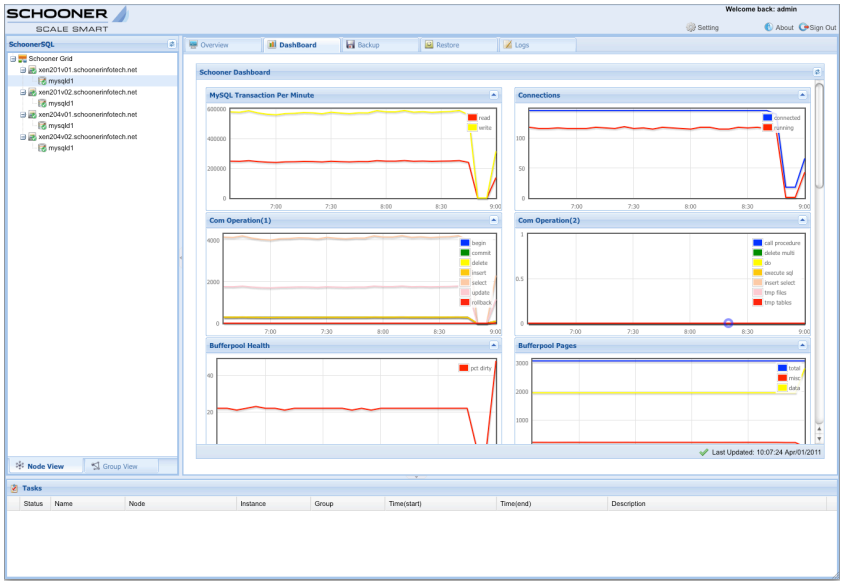
QuickStart Guide

SchoonerSQL™

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

Version 5.1



SCHOONER
SCALE SMART

Technical Support

Technical support for Schooner Informational 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-QG-01

SchoonerSQL™– QuickStart Guide

Issued November 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 QuickStart Guide provides a brief summary of the SchoonerSQL™, a full high-availability, high-performance build of MySQL and InnoDB. It describes its capabilities and features, along with pointers to the specific information needed to get SchoonerSQL™ up and running in a minimum amount of time.

Schooner Capabilities/Features

- Automated failover and recovery
- Advanced performance upgrades tuned for SSD
- Synchronous replication(Consistent data across cluster)
- Slave consistency
- Single click provisioning via simple GUI and CLI
- Instance migration
- Parallel appliers
- CLOCK and LRU replacement policies to match workloads
- Detailed monitoring of MySQL, server and cluster metrics
- Split brain handling
- Inter-operable with asynchronous replication
- Multiple instance support

Software Compatibility Overview

Compatibility

- Compatible with MySQL 5.1.52
- MySQL web site: **http://www.mysql.com/**
- InnoDB web site: **http://www.InnoDB.com/**

Incompatible Features

- Only InnoDB is supported by SchoonerSQL™
- Statement level and mixed mode replication may not be used with Schooner synchronous replication

Chapter 2: Getting Started

This section describes how to access the SchoonerSQL™ administration interfaces, basic set up tasks, backup features, verification tests and performance monitoring. Note that Appendix A describes the SchoonerSQL™ installation directories.

SchoonerSQL™ Database Directories

SchoonerSQL™ comes with a default `my.cnf` configuration file that includes settings tuned for best performance. Additionally, it defines the database directory paths as follows:

- `basedir = /opt/schooner/ac_5.1/mysql`
- `datadir = /schooner/data/db1`
- `tmpdir = /schooner/data/tmp/db1`
- `innodb_data_home_dir = /schooner/txlog/db1`
- `innodb_log_group_home_dir = /schooner/txlog/db1`

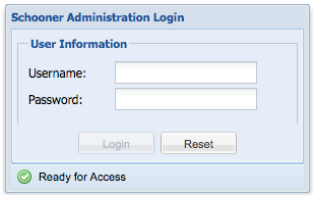
The `basedir` configuration must be as shown but you may modify any of the other paths. Please be sure that all directory paths are properly mounted before installing Schooner Active Cluster. If you are using flash storage devices, please be sure that the `datadir` and `tmpdir` paths map onto these devices.

Accessing the Schooner Administrator GUI

Before you can access the Schooner Administrator GUI, you must have completed the initialization procedure as described in the *Installation & Setup Guide*.

Once initialization has been completed, you may access the Schooner Administrator GUI at the following URL:

```
http://<server IP>/admin
```



The default login credentials are:

- Username: admin
- Password: admin

Start, Stop and Restart SchoonerSQL™ Service

The SchoonerSQL™ service consists of the SchoonerSQL™ database and a set of daemons that support administration and replication features. The service is

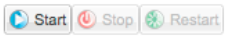
automatically started for you when you initialize SchoonerSQL™ using the CLI. It is also started upon reboot of your server.

- When the SchoonerSQL™ service is running, its node and instance icons will be marked in green:

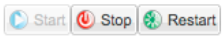
- When the SchoonerSQL™ service is running, its node and instance icons will be marked in red:

The GUI provides an interface to control the execution of SchoonerSQL™ instances:

- When SchoonerSQL™ is stopped, the start button is enabled:



- When SchoonerSQL™ is running, the stop and restart buttons are enabled:

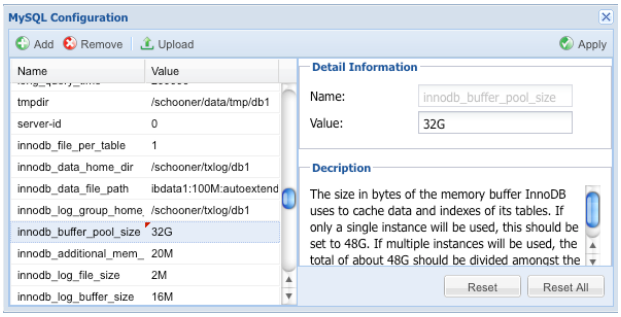


The CLI may also be used to control the execution of SchoonerSQL™. Please see the *Schooner Appliance for MySQL® Enterprise™ with InnoDB Application & Administration Guide* for instructions.

Configuring SchoonerSQL™

To modify the SchoonerSQL™ instance configuration (maintained in `/etc/my.cnf`), use the Config button to edit, add or remove attributes.

- Select attributes and edit their values. Apply to save all changes.



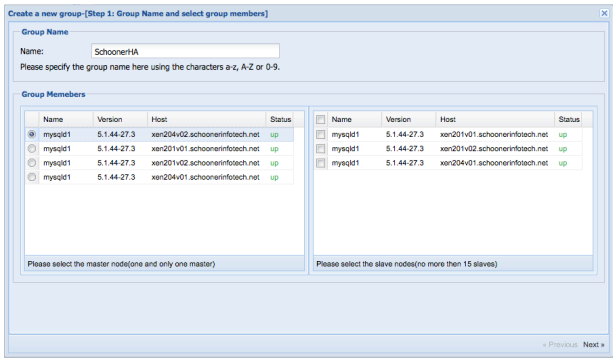
You must restart SchoonerSQL™ for the changes to take effect.

Schooner High-Availability

SchoonerSQL™ offers state-of-the-art high-availability using replication groups. A replication group consists of a set of SchoonerSQL™ instances that support synchronous replication, VIP fail-over and automatic recovery. The following explains how to configure a SchoonerSQL™ synchronous replication group.

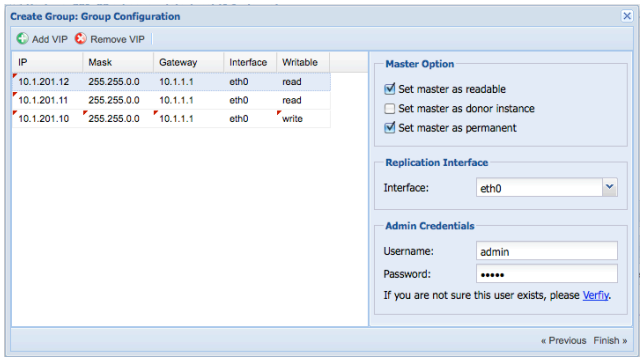
- To configure the SchoonerSQL™ replication group, click the add group button:

- Select the name of the group and select one instance to be the replication master:

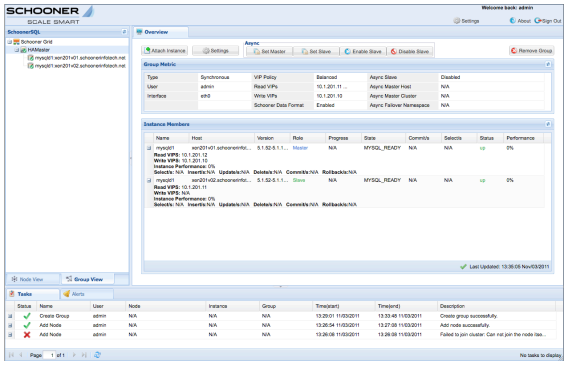


Also select those instances that are to be slaves to the replication master instance.

- Click next.
- Add the virtual IP addresses (VIP) to be used for read and write access to this replication group. SchoonerSQL™ automatically balances the VIPs across the executing instances and handles VIP fail-over as well.



- Set master options:
    - Set master as readable
        - This enables reads from the master instance.
    - Set master as donor instance
        - Use only the master as the recovery donor instance.
    - Set master as permanent
        - Use the initial master instance as the group master whenever it is alive.
- Enter the network interface to be used for replication traffic. This should optimally be a 10GE interface or a bonded set of 1GE interfaces.
- Enter the user and password to be used for replication group management. The default user is admin and password Is admin. Use the Verify button to ensure that the credentials are correct.
- Click Finish.
- The replication group will be created and the instances will begin synchronous replication.



Enabling SchoonerSQL™ Optimizations

Although SchoonerSQL™ is built and tuned to provide high out-of-the-box performance on recommended platform configurations, each database workload is different and may benefit from analysis and tuning. Tuning variables can be specified in the `my.cnf` file. Restart of mysqld is required if updated variables are static.

The following recommended optimizations have a global scope, are static variables and do not affect the database format:

```
innodb_log_file_size=4G                    # with 2 or more log files. 8G may also
                                             benefit
innodb_flush_method=O_DIRECT               # under heavy load, Linux virtual memory
                                             system may behave unpredictably with the
                                             file-system cache. The default is value
                                             is O_DIRECT.
innodb_page_replacement_algorithm=clock # Schooner buffer pool implementation for
                                             higher concurrency and scan resistance.
                                             The default value is clock
```

The following recommended optimizations have a global scope, are static variables and affect the database format (i.e. may not be compatible with stock MySQL at a file level, however `mysqldump` shall always work for dump/restore). Note that SchoonerSQL™ works with InnoDB files containing a mix of fast and normal (also referred as "old") checksums for convenience. However it is recommended that `ibd_sum` be used to convert checksums completely before using the database for performance testing or in production:

```
innodb_file_per_table=ON   # (default=ON) separate table/index data from
                              tablespace and store on separate storage volume

innodb_checksum_type=fast # (default=normal) reduce overhead for IO
innodb_extra_rsegments=16 # (default=0) add roll back segments to support higher
                              thread concurrency
```

When creating new transaction log files (`ib_logfilexx`) for an existing database, or loading a new database, the following recommended setting can be incorporated. To recreate transaction files with different settings (e.g. with different size, different number of files, with different block size), set variable `innodb_fast_shutdown=0` by a MySQL command such as "`set global innodb_fast_shutdown=0;`" before shutting down InnoDB. A safe shutdown will ensure that every committed modification in the transaction log file is reflected into the database files. Once a safe shutdown completes, the existing transaction log files can be archived and moved, and the database can be started with new log settings. The recommended setting is static

and effects the format of the transaction log file and is not compatible with stock MySQL at a file level:

```
innodb_file_per_table=ON   # (default=OFF) separate table/index data from
                             tablespace and store on
innodb_log_block_size=4096 # (default=512) larger block size for log IO
```

The SchoonerSQL™ default configuration includes specific tunings for high-performance platforms (that utilize flash memory for storage) and update-intensive workloads. Please see the *Schooner Appliance for MySQL® Enterprise™ with InnoDB Application & Administration Guide* for more information.

Using `ibd_sum` to Verify and Optimize Checksums

The `ibd_sum` tool has two main functions:

- It verifies and reports statistics on the checksums for each block of an InnoDB file. This can be used to determine consistency of the data by verifying the checksums. Note that correct checksums do not guarantee that data is not corrupted. However, they do minimize the probability of undetectable corruption after the checksum are updated on a page write. The tool works both on InnoDB's **file (.**`ibd`) and transaction log files (`ib_logfile`**)** in an off-line mode.

```
# cd /opt/schooner/ac_5.1/scripts/mysql-local/tools/
#
# ibd_sum Utility for InnoDB(tm) tablespace
  checksums. Ver: 5.1 Rev: . (c) 2010 Schooner Information Technology, Inc.  Usage:
  ibd_sum file_name [old/fast/xtradb/none [num-threads [skip-check|skip-convert]]]
#
# ibd_sum /schooner/data/db0/foo/bar.ibd old 1
  skip-convert  Space: /schooner/data/db0/foo/bar.ibd  Type: data  Page size: 16384
  File size: 98304 (6 pages)  Buffer size is 16384 pages  Threads count: 1
  Consistency check...success  Current format: old; Details: old=4;empty=2;  Max
  flushed LSN: 0  Min page LSN: 45004  Max page LSN: 48140
#
# ibd_sum /schooner/txlog/db0/ibdata1 old 1 skip-
  convert Space: /schooner/txlog/db0/ibdata1 Type: data Page size: 16384 File size:
  10485760 (640 pages) Buffer size is 16384 pages Threads count: 1 Consistency
  check...success Current format: old; Details: old=179;empty=461; Max flushed LSN:
  0 Min page LSN: 10675 Max page LSN: 48784 Checksums in required format already
#
# ibd_sum /schooner/txlog/db0/ib_logfile0 old 1
  skip-convert  Space: /schooner/txlog/db0/ib_logfile0  Type: log  Page size: 512
  File size: 134215680 (262140 pages)  Buffer size is 16384 pages  Threads count: 1
  Consistency check...success  Current format: old; Details: old=79;empty=262061;
  Log checkpoint LSN: 48146  Checksums in required format already
#
# ibd_sum /schooner/txlog/db0/ib_logfile0 old 1
  skip-convert  Space: /schooner/txlog/db0/ib_logfile0  Type: log  Page size: 512
  File size: 134215680 (262140 pages)  Buffer size is 16384 pages  Threads count: 1
  Consistency check...success  Current format: old; Details: old=79;empty=262061;
  Log checkpoint LSN: 48146  Checksums in required format already
```

- It converts checksums to a chosen format. Checksum calculations under heavy IO conditions are CPU-intensive and converting old checksums (current stock InnoDB implementation) to fast checksums (additional Schooner implementation choice) has potential to improve response time and CPU usage for a workload.

```
# ibd_sum /schooner/data/db0/foo/bar.ibd fast 1
  Space: /schooner/data/db0/foo/bar.ibd
  Type: data Page size: 16384  File size: 98304 (6 pages)
  Buffer size is 16384 pages Threads count: 1  Consistency check...success
  Current format: old; Details: old=4;empty=2;
  Max flushed LSN: 0  Min page LSN: 45004  Max page LSN: 48140
  Target format: fast  Converting…success
#
# ibd_sum /schooner/txlog/db0/ib_logfile0 fast 1
```

```
Space: /schooner/txlog/db0/ib_logfile0
Type: log  Page size: 512  File size: 134215680 (262140 pages)
Buffer size is 16384 pages  Threads count: 1  Consistency check...success
Current format: old; Details: old=79;empty=262061;  Log checkpoint LSN: 48146
Target format: fast
Converting...success
```

Cache Replacement Policies

Here is a brief overview of the two supported cache replacement policies.

Least Recently Used (LRU) – This policy tracks page usage over a relatively short time period, and discards the Least Recently Used page in the cache. The age of each cache line is tracked. A "hit" to a cache page moves that page to the Most Recently Used (MRU) position, with a subsequent change in the status of all of the other cache lines. A lock protects the MRU position. This lock can be a source of contention, because all cache hits are serialized behind it. Schooner has optimized the LRU policy, and made multi-threaded access more concurrent, by using fine-granularity locking.

Clock with Adaptive Replacement (CLOCK) – This policy combines the features of the "CLOCK" buffer pool replacement algorithm with those of ARC (Adaptive Replacement Cache), to create a cache replacement policy that considers frequency, as well as age, when determining which page to discard from the cache.

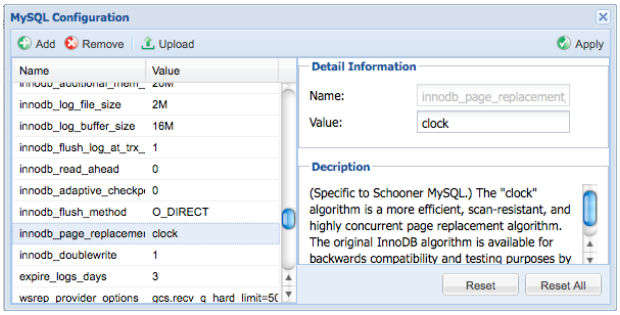
CLOCK maintains two pairs of lists: each pair consists of a list of the pages in the cache, and a corresponding "history" list, containing the pages recently discarded from the cache. The first pair of lists represents the most *recently* used pages. The second pair represents those pages most *frequently* used.

Queries that incorporate scans can severely effect concurrently executing transactions, by bringing "cold blocks" into the buffer pool. LRU is particularly susceptible to scans, because of the serial nature of its MRU lock. CLOCK, with no MRU and its ability to track frequency, provides improved scan resistance and better concurrent-transaction performance.

Upon selecting the cache replacement policy, SchoonerSQL™ must be restarted in order for the changes to take effect.

To select a cache replacement policy:

- In the GUI, select the SchoonerSQL™ instance to be modified.
- Click on the Config button to bring up the configuration editor:



The default setting is "clock". To change back to LRU, enter "lru" in the Value field, save and then apply the change.

You must then restart the SchoonerSQL™ instance by clicking the Restart button.

Integration with MySQL Asynchronous Replication

SchoonerSQL™ supports the standard MySQL asynchronous replication as well as Schooner synchronous replication. Asynchronous replication can be used to:

- Migrate an existing database cluster to a SchoonerSQL™ replication group (see next chapter).
- Provide replication from a SchoonerSQL™ replication group to a standard MySQL slave cluster.
- Provide replication service where synchronous replication is not required.

SchoonerSQL™ asynchronous replication greatly enhances stock MySQL replication by using synchronous replication groups as asynchronous masters and/or slaves:

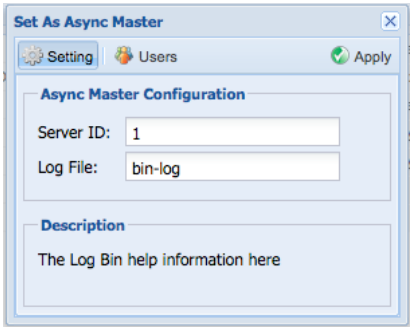
- When deploying SchoonerSQL™ synchronous replication groups as asynchronous masters, automated master failover is supported. This greatly improves the availability of asynchronous replication clusters as master downtime is eliminated. It also relieves the problem of migrating slaves to a newly selected master instance.

Creating an Asynchronous Master in SchoonerSQL™

The configuration of an asynchronous master using a SchoonerSQL™ instance is the same as in standard MySQL. The master enables binary logging and uses a non-zero server-id. Any SchoonerSQL™ or standard MySQL asynchronous slave may connect to this master.

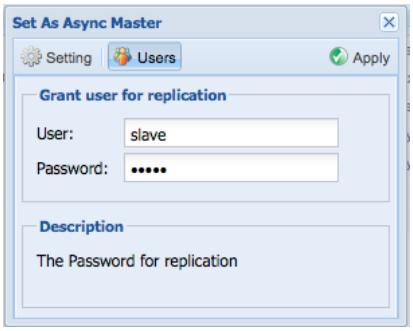
The Schooner Administrator simplifies configuration by providing an easy to use GUI:

- Select the master SchoonerSQL™ instance and click the “Set Master” button.



  Enter the server-id and binary log file.

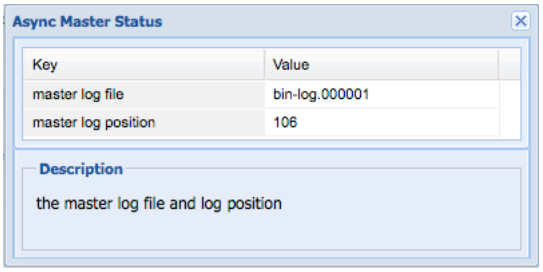
- Click the “Users” button to configure the replication user credentials.



Enter the replication user credentials and click “Apply”.

- The SchoonerSQL™ Instance Metric window will indicate that the instance is an asynchronous replication master:

- To see the name and position of the binary log, click the “Configured” hyperlink.



Configuring a Schooner Synchronous Replication Group as Asynchronous Master

A SchoonerSQL™ synchronous replication group may participate as a single entity in a MySQL asynchronous replication cluster. That is, the entire group can act as an asynchronous replication master and support automated master failover.

Each SchoonerSQL™ instance is created with binary logging (binlog) enabled in case it might need to take over asynchronous master duties. Each instance must also be assigned a unique server-id. Set these attributes for each instance using the Config dialog described in the “Configuring SchoonerSQL™” section above.

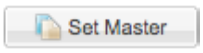
In order to support automated master failover, one or more failover namespaces are assigned to the group. A failover namespace specifies the SchoonerSQL™ instances within the group that are to be used as masters in case of master failure. For example:

- Assume that a synchronous replication group has 6 SchoonerSQL™ instances.
- A default namespace is automatically created that defines the initial master instance as the primary within the namespace and each of the other group instances as failover candidates.
- If the master instance should fail, the next instance in the namespace will be selected as the asynchronous master and all asynchronous slaves will be reconfigured to use this instance and its binary log.

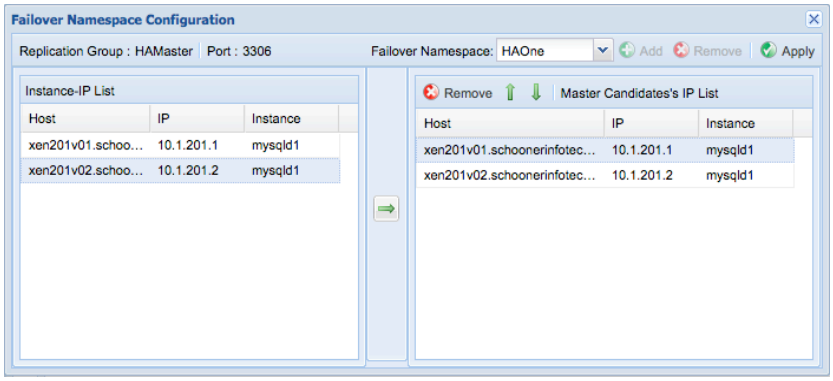
You may create other namespaces should you wish to limit a slave or set of slaves to connect with certain master instances. Remember that since each master instance is a member of a synchronous replication group, any master instance may be used as an asynchronous replication master.

To create a failover namespace:

- Click the synchronous replication group to be configured.
- Click the Set Master button.



- The Failover Namespace Configuration dialog will display.



- 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.
- When configuring asynchronous slaves, select the failover namespace.

Creating an Asynchronous Slave in SchoonerSQL™

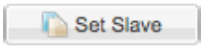
In SchoonerSQL™, asynchronous replication slaves may be configured as individual instances (as in stock MySQL) or as synchronous replication groups. The benefit of using a synchronous replication group as an asynchronous slave is that the synchronous replication group automatically handles failure of any slave.

The configuration of an asynchronous slave using SchoonerSQL™ is basically the same as in standard MySQL. The slave uses the `CHANGE MASTER` command to link to the master. The slave may connect to any SchoonerSQL™ or standard MySQL asynchronous replication master.

However, if a SchoonerSQL™ synchronous replication group is used as the asynchronous master, then you must also assign a failover namespace in addition to the `CHANGE MASTER` settings.

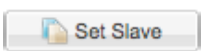
If the slave is a single SchoonerSQL™ instance:

- Select the instance in the Navigation Panel.
- Click the Set Slave button.



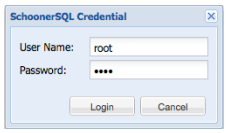
If the slave is a SchoonerSQL™ synchronous replication group:

- Select the group in the Navigation Panel.
- Click the Set Slave button.

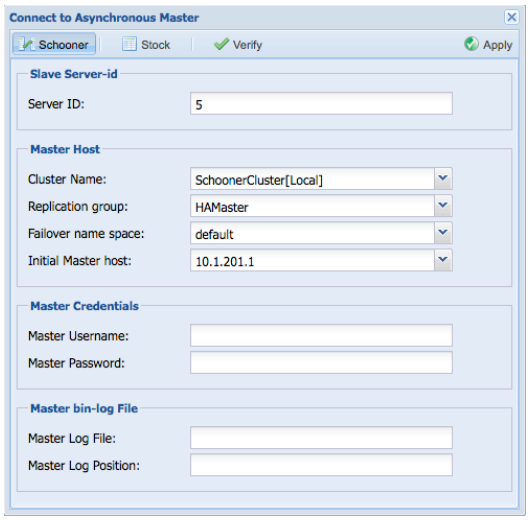


In both cases, the SchoonerSQL™ login dialog displays:

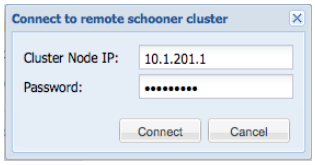
- Enter the credentials for the SchoonerSQL™ administrator.



- The Asynchronous Master Connection dialog displays:



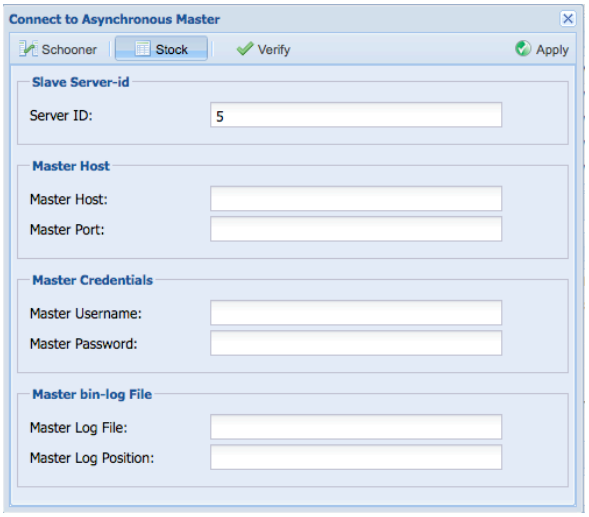
- If the master is a synchronous replication group:
  - Set the Server ID (must be unique within the replication cluster).
  - Select the Cluster Name:
    - If the master replication group is located on the local network select SchoonerCluster[Local].
    - If the master replication group is located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



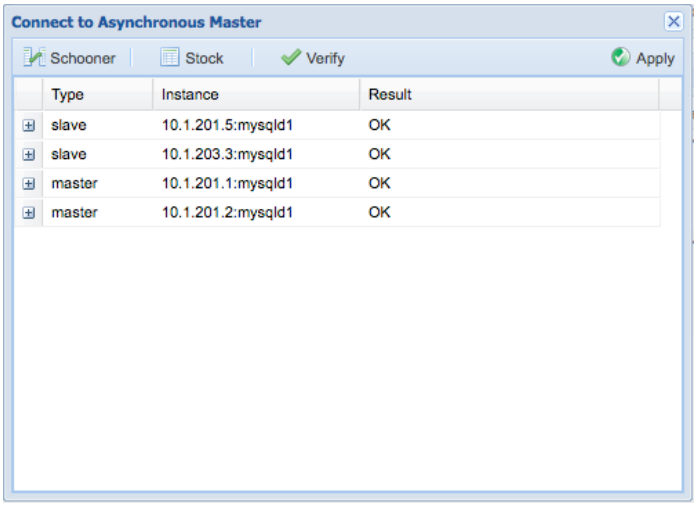
    - Enter the WAN IP address of the remote SchoonerSQL™ cluster node.
    - Click Connect.
    - The Master Host fields will be filled in with Failover Namespace information from the remote cluster.
  - Select the Replication Group to connect to.
  - Select the Failover Namespace.
  - Select the Initial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

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

- If the master is an independent SchoonerSQL™ or stock MySQL instance:
  - Click the Stock button.
  - The Stock Async Master Connect dialog displays.



  - Enter the Server ID and CHANGE MASTER parameters.
- You may click the Verify button to check your configuration:



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

Chapter 3: Migrating Asynchronous to Synchronous Replication

This chapter describes the different methods that may be used when converting from standard MySQL asynchronous replication to SchoonerSQL™ synchronous replication.

Common MySQL Replication Architectures

Basic Master-Slave Replication

In this configuration, at least two machines are configured, with one machine as a MySQL asynchronous replication master, and one or more slaves are connected to it and replicating.

Basic Master-Master Replication

Although MySQL asynchronous replication does not actually support true master-master replication, it is possible to use a special configuration of master-slave as a master-master system. This is a modification of the basic master-slave model by configuring the master to also be a slave of one of its slaves. This can be used either with single or dual write nodes.

Master-Master, Single Write Node

In this configuration, although both machines could be written to, writes are directed to only one node at a time, using client configuration, DNS, IP takeover, or an external load balancing device. Often the non-writable node is set explicitly read-only using the global read-only setting within MySQL.

Master-Master, Dual Write Modes

In this configuration, writes are allowed to both masters at the same time, and they may be load balanced using round-robin DNS or an external load balancing device. Special configuration is required if auto-incrementing columns are used, in order to avoid key conflicts.

MMM and Flipper

MMM and Flipper are two master-master replication management systems which implement “master-master, single write node” auto-configuration. While the two systems have many differences, a key similarity is that they both use IP takeover to implement the role migrations required.

The SchoonerSQL™ Architecture

The SchoonerSQL™ architecture is a synchronous replication model based on pre-commit-stage “certification” of each transaction processed on the master with any connected slaves. This process ensures that transactions aren’t lost in a master failure, as the slaves have already received the committed transaction.

Migration Strategy

Overview

The overall migration strategy is a stepwise move from the current MySQL asynchronous replication configuration into the new SchoonerSQL™ configuration. It is divided into three main phases. This allows testing to be done using the actual data and application before moving the system into production.

Phase One: Introduce Single SchoonerSQL™ Instance as a Slave

In this phase, a single SchoonerSQL™ instance is introduced as a slave (using MySQL asynchronous replication) to the current master.

Phase Two: Introduce Paired SchoonerSQL™ Instances

In this phase, a second SchoonerSQL™ instance is added as a slave (using SchoonerSQL™ replication) to the existing SchoonerSQL™ instance.

Testing the Application

It may be desirable to complete phases one and two, and then detach the now-functional and up-to-date SchoonerSQL™ replication group from the existing MySQL asynchronous replication system in order to do application testing (including writes) against SchoonerSQL™ alone. Writes should never be issued to the SchoonerSQL™ replication group while it is acting as a slave of MySQL asynchronous replication, as that may cause replication to the cluster to be broken irreparably, or may cause other undefined behavior.

Phase Three: Migrate Master role to SchoonerSQL™

In this phase, the primary system traffic is migrated to the now functional SchoonerSQL™ replication group.

Post-Migration

After migrating the master role to SchoonerSQL™, it is possible to keep some slaves using MySQL asynchronous replication. These slaves will not automatically be moved in the case of a SchoonerSQL™ failover, so they should be manually administrated. They should also be monitored to ensure that they keep up with SchoonerSQL™ and do not fall unreasonably behind in replication.

Phase One: Introduce Single SchoonerSQL™ Instance as a Slave

Data Copy from Current System

If the existing system has a simple method for setting up new slaves, that method can normally be used to set up the new SchoonerSQL™ instance. Typical methods available are: xtrabackup[1], InnoDB Hot Backup[2], LVM snapshots[3], and `mysqldump` (with the right set of options), or shutting down a slave to make a copy.

[1] https://launchpad.net/percona-xtrabackup/
[2] http://www.innodb.com/wp/products/hot-backup/

Recommended: Conversion to Schooner Optimized Format

It is recommended that the database be converted to Schooner's optimized format after copying to the new SchoonerSQL™ instance.

Set Up Replication

Replication should be set up using the `CHANGE MASTER` command as usual. The SchoonerSQL™ will additionally keep track of the replication position information within the database in order to replicate it to

Monitor Catch Up and Replication Performance

Once replication has been started, it should be monitored using SHOW SLAVE STATUS until it has completely caught up. It can then be monitored for performance and replication delays using the same command.

Phase Two: Introduce Paired SchoonerSQL™ Instances

Attach to Existing SchoonerSQL™ Instance

Following the *Application & Administration Guide,* attach the second SchoonerSQL™ instance to the SchoonerSQL™ replication group. This process will automatically copy data (preserving the optimized format), restore it on the new node, set up SchoonerSQL™ replication, and redistribute any configured read virtual IPs using IP takeover. Once the process is complete, the new SchoonerSQL™ instance is caught up and ready to take read traffic from clients.

Test Failover Between SchoonerSQL™ Instances

Once it has been configured it is possible to test the failover and recovery mechanisms provided by SchoonerSQL™.

Phase Three: Migrate master role to SchoonerSQL™

Migrate Master Role

A few considerations

Without potentially significant additional work, it is not possible (or at least advisable) to direct writing clients to both the old MySQL asynchronous replication-based system and the new SchoonerSQL™-based system at the same time. In order to make a clean transition with minimal risk of data loss, all write activity should be completely stopped on the old system before the transition is made to the new system. This can be achieved in a number of ways (with examples listed above) but the key goal is that there is no overlap at all.

After all necessary testing has been completed; the master role should be migrated to the SchoonerSQL™ virtual IPs. This can be done in one of a few ways, grouped into a few major groups, explained in detail in the following paragraphs.

[3] http://www.mysqlperformanceblog.com/2006/08/21/using-lvm-for-mysql-backup-and-replication-setup/

Move IP addresses directly

Migration of IP addresses directly can be risky due to not being able to have IP addresses configured in two places at the same time, requiring active reconfiguration of the clusters during the maintenance window.

- Migrate original IP addresses or virtual IPs to SchoonerSQL™. Migrating these IP addresses will require reconfiguration of the SchoonerSQL™ network configuration during the migration process.

Change client configurations directly or indirectly

Changing client configurations will require coordination of all clients to either wait for complete software deployment or complete cache refresh, depending on the solution chosen.

- Change client configurations to connect to new IP addresses. Changing client configuration may require significant downtime in large-scale environments, unless significant tools are in place already to make configuration deployments simple and fast.
- Change DNS names to cause clients to connect to new IP addresses. If DNS is used for all accesses to the master, this strategy may be relatively simple to achieve.

Use an intermediary

All connections are transitioned first from direct connections to the master to connecting through an intermediary. Once all connections are using the intermediary, it can be used to redirect the traffic simply.

- Use an external load balancer device to move clients to new IP addresses. If a load balancer is currently used (or could be easily implemented) it can provide a very clean and simple way to move all traffic simultaneously.
- Use an intermediate proxy between clients and the servers. Either a basic TCP proxy or a more complex solution such as MySQL Proxy4 could be used.

Optional: Replicate from SchoonerSQL™ Back to Old MySQL Server

It may be desired to continue to use MySQL asynchronous replication to replicate the SchoonerSQL™ traffic back to a standard MySQL server. This will allow the hardware to continue to be used and may allow more flexible handling of any unforeseen failure situations with SchoonerSQL™.

[4] http://forge.mysql.com/wiki/MySQL_Proxy

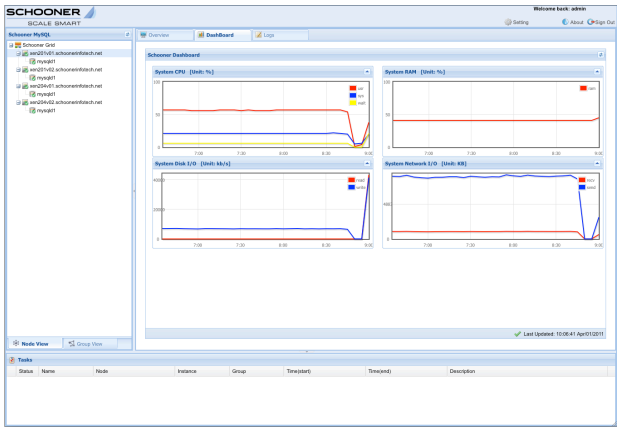
Chapter 4: Performance Monitoring

SchoonerSQL™ provides performance charts in the GUI dashboard. The charts are based on system and MySQL metrics collected by the "emt" monitoring system. This monitor samples metrics for cpu, memory, IO and MySQL at 5 minute intervals. Data is displayed for the past 2 hours.

The Adobe® Flash® Player is required to view SchoonerSQL™ charts.

The dashboard is context sensitive; it displays the metrics for either the host or the SchoonerSQL™ instance depending on what you are viewing at the moment.

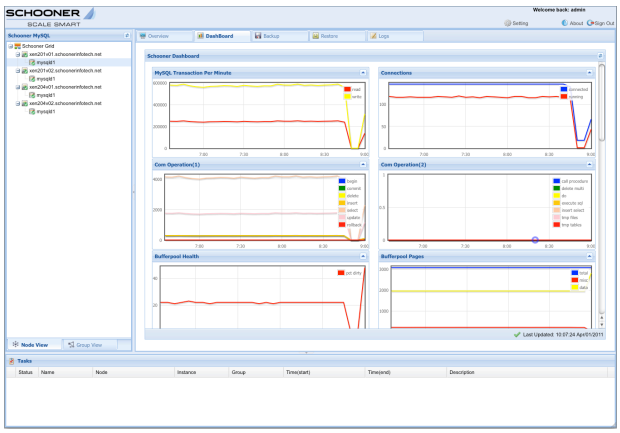
- To view the system dashboard, click on a node in the navigation panel and then click on the Dashboard tab:



The system dashboard displays the following metrics:

- CPU utilization across all cores.
- DRAM utilization.
- Storage device IO in kb/s.
- Network bandwidth in kb.

To view the MySQL dashboard, select a SchoonerSQL™ instance and then click on the Dashboard tab:



The MySQL dashboard displays the following metrics:

- MySQL read/write transactions per minute.
- Number of connections.
- MySQL statement execution:
    - begin
    - commit
    - delete
    - insert
    - select
    - update
    - rollback
    - call procedure
    - delete multi
    - do
    - execute sql
    - insert select
    - tmp file
    - tmp tables
- Buffer pool % dirty pages.
- Number of buffer pool pages.
- InnoDB data read/written.
- InnoDB pending reads/writes.
- InnoDB log writes.
- InnoDB log pending fsyncs and writes.
- Clock average flush loops.
- Modified age.
- InnoDB row operations.
- InnoDB page operations.
- Replicated Received
- Replicated Received Bytes
- Flow Control Paused
- Flow Control Sent/Received

Appendix A: Installation Directories

SchoonerSQL™ Installation Paths

SchoonerSQL™ is self-contained in terms of its installation directory paths, i.e., it does not conflict with existing stock MySQL installations.

Main Directory Tree

The main directory tree for SchoonerSQL™ starts at `/opt/schooner`:

```
drwxr-xr-x 6 root root  4096 Mar 15 16:41 ac_5.1
drwxr-xr-x 3 root root  4096 Mar 15 16:41 admin
drwxr-xr-x 2 root root  4096 Mar 15 16:41 bin
drwxr-xr-x 2 root root  4096 Mar 15 16:41 config
-rw-r--r-- 1 root root  4395 Mar 15 16:41 hwprofile
drwxr-xr-x 2 root root  4096 Mar 15 16:41 license
drwxr-xr-x 2 root root  4096 Jun 15 13:59 maps
```

- ac_5.1
  - SchoonerSQL™ binaries.
- admin
  - Schooner administration interface configuration files.
- bin
  - Various utilities for verification tests.
- license
  - Schooner and 3rd-party open source licenses.
- maps
  - Contains version information for rollback.

The "`hwprofile`" file contains the results of the Schooner hardware configuration check.

Configuration File Paths

SchoonerSQL™ looks for MySQL configuration settings only in the `/etc/my.cnf` file.

Monitor Directory Paths

SchoonerSQL™ installs the "emt" monitoring package and supports a set of performance monitoring charts based on this data. The directory paths for monitoring starts at `/opt/emt`:

```
drwxr-xr-x 2 root   root   4096 Jan 24 19:40 base
drwxr-xr-x 2 root   root   4096 Jan 24 19:40 bin
drwxr-xr-x 8 apache apache 4096 Jan 24 19:40 charts
drwxr-xr-x 9 root   root   4096 Jan 24 19:40 plugins
drwxr-xr-x 2 root   root   4096 Jan 24 19:40 sample
```

- base

    - emt PHP top-level classes.
- bin
    - emt utilities.
- charts
    - SchoonerSQL™ charting scripts.
- plugins
    - emt data collection plugins.
- sample
    - emt sample configuration file.