MySQL Referenzarchitekturen für hoch skalierbare Webinfrastrukturen

Mario Beck
Oracle
Berlin

Schlüsselworte
MySQL Referenzarchitektur MySQL-Cluster Hochverfügbarkeit Skalierbarkeit

Einleitung
Weltweit suchen Unternehmen ständig nach neuen Methoden, um die Möglichkeiten des Internets für mehr Geschäftswachstum zu nutzen. Diese Unternehmen wissen, dass sie ihre Technologienutzung optimieren und optimale Vorgehensweisen implementieren müssen, um sich gegenüber Mitbewerbern zu behaupten. Der Erfolg neuer webbasierter Initiativen hängt in hohem Maße davon ab, ob von Anfang an die richtigen Architekturen und Technologien ausgewählt werden. Nur so lassen sich langwierige Versuch- und Irrtum-Phasen vermeiden. Neun der zehn meistbesuchten Webpräsenzen basieren auf MySQL (einschließlich Google, Facebook und YouTube). Aus diesen Erfahrungen wurden datenbankgestützte Webarchitekturen entworfen, die bei minimalen Kosten und Risiken sowie bei minimaler Komplexität eine maximale Skalierbarkeit und Verfügbarkeit bieten. In diesem Vortrag werden vier Referenzarchitekturen vorgestellt, deren Grundlage optimale Vorgehensweisen sind, die bei der Zusammenarbeit mit den Betreibern der erfolgreichsten Websites der Welt entwickelt wurden.

Überlegungen zur Hochverfügbarkeit
Ein zentraler Punkt beim Entwurf der einzelnen hier beschriebenen Referenz-architekturen ist die Hochverfügbarkeitsstrategie. Wenngleich es wünschenswert
ist, sämtliche Architekturen für 100 % kontinuierliche Verfügbarkeit zu entwerfen, ist dieser Ansatz schlichtweg nicht praktikabel.

Wie in der folgenden Abbildung gezeigt, werden Ausfallzeiten durch höhere Verfügbarkeitsstufen erheblich gesenkt. Zu diesem Zweck werden Systeme mit höheren Redundanz- und Fehlertoleranzstufen bereitgestellt.

Da für höhere Redundanzstufen jedoch zusätzliche Hardware- und Software-komponenten bereitgestellt werden müssen, steigen auch die Gesamtkosten des Systems. Darüber hinaus sind für die Bereitstellung und Verwaltung der komplexeren Umgebungen größere Investitionen im Hinblick auf IT-Mitarbeiter, Prozesse und Dienste erforderlich.

Wie in der Abbildung unten gezeigt, bieten diese Architekturen unterschiedliche Verfügbarkeitsstufen. Da eine höhere Verfügbarkeit potenziell mit höheren Kosten und mehr Komplexität einhergeht, müssen diese Faktoren gegeneinander abgewägt werden. Die Bereitstellung einer hochverfügbaren Architektur allein ist noch keine Garantie dafür, dass tatsächlich Hochverfügbarkeit geboten wird. Es ist durchaus möglich, dass man mit einem Shared-Nothing-Cluster, der nicht ordnungsgemäß implementiert und verwaltet wird, eine geringere Verfügbarkeitsstufe erzielt als mit einer einfachen Datenreplikationslösung.

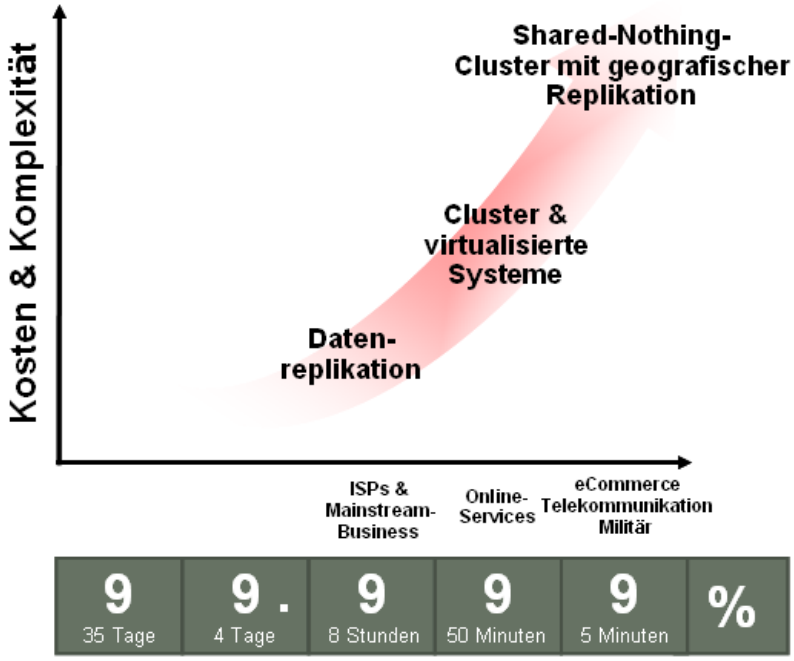

Abb. 1: Architekturen für Hochverfügbarkeit

Kleine Referenzwebarchitektur

Die nachfolgende Abbildung zeigt die empfohlene Topologie für „geringe“ Arbeitslasten.

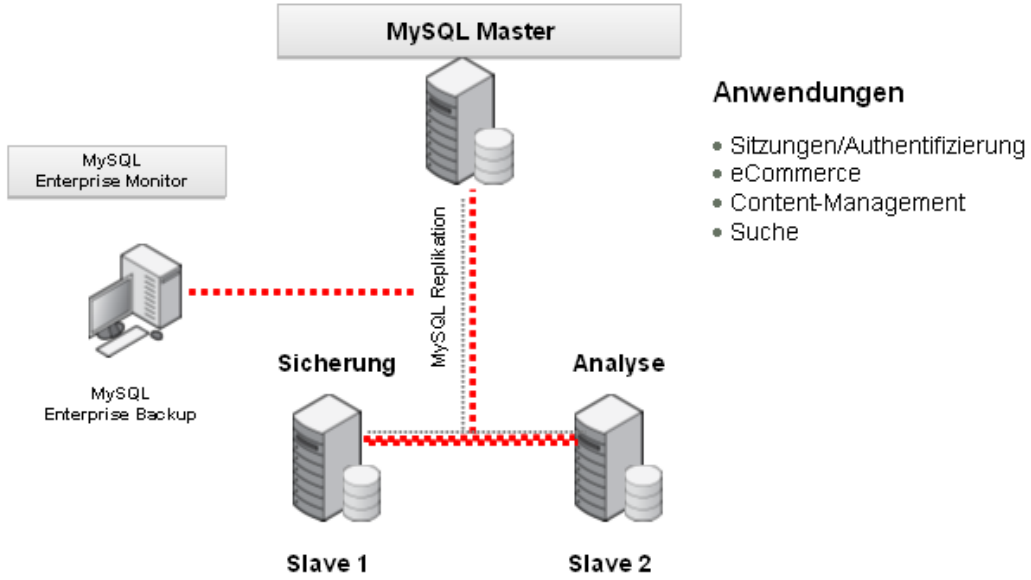

Abb. 2: Topologie für kleine Referenzarchitektur

In dieser Topologie wird ein einzelner MySQL Master-Server für sämtliche Anwendungen (einschließlich Session Management, eCommerce, Content-Management und Suche) bereitgestellt.

Um sicherzustellen, dass der MySQL Master ausreichend Ressourcen für die Webanwendungen nutzen kann, wird die Datenbank auf zwei Slaves repliziert. Ein Slave wird für Sicherungen, der andere für Analysefunktionen eingesetzt (Hinweis: MySQL Enterprise Backup kann Sicherungen Ihrer Datenbanken im laufenden Betrieb durchführen. Für sämtliche InnoDB-Daten können im laufenden Betrieb vollständige Sicherungen ausgeführt werden, ohne dass Abfragen oder Updates unterbrochen werden. Daher ist es bei Einsatz von MySQL Enterprise Backup nicht notwendig, einen dedizierten Slave zu verwenden).

Da die Replikation eine Standardfunktion der MySQL Datenbank ist, kann direkt bei der Installation eine Replikationsgruppe aus MySQL Master-/Slave-Servern bereitgestellt werden.

Die MySQL InnoDB-Speicher-Engine eignet sich für sämtliche Anwendungen, die Teil der kleinen Referenzwebarchitektur sind. Seit der Einführung von MySQL 5.5 ist InnoDB die standardmäßige Speicher-Engine. InnoDB bietet volle ACID-Konformität und nach Systemabstürzen schnelle, zuverlässige Wiederherstellungen ohne Verlust von Daten, für die bereits ein Commit ausgeführt wurde. InnoDB unterstützt Anwendungen mit einer großen Anzahl von gleichzeitigen Transaktionen, mit zeilenbasierten Sperren und MVCC-Unterstützung (Multi-Version Concurrency Control) sowie die Implementierung von Fremdschlüsseln und Constraints.

Wenn eine umfassende Erweiterung erwartet wird, sollten Benutzer jedoch mit der „mittelgroßen“ Konfiguration beginnen, die weiter unten beschrieben wird. Diese Konfiguration bietet eine höhere Kapazität, mehr Flexibilität bei sich ändernden Unternehmensanforderungen sowie bessere Verfügbarkeit.

Mittelgroße Referenzwebarchitektur
Die nachfolgende Abbildung zeigt die empfohlene Topologie für „mittelgroße“ Arbeitslasten.

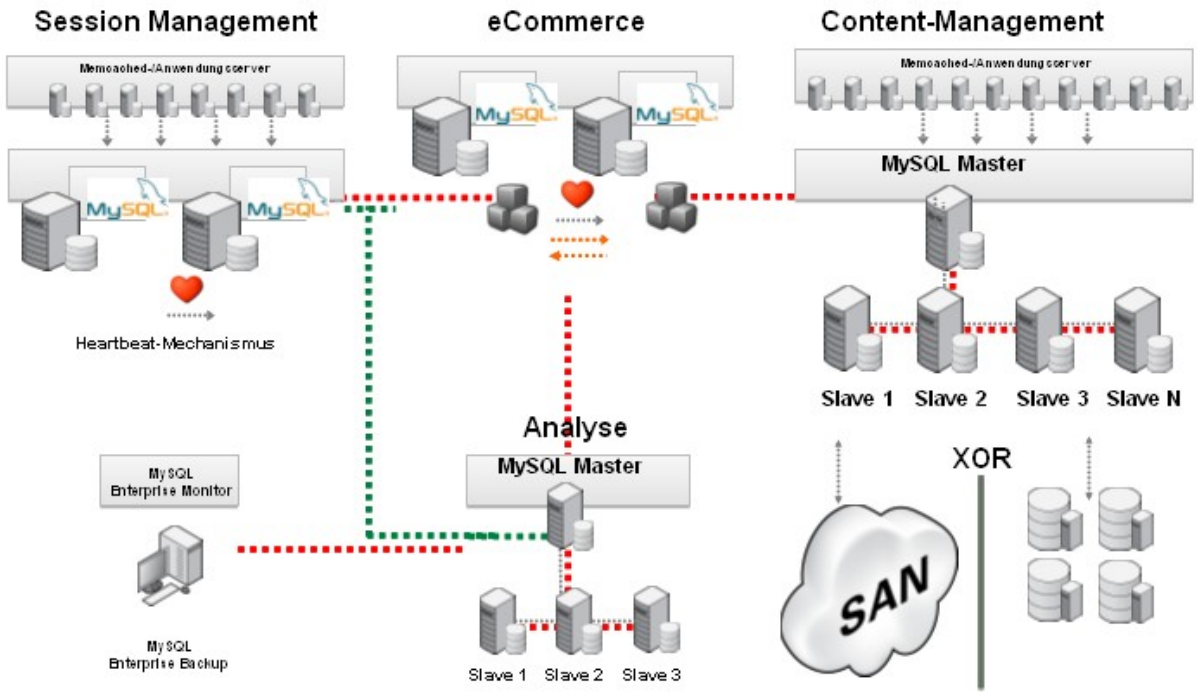

Abb. 3: Topologie für mittelgroße Webreferenzarchitektur

Im Vergleich zur oben beschriebenen kleinen Referenzwebarchitektur werden die Hauptfunktionen (Session Management, eCommerce, Content-Management und Analysefunktionen) in der mittelgroßen Topologie in separaten Server- und Speicher-infrastrukturen platziert, sodass diese unabhängig voneinander bereitgestellt, verwaltet und skaliert werden können.

Wenn ein Benutzer davon ausgeht, dass sein Webdienst im Laufe der Zeit skaliert wird, um die oben definierten Lasten zu unterstützen, sollte anfänglich diese mittelgroße Topologie bereitgestellt werden. Mit diesem Ansatz kann die Architektur bei steigenden Arbeitslasten und Unternehmensanforderungen einfacher erweitert und verwaltet werden.

Um die richtige Anzahl von Anwendungsservern für MySQL Server zu ermitteln, gilt folgende Faustregel: Jeder MySQL Server unterstützt ca. 8 Anwendungsserver, und bei der horizontalen Skalierung der Anwendungsserver in einer leseintensiven Umgebung werden weitere Slaves hinzugefügt. Für PHP-Anwendungen werden üblicherweise mehr Anwendungsserver benötigt, für Java-Anwendungen weniger.

Bei steigender Last der Verbindungen zu den MySQL Servern steigt auch die Notwendigkeit einer horizontalen Skalierung der einzelnen Komponenten innerhalb der Infrastruktur. Indem die MySQL Master-/Slave-Server für die einzelnen Hauptfunktionen unabhängig voneinander repliziert werden, können Entwickler und Datenbankadministratoren ihre MySQL Infrastruktur bei steigenden Anforderungen flexibler anpassen und besser steuern.

Für Session Management und E-Commerce wird die standardmäßige InnoDB-Speicher-Engine verwendet, um Transaktionsunterstützung und eine Wiederherstellungsfunktion bei Abstürzen zu bieten. Zur Bereitstellung von Hochverfügbarkeit verwenden beide Arbeitslasten zudem Linux Heartbeat mit semi-synchroner MySQL Replikation oder betriebssystembasierte Lösungen wie DRBD sowie MySQL Enterprise Backup.

Bei typischen Websites wird der Sitzungsstatus üblicherweise für eine Dauer von 45 bis 60 Minuten in einer dedizierten Datenbankpartition gespeichert. Um die betroffene Tabelle zum Löschen älterer Sitzungsdaten zu verwerfen, werden Rolling-Partitionen verwendet.

Da die Content-Management-Anwendung ein wichtiges Element des Webdienstes ist, ist die Skalierbarkeit dieser Anwendung unabdingbar. Um Skalierbarkeit für Lesevorgänge zu bieten, wird für jeden MySQL Master die MySQL Replikation eingesetzt und die Daten werden an typischerweise bis zu 30 Slaves repliziert. Bei einer normalen Content-Management-Arbeitslast sollte jeder Slave bis zu 3.000 gleichzeitige Benutzer unterstützen können.

Große Referenzwebarchitektur

Die nachfolgenden Abbildung zeigt eine Detailansicht der empfohlenen Topologie zur Unterstützung der „großen“ Arbeitslast.

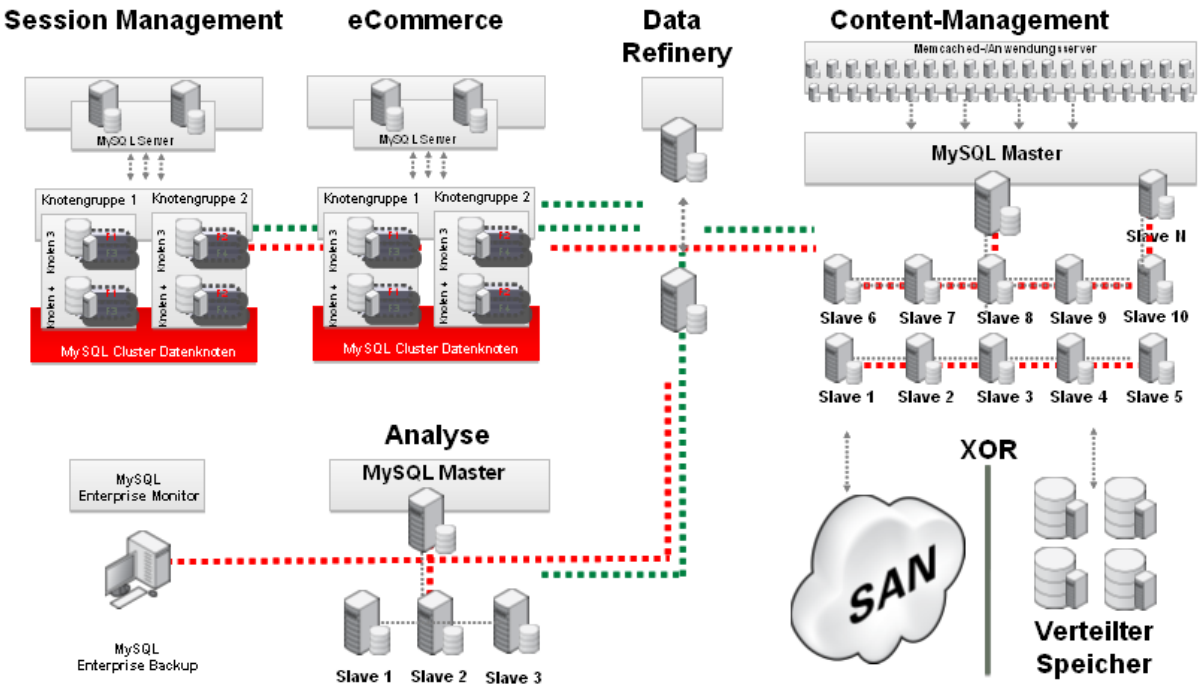

Abb. 4: Detailansicht – Große Referenzarchitektur

Zusätzlich zu den vorhandenen Arbeitslasten wird für umfangreichere Analysevorgänge und höhere Content-Management-Anforderungen eine Datenvorverarbeitung hinzugefügt.

Wie bei der mittelgroßen Referenzarchitektur werden Session Management- und eCommerce-Daten für Analysezwecke verwendet. In diesem Fall werden diese Daten jedoch zunächst in die Datenvorverarbeitung repliziert. Die Datenvorverarbeitung fasst Daten aus anderen Bereichen der Webinfrastruktur zusammen. Dazu zählt u. a. das Content-Management-System, in dem Daten bereinigt werden. Anschließend werden vollständige Datensätze und Data Warehouse-Dimensionen erstellt, die in die Analysedatenbank geladen werden. Die Data Refinery kann zudem zum Bereinigen von Daten verwendet werden, bevor diese in das Content-Management-System geladen werden.

Um die höheren Anforderungen im Hinblick auf Leistung und Verfügbarkeit zu erfüllen, die mit „großen“ Webarbeitslasten einhergehen, werden die Session Management- und eCommerce-Datenbanken auf MySQL Cluster bereitgestellt. Mit vier Datenknoten können 6.000 Sitzungen (Seitenabrufe) pro Sekunde unterstützt werden, wobei jeder Seitenabruf 8 bis 12 Datenbankoperationen generiert. Durch die Möglichkeiten zur horizontalen Skalierung von MySQL Cluster können die Session-Management- und eCommerce-Datenbanken in einem größeren Cluster kombiniert werden.

Sehr große Referenzarchitektur (Soziale Netzwerke)

Die Abbildung unten zeigt die empfohlene Topologie für die zu erwartenden Arbeitslasten von großen Sozialen Netzwerken.

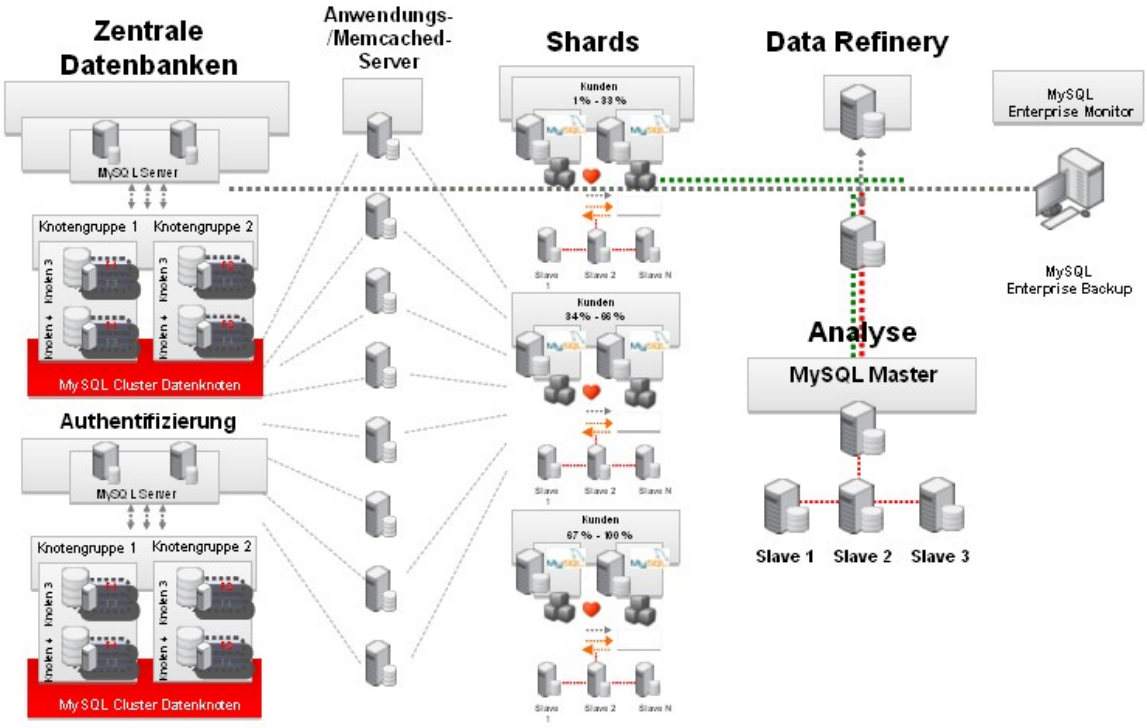

Abb. 5: Topologie der Referenzarchitektur für Soziale Netzwerke

In der Referenzarchitektur für Soziale Netzwerke werden viele Konzepte wiederverwendet, die in der mittelgroßen und großen Referenzarchitektur definiert wurden. Dies umfasst u. a. dedizierte Hardware für verschiedene Arbeitslasten innerhalb der Infrastruktur, den Einsatz von Memcached zur Senkung der Datenbankserverlast sowie ein Data Refinery (Datenvorverarbeitung), um Daten zu bereinigen, zu aggregieren usw.

MySQL Cluster wird zur Authentifizierung von Benutzern und zur Bereitstellung des Suchmechanismus eingesetzt, der von Anwendungen zum Weiterleiten von Lese- und Schreibvorgängen an die richtigen Partitionen oder Shards der Benutzerdaten-bank verwendet wird, wenn für die Suchvorgänge mehrere Schlüssel genutzt werden.

Sharding wird üblicherweise bei großen Websites verwendet, um die Skalierbarkeit von Schreibvorgängen innerhalb der Datenbank zu verbessern. In Web 2.0-Umgebungen mit hohem Datenverkehrsaufkommen (z. B. in Sozialen Netzwerk-Seiten) wird ein Großteil der Inhalte von den Benutzern selbst generiert. Folglich ist für Schreibvorgänge eine hohe Skalierbarkeit erforderlich, um die sich schnell ändernden Daten verarbeiten zu können. Es ist wichtig zu wissen, dass auch in Web 2.0-Umgebungen mehr Lese- als Schreibvorgänge ausgeführt werden, da die einzelnen Datensätze vor einer Aktualisierung typischerweise gelesen werden.

Darüber hinaus muss betont werden, dass Webanwendungen für mehrere Millionen von Benutzern bereitgestellt werden können, ohne dass ein Datenbank-Sharding vonnöten ist.

Beim Sharding – auch als Anwendungspartitionierung bezeichnet – wird die Datenbank von der Anwendung in kleinere Datensätze unterteilt und auf mehrere Server verteilt. Da bei steigenden Kapazitätsanforderungen kostengünstige Standardserver bereitgestellt werden können, ermöglicht dieser Ansatz eine äußerst kosteneffektive Skalierbarkeit. Darüber hinaus wirkt sich die Verarbeitung von Abfragen lediglich auf eine kleinere Untermenge der Daten aus, sodass die Leistung optimiert

wird.

Anwendungen müssen „shard-fähig“ sein, sodass Schreibvorgänge an die richtigen Shards weitergeleitet werden können und die Anzahl von JOIN-Operationen minimiert wird. Bei Verwendung optimaler Vorgehensweisen bietet Sharding eine effektive Möglichkeit zur Skalierung relationaler Datenbanken, die webbasierte, schreibintensive Arbeitslasten unterstützen.

Die effektivste Methode für das Daten-Sharding ist das Hashing einer einzelnen Spalte (Schlüssel). In der Referenzarchitektur für Soziale Netzwerke wurde die Benutzerdatenbank nach Benutzer-ID in Shards unterteilt. Shard 1 verarbeitet das erste Drittel der Benutzer, Shard 2 das zweite Drittel, und die übrigen Benutzer werden in Shard 3 gespeichert.

Dieser Ansatz ist anfänglich der logischste Sharding-Mechanismus und bietet beim Erweitern des Dienstes eine einfache horizontale Skalierung. Es ist jedoch wahrscheinlich, dass einige Shards über eine größere Anzahl aktiver Benutzer verfügen als andere Shards, sodass in der Zukunft möglicherweise eine Neuverteilung der Benutzer auf die Shards erforderlich ist. Sowohl die Anwendung als auch die Datenbank müssen die erforderliche Flexibilität bieten, um Änderungen an der Architektur durchführen zu können.

In der Referenzarchitektur wird DRBD mit Heartbeat eingesetzt, um Hochverfügbarkeit für die einzelnen Shards zu bieten. Zudem wird jeder Master-Server auf Slaves repliziert, um Skalierbarkeit für Lesevorgänge zu bieten. Im Gegensatz zu den anderen Referenzwebarchitekturen wird der Sitzungsstatus in der Architektur für Soziale Netzwerke in den Shards selbst verwaltet.

Da das Sharding auf Anwendungsebene implementiert wird, müssen bei Anwendungs- und Datenbankentwurf optimale Vorgehensweisen genutzt werden.

Durch die Bereitstellung von MySQL mit Sharding und Replikation konnten führende Soziale Netzwerke wie Facebook, Flickr, LinkedIn und Twitter ihre exponentiell steigenden Anforderungen im Hinblick auf die Verwaltung relationaler Daten erfüllen.

Der ideale MySQL Server

Bei der Größenbestimmung des Servers muss ein Profil mit der Größe und den Merkmalen der Datenbank erstellt werden, die ausgeführt werden soll. Gleichzeitig müssen die geschätzten Größenanforderungen bei Erweiterung der Arbeitslast in der Zukunft bestimmt werden. Bedenken Sie dabei, dass die für Skalierbarkeit von Lesevorgängen eingesetzten Slave-Server eine vergleichbar hohe Leistung bieten sollten wie Server, die als Master bereitgestellt werden. Folgende Spezifikationen werden empfohlen:

- 8 bis 16 x86-64-Bit-CPU-Cores (MySQL 5.5 und höher).
- 4 bis 8 x86-64-Bit-CPU-Cores (MySQL 5.1 und niedriger).
- 3- bis 10-mal mehr Arbeitsspeicher als aktive Daten.
- Linux-, Solaris- oder Windows-Betriebssystem.
- Mindestens 4 Festplattenlaufwerke. Bei Verwendung von 8 bis 16 Festplatten steigt die Leistung für E/A-intensive Anwendungen.
- Hardware-RAID mit batterieunterstütztem Cache.
  RAID 10 empfohlen. RAID 5 ist bei leseintensiven Arbeitslasten geeignet.

- 2 Netzwerkkarten und 2 Netzteile für Redundanz.

Aufgrund der verteilten Architektur von MySQL Cluster weichen die empfohlenen Serverkonfigurationen etwas von den Empfehlungen für einen MySQL Server ab, auf dem die MyISAM- oder InnoDB-Speicher-Engine ausgeführt wird.

Empfehlungen für Anwendungsknoten (MySQL Server):

- 4 bis 16 x86-64-Bit-CPU-Cores
- Mindestens 4 GB Arbeitsspeicher. Der Arbeitsspeicher ist auf Anwendungsebene von geringerer Bedeutung, und die Anforderungen hängen von den verfügbaren Verbindungen und Puffern ab.
- 2 Netzwerkkarten und 2 Netzteile für Redundanz.

Empfehlungen für Datenknoten:

- 8 x86-64-Bit-CPU-Cores. Verwenden Sie eine möglichst hohe Frequenz, um eine schnellere Verarbeitung von Nachrichten zu ermöglichen.
- Arbeitsspeicher pro Server = Datenbankgröße x Anzahl der Repliken x 1,25 (die erforderliche Gesamtgröße des Arbeitsspeichers ist durch Datenredundanz + Indizes bestimmt) / Anzahl an Datenknoten.

Beispiel: 10 GB Datenbank x 2 Repliken x 1,25 / 2 Datenknoten = 12,5 GB RAM pro Datenknoten.

- Linux-, Solaris- oder Windows-Betriebssysteme.
- 2 Netzwerkkarten und 2 Netzteile für Redundanz.

Die Daten- und Anwendungsknoten sollten in einem dedizierten Netzwerk (IP-Adressen ab 10.0.1.0) und unter Verwendung einer 1-Gigabit-Ethernet-Verbindung für den Transport (Mindestanforderung) bereitgestellt werden. Die MySQL Server werden dabei zusätzlich mit dem öffentlichen Netzwerk verbunden.

Um Ausfallsicherheit für das Netzwerk zu bieten, sollte jeder Server mit gebündelten und redundanten Netzwerkkarten konfiguriert werden, die mit redundanten Switches verbunden sind. Bei Bereitstellung mehrerer Datenknoten auf einem einzelnen Host sollten mindestens vier Netzwerkkarten gebündelt werden, um die höheren Anforderungen an die Netzwerkbandbreite zu erfüllen.

Zusammenfassung

Das Entwerfen, Verwalten und Skalieren webbasierter Infrastrukturen stellt eine große Herausforderung dar. Innovative Technologien werden heute schneller eingeführt als je zuvor, und mit noch größerer Geschwindigkeit ändern sich auch die Anwendungsanforderungen.

Das explosionsartige Wachstum der Daten, die verwaltet werden müssen, und die große Nachfrage nach neuen Diensten stellen erhebliche Hindernisse für Organisationen dar, die das Internet für mehr Geschäftswachstum nutzen möchten.

Wir hoffen, dass die in diesem Dokument beschriebenen optimalen Vorgehensweisen Ihnen einen

Ausgangspunkt bieten, um eine geeignete Architektur zur Bereitstellung Ihrer innovativen Webanwendungen zu erstellen.

Kontaktadresse:
Mario Beck
Oracle Deutschland
Komturstraße 18a
D-12099 Berlin

Telefon:	+49 (0) 30-74 70 96 - 879
E-Mail	mario.beck@oracle.com
Internet:	www.mysql.de