



ORACLE®

MySQL Cluster 7.1

Bernhard Ocklin

MySQL Cluster 7.0 Momentum

Leading Provider of Mobile, Fixed & Broadband Services across Europe & Asia

"Telenor has found MySQL Cluster to be the best performing database in the world for our applications."



Peter Eriksson, Manager, Network Provisioning

250% Increase in Download Rate

Windows Port

LDAP Interface

I/O Multi-Threading

2x Faster Record Handling

On-Line Cluster Scaling

4x Higher Throughput

Largest Entertainment Guide on the Mobile Web:

"go2 has the ability to efficiently scale our platform with the growth of the mobile web"



Dan Smith, Co-Founder & CEO, go2 Media

MySQL Cluster 7.1

Monitor

New real time SQL based monitoring of cluster internals with NDBINFO

Reliability

Faster restarts for higher availability

Manage

New Cluster Manager

Access

New high performance native Java access to cluster and OpenJPA plug-in

Platform

Windows version

Cluster on Windows

The image shows a Windows Task Manager window on the left and a terminal window on the right. The Task Manager window displays a list of running processes, with a callout box highlighting several MySQL Cluster-related processes. The terminal window shows the output of a command in a MySQL Cluster environment.

Task Manager Processes:

Name	herauslastung
mysql.exe	5.508 K
mysql.exe	1.464 K
mysql.exe	2.636 K
ndb_mgmd.exe	4.120 K
ndbd.exe	3.052 K
ndbd.exe	16.248 K
ndbd.exe	1.380 K
ndbmtbd.exe	28 K
ndbmtbd.exe	6.284 K
perl.exe	968 K
perl.exe	1.060 K
perl.exe	980 K
perl.exe	1.060 K
perl.exe	1.060 K
mysql.exe	7.732 K
mysql.exe	7.964 K
ndb_mgmd.exe	4.112 K
ndbd.exe	187.660 K
ndbmtbd.exe	244.556 K
perl.exe	13.904 K
perl.exe	12.656 K
perl.exe	12.636 K
perl.exe	14.380 K
perl.exe	12.636 K
perl.exe	12.636 K
perl.exe	12.656 K
services.exe	3.216 K
smss.exe	416 K
spoolsv.exe	4.440 K
svchost.exe	3.384 K
svchost.exe	3.068 K
svchost.exe	3.920 K
svchost.exe	24.784 K
svchost.exe	3.468 K
svchost.exe	3.668 K
System	236 K
taskmgr.exe	4.072 K
VBoxService.exe	3.044 K
VBoxTray.exe	2.768 K

Terminal Output:

```
mysql-cluster/ p.p1 p2.p
b_mgmd.1.1 - pid: 324, winpid: 324]
```

Real time monitoring with NDBINFO

- Real time infrastructure
- SQL based
- Access via MySQL Server (and its connectors)

```
mysql> select * from logbuffers;
```

node_id	log_type	log_id	log_part	total	used
1	REDO	0	1	33554432	262144
2	REDO	0	0	33554432	262144

```
2 rows in set (0.04 sec)
```

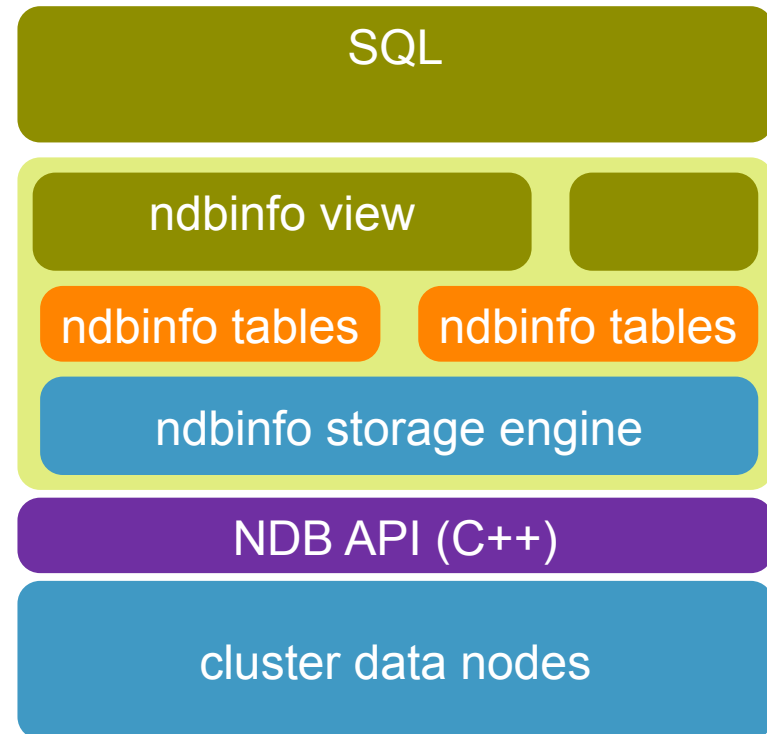
Simplified Monitoring

```
mysql> use ndbinfo
mysql> show tables;
+-----+
| Tables_in_ndbinfo |
+-----+
| blocks             |
| config_params     |
| counters          |
| logbuffers        |
| logspaces         |
| memoryusage       |
| nodes             |
| resources         |
| transporters      |
+-----+
```

- Simple & consistent access to real-time status & statistics
 - Info from data nodes
 - Data presented in SQL views
- Monitor & optimize cluster performance & availability
 - Monitor resource consumption (i.e. memory & log buffers)
 - Monitor operations performed on the database
 - Monitor node and connection status
- No log trawling

ndbinfo internals

- Simple structs in data nodes
- Hidden tables as ndbinfo storage engine
- End user views on top



Optimized faster restarts



- Dramatically reduced restart times by
 - better caching
 - better internal communication
- Faster recovery, higher availability

New Java Connector for Cluster

- 2 new Java interfaces into MySQL Cluster
- No SQL - Avoiding transformations into SQL
- avoiding MySQL Server's performance overhead

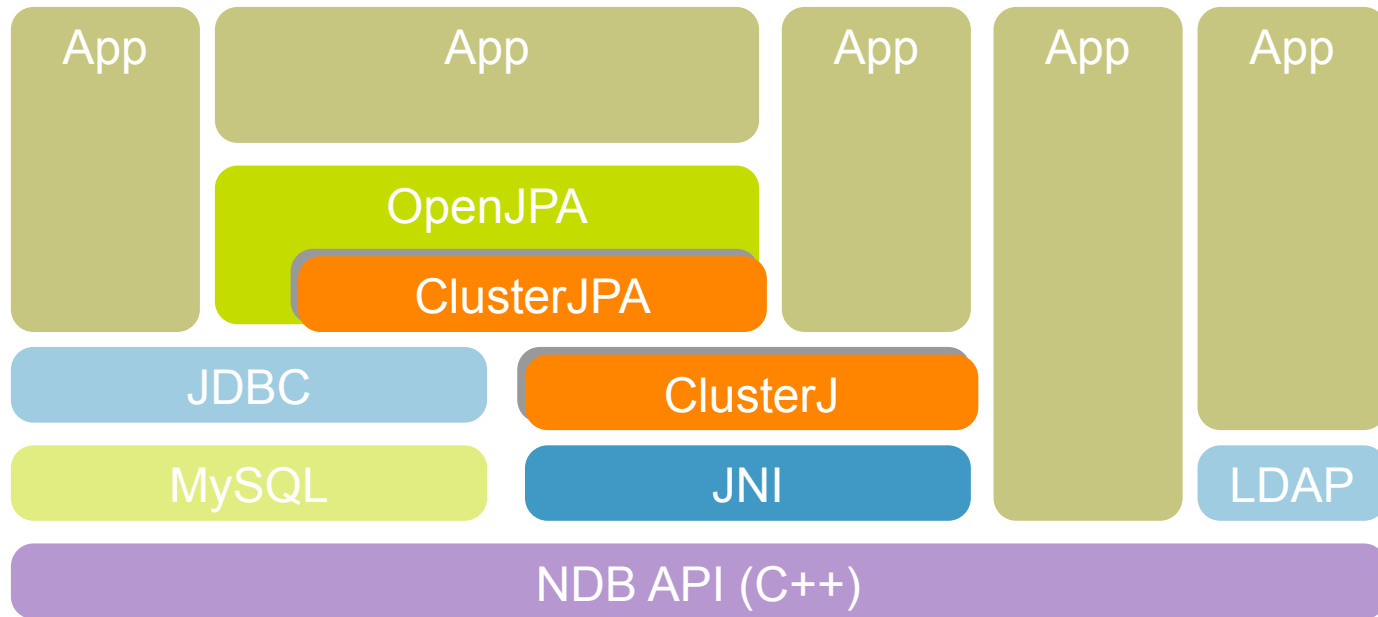
- Domain Object Model familiar to most Java developers
- Embedded into Java applications
- „Native“ JNI based Java Connector

ClusterJ

- OpenJPA plug-in
- ClusterJ used for primary key reads, insert, deletes & updates
- JDBC used for remaining operations
- JPA very popular (e.g. JavaWorld's all time top 3)

ClusterJPA

New native Cluster Connector for Java



ClusterJ interface table mapping

```
@PersistenceCapable(table="t_employees")  
public interface Employee {
```

```
@PrimaryKey
```

```
int getId();  
void setId(int id);
```

```
@Column(name = "Lastname")
```

```
@Index(name="idx_unq_hash_name")
```

```
String getName();  
void setName(String name);
```

```
...
```

```
};
```

```
CREATE TABLE t_employees (
```

```
id int primary key,
```

```
Lastname varchar(255)  
unique index using hash,
```

```
...
```

```
) ENGINE = ndbcluster;
```

ClusterJ configuration

```
com.mysql.clusterj.connectstring=localhost:1186
```

```
com.mysql.clusterj.database=test
```

```
com.mysql.clusterj.connect.retries=4
```

```
com.mysql.clusterj.connect.delay=5
```

```
com.mysql.clusterj.connect.verbose=1
```

```
com.mysql.clusterj.connect.timeout.before=30
```

```
com.mysql.clusterj.connect.timeout.after=20
```

```
com.mysql.clusterj.username=berni
```

```
com.mysql.clusterj.password=inreb
```

```
com.mysql.clusterj.max.transactions=1024
```

ClusterJ start

```
SessionFactory factory =  
    ClusterJHelper.getSessionFactory(props) ;  
  
Session session = factory.getSession() ;
```

ClusterJ example

```
Employee theEmployee =  
    session.find(Employee.class, 4711);  
  
// Create and initialise an Employee  
Employee newEmployee =  
    session.newInstance(Employee.class);  
newEmployee.create(988, "Bernd", "Ocklin", "01/09/2005", 67);  
  
// Write the Employee to the database  
session.persist(newEmployee);  
  
// change Employee & write back to the database  
theEmployee.setDepartment(777);  
theEmployee.setCity("Stockholm");  
session.updatePersistent(theEmployee);
```

ClusterJ query

```
// Retrieve the set all of Employees in department 777
QueryBuilder builder = session.getQueryBuilder() ;

QueryDomainType<Employee> domain =
    builder.createQueryDefinition(Employee.class) ;

domain.where(
    domain.get("department").equal(domain.param("department")) ) ;

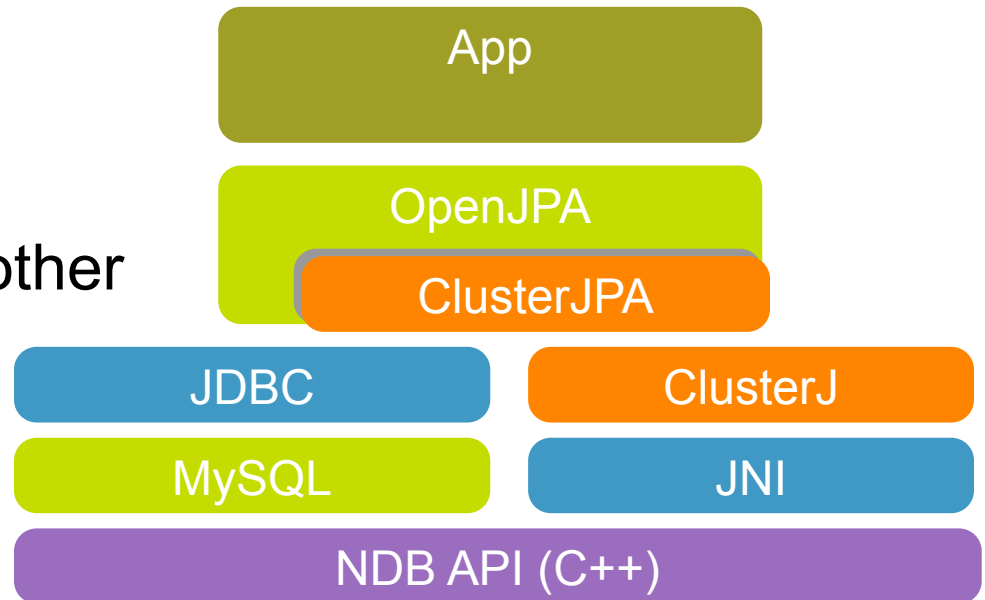
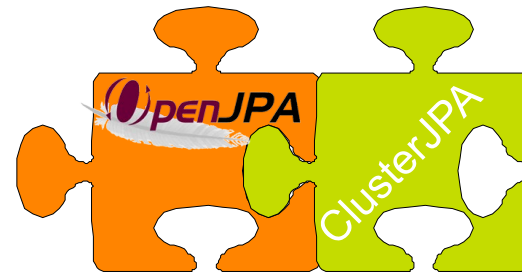
Query<Employee> query = session.createQuery(domain) ;

query.setParameter("department", 777) ;

List<Employee> results = query.getResultList() ;
```

ClusterJPA

- plugs into OpenJPA
- uses native access path via ClusterJ for
 - find()
 - insert
 - delete
 - update
- transparently routes all other to JDBC



ClusterJPA configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence ...>
  <persistence-unit name="ndbjpa" transaction-type="RESOURCE_LOCAL">
    <provider>org.apache.openjpa.persistence.PersistenceProviderImpl</provider>
    ...
    <properties>
      <property name="openjpa.ConnectionDriverName"
                value="com.mysql.jdbc.Driver" />
      <property name="openjpa.ConnectionURL" value="jdbc:mysql://host:3306/test" />
      <property name="openjpa.ConnectionUserName" value="root" />
      <property name="openjpa.ConnectionPassword" value="" />
      <property name="openjpa.BrokerFactory" value="ndb" />
      <property name="openjpa.ndb.connectString" value="localhost:1186" />
      <property name="openjpa.ndb.database" value="test" />
    </properties>
  </persistence-unit>
</persistence>
```

ClusterJPA example

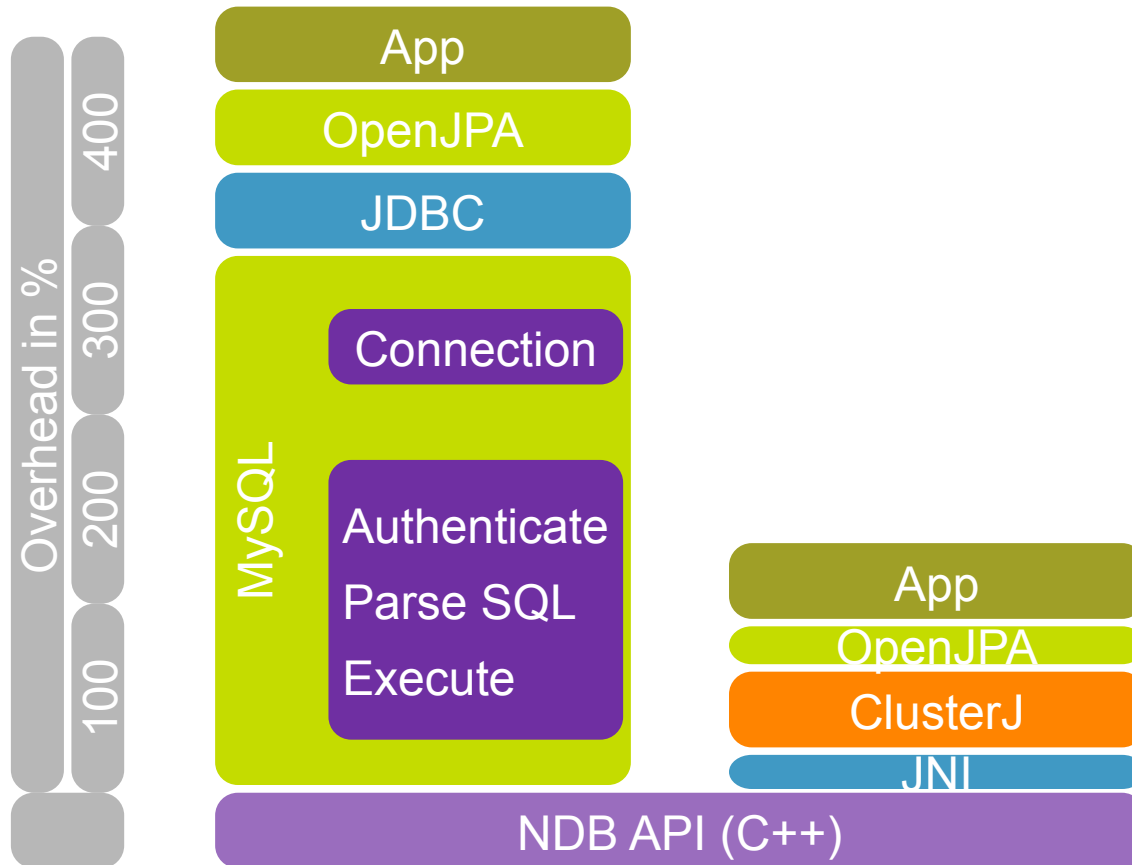
```
EntityManagerFactory entityManagerFactory =
    Persistence.createEntityManagerFactory("ndbjpa");
EntityManager em = entityManagerFactory.createEntityManager();
EntityTransaction userTransaction = em.getTransaction();

userTransaction.begin();
Query q =
    em.createQuery("select x from Fish x where x.tastiness > 12");

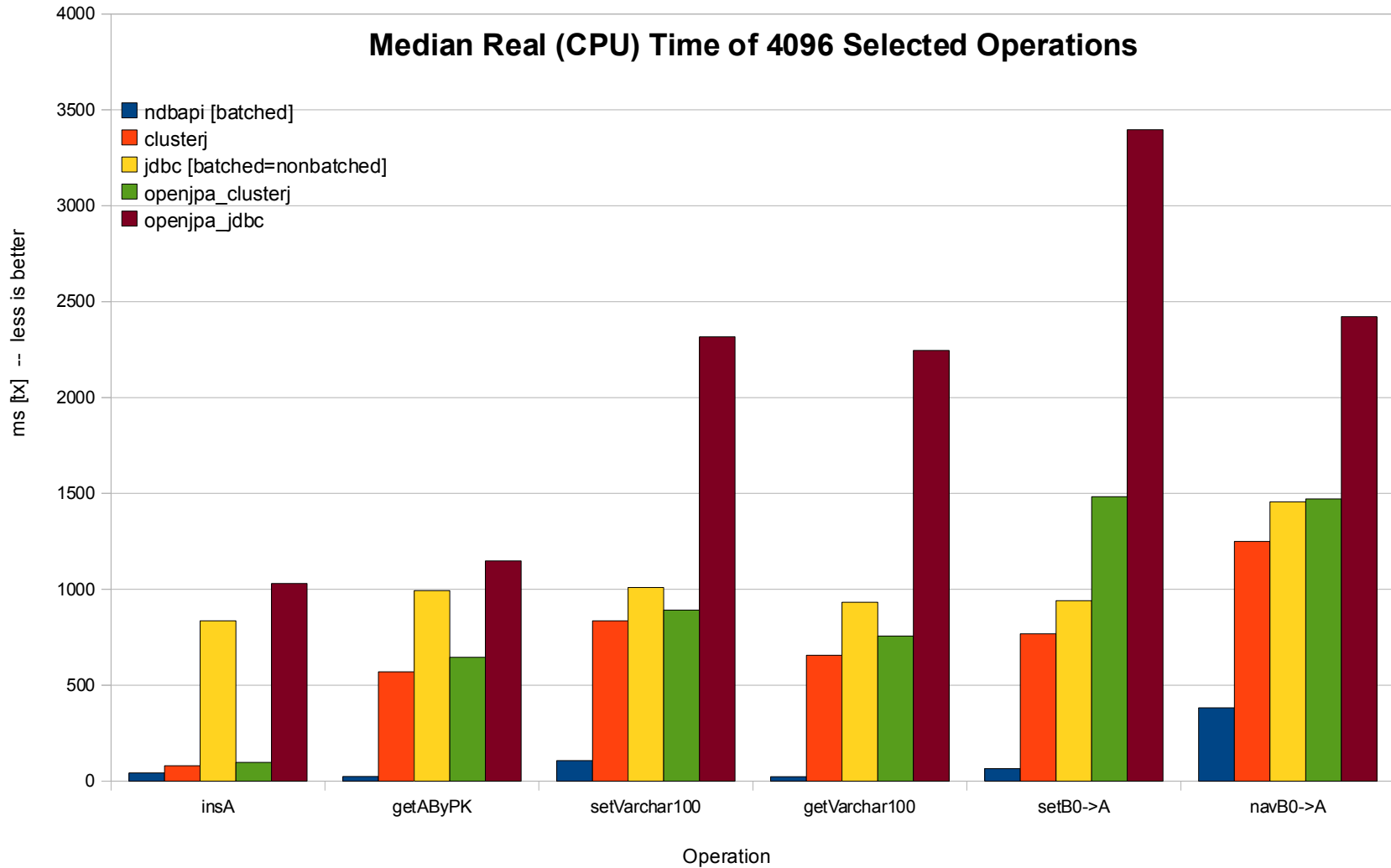
for (Fish m : (List<Fish>) q.getResultList()) {
    System.out.println(m.toString());
}
userTransaction.commit();

em.close();
entityManagerFactory.close();
```

MySQL OpenJPA overhead

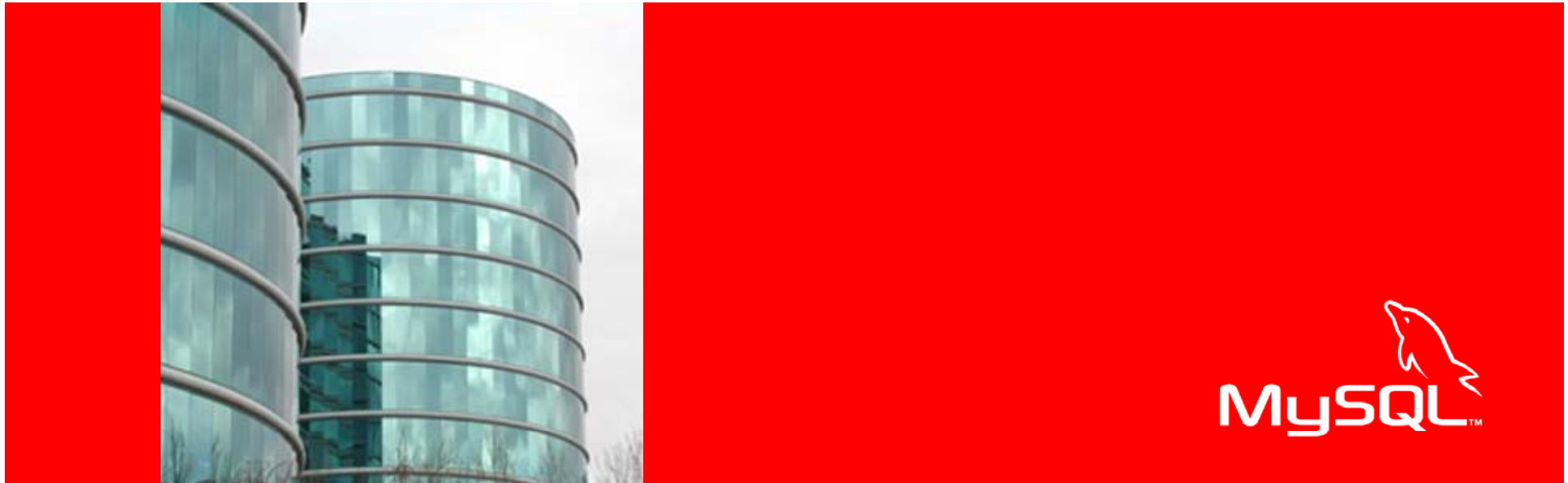


ClusterJ/JPA reducing latency



What are we working on now?

- Today's & tomorrow's hardware
 - large RAM boxes
 - disk speed
 - thread contention
 - DD steal
 - disk index
 - multi-threaded transaction coordinator
- Raising the performance bar, again
 - join performance
- More columns, large columns
- Basing Cluster on MySQL Server 5.5



ORACLE[®]

MySQL Cluster 7.1 (MySQL Cluster Manager 1.0)

Andrew Morgan
MySQL Cluster Product Manager

How Does MySQL Cluster Manager Help ?

Automates and Simplifies MySQL Cluster Management



<http://www.flickr.com/photos/thebbp/>

Reduces management complexity, increases administrator efficiency and reduces the risk of downtime

MySQL Cluster Manager 1.0 - Features

Automated Management

- Cluster-Wide Management
- Process Management
- On-Line Operations (Upgrades / Reconfiguration)

Monitoring

- Status Monitoring & Recovery

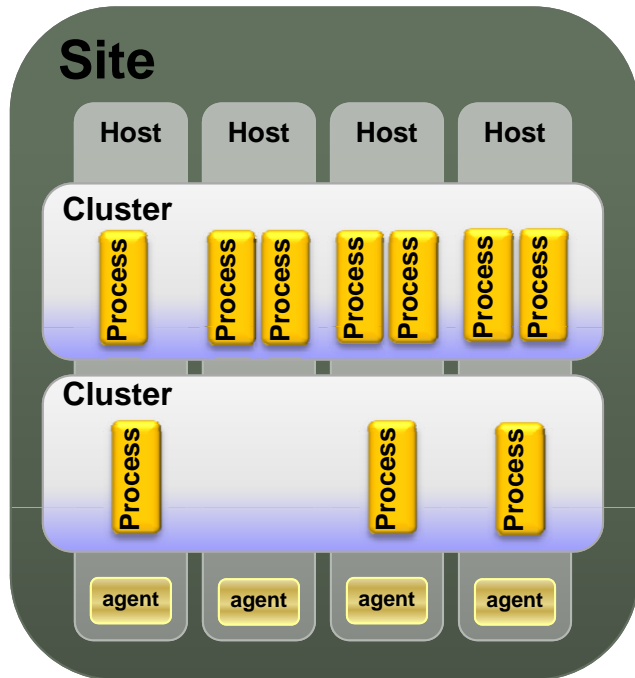
HA Operations

- Disk Persistence
- Configuration Consistency
- HA Agent Operation



Cluster Manager

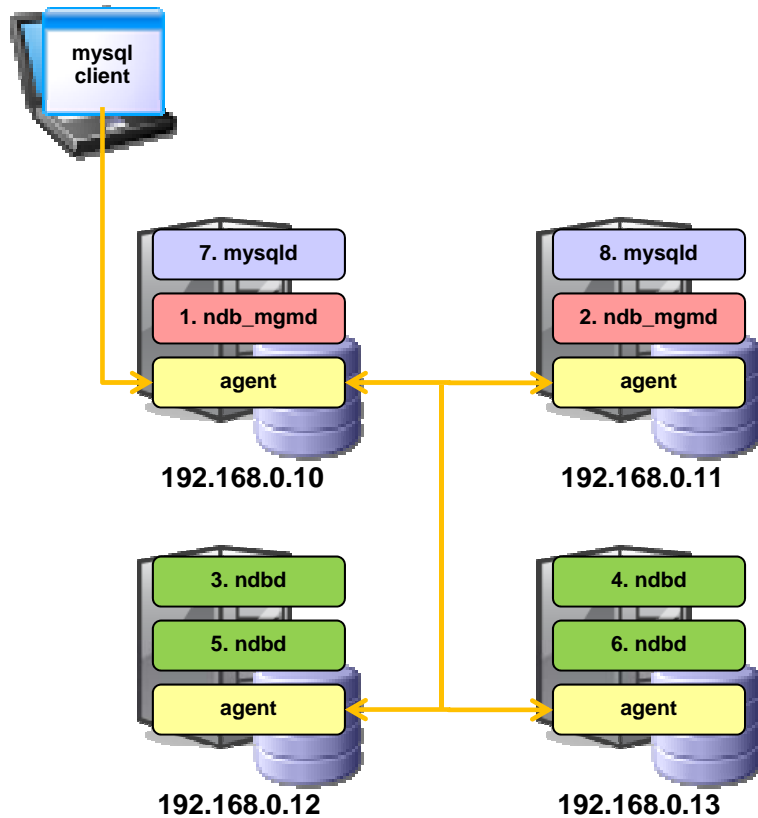
Terms used by MySQL Cluster Manager



- **Site:** the set of physical hosts which are to run Cluster processes to be managed by MySQL Cluster Manager. A site can include 1 or more clusters.
- **Cluster:** represents a MySQL Cluster deployment. A Cluster contains 1 or more processes running on 1 or more hosts
- **Host:** Physical machine, running the MySQL Cluster Manager agent
- **Agent:** The MySQL Cluster Manager process running on each host
- **Process:** an individual MySQL Cluster node; one of: `ndb_mgmd`, `ndbd`, `ndbmtd`, `mysqld` & `ndbapi`*
- **Package:** A copy of a MySQL Cluster installation directory as downloaded from `mysql.com`, stored on each host

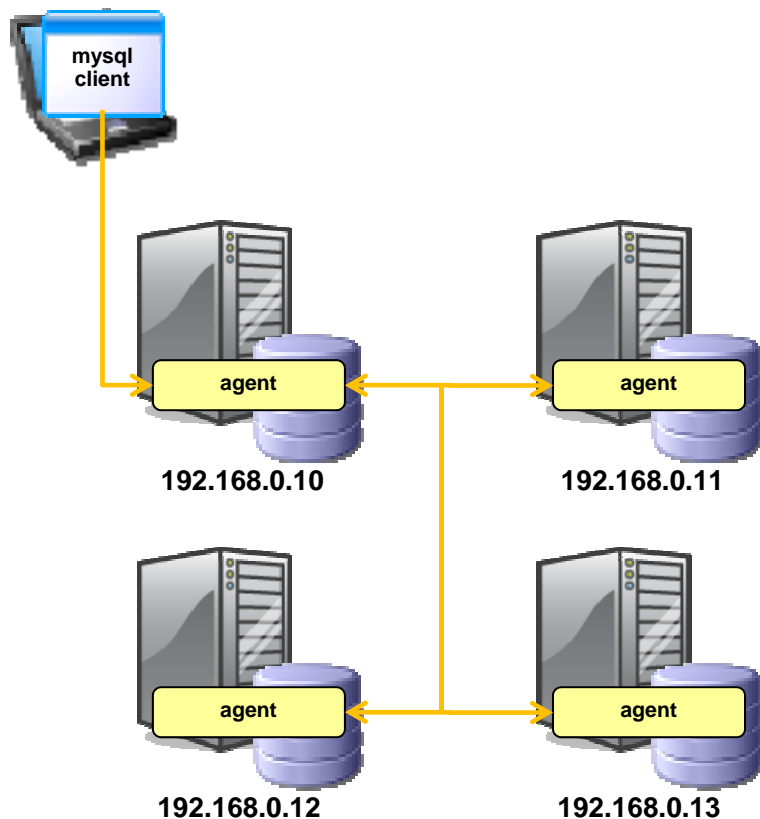
* `ndbapi` is a special case, representing a slot for an external application process to connect to the cluster using the NDB API

Example configuration



- MySQL Cluster Manager agent runs on each physical host
- Each host can run multiple MySQL Cluster nodes (processes)
- No central process for Cluster Manager – agents co-operate, each one responsible for its local nodes
- Agents are responsible for managing all nodes in the cluster
- Each node allocated an ID
- Management responsibilities
 - Starting, stopping & restarting nodes
 - Configuration changes
 - Upgrades
 - Host & Node status reporting
 - Recovering failed nodes

Installing, configuring, running & accessing MySQL Cluster Manager



- The agent must be installed and run on each host in the Cluster:

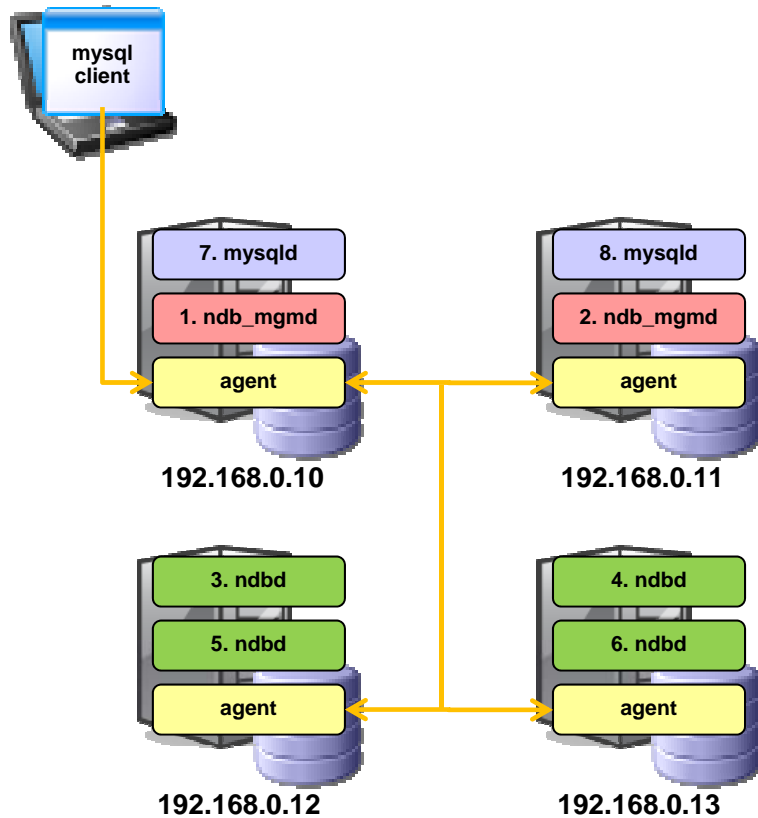
1. Expand the tar-ball into a known directory (/usr/local/mcm)
2. Copy /usr/local/mcm/etc/mysql-cluster-manager.ini to /home/billy/mcm and edit:

```
[mysql-proxy]
plugins=manager
manager-port = :1862
log-file = mysql-manager-agent.log
log-level = message
manager-directory=/home/billy/mcm/manager
```

3. “plugins=manager” should always be used
4. Launch the agent process:
5. Access any of the agents from any machine (that has the mysql client installed):

```
C:\Users\am233268>mysql -h 192.168.0.10 -P 1862 -u
admin -psuper
mysql>
```

Creating & Starting a Cluster



1. Define the site:

```
mysql> create site --hosts=192.168.0.10,192.168.0.11,  
-> 192.168.0.12,192.168.0.13 mysite;
```

2. Expand the MySQL Cluster tar-ball(s) from mysql.com to known directory

3. Define the package(s):

```
mysql> add package --basedir=/usr/local/mysql_6_3_26 6.3;  
mysql> add package --basedir=/usr/local/mysql_7_0_7 7.0;
```

Note that the basedir should match the directory used in Step 2.

4. Create the Cluster

```
mysql> create cluster --package=6.3  
-> --processhosts=ndb_mgmd@192.168.0.10,ndb_mgmd@192.168.0.11,  
-> ndbd@192.168.0.12,ndbd@192.168.0.13, ndbd@192.168.0.12,  
-> ndbd@192.168.0.13,mysqld@192.168.9.10,mysqld@192.168.9.11  
-> mycluster;
```

This is where you define what nodes/processes make up the Cluster and where they should run

5. Start the Cluster:

```
mysql> start cluster mycluster;
```

Check the status of the Cluster



<http://www.flickr.com/photos/bludgeoner86/>

1. Request the status of the Cluster processes

```
mysql> show status --process mycluster;
```

Id	Process	Host	Status	Nodegroup
1	ndb_mgmd	192.168.0.10	running	
2	ndb_mgmd	192.168.0.11	running	
3	ndbd	192.168.0.12	running	0
4	ndbd	192.168.0.13	running	0
5	ndbd	192.168.0.12	running	1
6	ndbd	192.168.0.13	running	1
7	mysqld	192.168.0.10	running	
8	mysqld	192.168.0.11	running	

2. Get the status of the hosts & agents

```
mysql> list hosts mysite;
```

Host	Status	Version
192.168.0.10	Available	1.0.1
192.168.0.11	Available	1.0.1
192.168.0.12	Available	1.0.1
192.168.0.13	Available	1.0.1

Checking Cluster parameters

- Fetch all parameters that apply to all data nodes, including defaults:

```
mysql> get -d :ndbd mycluster;
```

Name	Value	Process1	Id1	Process2	Id2	Level	Comment
__ndbmt_lqh_threads	NULL	ndbd	3			Default	
__ndbmt_lqh_workers	NULL	ndbd	3			Default	
Arbitration	NULL	ndbd	3			Default	
.....							
__ndbmt_lqh_threads	NULL	ndbd	4			Default	
__ndbmt_lqh_workers	NULL	ndbd	4			Default	
Arbitration	NULL	ndbd	4			Default	
ArbitrationTimeout	3000	ndbd	4			Default	
.....							
__ndbmt_lqh_threads	NULL	ndbd	5			Default	
.....							
__ndbmt_lqh_threads	NULL	ndbd	6			Default	
.....							

- Fetch the values of parameters (excluding defaults) for mysqld with ID=7:

```
mysql> get :mysqld:7 mycluster;
```

Name	Value	Process1	Id1	...
datadir	/home/billy/mcm/alpha/manager/clusters/mycluster/7/data	mysqld	7	...
HostName	ws1	mysqld	7	...
ndb-nodeid	7	mysqld	7	...
ndbcluster		mysqld	7	...
NodeId	7	mysqld	7	...

- Fetch the port parameter to connect to mysqld with ID=7:

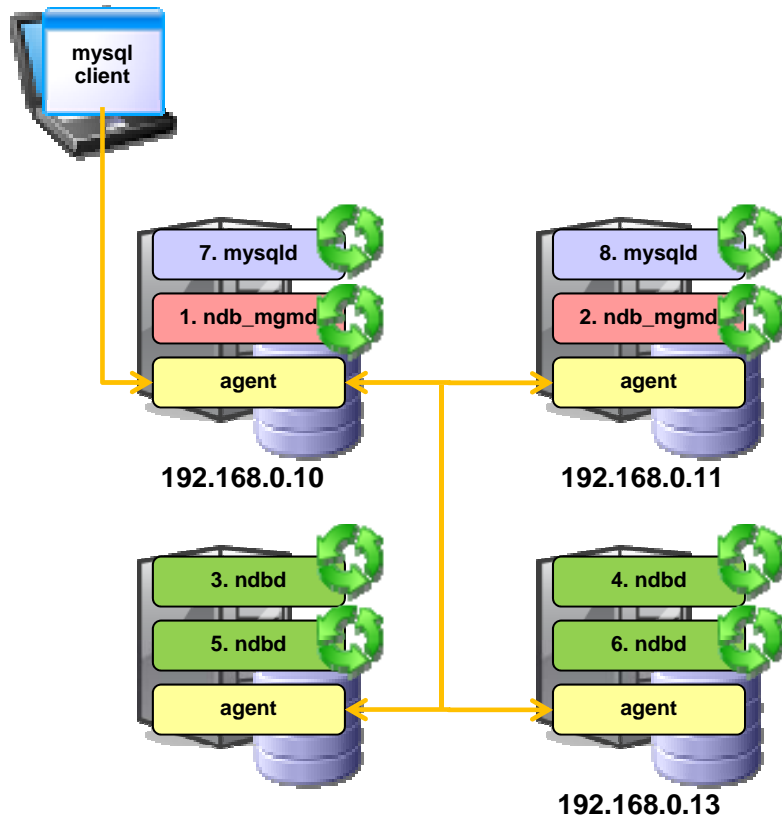
```
mysql> get -d port:mysqld:7 mycluster;
```

Name	Value	Process1	Id1	Process2	Id2	Level	Comment
port	3306	mysqld	7			Default	



<http://www.flickr.com/photos/jonathanbeard/>

Setting Cluster parameters



- Example: Turn off privilege checking for all MySQL Servers and change the port for connecting to the mysqld with ID = 8 to 3307. Allow data nodes to be automatically restarted after they fail:

```
mysql> set skip_grant_tables:mysqld=true,port:mysqld:8=3307,  
--> StopOnError:ndbd=false mycluster;
```

MySQL Cluster Manager automatically determines which nodes (processes) need to be restarted and in which order to make the change take effect but avoid loss of service

Upgrade Cluster



<http://www.flickr.com/photos/geminidustin/>

- Upgrade from MySQL Cluster 6.3.26 to 7.0.7:

```
mysql> upgrade cluster --package=7.0 mycluster;
```

- Automatically upgrades each node and restarts the process – in the correct order to avoid any loss of service
- Without MySQL Cluster Manager, the administrator must stop each process in turn, start the process with the new version and wait for the node to restart before moving onto the next one

HA Provided by MySQL Cluster Manager



www.flickr.com/people/belgapixels/

- Service is maintained when any Cluster process fails but the processes need to be restarted as soon as possible afterwards to restore redundancy
- If a process dies then it will be detected by the local agent
- Without MySQL Cluster Manager, an angel process would recreate failed `nbd` processes but not `mysqld` or `ndb_mgmd` processes
- Cluster continues to function if one or more agents fail or is stopped (e.g. To upgrade it)
 - To restore management functions, the agent should be restarted




MySQL Cluster Manager & Management Node

- Configuration and state changes should now be performed through MySQL Cluster Manager
- MySQL Cluster Manager does not remove the need for the management node (ndbd_mgmd); its continuing responsibilities are:
 - When data nodes start up (or are restarted) they connect to the management node(s) to get their configuration data
 - When stopping or restarting a data node through the Cluster Manager, the state change is actually performed by ndb_mgmd
 - The management node(s) can continue to act as arbitrators (avoiding a split-brain scenario)
 - Some reporting information (e.g. Memory usage) is not yet available in the Cluster Manager and can still be performed using the ndb_mgm tool
- Important to continue to host the management node on an independent host (from the data nodes) and consider using 2 for redundancy.



Resources

- MySQL Cluster 7.1, Architecture and New Features:
 - www.mysql.com/why-mysql/white-papers/mysql_wp_cluster7_architecture.php (http://bit.ly/7_1_wp)
- MySQL Cluster Connector for Java Whitepaper:
 - www.mysql.com/why-mysql/white-papers/mysql_wp_cluster_connector_for_java.php (http://bit.ly/clusterj_wp)
- MySQL Cluster Manager Whitepaper:
 - www.mysql.com/why-mysql/white-papers/mysql_wp_cluster_manager.php (http://bit.ly/mcm_wp)
- Getting started with MySQL Cluster:
 - www.mysql.com/products/database/cluster/get-started.html
- Download MySQL Cluster:
 - dev.mysql.com/downloads/cluster/
- Blogs:
 - Bernd: ocklin.blogspot.com
 - Andrew: www.clusterdb.com



The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.