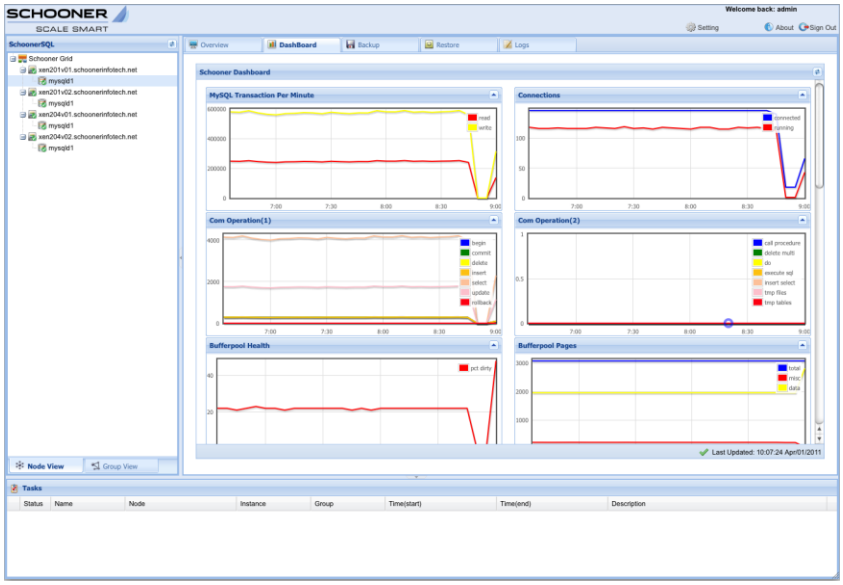
Deployment Guide

SchoonerSQL™

A Full High-Availability High-Performance Build of MySQL and InnoDB

Version 5.1

SCHOONER


SCHOONER
SCALE SMART
Welcome back: admin
Setting
About
Sign Out
SchoonerSQL
Schooner Grid
xen201v01.schoonerinfotech.net
mysqld1
xen201v02.schoonerinfotech.net
mysqld1
xen204v01.schoonerinfotech.net
mysqld1
xen204v02.schoonerinfotech.net
mysqld1
Node View
Group View
Overview
DashBoard
Backup
Restore
Logs
Schooner Dashboard
MySQL Transaction Per Minute
Connections
Com Operation(1)
Com Operation(2)
Bufferpool Health
Bufferpool Pages
Last Updated: 10:07:24 Apr/01/2011
Tasks
Status
Name
Node
Instance
Group
Time(start)
Time(end)
Description

Technical Support

Technical support for Schooner Information Technology products in North America is available from the following sources:

- Phone: (877) 888-5064; (408) 888-1619
- Fax: (408) 736-4212
- Email: support@schoonerinfotech.com
- Website: www.schoonerinfotech.com/support

Documentation ID: SSQL-v5.1-DG-011

SchoonerSQL™– Deployment Guide

Issued December 2011

Schooner Information Technology
501 Macara Ave., Suite 101
Sunnyvale, CA 94085, USA
Tel: (408) 773-7500 (Main)
(877) 888-5064 (Sales and Support)
Fax: (408) 736-4212

Contents

Chapter 1: Introduction

This guide explains how to solve your database problems by deploying SchoonerSQL™. We first cover SchoonerSQL™ replication concepts, what is and is not supported in terms of MySQL features and then present SchoonerSQL™ deployment topologies, requirements and instructions on how to configure them.

Chapter 2: SchoonerSQL™ Replication Concepts

This chapter describes SchoonerSQL™ replication and recovery at a conceptual level. The purpose is to provide you with a framework for understanding the motivations for the various deployment scenarios and their configuration.

Synchronous Replication

SchoonerSQL™ synchronous replication provides immediate updates of all nodes in a cooperating cluster (called a synchronous replication group).

Replication Group

Synchronous replication in SchoonerSQL™ applies to all instances within a replication group. A replication group is defined by:

- A set of up to 8 SchoonerSQL™ instances. One instance will be assigned as master and will support write access. It may also support read access. The remaining instances are slaves and support read access.
- A MySQL application port. All instances are assigned the same port, which means that a given node may host only 1 instance per replication group. However, a node may host multiple replication groups.
- A consistent read port (a MySQL application port associated with the main MySQL application port). See discussion on read consistency.
- A set of virtual IP (VIP) addresses.
- A set of failover namespaces (see discussion below).

Replication groups support the following features:

- Synchronous replication of updates across all members of the group.
- Automatic failover of the synchronous master.
- Automatic recovery and synchronization of failed, stopped or newly added group instances.
- Automatic detection and handling of split-brain conditions.

Instance Recovery

When an instance is added to a group or a stopped instance is restarted, database recovery will be required. A new instance must be synchronized using a full backup of a donor instance. An existing, restarted instance be synchronoized using an incremental backup of the donor instance.

SchoonerSQL™ makes use of hot backup utilities to perform instance recovery. Recovery is fully automated.

Consistent Read Port

SchoonerSQL™ instances belonging to a replication group will provide consistent reads across all members of the group when the assigned MySQL

application port is used. However, the data provided via this port may be buffered and not physically applied to permanent storage.

A second, associated MySQL application port is automatically assigned to each SchoonerSQL™ instance that provides consistent, permanent data.

Failover Namespace

A failover namespace is simply an ordered list of replication group instances to be used in case of group master failure. A set of failover namespaces is configured when the group is created (or later modified). Slaves must specify which namespace is to be used in the event of master failure. Only one namespace should be active at any time.

Split-Brain Detection

SchoonerSQL™ provides automatic split-brain detection among replication group instances. However, it does not provide split-brain detection at the application level.

A split-brain event occurs whenever members of a replication group lose the ability to communicate with one or more of the other group members. Network partitioning is the typical cause.

The result of a network partition is two or more sub-groups that are unable to communicate with other sub-groups. In order to continue operation, one of the sub-groups must take on the group master responsibility.

In SchoonerSQL™, an existing group master will remain as such and all other instances in other partitioned sub-groups will be halted. This ensures that only one sub-group will be able to perform updates and thus avoid potential data inconsistencies.

SchoonerSQL™ cannot detect network partitioning at the application level and therefore other methods must be employed for this case.

Asynchronous Replication

SchoonerSQL™ supports the standard MySQL asynchronous replication model with the following enhancements:

- Automated failover of master instances and automatic re-connect of slaves.
- Parallel slave log applier threads.

Automated Master Failover

While standard MySQL replication supports master instance failover, the process is manual and may require significant downtime due problems in reconfiguring slaves for the new master's binlog file and position.

SchoonerSQL™ asynchronous replication elminates this problem by automatically reconfiguring slaves for a new master's binlog file and position.

Parallel Slave Log Applier Threads

Slave lag in stock MySQL replication environments is greatly exacerbated through the use of single-threaded slave log appliers. This serialization of updates at each slave can lead to large inconsistencies among all members of the asynchronous cluster. Semi-synchronous and per-database slave log appliers reduce the problem somewhat but do not eliminate it.

SchoonerSQL™ asynchronous slaves employ multiple, parallel log applier threads and therefore maintain a much more consistent database state. Slave lag is minimized.

Chapter 3: Limitations, Issues and Requirements

SchoonerSQL™ in general supports the standard MySQL feature set. This chapter identifies those features that are not supported due to the nature of SchoonerSQL™ replication functions.

Limitations

Database Engines

SchoonerSQL™ supports the InnoDB engine. No other engine types are supported. (An exception are MySQL tables supported by the MyISAM engine. See below for a discussion on these table types.)

The reason that SchoonerSQL™ supports only InnoDB is that SchoonerSQL™ replication and automated recovery is based on a transactional model. Non-transactional table types are not amenable to these features.

MySQL System Tables

The following MySQL system tables are supported by the MyISAM engine:

- columns_priv
  - Column level privileges.
  - Only GRANT/REVOKE commands are supported.
- db
  - Database level privileges.
  - Only GRANT/REVOKE commands are supported.
- event
  - Event information.
  - Only CREATE/ALTER/DROP commands are supported.
- func
  - User defined functions.
  - User defined functions are fully supported.
- general_log
  - General query log table.
  - General query log is fully supported.
- help tables
  - help_category
  - help_keyword
  - help_relation
  - help_topic

    - HELP is fully supported.
- host
    - Obsolete.
- ndb_binlog_index
    - NDB is not supported.
- plugin
    - Engine plugins.
    - SchoonerSQL™ InnoDB plugin is supported.
    - No other plugins are supported.
- stored procedure tables
    - proc
    - procs_priv
    - Stored procedures are fully supported.
    - Stored functions are not supported.
- servers
    - Federated servers.
    - Not supported.
- slow_log
    - Slow query log table.
    - Slow query log is fully supported.
- tables_priv
    - Table level privileges.
    - Only GRANT/REVOKE commands are supported.
- time zone
    - time_zone
    - time_zone_leap_second
    - time_zone_name
    - time_zone_transition
    - time_zone_transition_type
    - Timezones are fully supported.
- user
    - User accounting, privileges.
    - Only the following commands are supported:
        - CREATE USER
        - DROP USER
        - GRANT

        - RENAME USER
        - REVOKE
        - SET PASSWORD
    - Arbitrary queries such as INSERT, DELETE, etc., are not supported.

Replication

SchoonerSQL™ supports both standard MySQL asynchronous replication and SchoonerSQL™ synchronous replication. This section describes the limitations on each type of replication.

Asynchronous Replication

The following are not supported by SchoonerSQL™ asynchronous replication:

- Tables other than SchoonerSQL™ InnoDB.
- STATEMENT based logging.
- Stored functions.
- SAVEPOINT.
- Transactions > 100G.

  A very large transaction may cause asynchronous slaves to fail with the error "log event entry exceeded max_allowed_packet; Increase max_allowed_packet on master".
- Multi-master asynchronous replication between two synchronous replication groups does not work.

Synchronous Replication

The following are not supported by SchoonerSQL™ synchronous replication:

- Tables other than SchoonerSQL™ InnoDB.
- STATEMENT based logging.
- Stored functions.
- SAVEPOINT.
- If the MySQL temp directory is within the data directory, a full recovery of a synchronous instance removes it which would cause the instance to fail on startup. Make the the MySQL temp directory is located outside of the data directory.

Issues

Database Engine

The following issues exist in the SchoonerSQL™ 5.1 InnoDB engine:

- If the replication and client network interfaces are in same subnet in a SchoonerSQL™ node, the replication traffic may still go to client network interface. The workaround for this issue is to add static route for all peer

nodes replication IP in each SchoonerSQL™ node to route replication traffic through the replication interface.

- In certain conditions where a network is completely unstable, if a recovering slave splits from group before finding a donor instance and tries to rejoins again with the original group, all the mysqld instances in the cluster will abort. The workaround is to start the instance one by one if the network is unstable.
- SchoonerSQL™ synchronous slave recovery and provisioning will fail If DDL statements are executed during the recovery process.
- A write transaction may be aborted when a slave joins or leaves the cluster. The MySQL client may fail upon this abort. The workaround is the client to retry the same transaction again.
- If a transaction contains both InnoDB and MyISAM statements, binlog entry won't be generated for the corresponding transaction. This will result in inconsistent data in the asynchronous slaves replicating from SchoonerSQL™ cluster.
- In conditions involving heavy DDL transactions along with heavy DML transactions the synchronous slave may abort.
- In certain conditions, some of the SchoonerSQL™ recovery process might not be cleaned up on the donor instance which may cause new slave recovery to hang. The workaround is to manually kill the sticky processes and restart the slave.
- Under rare circumstances, a SchoonerSQL™ instance may abort during shutdown.
- A recovering synchronous slave will fail if ALTER TABLE is executed during the recovery.

Administration

- Asynchronous slaves may not failover to a new asynchronous master if a node in the failover namespace is removed from a SchoonerSQL™ cluster and added to a new SchoonerSQL™ cluster with the same IP address. The workaround is stop asynchronous slave and reconfigure it.
- If a replication group does not have any live instances and more than one instances are started at the same time through CLI, all instances except the master instance may be brought down. The workaround is start master instance first and wait for it to become READY and then start all slave instances.
- If SchoonerSQL™ is stopped on more than one node in a particular order and started in the reverse order one by one, *the SchoonerSQL™ Administrator* will show incorrect states for the mysqld instances until SchoonerSQL™ is started on all the nodes.
- Changing configuration while the administration network is partitioned due to network split would result an inconsistent management view and

configuration. The workaround is to not change the configuration while thje administration network is in a partitioned condition.

- Changing the replication interface of a replication group while a SchoonerSQL™ node is rebooting will fail and will result in an inconsistent configuration. The workaround is to not change the replication interface while one or more SchoonerSQL™ nodes are rebooting.
- Attaching instance while a SchoonerSQL™ node is rebooting may fail. The workaround is to wait until the rebooting node comes up and then attach the instance.
- If the `/etc/sudoers` file does not have entry "`root ALL=(ALL) ALL`", the *SchoonerSQL™ Administrator* will not work.
- If a donor instance dies during slave recovery, the slave will abort and have corrupted data. Restarting the slave again while one or more new donors are available may keep failing. The workaround is manually clean the txlog directory and restart the slave.
- Changing configuration through simultaneous *SchoonerSQL™ Administrator* sessions is not supported.
- Instance creation will fail if the file `/root/.my.cnf` exists.

Deployment Requirements

The following are deployment requirements for SchoonerSQL™:

- Employ separate network connection for client and replication traffic.
- MyISAM is not supported (for numerous reasons inherent to MyISAM).
- Do not use NFS on Schooner sync nodes (NFS can cause 5+ seconds of delays in processing disjoint network packets, causing unnecessary cluster splits).
- Do not turn off binary logging and "--log-slave-updates" if using asynchronous replication.
- Do not increase network transfer thresholds as it causes network congestion.
- Do not change MySQL settings in production, test these in staging first and/or check with Schooner support first.
- In case MAN connections go down, two or more writable islands are possible.
- Synchronous replication performance and up time of the cluster (failure detection, heartbeats, etc.) depend on network quality. Intermittent network congestion (even 5 seconds) can trigger failure detection logic, thus we need to monitor at a granular level for such events.

Chapter 4: Topologies

This chapter describes the most common topologies for SchoonerSQL™.

- Master-Master (single writer)
- HA Master + High-Speed Asynchronous Slaves
- Consisten Read Scale-Out
- Metro Area Cluster
- Wide Area Cluster

Master-Master

Common methods for supporting master-master (single writer) MySQL deployments are:

- 2 MySQL instances, each configured as both asynchronous master and slave to one another.
- 2 MySQL master instances sharing storage via DRBD.

Each of these methods has issues in terms of maintenance, performance and reliability:

- Recovery of a MySQL master may require manual reconfiguration of its asynchronous slave.
- MySQL asynchronous replication is single-threaded. Large slave lags are not uncommon.
- DRBD is a low-level protocol that has no understanding of MySQL traffic. Performance can be very poor and recovery very slow.

SchoonerSQL™ synchronous replication groups provide a simple, high-performance and robust method for deploying a 2-node master-master topology:

- Configuration of a synchronous replication group is as simple as creating a SchoonerSQL™ instance and adding it to a synchronous replication group. All initialization and data transfer is handled automatically by the *SchoonerSQL™ Administrator*.
- SchoonerSQL™ automatically handles failover of instances within a synchronous replication group.
- SchoonerSQL™ automatically handles migration of instances within a synchronous replication group.

The figure below shows a 2-node SchoonerSQL™ master-master deployment.

The figure depicts a SchoonerSQL™ synchronous replication group consisting of a synchronous master and a synchronous slave. Both instances support concurrent reads. Only the master instance supports writes.

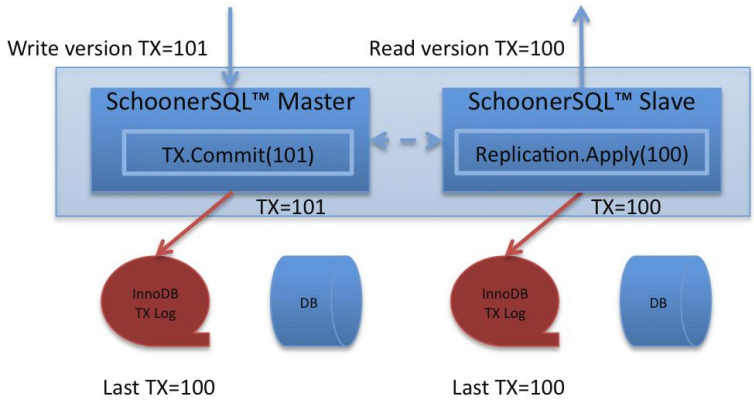


Figure 4-1: Master-Master

As stated in the previous chapter, the communications links for cluster management and replication traffic should be separated in order to avoid loss of heartbeat messages that could lead to a false split-brain scenario.

Also, network communications paths should be redundant at all levels (host interfaces, bonded links, redundant switches and routers, etc.) in order to provide the highest levels of availability.

In order to perform instance recovery after instance stoppage (scheduled or otherwise) or when adding a new instance to a group, sufficient storage space is required to perform the recovery. This temporary space should be separated from the database directory as the recovery process will remove and recreate the temporary space each time it is required.

Create a Synchronous Replication Group

To create a SchoonerSQL™ synchronous replication group:

- Click on Schooner Grid in the navigation panel.
- Click on Add Group in the overview tab.

Figure 4-2: Add Group Button

- Enter the name of the replication group.

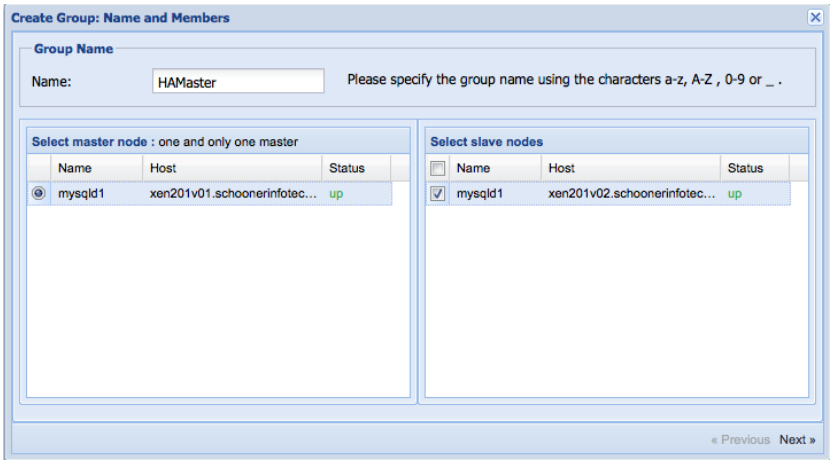


Figure 4-3: Group Name and Instance Configuration

- Select the master replication instance.
- If you are not configuring the group as an asynchronous replication slave, you may select one or more synchronous replication slave instances. Otherwise you may only add the master instance at this time.
- If you wish to make this instance the master any time it is active, click on the "Set master as permanent" checkbox. This ensures that a single instance will always be assigned the master role any time it is active.
- Click Next.

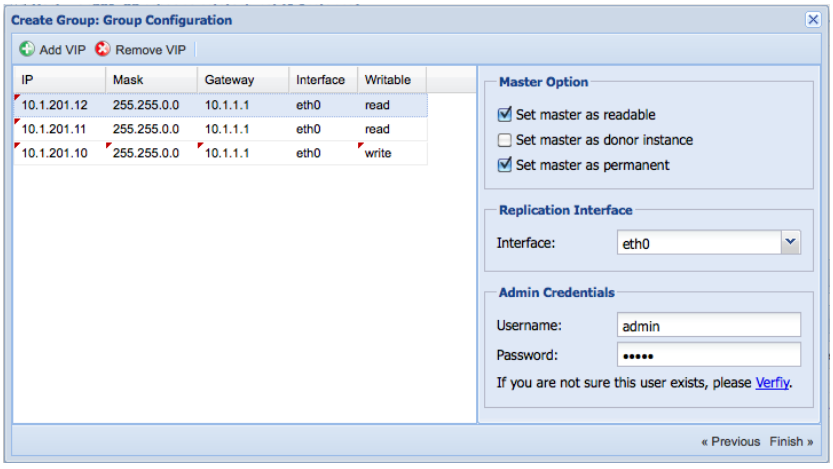


Figure 4-4: Group VIP, Interface, Login and Asynchronous Replication

- Add virtual IP addresses (VIP) for the group.
  - Generally, you need one write VIP for the master and one read VIP for the master and for each slave instance.
- Enter the network interface used for replication traffic.
  - Ideally, this will be a 10GE interface or N x 1GE bonded interface.
- Enter the database credentials to be used for replication access.
- If the replication group is to be used as a slave in a standard asynchronous replication cluster, click on Enable Async Replication.

- Configure asynchronous replication as you would for a standard MySQL slave instance, that is, enter the `CHANGE MASTER` parameters.

  NOTE: Each SchoonerSQL™ instance must have the `log-bin`,`log-slave-updates` and `server-id` parameters configured before the instance is added to the replication group. This is required in case the replication group must take over as master for the asynchronous replication cluster.
- NOTE: By default, the master instance will not be assigned a read VIP, it will only have the write VIP. If you wish to assign a read VIP to the master instance, click the "Set master as readable" checkbox. Be sure to create as many read VIPs as there are instances.
- Click Finish to create the replication group.
- The message panel will display the status of the group creation.
- On completion, the group panel will display each instance.

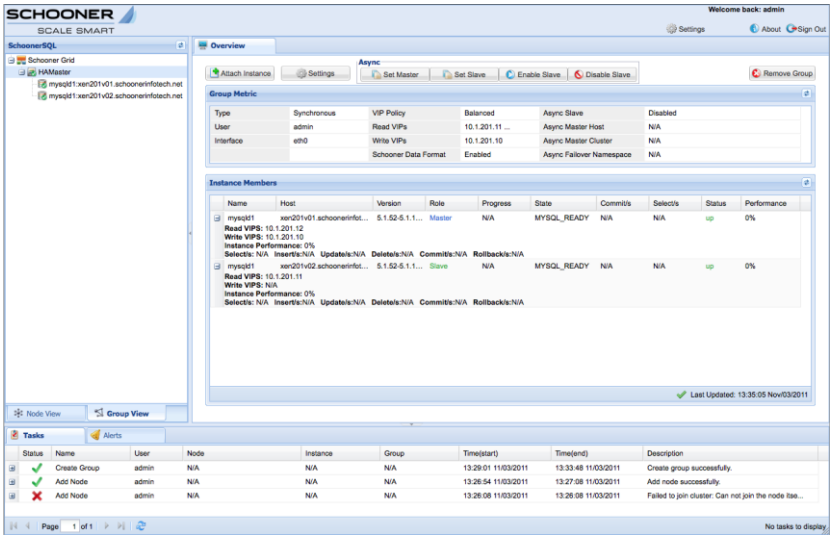


Figure 4-5: Replication Group Panel

Attach Instance to Synchronous Replication Group

To add an instance to a synchronous replication group:

- Click on Group View in the navigation panel.
- Select the replication group.
- Click on Attach Instance.

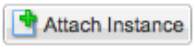


Figure 4-6: Attach Instance Button

- Select the instance to add and click Apply.

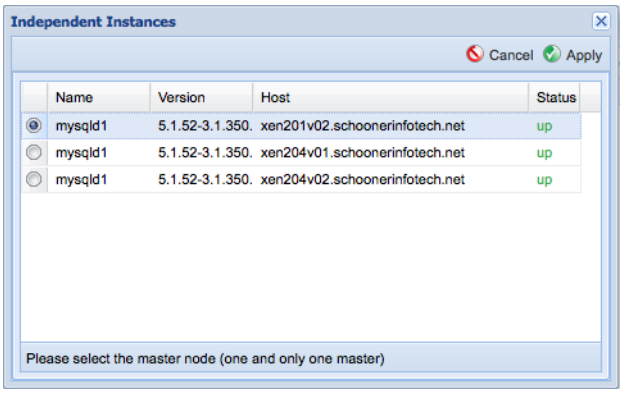


Figure 4-7: Attach Instance Dialog

- The message panel will display the status of the operation. Once complete, the instance will be displayed in the replication group panel.

HA Master + High-Speed Asynchronous Slaves

The most common method of replication using standard MySQL instances is asynchronous (binlog) replication. While MySQL asynchronous replication is widely used and is robust, the local nature of the binary log files makes failover and recovery difficult at times:

- Failover of a MySQL master may require stoppage of slaves and manual fix up of binary log configuration as well as clean up of transaction inconsistencies.
- Failover of slaves may require a proxy or externally supplied virtual IP (VIP) mechanism such as MMM.
- MySQL asynchronous replication is single-threaded. Large slave lags are not uncommon.

SchoonerSQL™ synchronous replication groups support the creation of high-availability asynchronous masters:

- Configuration of a synchronous replication group is as simple as creating a SchoonerSQL™ instance and adding it to a synchronous replication group. All initialization and data transfer is handled automatically by the *SchoonerSQL™ Administrator*.
- SchoonerSQL™ automatically handles failover of synchronous masters.
- SchoonerSQL™ automatically handles failover of SchoonerSQL™ asynchronous slaves to newly assigned SchoonerSQL™ asynchronous masters. No manual intervention is required to align slaves with a new master's binary log configuration.

Dividing your database between a SchoonerSQL™ HA master and multiple SchoonerSQL™ asynchronous slaves allows you to:

- Expand read capacity without sacrificing synchronous group bandwidth.
- Locate read instances in different networks.

- Make use of more moderately priced servers for slaves.
- Use replication filters to balance load among slaves.

The following diagram shows a 2-node SchoonerSQL™ HA master supporting 3 SchoonerSQL™ high-performance asynchronous slaves.

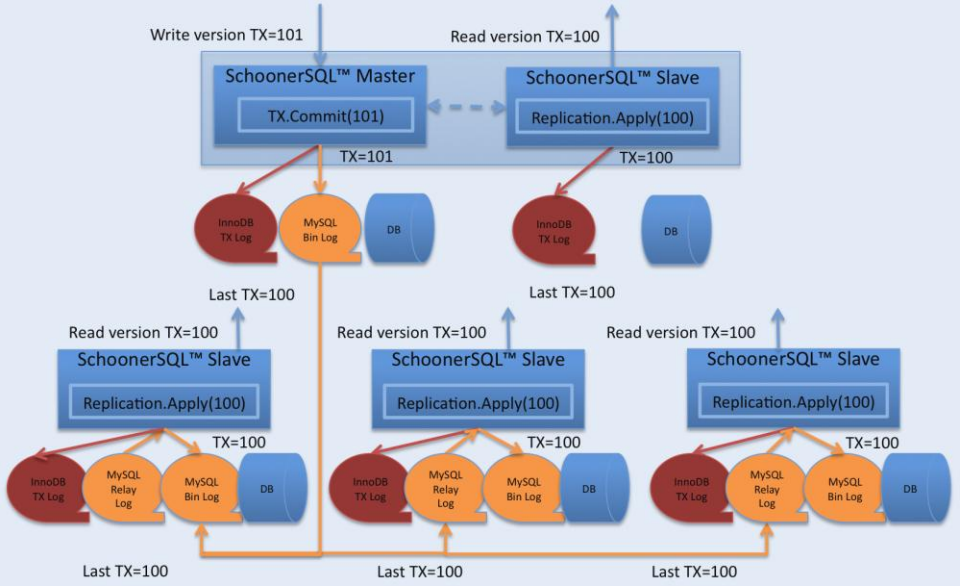


Figure 4-8: HA Master + Asynchronous Slaves

In this deployment, 2 SchoonerSQL™ nodes are configured in a synchronous replication group. The SchoonerSQL™ Master node acts as both the synchronous and asynchronous master.

In the event of a failure of the SchoonerSQL™ Master:

- The synchronous slave will automatically take over as the synchronous and asynchronous master.
- The asynchronous slave nodes will automatically be reconfigured to connect with the new asynchronous master and its associated binary log and position.

SchoonerSQL™ asynchronous slaves support parallel log applier threads which greatly increases the throughput of slaves and greatly reduces or eliminates slave lag.

The requirements for the SchoonerSQL™ HA master replication group are the same as for a simple, stand-alone replication group. The node-to-node communication paths must have enough capacity and redundancy to tolerate max traffic loads and network failures.

Configure Synchronous Replication Group as Asynchronous Replication Master

Synchronous replication groups may be configured as asynchronous replication masters. This involves setting up failover namespaces:

- Click the Set Master button.

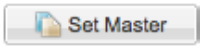


Figure 4-9: Set Master button

- The Failover Namespace Configuration dialog will display.

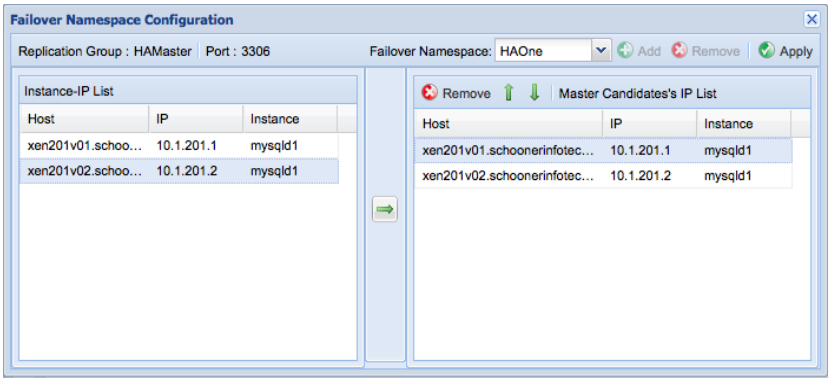


Figure 4-10: Failover Namespace Dialog

- Create a new failover namespace by clicking Add, typing its name in the Failover Namespace text field and hitting enter.
- Select the host IP addresses from the left panel and click the arrow to add them to the failover list.
- Click Apply to save the list.

Create an Asynchronous Replication Slave Instance

You may configure an independent SchoonerSQL™ instance as an asynchronous replication slave to either a SchoonerSQL™ replication group or a SchoonerSQL™ master instance:

- Click on Set Slave in the instance overview tab.

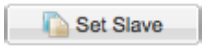


Figure 4-11: Set Slave Button

- Enter the administrator credentials for this instance.

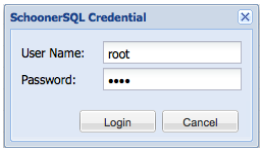


Figure 4-12: Instance Loging

- The Asynchronous Master Connection dialog displays:

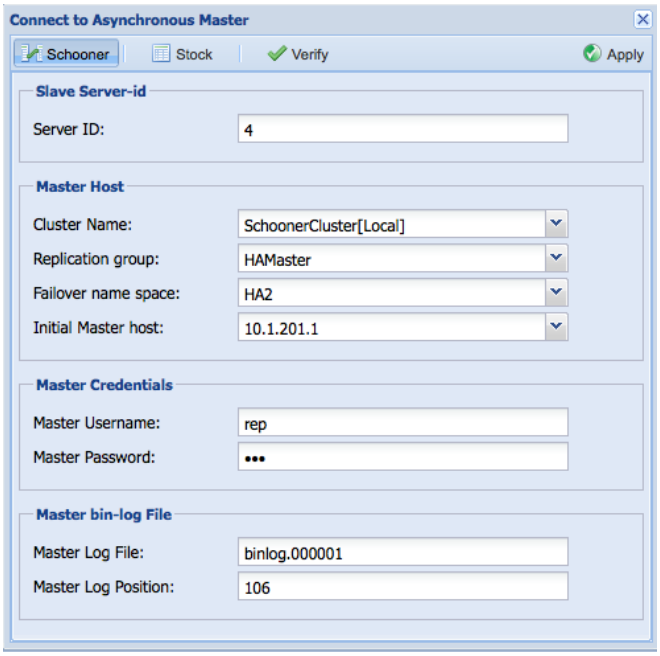


Figure 4-13: Asynchronous Master Connection Dialog

- Set the Server ID (must be unique within the replication cluster).
- Select the Cluster Name:
  - If the master replication group is located on the local network select SchoonerCluster[Local].
  - If the master replication group is located on a remote (WAN) network, select Change Cluster…the Remote Schooner Cluster dialog displays:

Figure 4-14: Cluster Login Dialog

  - Enter the WAN IP address of the remote SchoonerSQL™ cluster node.
  - Click Connect.
  - The Master Host fields will be filled in with Failover Namespace information from the remote cluster.
- Select the Replication Group to connect to.
- Select the Failover Namespace.
- Select the Initial Master host within the failover namespace.
- Enter the CHANGE MASTER parameters:
  - Replication user.

- Replication user password.
- Binary log file.
- Binary log file position.

- You may click the Verify button to check your configuration:

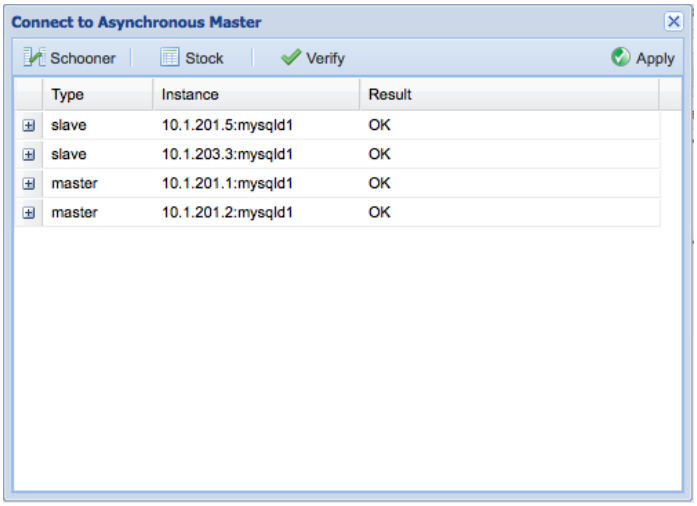


Figure 4-15: Asynchronous Replication Verification Dialog

- Click Apply to save the configuration and enable slave replication.

Consistent Read Scale Out

Certain applications may depend on consistent read data that cannot be supported by asynchronous replication methods due to slave lag. SchoonerSQL™ replication groups provide an effective method for consistent read scale out.

The figure below depicts an 8-node replication group.

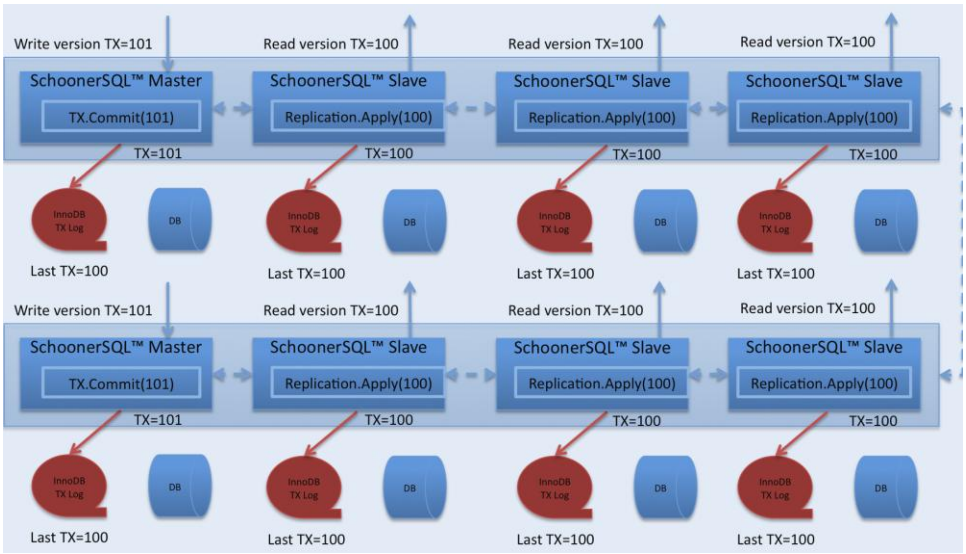


Figure 4-16: 8-Node Replication Group

In the figure, an 8-node replication group has been configured. The synchronous master accepts all write traffic and replicates updates to the slaves.

Metro Area Cluster

SchoonerSQL™ synchronous replication groups may be deployed across a metro area network (MAN) provided that the latency induced by the network connection between MAN data centers is sufficiently low to meet peformance requirements.

In the figure below, a replication group has been configured that spans two MAN data centers with a high-speed link connecting them.

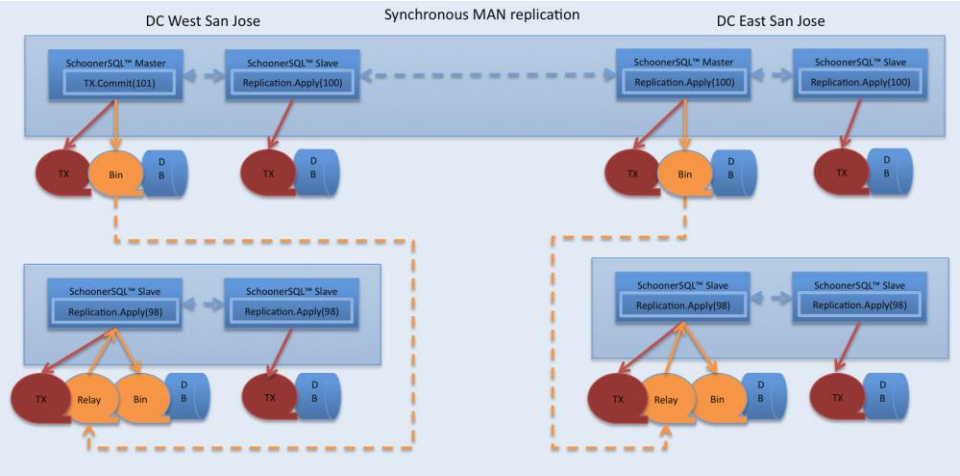


Figure 4-17: MAN Cluster

The 4 SchoonerSQL™ instances act as the asynchronous master to a set of asynchronous slaves within each data center. Deployment of the asynchronous slaves is dependent on the consistency requirements of your applications. Some applications require strict read consistency and would read only from the replication group instances. Other applications that require less strict read consistency would read from the asynchronous slaves.

Configuration of the synchronization group is the same as described in previous sections. You must ensure that the MAN link meets the requirements for bandwidth, latency and availability required by your applications. Additionally, care must be taken to ensure the constant delivery of SchoonerSQL™ heartbeat messages across the MAN link.

Wide Area Network Cluster

Full disaster recovery requires the deployment of databases across data centers connected via wide area networks (WAN).

The figure below depicts a pair of SchoonerSQL™ synchronous replication groups connected via WAN links. Within each data center, a set of asynchronous slaves is fed by the replication group. The deployment of asynchronous slaves is optional and is dependent on the needs of the applications.

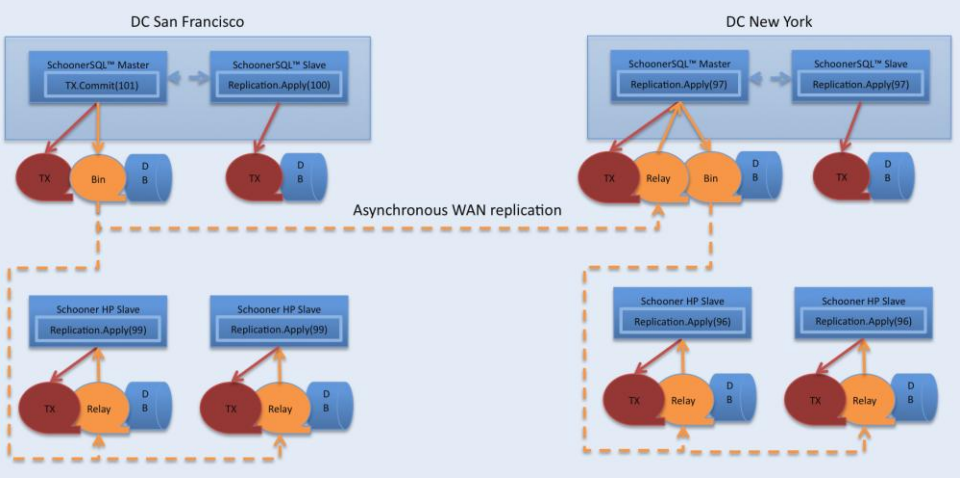


Figure 4-18: WAN Cluster

Because of the potentially long latency of WAN transmissions, synchronnous replication may not be feasible. The network latency could have a limiting effect on the total throughput of the replication group.

An asynchronous replication connection can be used effectively over the WAN since masters and slaves operate independently of one another.

The configuration of the above topology follows that of the previous sections with the exception of the asynchronous WAN connection between data centers.

Configure Synchronous Replication Group as Asynchronous Replication Slave

Synchronous replication groups may be configured as asynchronous replication slaves to synchronous replication groups acting as asynchronous replication masters. This involves identification of the synchronous replication group (either local or remote), selection of the failover namespace and configuration of CHANGE MASTER parameters:

- Click the Set Slave button.

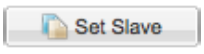


Figure 4-19: Set Slave Button

- Enter the credentials for the SchoonerSQL™ administrator.

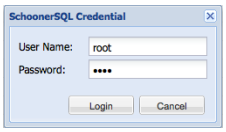


Figure 4-20: Instance Login Dialog

- The Asynchronous Master Connection dialog displays:

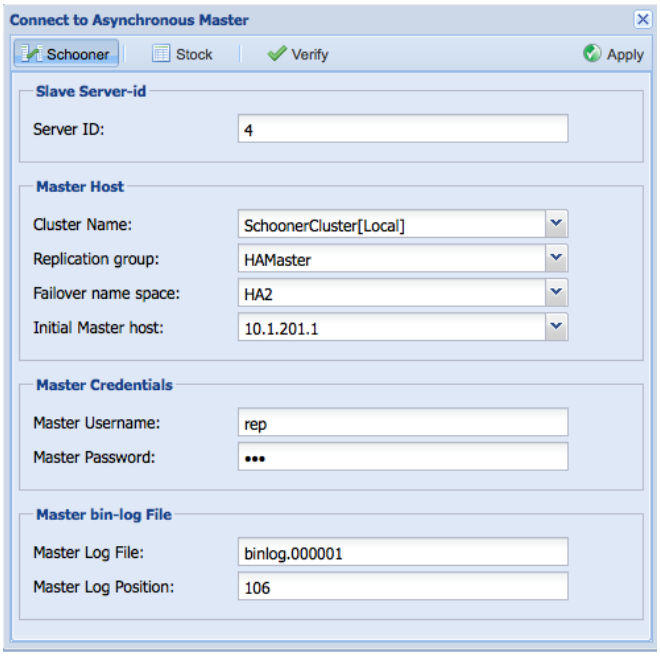


Figure 4-21: Asynchronous Master Connection Dialog

- Set the Server ID (must be unique within the replication cluster).
- Select the Cluster Name:
  - Change Cluster…the Remote Schooner Cluster dialog displays:

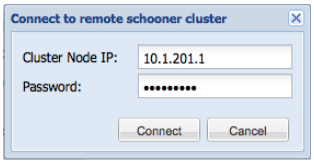


Figure 4-22: Cluster Login Dialog

  - Enter the WAN IP address of the remote SchoonerSQL™ cluster node.
  - Click Connect.
  - The Master Host fields will be filled in with Failover Namespace information from the remote cluster.
- Select the Replication Group to connect to.
- Select the Failover Namespace.
- Select the Initial Master host within the failover namespace.
- Enter the CHANGE MASTER parameters:
  - Replication user.
  - Replication user password.
  - Binary log file.
  - Binary log file position.

- You may click the Verify button to check your configuration:

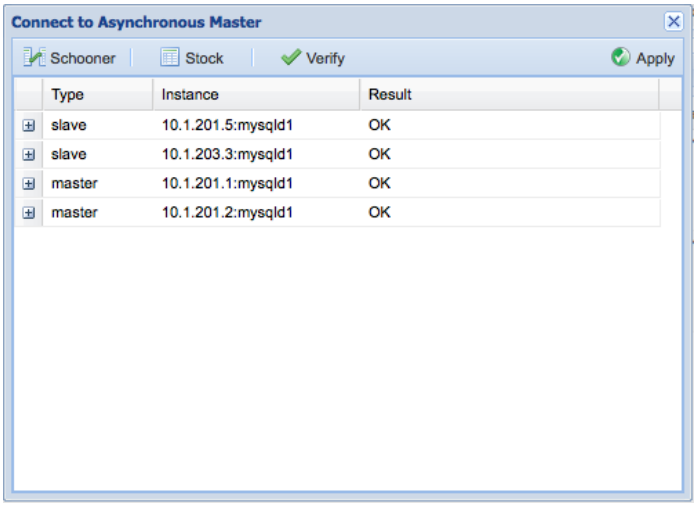


Figure 4-23: Asynchronous Replication Verification Dialog

- Click Apply to save the configuration and enable slave replication.

Appendix

Storage I/O Considerations

- Selecting the proper I/O Scheduler for your HDD and SSD devices can make a significant impact on your I/O throughput depending on the configuration. The default I/O Scheduler for CentOS and RedHat is "Completely Fair Queuing" or "cfq". The cfq I/O Scheduler attempts to distribute the available I/O bandwidth equally among all I/O requests.

  Schooner has observed that "cfq" can exhibit sub-optimal performance and we therefore recommend changing your data devices, backup devices, and transaction log devices to the "deadline" I/O Scheduler. The "deadline" I/O Scheduler uses a deadline algorithm to minimize I/O latency for a given I/O request. This leads to near real-time behavior while being fair to multiple I/O requests to avoid process starvation.

  To check the current I/O scheduler for a given device (example: /dev/sdb):

```
# cat /sys/block/sdb/queue/scheduler
noop anticipatory deadline [cfq]
```

  To change the I/O scheduler for a given device:

```
# echo deadline /sys/block/sdb/queue/scheduler
deadline /sys/block/sdb/queue/scheduler
```

Backup considerations

- If scheduling a recurring automated backup using the Schooner CLI or GUI, please be aware that the amount of space on the backup partition must be calculated using the following equation:

  (size of backup) x rotation count + (size of backup)

  For example, if the backup size is 200g and your rotation count is "3" then 800g of free space must exist on the backup partition. This is because the Schooner backup utility will only purge the oldest backup after the most recent backup has successfully completed.