MySQL High-Availability and Scale-Out architectures

Oli Sennhauser
Senior Consultant
osennhauser@mysql.com
Introduction

• Who we are?
• What we want?
Table of Contents

• Scale-Up vs. Scale-Out
• MySQL Replication
• MySQL HA solution
• MySQL Cluster
• Application architectural solutions
• Other architectural stuff
• It's your turn...
• Let us build a replication
When do we think about architecture?

• Performance problems
• HA requirements
Scale-up vs. Scale-Out

- Scale-up

- Scale-out

Relaxation of constraints
MySQL Replication

• The MySQL answer: Master-Slave-Replication:

Even Oracle uses the expression “scale-out” now!
MySQL Replication in detail

- wide distance possible!
- thin line possible!
MySQL Replication varieties

• Cascaded replication:

for example Yahoo!
MySQL Replication varieties

• HA with “hot standby” Slave:

  - **Appl**
  - **VIP**
  - **S\textsubscript{backup}**
  - **M**
  - **S\textsubscript{standby}**
  - **S\textsubscript{1}**
  - **S\textsubscript{2}**
  - **S\textsubscript{3}**

  HA solution

• active – active fail over!
MySQL HA solution

- HA with MySQL:
  - 99.99% HA (four nine)
  - active – passive fail over!

VIP

Appl

M

M'

S_1  S_2  S_3

DRBD

heartbeat
MySQL HA solution

• HA with MySQL and SAN:

```
+---+---+---+
| Appl | VIP | M' |
+-------+-----+-----+
|       |     |     |
|       |     |     |
|       |     |     |
|       |     |     |
+---+---+---+
| S_1 | S_2 | S_3 |
|     |     |     |
|     |     |     |
|     |     |     |
|     |     |     |
+---+---+---+
|     |     |     |
|     |     |     |
| SAN |     |     |
```

heartbeat
Multi-Master Replication

• actually Master-Master Replication

• This architecture does NOT solve your write problems!!!
Multi-Master Replication

• HA Multi-Master Replication
Shared disk cluster

- This is in 99.9% of the cases NOT your solution!!!
- It looks like Oracle RAC but MySQL does NOT (yet) have the instruments needed!
- Can be useful in some special kinds of reporting.
Where are we now?

• The architectures above solved:
  – your read problems.
  – some HA requirements.

• But they did not solve:
  – your write problems!

• And what when we need:
  – higher HA?
  – synchronous replication?
  – more write speed?
MySQL Cluster

- Shared-nothing architecture
- Synchronous replication (2-Phase commit)
- Fast automatic fail over
- High performance (also writing)
- High transactional throughput
- No special components required
- In-Memory database (in 5.1 also disk support)
- Scalable, 1000's of transactions per second
- 99.999% HA (five nine)
- On-line upgrade path (at least on GA within same version)
MySQL Cluster architecture

The diagram shows the architecture of MySQL Cluster, which includes:

- **Application Nodes**: These are the application nodes where the applications interact with the database.
- **SQL Nodes**: These nodes are responsible for handling SQL queries.
- **Data Nodes**: These are the nodes that store the actual data.
- **Mgmt Nodes**: These nodes handle the management and configuration of the cluster.

The MySQL Server is present in each node, and they are connected through a network to form the cluster. The `my.cnf` and `config.ini` files are also shown, indicating configuration settings for the cluster.
MySQL Cluster HA features
Cluster with Replication

• for read scale-out (Reporting):

![Diagram showing a cluster with replication setup using MySQL and NDBD. The diagram includes an application (Appl) connected to multiple MySQL and NDBD processes, which are then connected to a reporting section labeled 'Reporting'.]
Cluster with Replication

• Cluster-Cluster replication for disaster fail over (MySQL 5.0):

```
Appl
  "TW"
  mysql
    mysql
      ndbd
      ndbd
      ndbd
      ndbd
```

```
mysql
  mysql
    mysql
      ndbd
      ndbd
      ndbd
      ndbd
```

```
fail over
```
Cluster with Replication

• Cluster-Cluster replication for disaster fail over (MySQL 5.1):
Cluster with Replication

- circular Cluster-Cluster replication (>= MySQL 5.1.18):
Cluster examples

• Session handling
• Telecom (Mobile)
• VoIP
• RSS-Feed aggregation
• Mail
• On-line Games

• Use cluster where
  – you need HA
  – you have high write load
  – you do little Joins and Grouping
How to go on?

• Read is a caching problem!
  ➔ More RAM or scale-out.

• Write is a batching problem!
  ➔ Batch your load,
  ➔ Buy stronger I/O system.
  ➔ Use MySQL Cluster.
  ➔ SSD!

• What then?
SSD disruption

• I/O system without any movable parts!
• SSD = Solid State disk (Flash memory, NAND, NOR chips, RAM-SAN).
• During the last year(s)
• Price from 15'000 USD / 160 Gbyte -> 100 USD / 32 Gbyte
• 10-50 times faster than mechanical I/O systems
• 1 Mio write cycles dead :-( (special FS!)

• RAM: Huge amount of memory is cheap!

→ This will disrupt the database world!
What then?

• With or without SSD we will reach a physical limit. What then?

• Application partitioning:
  – Split applications
    • OLTP vs. OLAP
    • all in one
  – Segment your application
Application partitioning

• OLTP vs. OLAP
  – hot business data (trx) vs. “old” reporting data

• “All in one”
  – Sessions, user tracking, ads, chat, booking

• Segment
  – split by for example 1 Mio users (split by user_id).
Architecture examples
Architecture examples
Architecture examples
Architecture examples

Failover Standort

AS01
eth0
eth1
bond0:0.101

AS02
eth0
eth1
bond0:0.102

AS03
eth0
eth1
bond0:0.103

AS04
eth0
eth1
bond0:0.104

VIP .113

bund0:109
bund1:10
bond1:20
bond0:116

DB01 (Blade 6/2 bays)
RDAC
FC1
FC2

SCSI
local
SAN

Disaster Standort

AS05
eth0
eth1
bond0:0.105

AS06
eth0
eth1
bond0:0.106

Replication

bund0:126

DB03 (Blade 12/1 bay)
RDAC
FC1
FC2

SCSI
local
SAN

Copyright 2007 Oli Sennhauser
The World’s Most Popular Open Source Database
Some more architecture stuff?

• Databases are slow!!!
  ➔ Whenever possible try to avoid databases! Use memcached for example.

• The SP trap.
  ➔ Stored Procedures are a lock in! Try to avoid SP!

• Use Materialized Views (MV) and/or shadow tables.

• VM/SAN is nice for consolidation but not for performance/scale-out!

• Backup and Staging

• MySQL – Proxy
Now it's your turn...

• Your problems?

• Let us build a replication...