台服务器的slave。

3：互为主从（被动双主Master-Master）

就是所有的写操作只在其中一个主上进行，另外一个充当从服务器，只有当主服务器挂了，另外一个才转为主服务器

4：级联复制（Master-Slaves-Slaves）

也就是打开log-slave-update选项，然后通过二级（或者是更多级别）复制来减少Master端因为复制所带来的压力，相当于从服务器再带从服务器器

MySQL的主从复制-3

- 复制体系结构的基本原则

1：每个slave只能有一个master

2：每个slave只能有一个唯一的服务器ID

3：每个master可以有很多slave

4：如果你设置log_slave_updates，slave可以是其它slave的master

5：MySQL不支持多主服务器复制，即一个slave可以有多个master

- 复制的最大问题——复制延时

由于所有的写操作都是先在Master上操作，然后同步更新到Slave上，所以从Master同步到Slave机器有一定的延迟，当系统很繁忙的时候，延迟问题会更加严重，Slave机器数量的增加也会使这个问题更加严重。

MySQL的主从复制-4

- 配置使用

1：修改master配置

（1）设置log-bin=mysql-bin ，也就是启用二进制日志，必须

（2）设置server_id=1-231的整数，默认是1，要做到全局唯一

2：修改从slave配置

（1）设置log-bin=mysql-bin ，不必须；（2）设置server_id=任意的整数

3：启动两个MySQL

4：在master上建立帐户，并授权这个账户能操作slave，最好是单独建立一个账户：GRANT REPLICATION SLAVE ON *.* to 'root'@'%' identified by ''

5：查询master的状态：show master status;

6：为slave配置master：change master to master_host='localhost',master_port=3306,master_user='root',master_password='', master_log_file='mysql-bin.000001',master_log_pos=318;

7：启动从服务器的复制功能：start slave; ，关闭是 stop slave

8：检查从服务器复制功能状态：show slave status ，要求Slave_IO_Running和Slave_SQL_Running 必须都为yes

MySQL的读写分离

- 概述

  数据库经过复制后，多台机器拥有一样的数据，这样就可以同时利用这些机器来分担应用的请求，通常的方式是，主服务器负责处理写的请求，而其余从服务器负责处理读的请求，这就是所谓的读写分离。

- 读写分离的实现

  核心任务就一个：决定请求交给哪一个服务器来处理

- 开源方案

- 自行实现的基本思路

Galera for MySQL的集群-1

- 概述

  Galera Cluster for MySQL是一套基于同步复制的、多主的MySQL集群解决方案。使用简单，没有单点故障，可用性高，读写性能高，可扩展性好。

- 主要特点

  同步复制，主备无延迟

  支持多主同时读写，保证数据一致性

  真正行级别的并发复制

  集群中各节点保存全量数据

  可以在任意节点上进行读写，无需再读写分离了

  自动添加新节点和剔除故障节点，高HA

  不需要写binlog

  对应用透明，客户端连接跟操作单台MySQL数据库的体验一致

- Wsrep（Write Set Replication集合写入式复制）

  Wsrep定义了一系列应用回调和复制调用库，来实现事务数据库同步写集的复制，实现抽象的，隔离的复制。虽然这个接口的主要目标是基于认证的多主复制，但同样适用于异步和同步的主从复制。 Galera本质是一个wsrep提供者（provider）。

Galera for MySQL的集群-2

- GTID(Global Transaction ID)

  WSREP模型中，用状态来表示数据库内容被修改过，GTID就用来标识状态的改变，这种改变的状态表示为一系列的原子变化，集群中的所有节点通过同步复制具有相同的状态并以相同的顺序应用这些状态。

- 状态快照转移(SST)

1：mysqldump：此方法是通过定义阻塞，阻止修改自身状态转移的持续时间。这也是最慢的方式，可能会带来高负载的问题。

2：直接拷贝数据文件：rsync，xtrabackup等方法都属于这类。这些方法比mysqldump快，但有一定的局限性，发送和接收服务器配置参数需要高度相似，xtrabackup可以无阻塞发送

- 仲裁线程（garbd守护线程）

  这个线程是个不存储数据的Galera节点，garbd节点可以帮助我们在最少两个节点的集群中避免主节点分裂的情况

- Donor和Joiner

  新接入的节点叫Joiner,给joiner提供复制的节点叫Donor。

  在生产环境中，建议设置一个专用的donor，这个专用的donor不执行任何来自客户端的SQL请求，这样做有以下几点好处：

1：数据的一致性：因为donor本身不执行任何客户端SQL，所以在这个节点上发生事务冲突的可能性最小，因此，如果发现集群有数据不一致时，donor上的数据应该是整个集群中最准确的。

Galera for MySQL的集群-3

2：数据安全性：

因为专用donor本身不执行任何客户端SQL，所以在这个节点上发生灾难事件的可能性最小，因此当整个集群宕掉的时候，该节点应该是恢复集群的最佳节点。

3：高可用性：

专用donor可以作为专门的state snapshot donor。因为该节点不服务于客户端，因此当使用此节点进行sst的时候，不影响用户体验，并且前端的负载均衡设备也不需要重新配置

- 基本的复制过程

1：当客户端发起commit命令时（此时并没有发生真正的commit），所有本事务内对数据库的改动与改动数据行的主键都会被收集到一个写入集（writeset）中

2：该写入集随后会被复制到其它节点

3：该写入集会在每个节点上进行确认性认证测试，来判断该写入集是否可以被应用。如果认证测试失败，写入集会被丢弃并且原始事务会被回滚，如果认证成功，事务被提交并且写入集会在剩余节点进行应用。

4：认证测试在Galera集群中的实现取决于全局事务顺序，每个事务在复制期间都会被指派一个全局顺序序列；当一个事务到达提交点时，该事务会知道当前与该事务不冲突的最新已提交事务的顺序序号，在这两个事务的全局顺序序列之间的间隔是不确定区域，在该区域间的事务相互是“看不到”对方的影响的，但所有在这间隔之间的唯物都会被进行主键冲突检测

Galera for MySQL的集群-4

- 使用注意

1：使用Galera必须要给MySQL-Server打wsrep补丁，可以直接使用官方提供的已经打好补丁的MySQL安装包，如果服务器上已经安装了标准版MYSQL，需要先卸载再重新安装

2：MySQL/Galera集群只支持Innodb / XtraDB / Maria存储引擎，如果你的数据表使用的MyISAM，需要转换为InnoDB，否则记录不会在多台复制

3：MySQL 5.5及以下的InnoDB引擎不支持全文索引，如果之前使用MyISAM并建了全文索引字段的话，只能安装MySQL 5.6 with wsrep patch

4：所有数据表必须要有主键（PRIMARY）,如果没有主键可以建一条AUTO_INCREMENT列

5：MySQL/Galera集群不支持下面的查询：LOCK/UNLOCK TABLES，不支持下面的系统变量：character_set_server、utf16、utf32及ucs2

6：数据库日志不支持保存到表，只能输出到文件，不能设置binlog-do-db、binlog-ignore-db

7：跟其他集群一样，为了避免节点出现脑裂而破坏数据，建议Galera集群最低添加3个节点

8：在高并发的情况下，多主同时写入时可能会发生事务冲突，此时只有一个事务请求会成功，其他的全部失败。可以在写入/更新失败时，自动重试一次，再返回结果。

9：不支持XA事务

10：允许最大的事务大小由wsrep_max_ws_rows和wsrep_max_ws_size定义，任何大型操作将被拒绝

11：节点中每个节点的地位是平等的，没有主次，向任何一个节点读写效果都是一样的

12：第一台启动的服务器以空地址启动如：mysqld_safe -wsrep_cluster_address=gcomm:// >/dev/null &

Galera for MySQL的集群-5

n 安装配置

一：安装python2.7.6，这个在前面讲Fabric的时候已经安装过了

二：安装scons2.3.6

1：http://sourceforge.net/projects/scons/files/scons/2.3.6/scons-2.3.6.tar.gz

2：解压，进入解压后的目录，然后：python setup.py install

三：安装cmake3.3.1

1：http://www.cmake.org/download/

2：解压，进入解压后的目录，然后依次：./bootstrap， make，make install

四：安装mysql-wsrep

1：http://galeracluster.com/downloads/ 下载源码

2：解压，进入解压后的目录，然后依次：

（1）：BUILD/compile-pentium

（2）：cmake -DWITH_WSREP=ON -DWITH_INNODB_DISALLOW_WRITES=1

（3）：make

（4）：make install

3：安装后，依次执行如下步骤：

（1）：cd /usr/local/mysql/

Galera for MySQL的集群-6

（2）：groupadd mysql
（3） ：useradd -r -g mysql mysql
（4）：chown -R mysql:mysql .
（5）：./scripts/mysql_install_db --no-defaults --datadir=/usr/local/mysql/data/ --user=mysql
（6）：chown -R root .
（7）：chown -R mysql /usr/local/mysql/data
（8）：echo "export PATH=$PATH:/usr/local/mysql/bin" >> /etc/profile
（9）：source /etc/profile
（10）：mysqld_safe --wsrep_cluster_address=gcomm:// >/dev/null &
（11）：启动后，可以登录mysql客户端，检查一下是否能操作，然后关闭服务器：mysqladmin -uroot -p shutdown

五：安装Galera复制插件
对于CentOS/RHEL5如下：其它的请查看相关安装文档

1：安装gcc44 gcc44-c++
（1）yum install gcc44 gcc44-c++
最好先删除本机已安装的gcc和g++，先rpm -qa | grep gcc查看，然后rpm -e 文件名，一个一个的删除掉，实在删不掉的就留着
（2）export CC=gcc44
（3）export CXX=g++44
（4）到/usr/bin下面，建立软链：ln -s gcc44 gcc ， ln -s g++44 g++

Galera for MySQL的集群-7

2：安装boost141

（1）删除已经安装的boost，直接删除相关文件夹即可，如：rm -rf /usr/include/boost/

（2）http://www.boost.org/ ，CentOS/RHEL5是用141版本的

（3）然后依次执行：./bootstrap.sh ，./bjam ，./bjam install --prefix=/usr/local

3：然后安装Check unit test library和OpenSSL：

```
yum install check-devel openssl-devel
```

4：安装galera

（1）http://galeracluster.com/downloads/ 下载源码

（2）解压，进入解压后的目录，然后执行：scons

如果执行的时候，遇到“undefined reference to ‘___assert_fail’ ”之类的错误，基本是galera的单元测试代码引起的，进到每个模块里面，修改SConscript文件，主要是把里面涉及的test部分去掉就可以了。当然原因主要是机器里面有各种版本的lib，导致引用某些包的时候不正确。

大致有：galera、galerautils、gcache、gcomm、gcs/unit_tests，具体的你可以看错误信息提示，会明确告诉你是哪个模块的哪个文件出的问题

（3）把插件复制到相应的位置

```
cp garb/garbd /usr/local/mysql/bin/
cp libgalera_smm.so /usr/local/mysql/lib/plugin/
```

Galera for MySQL的集群-8

5：初始化配置

（1）mkdir -p /etc/mysql/conf.d/

（2）cp /usr/local/mysql/support-files/mysql.server /etc/init.d/mysql

（3）/usr/local/mysql/bin/mysqld_safe --wsrep_cluster_address=gcomm:// >/dev/null &

（4）进入mysql客户端，依次执行：

```
SET wsrep_on=OFF;
GRANT ALL ON *.* TO galera@'%' IDENTIFIED BY 'cc';
```

接下来修改root的帐号密码，mysqladmin -u root password 'cc';

```
GRANT ALL PRIVILEGES ON * . * TO 'root'@'%' IDENTIFIED BY 'cc' WITH GRANT OPTION
    MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_UPDATES_PER_HOUR 0
    MAX_USER_CONNECTIONS 0;
```

（5）关闭服务器：mysqladmin -uroot -p shutdown

（6）修改mysql的配置文件：/usr/local/mysql/my.cnf，添加如下内容：

```
basedir = /usr/local/mysql
datadir = /usr/local/mysql/data
port = 3306
server-id=101
socket = /tmp/mysql.sock
#pid-file=/data/mydata/mysql.pid
```

Galera for MySQL的集群-9

```
wsrep_node_name = mysql1
wsrep_provider = /usr/local/mysql/lib/plugin/libgalera_smm.so
wsrep_sst_method = rsync
wsrep_sst_auth=galera:cc      #使用sst的用户和密码，这里如果开启，需要在mysql上创建该用户，并授
    予其足够的权限
wsrep_cluster_address=gcomm://192.168.1.202:4567
default_storage_engine=InnoDB
innodb_autoinc_lock_mode=2
innodb_locks_unsafe_for_binlog=1
innodb_flush_log_at_trx_commit=1
innodb_file_per_table=1
binlog_format=ROW
log-bin=mysql-bin
relay-log=mysql-relay-bin
log-slave-updates=1
（7）再次启动mysql：  /usr/local/mysql/bin/mysqld_safe --wsrep_cluster_address=gcomm://
    >/dev/null &
启动第二个时：/etc/init.d/mysql start --wsrep_cluster_address=gcomm://192.168.1.202:4567
（8）检查端口，应该有3306和4567两个端口，如：  netstat -tunlp |grep 4567，还可以在mysql客户端，
    查看是否有warep数据，如：show status like 'wsrep%'
```

Galera for MySQL的集群-10

- 常用的监控管理命令

1：查看wsrep版本：SHOW GLOBAL STATUS LIKE 'wsrep_provider_version ‘

2：查看wsrep有关的所有变量：SHOW VARIABLES LIKE 'wsrep%' \G

3：查看Galera集群状态：show status like ‘wsrep%’，参数说明如下：

（1）集群完整性检查

wsrep_cluster_state_uuid：在集群所有节点的值应该是相同的，有不同值的节点，说明其没有连接入集群

wsrep_cluster_conf_id：正常情况下所有节点上该值是一样的，如果值不同，说明该节点被临时”分区”了，当节点之间网络连接恢复的时候应该会恢复一样的值

wsrep_cluster_size：如果这个值跟预期的节点数一致，则所有的集群节点已经连接

wsrep_cluster_status：集群组成的状态。如果不为”Primary”,说明出现”分区”或是”split-brain”状况

（2）节点状态检查

wsrep_ready：该值为ON，则说明可以接受SQL负载；如果为Off，则需要检查wsrep_connected

wsrep_connected：如果该值为Off，且wsrep_ready的值也为Off，则说明该节点没有连接到集群

wsrep_local_state_comment：如果wsrep_connected为On，但wsrep_ready为OFF，则可以从该项查看原因

Galera for MySQL的集群-11

（3）复制健康检查

wsrep_flow_control_paused：表示复制停止了多长时间，即表明集群因为Slave延迟而慢的程度，值为0~1，越靠近0越好，值为1表示复制完全停止。

wsrep_cert_deps_distance：有多少事务可以并行应用处理，wsrep_slave_threads设置的值不应该高出该值太多

wsrep_flow_control_sent：表示该节点已经停止复制了多少次

wsrep_local_recv_queue_avg：表示slave事务队列的平均长度，slave瓶颈的预兆。最慢的节点的wsrep_flow_control_sent和wsrep_local_recv_queue_avg这两个值最高，这两个值较低的话，相对更好。

（4）检测慢网络问题

wsrep_local_send_queue_avg：网络瓶颈的预兆，如果这个值比较高的话，可能存在网络瓶颈

（5）冲突或死锁的数目

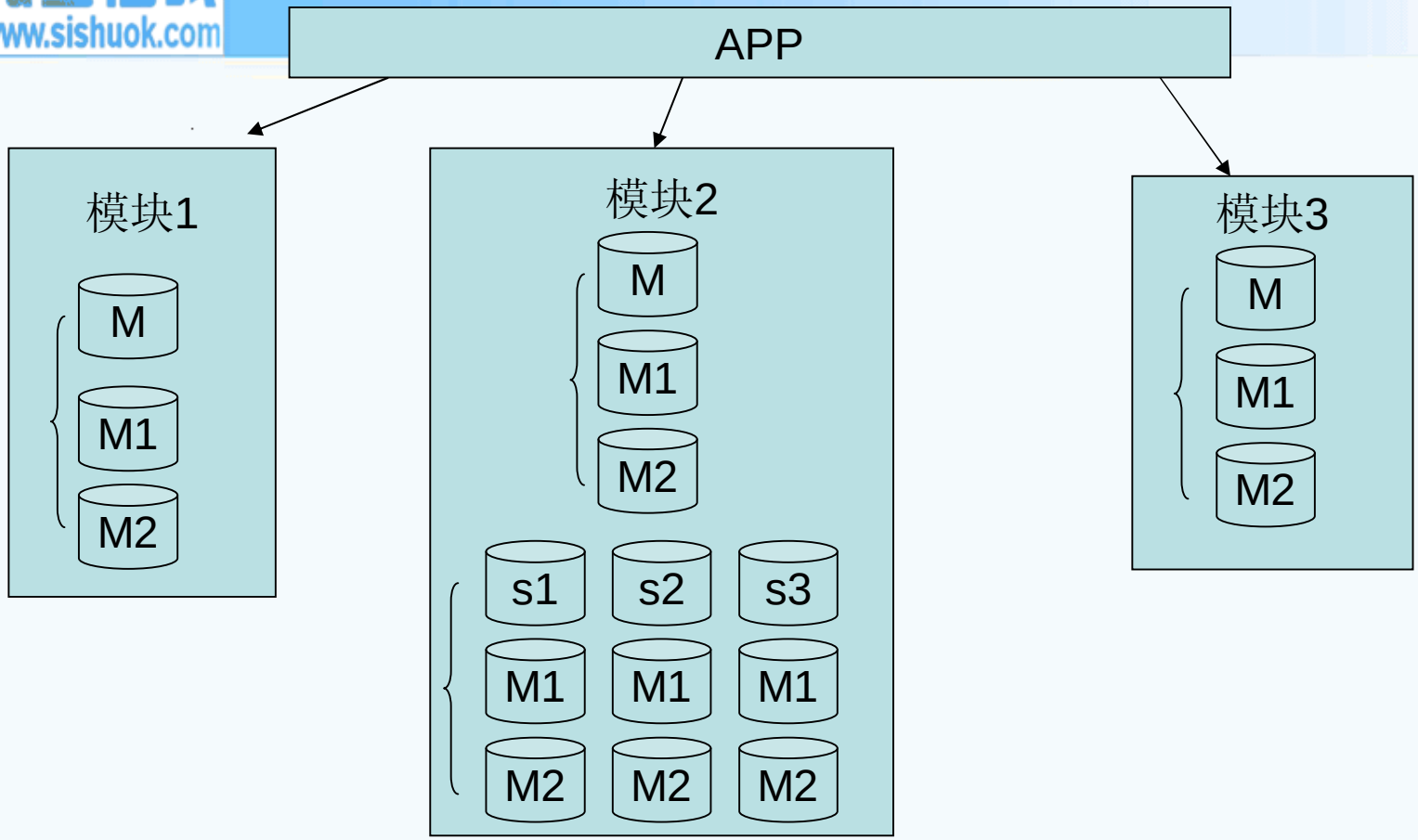
wsrep_last_committed：最后提交的事务数目

wsrep_local_cert_failures和wsrep_local_bf_aborts：回滚，检测到的冲突数目

MySQL的HA和负载均衡

- 为何需要
- 常见的架构
- 如何实现

  基本方案：Keepalived + LVS + MySQL + Galera


APP
模块1
M
M1
M2
模块2
M
M1
M2
s1
s2
s3
M1
M1
M1
M2
M2
M2
模块3
M
M1
M2

常见的MySQL服务器配置优化-1

- key_buffer_size：为索引键设置缓冲区大小，使用时才真正分配
- innodb_buffer_pool_size：定义InnoDB缓冲池的大小
- innodb_additional_mem_pool_size：定义InnoDB的数据字典和内部数据结构的缓冲池的大小
- query_cache_size：设置查询缓存的大小
- max_heap_table_size：定义一个Memory存储引擎表的最大容量
- tmp_table_size：设置临时表能用的内存大小
- join_buffer_size：全表连接操作所使用的内存缓冲区大小
- sort_buffer_size：设置用于排序的缓存大小
- read_buffer_size：在查询有需要的时候，会为该缓存分配内存，一次性分配指定的大小
- read_rnd_buffer_size：和上个参数的区别是：只会分配需要的内存大小
- table_cache_size：缓存表的数量
- thread_cache_size：缓存线程的空间大小
- innodb_max_dirty_pages_pct：设置在缓存池中保存的最大的脏页的数量
- innodb_log_file_size：日志文件大小
- innodb_log_buffer_size：日志缓存区大小

常见的MySQL服务器配置优化-2

- innodb_flush_log_at_trx_commit：设置日志缓冲刷新到文件的机制（0-每秒刷新一次，1-每次事务刷新一次，2-每次提交时把缓存写到文件，但不刷新）
- innodb_flush_method：设置如何跟文件系统交互
- innodb_file_per_table：每个表一个文件
- innodb_thread_concurrency：设置一次性有多少线程进入内核
- max_length_for_fort_data：用于排序数据的最大长度，可以影响MySQL选择那个排序算法
- optimizer_switch：设定MySQL优化器中哪个高级索引合并功能被开启
- default_storage_engine：默认的存储引擎
- max_allowed_packet：结果集的最大容量
- sql_mode：支持的各种服务器SQL模式
- innodb_strict_mode：定义一个专门为InnoDB插件提供的服务器SQL模式级别
- max_connections：最大连接数

……

NoSQL的应用

- 使用Redis配合MySQL
  做持久化要求不高部分数据的持久化

- 使用MongoDB配合MySQL

1：做读多写少数据的持久化
2：做数据量大的数据的持久化