

MySQL集群评估手册

上海爱可生-中国最领先的MySQL服务提供商

目录

- 1 简介
- 2 MySQL 集群是什么
- 3 使用 MySQL 集群可带来的效益
- 4 MySQL 电信级集群 7.0 的主要特性
- 5 评估前的注意点
- 6 设置与评估指南
- 7 配置文件的注意事项
- 8 正确性检查

1 简介

编写此文档的目的：在您评估 MySQL 集群或是 MySQL 电信级集群时，需要引起您注意的一些地方。利用此文档可以帮您有效地对 MySQL 集群进行评估，来确定它是否适用于您的项目。

2 MySQL 集群是什么

2.1 MySQL 集群架构

MySQL Cluster 是一个以独特的无共享体系架构和标准 SQL 接口构建的高可用数据库产品。系统由一系列的通信进程，或是分布于各机器上的节点构成，哪怕在服务器出现故障或是网络故障时，都可以提供一个持续可用的系统。MySQL Cluster 使用专有的存储引擎来存取数据，这套引擎由一组数据节点构成，可以通过 MySQL Server 用标准 SQL 来访问或是通过 NDB API 进行实时的访问。NDB API 是 MySQL Cluster 使用的面向对象的应用程序接口，它实现了索引，扫描，事务，事件处理。NDB 事物是遵循 ACID 准则的。集群正是通过这种方式提供了将多个操作组成一组，要么全部执行成功（提交），要么作为一个整体失败（rollback）。

MySQL Cluster 容许几个数据节点同时出现故障，并且在重新配置集群的设置之后可以屏蔽掉这些故障。这种自我修复的特性、集群数据存储分布和按应用类别分区存储的透明性形成了一个简洁的编程模型，这个模型使数据库开发人员在无需的底层代码编写情况下，很容易在他们的应用程序中获得系统的高可用性。

MySQL 集群由三种不同的节点组成，每一种都提供特点的服务。

数据节点 (Data Nodes) 集群中的主要的节点，提供如下的功能：

- 数据存储和管理内存和磁盘数据
- 自动或是用户自定义分区数据
- 在数据节点之间进行同步的复制
- 事务处理和数据检索
- 自动故障恢复
- 故障后进行异步

使数据分布存储于无共享架构，如此一来，如果数据节点故障，总是至少会有一个备份的数据节点存储了相同的信息，于是请求将会被满足，而觉察不到中断，数据节点可以在线添加。

应用节点 (Application Nodes) 即连向数据库的应用。它可以是使用 NDB API 的应用程序或是就是 MySQL 服务器。如此一来应用可以利用丰富的接口（比如：SQL，LDAP，WEB 服务）并发的访问 MySQL 集群中的数据，应用节点可以在线添加。

管理节点 (*Management Nodes*) 负责管理集群，并使配置信息对于其它的节点都可用。管理节点用于启动集群或是在系统经过重新配置的情况下。管理节点可以停启而不影响数据节点和应用节点的运行。默认情况下，管理节点还提供仲裁服务。

2.2 MySQL 集群的变量

MySQL 集群数据库可以在社区下载页面下载到，其遵循 GPL 协议。商用版有以下两种：

- MySQL 标准版集群 (SE)
- MySQL 电信级集群 (CGE)

图 1 列出了 MySQL 标准版集群与 MySQL 电信级集群的区别

MySQL Cluster Feature	SE	CGE
Acid Compliant, Transactional Database	X	X
In-Memory Index and Data	X	X
Disk-Based Data	X	X
Distributed, Shared Nothing Architecture	X	X
Synchronous Data Replication	X	X
Automatic Sub-Second Failover & Self-Healing	X	X
Variable Sized Records	X	X
User-Defined Partitioning	X	X
On-Line Schema Updates and Systems Maintenance	X	X
Scale Up and Scale Out on COTS Systems	X	X
On-Line Back-Up	X	X
SQL Access	X	X
NDB Access (C / C++ / Java)		X
On-Line Add Node		X
Data Store for LDAP Directories		X
Geographic Replication		OPTION
Custom Feature Development		OPTION

图 1

3 使用MySQL集群可带来的效益

3.1 可扩展性

MySQL Cluster 在五个不同的级别提供可扩展性

- 如果需要更多的存储或是容量，那么可以在不影响服务的情况下动态地添加数据节点。
- 可以动态的添加应用节点来提升性能和并行处理能力。
- 连接至应用节点的客户端可以动态的添加。
- 数据节点上额外的 CPU，核，线程可通过使用多线程的 ndb 进程来得到充分利用
- 数据库可以复制到其它的数据库（比如，MyISAM）以用于读操作或是用于生成复杂报表。

3.2 性能

MySQL Cluster 在五个不同的级别提供可扩展性，在与以下特性相结合时，可得到极大的性能的提升：

- NDB API 或是 NDB/J
- 主键查找
- 预知分布应用设计
- 用户自定义分区
- 并行化
- 批量事务
- 高性能的网络接口 (SCI)

3.3 高可用性

数据节点可以故障，但是可以自动重新同步，而不影响服务或是强制应用节点重新连接。再者，可以添加多个管理节点和应用节点以提供最大限度的服务。亦可以在不同的地理位置上进行集群间的复制。

4 MySQL电信级集群 7.0 的主要特性

MySQL 电信级集群引入了许多新的特性，这些使得完全可以用它构建一个高性能，高可扩展性和高可用的系统。这些特性包括：

- 多线程的数据节点：一个单一的数据节点可以有效的利用 8 个 CPU/核/线程
- 在线添加节点：通过添加节点组提供可扩展的性能与容量而不影响服务
- 多线程的磁盘数据文件访问：基于磁盘的数据的性能由此提高。
- 改善的大记录的处理能力：提升了性能。

5 评估前的注意点

5.1 使用 MySQL 集群的应用程序可以得到比使用其它的数据库更快的速度吗？

事实情况是，必须要在应用和数据库做一些修改，才有可能达到这种效果。根据此白皮书的指南和最佳实践建议，可以得到比其它的数据更好的性能。

5.2 有没有一种更方便快捷的方式来将所有的东西都安装好？

Sun Microsystems提供RPM及非RPM形式的安装包以用于安装集群二进制包。MySQL参考手册(<http://dev.mysql.com/doc/refman/5.1/en/index.html>)包含了安装集群和MySQL Server组件的指南。

可以将安装和配置MySQL集群的过程写成一个脚本。但是，Sun Microsystems并没有发布此类的脚本，用户可以自己创建，或是从第三方得到。一个第三方（并且是免费）的工具可以在以下地址得到<http://www.severalnines.com/config/>

使用这脚本可以在几分钟内完成一个集群的安装，启动，运行。注意：Sun Microsystems 并不对此工具提供支持。

5.3 数据库需要运行于一个特定的操作系统吗？

MySQL 集群的所有的核心组件，即，MySQL Server，数据节点，管理服务器/客户端必须运行于支持集群的 Linux 或是 Unix 操作系统。MySQL 集群 7.0 开始在开发系统（而不是生产系统）中提供对 Windows 平台的支持。

集群中的所有的机器必须要有相同的架构。即运行节点的所有机器必须同为 big-endian 或是 little-endian，两者的混合是不允许的。比如，将管理节点运行于 SPARC 主机上，而将数据节点运行于 x86 机器。这种限制不延伸到那些只是连接至 SQL 节点的简单 mysql 或是客户端主机。

最近支持平台，请参考：

<http://www.mysql.com/support/supportedplatforms/cluster.html>

5.4 整个数据库都要求放在内存中吗？

MySQL 集群支持基于磁盘和基于内存的数据。但是，在当前的版本下基于磁盘存储还是有一些限制的。比如，没有经过索引的数据可以存于磁盘，但是索引过的列必须存于内存当中。数据库越大，就意味着你需要更多的内存和硬件。有关基于磁盘实现的更多的资料，请参见：

<http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-disk-data.html>

有一些基本的工具可以用于帮助你计算需要多少内存。首先，我们推荐您使用 `ndb_size.pl`（该工具是 MySQL Server 本身自带的）来计算将当前的数据库转换为 NDB 存储引擎时，需要多少的内存。请注意，该工具假设将当前的数据库全部转换为基于内存的数据库。更多此方面的信息，请参见：

<http://dev.mysql.com/doc/refman/5.1/en/mysql-cluster-utilities-ndb-size.html>

另一个替代的工具来自 <http://www.severalnines.com/sizer/index.php> 此工具由第三方提供，Sun Microsystems并不提供支持。

如果你是新设计数据库，可以使用以下的算式来得到每个数据节点需要的内存量：

$$\text{Data Size} * \text{Replicas} * 1.25 = \text{Total Memory}$$

举例: $2 \text{ GB} * 2 * 1.25 = 5 \text{ GB}$

$(\text{Data Size} * \text{Replicas} * 1.25) / \text{Nodes} = \text{每个数据节点需要的内存量}$

举例: $(2 \text{ GB} * 2 * 1.25) / 4 = 1.25 \text{ GB}$

5.5 应用程序可以使用完整的 JOIN 或是全表扫描吗?

如果使用的话, 除非你重写你的应用, 否则你极有可能遭遇极差的性能。因为在集群中数据是分区并分布于多个数据节点的。使用主键查找方法的应用程序可以从集群的分布式数据中获取最大的效益。

一个策略是考虑使用存储过程以简化 join 操作带来的影响。这可以减少应用程序的修改。以后的 MySQL 集群发布版本将在这方面改善性能。

5.6 数据库可以使用外键吗?

当前, NDB 存储引擎不支持外键。如果使用 MySQL 服务器访问数据节点, 那么, 这个不足可以通过使用触发器来弥补 (那些使用 NDB API 的客户端无法使用这一特性)。该方法, 请参见

<http://forge.mysql.com/wiki/ForeignKeySupport#Appendix A: Triggers implementing foreign key constraints>

5.7 支持应用程序可以全文检索吗?

当前, NDB 存储引擎不支持全文检索, 一个普遍的做法是将集群数据库复制到一个只能的 MyISAM 从机, 以提供全文检索。

5.8 NDB 引擎与其它存储引擎相比性能会更好吗?

在 MySQL 中执行 INSERT 比在 MEMORY 中花费的时间要多?

我们有一个巨大的 INNODB 表, 将其改为 MyISAM 只需要花费几秒钟的时间。而将其改为 NDB 表时, 花费的时间是前面的五倍, 在我们的希望中, 这应该是很快的。

我们针对 INNODB 和 NDB 执行相似的查询, INNODB 的反应往往是很快的, 而 NDB 则不是, 为什么?

在以上的三个场景中, 用户忘了一点, NDB 存储引擎架构是与其它的 MySQL 存储引擎不一样的。MySQL 集群是一个分布式的, 无共享的数据存储的架构。在其中, 大量的查询都是使用主键访问, 访问速度极快, 且是性能可扩展的。但是, 在 MySQL 节点和数据节点网络之间的通信将是有一定压力的。图 2 中你可以注意到为了满足查询, 对数据的检索需要多个数据节点的参与。

Query:

```
SELECT fname, lname  
FROM author  
WHERE authid BETWEEN 1 AND 3;
```

Result:

```
Albert Camus  
Ernest Hemingway  
Johan Goethe
```

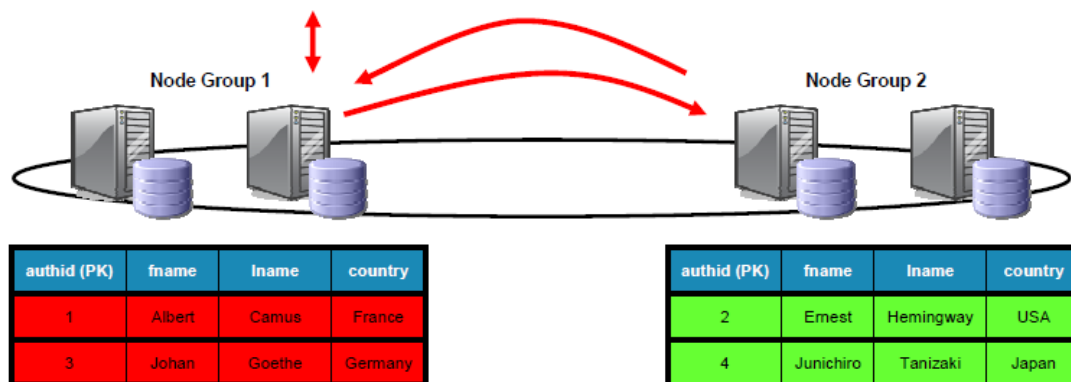



图 2

在图 3 中, 我们可以看到传统的存储引擎在执行查询的时候, 并没有网络间通信, 因为整个的库和表都在同一个主机上面。

Query:

SELECT fname, lname

FROM author

WHERE authid BETWEEN 1 AND 3;

Result:

Albert Camus

Ernest Hemingway

Johan Goethe

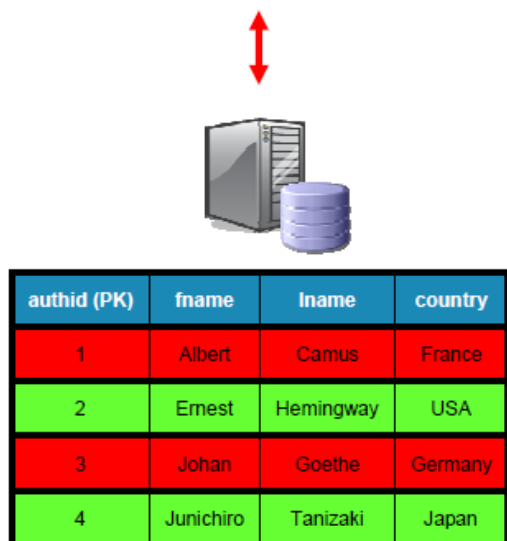


图 3

6 设置与评估指南

6.1 硬件

你有足够的计算机和资源用于这次的评估吗？

MySQL 推荐使用以下的硬件配置

数据节点 (Data Nodes)

- 双核或是双 CPU 机器为起点；从 MySQL Cluster 7.0 起，一个单一数据节点可以有有效的使用 8 核/CPU
- 足够大内存的 64 位机器以存储基于内存的数据集
- 大 CPU 缓存对于优化性能是非常重要的。
- 一个快速的磁盘系统，因为这会影响检查点和备份。

在集群中的所有的节点都应该运行于局域网内的机器上；在不同的地理上，使用集群复制可以达到地域的冗余。

MySQL 节点/应用节点 (MySQL Servers/Application Nodes)

- 大缓存 CPU
- RAM 的重要性并不和数据节点一样（一般 4GB 就可以）

另外，高宽带，低延迟的网络连接亦可提高集群的性能——通过使用 G 比特的以太网或是 SCI 连接。

性能的提高也可以通过把 MySQL 节点和数据节点放于同一个机器上，这样的话，就去除了网络的延迟。

请确保你的系统不会经常的进行交换 (swapping)。因为对于数据节点来说，这是非常糟糕的。根据以往经验，磁盘空间的大小至少要是 DataMemory 的 12 倍。这个空间被用于存放三个 Local Checkpoints (LCPs) 和 redo log. 如果需用备份或是使用基于磁盘的数据，那么，还需用分配额外的空间给磁盘。当进行 MySQL 集群评估时，最小的推荐配置是

- 两台计算机各自跑一个数据节点
- 两台计算机分别跑一个应用节点和管理节点

以上配置在包含 MySQL 集群所有进程的情况下提供了最小的冗余。

6.2 性能参数

要求每秒多少个事务？要求的反应时间是多少？

6.3 测试工具

哪些工具可以用以测试性能指标？

6.4 程序接口

针对严格要求性能的实时系统，强烈推荐考虑使用 API，比如 NDB API (C++ API)。

使用 NDB API 和使用 SQL 相比，具有的优点：

- 低延迟和低负载—无需解析或是优化查询
- 有利于应用节点和数据节点间的控制

- 批量处理插入，更新，读取，删除。针对一个或是多个表可以定义批量操作。
- 使用 NDB API，执行的速度是 SQL 的三倍，通常情况下是根据请求来的，也和是否使用批量处理有关。

需要严格要求性能的客户使用 API 实现他们的网络请求，而用 SQL 接口执行监控维护工作。

6.5 数据模型和查询设计

数据模型是专为 MySQL 集群开发的还是就是原先的？

用于评估的数据能够反映真实环境下的数据吗？

用户案例是真实的还是边缘测试？

良将的数据模型和良好的查询语句对于良好的性能是非常重要的。除非对网络延迟，数据运算法则和数据结构的检索有基本了解，否则评估是不会通过的。

对于数据库的设计者，这意味着选择正确的索引结构和访问方法是非常重要的。我们强烈推荐那些需要高性能的应用请求被设计成使用主键检索。这是因为在单一的数据节点中，在一个哈希表中检索索引数据比在树状的索引中检索快的多，设计数据模型的时候，考虑到这点是非常重要的。选择一个良好的主键定义也是非常重要的。

如果需用使用顺序索引，那么表应该被分为如下的效果：只有一个数据节点会被扫描到。

6.6 使用基于磁盘的表或是基于内存的表

如果你的数据量比 RAM 要大很多，那么你可以进行定义将表数据存储在磁盘上。

非索引列全部放置于磁盘，但是索引列从来都是放在内存的。数据库越大，一个表上的索引越多，就意味着你需要更多的内存或是主机。和大多数的基于磁盘的数据库系统类似，使用 LRU（最近最少使用）来缓存热页。当读取一个基于磁盘数据的记录时，检索是先在缓存中查看，该页是否存在。如果不存在，该记录就需要从磁盘中取。

在有些情况下，在缓存中的所有的脏页会被写回到表空间。表空间和可用于索引列的 RAM 决定了可以在基于磁盘的表上存储多少的数据。

这意味着，和传统的基于磁盘的数据库系统一样，MySQL 集群在性能方面同样也是有着限制的。缓存越大越好，因为可能存在着磁盘 I/O 瓶颈。对于那些需要大量随机访问的表，并且对性能和反应时间有较高要求的表最好设计成使用内存表，以避免磁盘 I/O 瓶颈。

6.7 用户定义分区

图 5 和图 6 当中，我们分别可以看到分布对应用不透明和对应用透明。比较两者可以发现使用后者可以带来内部通信量的大幅降低。

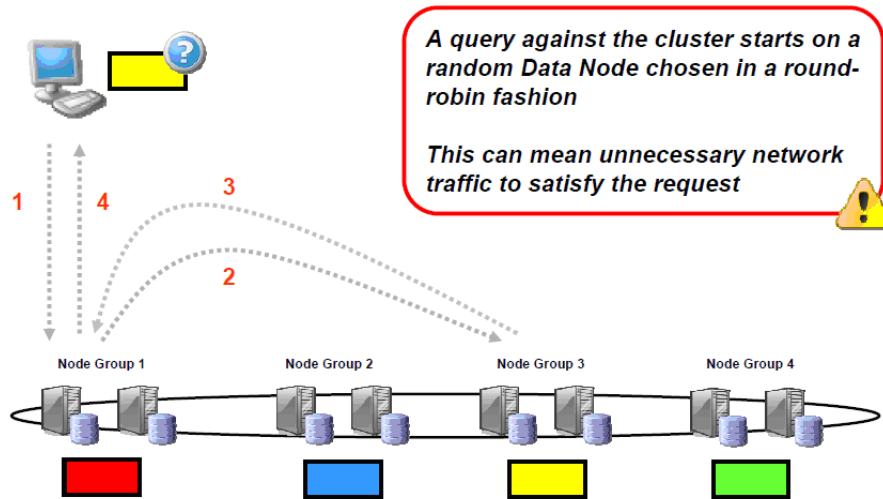


图 5

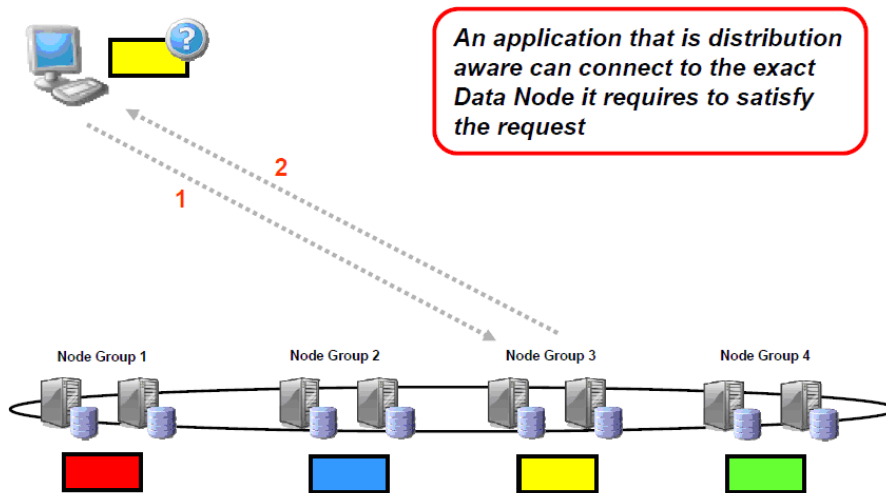


图 6

如果知道分区分布，并且加以应用，将会带来性能上的巨大提升，如图 7 所示，更多信息，请参见：
<http://www.mysql.com/why-mysql/benchmarks/CGE-Intel-Dolphin.html>

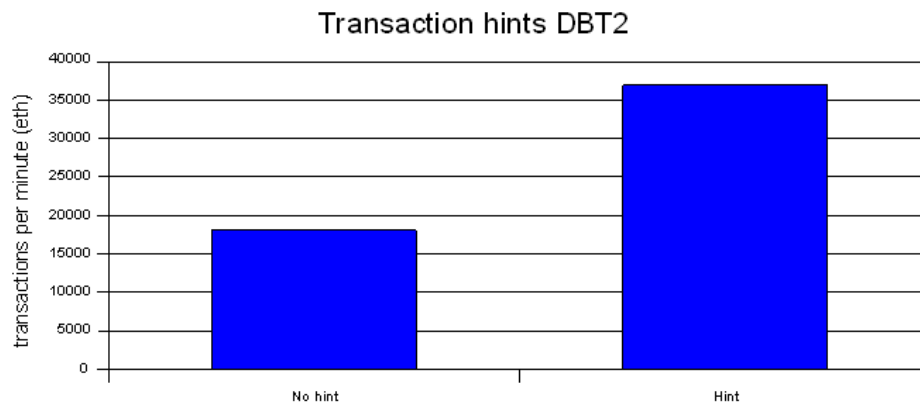


图 7

应用程序通过SQL节点访问数据库，如果是不知道分区的，那么会尽可能的使用

到主键，那么没有用到主键查询的语句利用 WHERE 表达式，来缩小查询到一个分区。应用程序使用的 NDB API，在 MySQL Cluster API Developer Guide 中介绍了怎样才能做到分布对应用不透明，请参见 <http://dev.mysql.com/doc/ndbapi/en/index.html>

6.8 应用程序平行访问与其它小技巧

之前已经提到，MySQL 集群的特性之一是平行数据存储。这意味着经常会有多于一个以上的数据节点（每个数据节点有多个线程）在平行的工作以满足应用程序的请求。

另外，MySQL 集群 7.0 也引入了多线程的数据节点，它可以在一个核上利用多达 8 个的线程，CPU 或是核。为了使用这个新的特性，数据节点应该使用新的 ndbmtl 二进制包，而不是使用 ndb，另外，在 config.ini 中要正确的进行设置。

使你的应用程序达到高性能的关键因素之一是尽可能利用这项特性。在 MySQL cluster 当中，平行化可以通过以下的几个方面获得：

- 添加更多的应用节点
- 使用多线程的数据节点
- 批量的请求
- 在连接至数据节点的不同的应用节点上平行的进行工作
- 使用 NDB API 的异步同步特性。

使用多少的线程和多少的应用以进行负载测试是由基准测试工具进行的。方法之一是一次只连接至一个应用节点，然后增加线程。当一个应用节点不能产生更多的负载的时候，再添加另一个应用节点。建议从两个数据节点开始，然后逐步添加数据节点。如果已经完成对应用查询的设计，并且数据模型也是根据在此白皮书中的最佳实践来的，那么可以预期，四个节点的情况下，四个节点的吞吐量是两个节点的两倍。

尽可能利用多线程，并且尽可能在 MySQL 节点上进行负载均衡。在 MySQL 电信级集群中，可以得到更多的性能的提升。因为它允许 MySQL 集群在多 CPU/核的系统下尽可能的利用多个 CPU。这些包括了：

- 一个 MySQL 节点连接至数据节点。降低了从 MySQL 节点连出的多个连接的锁竞争。（--ndb-cluster-connection-pool=X）
- 将该线程设置为实时优先级。
- 将数据节点锁定（内核线程，将该线程与 CPU 直接关联）
-

7 配置文件的注意事项

以下是 MySQL 集群的一个模版。多数的参数可以按初始默认进行设置，但是需用注意以下若干重要参数：

```
config.ini
[TCP DEFAULT]
SendBufferMemory=2M
```

```
ReceiveBufferMemory=1M
```

```
[NDB DEFAULT]
```

```
NoOfFragmentLogFiles= 6 * DataMemory (MB) / 64
```

以上只是一个启发式的例子，如果你知道每秒有多少的数据量被写入，即可计算出来这个准确的数值。

```
MaxNoOfExecutionThreads=8
```

该数据与该数据节点的核有关系。对于一个单核系统，这个值不是必须的。但是，对多 2 个核，设为 3，4 个核设为 4，8 个核设为 8。

```
RedoBuffer =32M
```

```
LockPagesInMainMemory=1
```

```
my.cnf
```

```
[mysqld]
```

```
ndb-use-exact-count=0
```

```
ndb-force-send=1
```

```
engine-condition-pushdown=1
```

8 正确性检查

8.1 做如下基本的测试以检测集群是否已经准备完毕

以下是一些基本的测试以确定 MySQL Cluster 已经完成可以操作了。

创建一个基本的表：

```
CREATE TABLE t1 (a integer, b char(20), primary key (a)) ENGINE=NDB;
```

插入一些数据：

```
INSERT INTO t1 VALUES (1, 'hello' );
```

```
INSERT INTO t1 VALUES (2, 'hello' );
```

```
INSERT INTO t1 VALUES (3, 'hello' );
```

```
INSERT INTO t1 VALUES (4, 'hello' );
```

从表中选取出数据：

```
SELECT * FROM t1;
```

从管理节点 (ndb_mgm) 上重启数据节点 (ndbd)

```
3 restart -n
```

检查节点是否已经启动，重启加入集群，所有的节点已经连接

```
show all status
```

从表中选取出数据：

```
SELECT * FROM t1;
```

在每个数据节点上执行以上的过程。

如果你不能完成以上的步骤，返回数据有问题或是停启数据节点时出现故障，那么请查看错误日志，网络和防火墙。