



Managing MySQL Scale Through Consolidation

Percona Live 04/15/15

Chris Merz, @merzdba
DB Systems Architect, **SolidFire**

Enterprise Scale MySQL Challenges

- Many MySQL instances (10s-100s-1000s)
- Often 100s of GB or multi-TB range
- Capacity planning and resource management
- Quickly respond to changing requirements
- Ability to quickly scale in real-time

Confidently Planning for the Future

- Understand application data storage profile
- Predict growth trajectory for MySQL platform
- Capacity planning and resource management
 - Compute Resources
 - Memory Allocation
 - Storage Resources
- Growth and scaling plan for application
- Public cloud TCO tipping-point plan

Planning for the Future: Compute

- Compute Resources
 - What is contributing to total CPU consumption? (query processing, i/o wait, virtualized steal?, etc)
 - Is CPU utilization truly driven from mysqld churning and processing data (rather than wait, steal, network wait, etc)?
 - What is the CPU growth rate for your systems?

Planning for the Future: Memory

- Memory Resources
 - Memory bound?
 - ‘Hot set’ of data fit in memory?
 - Are queries optimized to only pull required data?
 - Indexes? Over indexed? Under indexed?
 - When is the next RAM increase required?

Planning for the Future: Storage

- Storage Resources
 - Disk resource heavy?
 - IOPS bound?
 - Max IOPS ceiling for disk configuration?
 - Disk latencies within acceptable tolerance?
 - Sequential reads, large chunk processing?
 - More random in nature? Disk heads thrashing?
 - Is flash memory required?
 - Disk usage consumption rate?
 - When is the next storage addition required?
 - How will you add that capacity?
 - Extend existing filesystem natively? (xfs_grow)
 - Filesystem-per-database strategy (symlinks)?

Planning for the Future: Topology

- MySQL topology scaling plan
 - Master cluster servers (Percona Cluster, Galera)
 - Master shard servers (Homegrown, ClusterixDB, etc)
 - Slave servers (read-heavy environments)
 - DevTest instances that require prod db copies
 - Reporting and Data Warehouse instances
 - Public vs Private Cloud
 - Orchestration, OpenStack, DBaaS: Trove

Instrument and Gather

- System Performance Data is Essential
 - Quantify change rate over time
 - Monitor every layer, from app to storage
 - Zabbix, Zenoss, Munin, Cacti, Nagios, Graphite
 - Critical to real-time troubleshooting

“Trust in God, all others bring Data”

Larger Data Sets == Larger Challenges

- Automation increasingly important
- Leverage software defined infrastructure
- Quickly react to increase performance
- Deploy additional slaves for read scaling
- Refresh DevTest, QA, Business copies
- Improve Backup and Restore times

Leverage Point: Pivot on the Storage Layer

- Invert the Dominant Paradigm
- Data Gravity and Storage Jiu-jitsu
- Move the platform around the data
- Modern shared storage capabilities
 - Snap/clone, writable snapshots
 - De-duplication, QoS allocation, Scale-out
- Efficient use of storage resources

Shared Storage: MySQL Ops Secret Weapon

- Real-time resource allocation
 - Extend volume capacity
 - Designate Min/Max/Burst IOPS
- Dev/Test secondary copies
 - Deployment for growing teams
 - Refreshes for faster iterations
- Replication slave creation
 - MySQL read arrays
 - HA warm standby copies
- Decrease/eliminate backup windows
- Accelerate restore scenarios

Real-time resource allocation

- Modern storage virtualizes resources
- Allows for dynamic allocation
- Increase capacity on the fly
- Change QoS settings (Min/Max/Burst IOPS)
- Scale 'up' instantly without lead time
- Scale out – horizontal storage growth

Deploy Secondary Copies in Seconds

- Snapshot the prod MySQL storage volume
- Create a volume clone from the snapshot
- Attach the cloned volume to a MySQL VM
- `service mysqld start`
- For a 1TB dataset: ~6 hrs -> ~90 seconds

Deploy Replication Slaves in Seconds

- Flush the target master, `SHOW MASTER STATUS`
- Snapshot the prod MySQL storage volume
- Create a volume clone from the snapshot
- Mount the cloned volume to a MySQL instance
- Increment `server_id`; remove `auto.cnf`
- Script in the `CHANGE MASTER` config
- `service mysqld start`
- `start slave`
- For a 1TB dataset: ~9+ hrs -> ~100 seconds

Decrease or Eliminate Backup Windows


- Leverage instant snapshots
- Creates crash consistent backups
- Zero impact to production performance
- Multiple volumes? Group snapshot
- Suitable for many use cases
- Efficient storage utilization (meta data only)

Accelerate Restore Scenarios

- Snapshot Backups are crash consistent
- Ideal time-sensitive restore operations
- Snapshots applied to volumes instantly
- Revert in seconds:
 - Stop mysqld
 - Unmount /var/lib/mysql
 - Restore storage volume from snapshot (instant)
 - Remount /var/lib/mysql
 - Start mysqld
- Key: block storage metadata manipulation

Managing Scale Through Consolidation

- Enterprise/Web Scale MySQL: new set of challenges
- Understanding growth patterns is essential
- Virtualization and Orchestration ecosystems
- Data Gravity requires a capable storage layer
- Pivot on storage to avoid data transfer

Thank You 



Come visit us at Booth #211

We're also hiring...