MySQL Cluster

sehr große und schnelle Datenbanken auf Aldi-PCs

PHP Conference | 9.11.2005 | Alex Aulbach

Der Referent

- Alexander Aulbach
  - 37 Jahre alt
  - Angestellt bei der Mayflower GmbH
  - Webapplikationsprogrammierung seit 9 Jahren
  - Datenbankentwicklung seit 7 Jahren

Mayflower GmbH

- Gegründet 1997
- München und Würzburg
- zahlreiche Projekte bei europäischen Firmen/Konzernen
  - Vaillant
  - Telefonica
  - HypoVereinsbank
- starkes Wachstum
- thinkPHP bündelt die PHP und LAMP Aktivitäten von MAYFLOWER
  - Core Developer PHP und Apache
  - Starkes Open Source Engagement
    - PHProjekt
    - Lighttpd
    - PHP-Support

Inhalt

- 17 Folien
- Cluster: Wozu und für wen?
- Begriffe
- Features
- Funktionsweise (möglichst einfach erklärt)
- NDB-Storage-Engine
- Vergleich mit Oracle-RAC
- Hardware-Anforderungen
- Einschränkungen
- Praktische Vorführung

Man will hoch hinaus...

Some look at MySQL's shared-nothing clustering as a viable enterprise-class feature. [Oracle CEO] Larry Ellison has said he'd be happy if 10 percent of customers adopted clustering. MySQL's [vice president of marketing] Zak Urlocker has said he's happy to pick up the other 90 percent.

(http://www.eweek.com/article2/0,1895,1605776,00.asp
Oracle VP: MySQL Cluster Not a Threat)

Schlüssel-Eigenschaften

- Hochverfügbarkeit
  durch parallele Server-Architektur 99,999% Verfügbarkeit (five-nines), also weniger als 5 Minuten Ausfall pro Jahr
- Dynamische Skalierbarkeit
  man erweitert den Cluster einfach indem man mehr Rechner hinstellt; NDB skaliert nahezu linear
- Hohe Performance
  100.000 Transaktionen pro Sekunde bei weniger als 5 ms Antwortzeit bei 4 CPUs. 380.000 Schreibetransaktionen/sek, 1,5 million Lesetransaktionen/sek (128 Byte records) bei 72 CPUs
- Niedrige Kosten
  billige Hardware benutzen (comodity-hardware)
- Es geht beim MySQL-Clustering in erster Linie um Geschwindigkeit und Verfügbarkeit bei niedrigen (Hardware-)Kosten.

Zielgruppen

- Bereits existierende MySQL-Nutzer
  Durchsatz bei eine geschäftskritischen Applikation reicht nicht mehr aus.
- Telekomunkations-Firmen
  Bisherige kommerzielle oder selbstgeschriebene Lösungen ersetzen.
- Regierungen
  Haushaltgünstige Lösungen für Regierungen/Städte, die Open-Source-Software einsetzen wollen.
- Unternehmen
  Jede Organisation, die Hochverfügbarkeit benötigt, um die Kosten eines Ausfalls zu reduzieren.
- Spielkinder/Entwickler
  Ohne dafür zu etwas vorab zu bezahlen, einen Cluster zum Laufen bringen und mal alle Stecker rausziehen um zu sehen, was dabei ausfällt.

Begriffe

- NDB: Netzwerk-Datenbank (network database)
- Nodes: Eine Node ist einfach ein Prozess auf einem Rechner.
- Es wird unterschieden zwischen der
  - MGM-Node (Management), der
  - Data-Node (die eigentliche Speicherung und Replikation, also dem eigentlichen Cluster) und der
  - SQL-Node (also dem mysqld)
- Cluster: In unserem Fall die Data-Nodes.
  Oder anders erklärt:
- Cluster: Physikalischer Speicher der NDB-Storage-Engine in MySQL.
- MySQL-Cluster: Kombination aus MySQL und NDB-Storage-Engine

Features

- Transaktionen
- Synchrone Replication
- Auto-sync beim Hochfahren einer NDB node
- Restore von Checkpunkt (falls Cluster cache an)
- Online Backup (ohne vorheriges runterfahren)
- nachvollziehen von Änderungen einer Zeile
- Es gibt zwei Indizes (Eindeutiger Hash und T-tree Sortierung)
- Indexerzeugung zur Laufzeit
- Softwareupdate zur Laufzeit

Wie funktioniert der MySQL Cluster?

- Hat nichts, aber wirklich gar nichts mit Replikation zu tun!
  (man kann natürlich einen Cluster replizieren, das macht sogar Sinn wenn man die Daten auf „langsamen“ Datenwegen nach außen replizieren will)
- In etwa gleiches Prinzip wie Raid

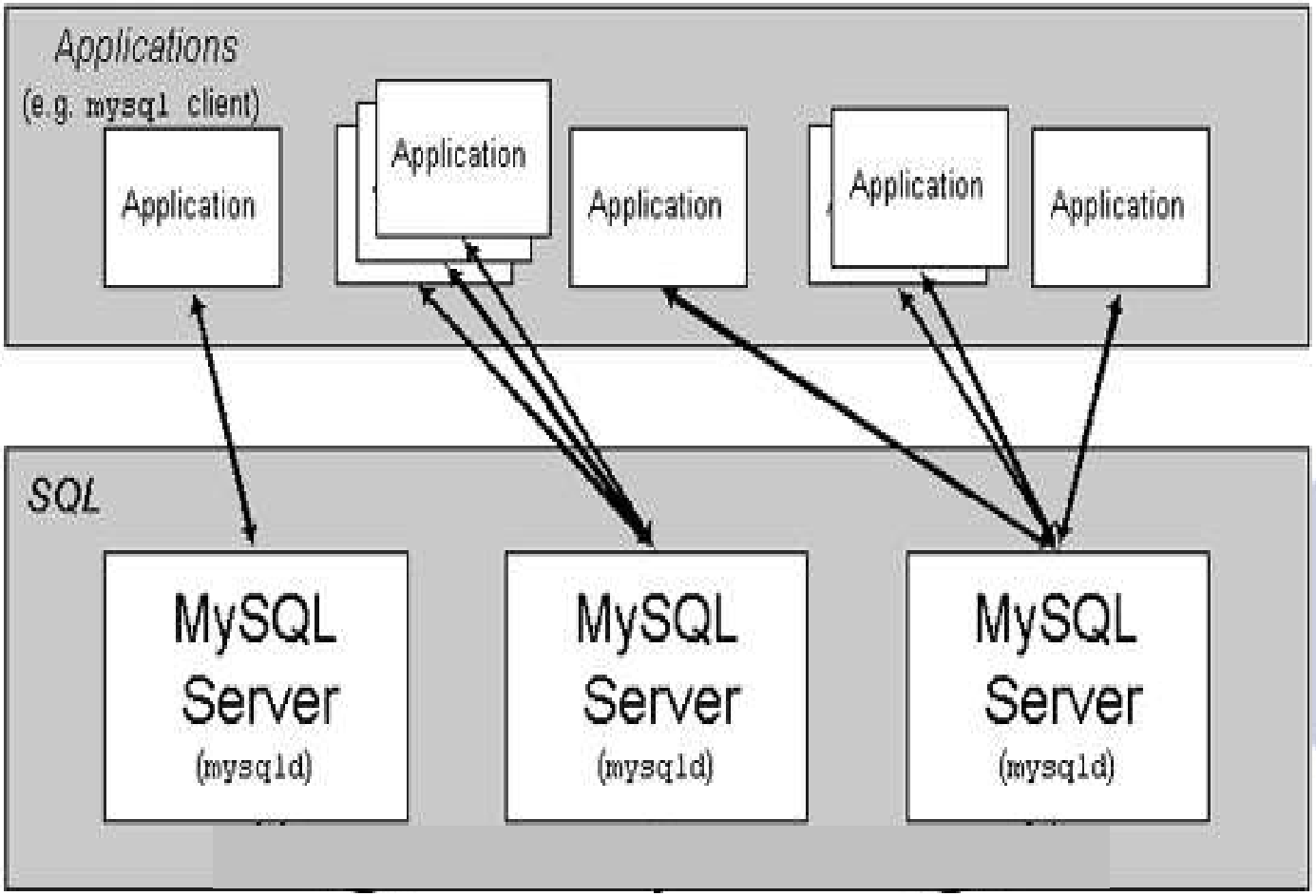
  Zur Erinnerung:

- Raid 0: Striping (Aneinanderhängen)
  mehrere physikalische Platten werden zu einer großen Platte zusammengefasst
- Raid 1: Mirroring (Spiegeln)
  gleiche Inhalte werden auf mehreren Festplatten verteilt, so dass der Ausfall einer Platte verkraftet werden kann

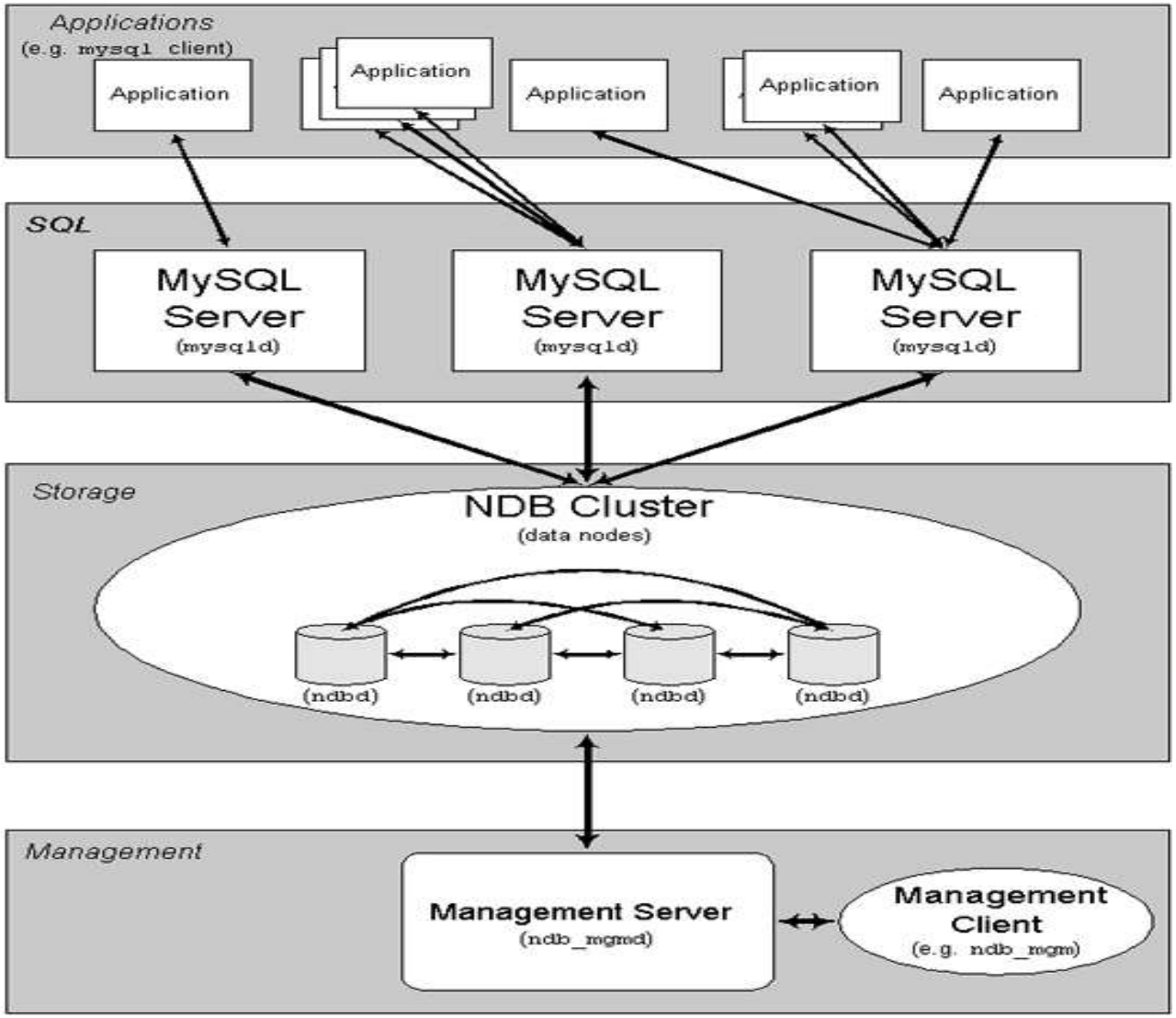
Analogie: MySQL-Cluster zu Raid

- MySQL-Cluster funktioniert nach den gleichen Prinzipien
- ersetze „Platte“ durch „Data-Node“ (oder auch durch „CPU-Prozess“)
- ersetze „Raid“ durch „NDB Cluster“
- Raid 0 oder Raid 1 wird nicht weiter unterschieden, der Zusammenhang ergibt sich stets von selbst, wenn mehrere Nodes auf unterschiedlichen Rechnern laufen.
- Es gibt noch einen „Oberchef“ (MGMD), der den Ausfall eines oder meherer Server managen kann und den data-Nodes entsprechende Befehle gibt.

„Herkömmliche“ Datenbankinstallation



MySQL Cluster



NDB-Storage-Engine

- In MySQL inzwischen „üblich“: Einfach eine neue Storage-engine.
- Die Engine verbindet sich „automatisch“ mit dem Cluster. (Wird über die MGM-Node konfiguriert).
- Queries an MySQL-Server die eine NDB-Tabelle betreffen, werden über NDB-Cluster abgearbeitet. Der MySQL-Server „leitet nur durch“.
- Möglich: NDB-Cluster als eigenständige Datenbank. MySQL nicht zwingend notwendig, aber praktisch.
- Cluster verwaltet seine Data-Nodes komplett selbst.
- MGM ist nach Start im Prinzip nicht mehr erforderlich.
- Aber handelt Ausfälle: wenn ein Teil des Clusters ausfällt oder sich nicht mehr gegenseitig sieht usw. (Über eine Art Watchdog)

Was geht (noch) nicht?

- Absolut sicher – es fehlt z.B. eine 100%ige Absicherung vor Datenverlust bei Ausfall aller Cluster gleichzeitig (Stromausfall).
- Noch nicht möglich für sehr große Datenbanken (hundert Gigabyte z.B.) – weil alles im RAM gehalten wird.
  (soll ab V5.1 gehen)

MySQL Cluster vs Oracle RAC?

- Oracle RAC benötigt entsprechende Hardware
- Investionen in SAN (storage area network, sauteueres Netzwerk)
- Spezialisten, die in der Lage sind, das zu bauen
- MySQL-Cluster ist nicht für die ganz speziellen Anforderungen bestimmter Kunden gedacht
- Produkt für den Massenmarkt
- NDB-Cluster ist bei Grundinstallation (MySQL ab V4.1) schon mit dabei (muss nur konfiguriert werden)
- “erst” 1,5 Jahre nach Einführung: Stabilität?!
- wird ständig weiterentwickelt
- Kunden sind oft nicht bereit mehr als notwendig auszugeben
- man kann guten Gewissens für 90% aller Fälle einen MySQL Cluster empfehlen (siehe erste Folie)

Anforderungen an die Hardware (1)

- Momentan wird alles in RAM gespeichert, das heißt man benötigt mindestens so viel RAM wie die Datenbank groß ist:

  *(SizeofDatabase * NumberOfReplicas * 1.1) / NumberOfDataNodes*

  Die genaue notwendige Größe zu berechnen ist ziemlich schwierig (der Primary Key wird zum Beispiel zusätzlich als Hash gespeichert usw.), daher sollte man hier lieber mehr RAM-Verbrauch kalkulieren.
- Damit der Cluster gut läuft sollte man schon fette Maschinen hinstellen, z.B. pro Maschine 2 x Xeon, 16 GB Ram, 4 x 73 GB Raid, Gigabit Ethernet
- Um den Durchsatz dann zu steigern kann man mehr RAM, mehr CPU's oder einfach mehr Maschinen aufbauen. (2, 4, 8, 16 usw.)

Anforderungen an die Hardware (2)

- Ordentlicher Cluster fängt bei 4 Data-Nodes an.
- Mindestens 2 Data-Nodes notwendig (vorher ist ein Cluster sinnlos) plus dritter Server (kann nebenbei irgendwo anders laufen, da kaum Ressourcen verbraucht werden) für die MGM-Node!
- Man benötigt also mindestens drei (3) Rechner, um einen Cluster sinnvoll (ausfallsicher) laufen zu lassen!
- 100 MBit Netzwerk ist absolute Mindestvorraussetzung. Besser GBit-Netzwerk, SCI (Scalable Coherent Interconnect) oder andere hochschnelle Verbindungen.
- Das Cluster-Netzwerk sollte möglichst auf seinen eigenen Leitungen laufen.

Planung?

- Ein Cluster ist nicht „mal eben schnell“ aufgesetzt!
  Auch nicht zum spielen (man sollte einen halben bis einen Tag dafür ansetzen, außer man will es partout mit nur einem Server machen, was prinzipiell auch geht, aber sinnlos ist).
  Gründliche Planung ist notwendig, vorher mal ausprobieren!
- Die Hardware muss harmonieren (Datendurchsatz).
- Backup-Konzept – notwendig? Wie wichtig sind die Daten?
- Was nutzt die der beste Cluster, wenn die Stromversorgung/Wärmeabfuhr Müll ist?
- Cluster-Konzept steht und fällt mit der Netzwerkgeschwindigkeit.

Weitere Einschränkungen

- Jede Tabelle benötigt Primary Key (evtl. automatisch erzeugt)
- FULLTEXT und Prefix-Indexe (noch) nicht indizierbar
- Alle Character-Sets und -Collations werden ab V5.0 unterstützt
- Spatial Extensions (also z.B. geometrie-Datentypen) werden nicht unterstützt
- keine partiellen Rollbacks
- Maximal 128 Attribute pro Tabellenzeile; Attributnamen nicht länger als 31 Zeichen; maximale Länge von Datanbankname + Tabellenname 122 Zeichen; maximal 1792 Tabellen pro Cluster-DB
- Maximale Größe einer Tabellenzeile ist 8 KiB (ohne BLOBs).
- Keine Foreign-Keys
- Kein Query-Caching (logisch)
- Tabellen werden nur in fester Länge gespeichert

Andere Limitierungen

- Da alles sequentiell abgearbeitet wird, sind Suchen nach Bereichen (z.B. BETWEEN) langsam
- Der Query-Optimizer arbeitet noch nicht, weil „records in range“ noch nicht funktioniert. Stattdessen mit „USE/FORCE INDEX“ arbeiten.
- Alle NDB-Nodes (Maschinen) müssen die gleiche Architektur haben (BIG/LITTLE-Endian!)
- Cluster muss bei allen Änderungen der Nodes restartet werden (Online adding/dropping geht nicht)

Vorführung

- Test-Setup
  - 3 virtuelle Linux-Server (VM-Ware)
  - Einer als MGM-node und zwei als data-node und gleichzeitig MySQL-Node
- Konfigurationsfiles
- Starten der Dienste (MGMD, data-Nodes, Datenbanken)
- Runter- und wieder hochfahren einer data-Node
- Radikales Abschalten einer Node und Wiedereinbinden in Cluster
- Keine Performance-Tests (bringt in dieser speziellen Konfiguration nix, weil VM-Ware greift ja trotz Cluster auf gleiche physikalische Hardware zu)
- Falls Zeit: Einbinden des vierten Servers als eigenen MySQL-Cluster (abschalten der beiden anderen)
- „rumspielen“, falls Zeit

think php

by MAYFLOWER

Vielen Dank für Ihre Aufmerksamkeit

Alex Aulbach

Mayflower GmbH
Pleichertorstr. 2
97070 Würzburg

+49 (931) 35 9 65 - 23

aulbach@mayflower.de