

INTERSYSTEMS CACHÉ
ALS ALTERNATIVE ZU IN-MEMORY-DATENBANKEN

INTERSYSTEMS CACHÉ

ALS ALTERNATIVE ZU IN-MEMORY-DATENBANKEN

Einleitung

Um die Performanz-Einschränkungen herkömmlicher relationaler Datenbanken zu überwinden, nutzen Applikationen, unabhängig davon, ob sie nun auf einem einzelnen System oder in großen, interkontinentalen Grids laufen, oftmals In-Memory-Datenbanken zur Zugriffsbeschleunigung. In-Memory-Datenbanken und Caching-Lösungen beschleunigen den Datendurchsatz, bringen aber ihre eigenen Einschränkungen, darunter mangelnde Eignung für sehr große Datenmengen, hohe Hardwareanforderungen und begrenzte Skalierbarkeit, mit.

InterSystems Caché® ist eine hochperformante Objektdatenbank mit einer einzigartigen Architektur, die sie für Applikationen interessant macht, die normalerweise eine In-Memory-Datenbank verwenden würden. Cachés Performanz entspricht der einer In-Memory-Datenbank, bietet aber zudem:

- Persistenz – Daten gehen im Fall des Abschaltens oder eines Absturzes nicht verloren
- schnellen Zugriff auf sehr große Datenvolumen
- eine Skalierbarkeit, die Hunderte Computer und Tausende Anwender erlaubt
- simultanen Datenzugriff per SQL und als Objekt: Java, C++, .NET usw.

Dieses White Paper erläutert, warum Caché für Unternehmen, die Hochgeschwindigkeitszugriff auf sehr große Datenvolumen benötigen, eine attraktive Alternative zu In-Memory-Datenbanken darstellt.

Eine einzigartige Daten-Engine ermöglicht Persistenz und höchste Performanz

Caché ist eine persistente Datenbank, was sich darin äußert, dass der Inhalt des RAM durch im Hintergrund ablaufende Prozesse auf Festplatten geschrieben wird. Wie gelingt es Caché also, eine Performanz zu bieten, die der von In-Memory-Datenbanken entspricht, die nur gelegentlich Daten in Permanentspeichern ablegen?

Ein Teil der Antwort liegt in Cachés einzigartiger Architektur. Anstelle der Spalten und Zeilen herkömmlicher Datenbanken nutzt Caché multidimensionale Datenfelder, deren Struktur durch die Definition der Objekte bestimmt wird. Daten werden in der vom Architekten festgelegten Form gespeichert und dieselben Strukturen, die der In-Memory-Cache nutzt, finden sich auch auf der Festplatte. Daten, die zusammen gespeichert werden sollten, werden zusammen gespeichert. Das Ergebnis ist, dass Caché sehr schnell auf Festplattendaten zugreifen kann.

Ein weiteres Element, das die Performanz vieler verteilter Caching-Lösungen mindert, ist die Notwendigkeit, die multiplen In-Memory-Caches bei jeder Aktualisierung der Daten zu synchronisieren. In Caché ist die Aktualisierung der Daten von der Verteilung der Daten an die Caches logisch getrennt. Dies führt zu einem deutlich einfacheren Arbeitsablauf und dadurch zu überlegener Performanz.

Caché bietet zudem „In-Process-Bindings“ für C++ und Java, die es Applikationen, die in diesen Sprachen entwickelt wurden, erlauben, Cachés interne Datenstruktur direkt zu bestücken.

Die Vorteile der Persistenz

Angesichts der Tatsache, dass Caché eine vergleichbare Performanz erreicht, bietet seine Fähigkeit, auf Festplattendaten zuzugreifen, im Vergleich zu In-Memory-Datenbanken einige signifikante Vorteile. Der offensichtlichste ist die Tatsache, dass die Notwendigkeit für einen separaten Permanentenspeicher entfällt. Caché ist dieser Permanentenspeicher und gleichzeitig jederzeit auf dem aktuellsten Stand. Daten gehen beim Abschalten oder Absturz des Systems nicht verloren.

Ein weiterer Vorteil von Caché besteht darin, dass die Größe eines Datensatzes nicht mehr von der Menge des verfügbaren RAMs abhängig ist. Falls Informationen nicht im lokalen Cache vorliegen, werden sie entweder aus einem Remote-Cache oder nahtlos von der Festplatte bezogen. Da es keine Beschränkung durch den RAM mehr kennt, ist ein Caché-basiertes System in der Lage, Petabytes an Daten zu handhaben, was mit In-Memory-Datenbanken nicht geht.

Ein System zur Leistungssteigerung um zusätzlichen RAM zu erweitern ist teurer, als dies mittels Festplattenkapazität zu erreichen. (Ein Terabyte Festplattenspeicher ist preiswerter als ein Terabyte RAM.) Zudem erfordern viele In-Memory-Systeme als Schutz gegen die Auswirkungen eines Systemabsturzes, dass redundante Kopien der Daten auf separaten Rechnern vorgehalten werden. Das Betreiben von verteilten Cache-Lösungen mit einer persistenten Datenbank wie Caché resultiert oftmals in einer Reduzierung der Hardwarekosten.

Nahtloser Datenzugriff per SQL oder als Objekt

Ein Problem, das die meisten In-Memory-Datenbanken teilen, liegt darin, dass aufgrund der für hohe Verarbeitungsgeschwindigkeit optimierten Datenstruktur die Daten für SQL-Abfragen nur schwer zugänglich sind. Um kompatibel zu den üblichen Analyse- und Reporting-Tools zu werden, müssen die Daten zunächst per „Mapping“ in relationale Tabellen konvertiert werden. Dies geschieht in der Regel, wenn Daten aus der In-Memory-Datenbank auf die Festplatte übertragen werden, und beinhaltet üblicherweise einen ETL-Prozess (extract, transform and load). (Dieser Verwaltungs-Overhead und der zusätzliche Zeitaufwand für das Mapping sind die wesentlichen Gründe, warum relationale Datenbanken zu langsam für verteilte Hochgeschwindigkeits-Applikationen sind und warum an ihrer Stelle In-Memory-Datenbanken zum Einsatz kommen.)

Ein paar der In-Memory-Datenbanken basieren auf einem relationalen Modell und bieten Zugriffsmöglichkeiten per SQL. Derartige Systeme leiden unter dem umgekehrten Problem. Objektorientierte Technologien, die typischerweise zur Entwicklung solcher Applikationen genutzt werden, können nicht so einfach auf die Daten zugreifen. Zudem sind die meisten relationalen In-Memory-Datenbanken nicht für Multi-Computer-Konfigurationen ausgelegt. Sie laufen nur auf einem System und haben Einschränkungen beim RAM.

Caché ist anders. Auf die multidimensionalen Arrays kann gleichzeitig als relationale Tabelle und als Objekt zugegriffen werden. Cachés Unified Data Architecture™ hält beide Sichten, die relationale wie die auf die Objekte, zu jeder Zeit aufrecht – und das ohne Mapping.

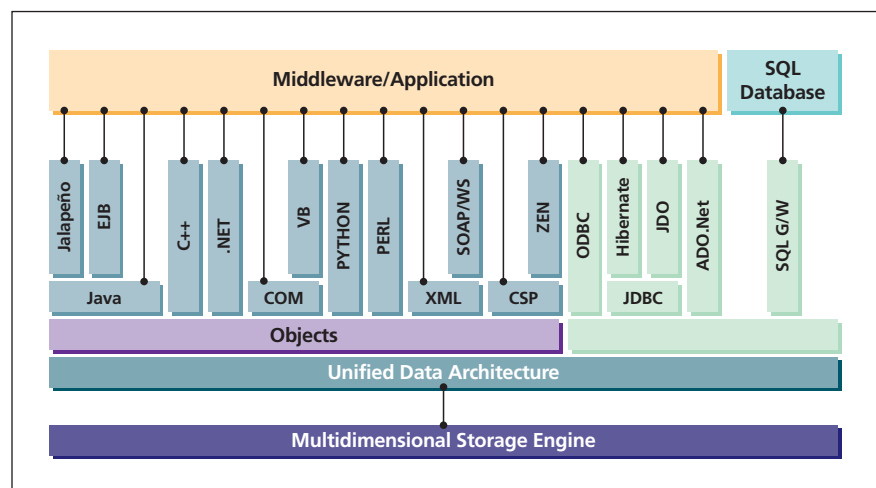


Abb. 1: Cachés Unified Data Architecture erlaubt vielfältige Möglichkeiten, um auf Daten zuzugreifen.

Cachés SQL-Zugriff ist sowohl zu ODBC als auch JDBC kompatibel. Bei den Objekten bietet Caché Bindings zu nahezu jeder objektorientierten Programmiersprache, einschließlich Java, .NET und C++. Cachés Objektdarstellung ist vollumfänglich und unterstützt objektorientierte Konzepte wie Vererbung, Polymorphie und Kapselung.

Enterprise Cache Protocol

Bei verteilten Applikationen pflegt Caché automatisch die verschiedenen Caches mit Hilfe seines Enterprise Cache Protocol™ (ECP).

Mit Hilfe von ECP können Caché-Instanzen als Datenserver und/oder als Applikationsserver konfiguriert werden. Jeder einzelne Datensatz ist Eigentum eines Datenservers. Applikationsserver wissen, wo die Daten liegen, und unterhalten lokale Caches für kürzlich benötigte Daten. Falls ein Applikationsserver die Anfragen nicht aus dem eigenen Cache bedienen kann, fordert er die Daten von dem entsprechenden Datenserver an. ECP hält automatisch die Konsistenz der Caches aufrecht.

ECP erfordert keinerlei Veränderung an den Applikationen. Applikationen behandeln die gesamte Datenbank einfach so, als wäre es eine lokale Datenbank. Dies ist ein wesentlicher Unterschied zu einigen verteilten Cache-Systemen, bei denen jeder Client spezifizieren muss, an welcher Untermenge der Daten er interessiert ist, bevor irgendeine Anfrage starten kann.

Ein Rechner, ein Cache

Ein anderes elementares Unterscheidungskriterium zwischen Caché und anderen verteilten Cache-Lösungen liegt darin, dass diese Produkte in der Regel einen separaten Cache für jeden Prozess auf dem Rechner unterhalten. Hat ein System zum Beispiel acht Clients, dann werden auch acht Caches auf dem System unterhalten.

Im Gegensatz dazu hält Caché seine Caches in einem gemeinsamen Speicher und unterhält lediglich Bindings, um Prozessen, die in ihrem eigenen Speicher-Adressraum ablaufen, Zugriff auf Daten zu ermöglichen. Auf Daten kann simultan per TCP-basierenden Protokollen wie JDBC, über Language Bindings oder, für höchste Performanz, per Bindings, die Applikationen einen direkten Zugang zum Speicher erlauben, zugegriffen werden.

Es mehreren Clients zu erlauben, sich einen einzigen Cache zu teilen, bietet eine Reihe von Vorteilen. Einer liegt darin, dass ein gemeinsam genutztes Cache-System geringere Speicheranforderungen hat. Wenn, wie es oftmals der Fall ist, einzelne Clients auf gemeinsame Daten zugreifen, halten viele andere Cache-Lösungen multiple Kopien derselben Daten vor. Mit Caché muss lediglich eine einzelne Kopie der Daten pro System vorgehalten werden.

Auch die Last im Netzwerk profitiert von einem einzigen Cache pro System. In hochperformanten Systemen kann der Netzverkehr zur Aufrechterhaltung der Cache-Integrität zu einer ernsthaften Herausforderung werden. Bei einem einzelnen Cache pro System hingegen muss auch nur dieser eine Cache aktualisiert werden, wenn sich die zugrunde liegenden Daten ändern. Die Notwendigkeit, identische Datensätze in multiplen Caches zu aktualisieren, entfällt.

Selbst beim Einsatz von Multi-Kern-Prozessoren kommt ein Caché-basierendes System mit einem gemeinsamen Cache pro Rechner aus, was in überlegener Skalierbarkeit im Vergleich zu anderen verteilten Cache-Lösungen mündet. So müssen in einem Caché-basierenden Netzwerk mit 250 Systemen, jedes mit einem 8-Kern-Prozessor, lediglich 250 Caches miteinander kommunizieren, um Speicherkohärenz zu gewährleisten. Eine Lösung, die separate Caches für jeden Kern erfordert, müsste in diesem Fall 2.000 Caches koordinieren. Da moderne Computer acht, sechzehn oder noch mehr Kerne besitzen können, wird der Vorteil von Caché bei der Skalierbarkeit immer wichtiger.

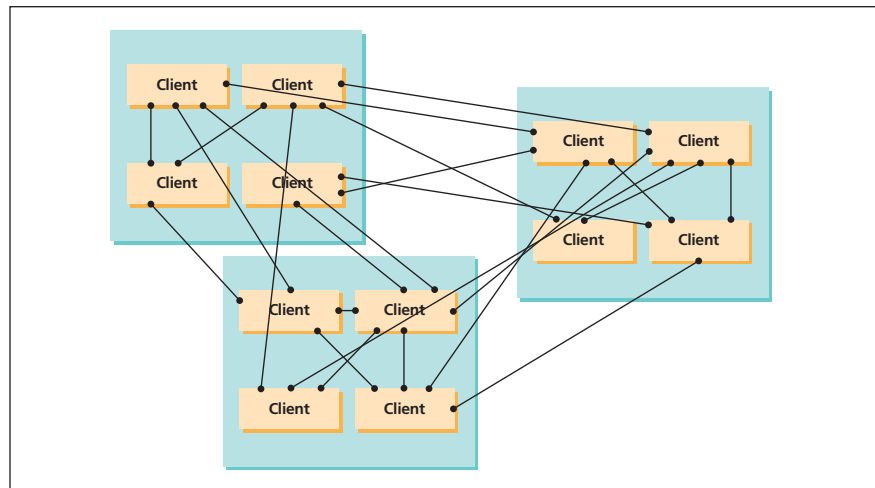


Abb. 2a: Speicherkohärenz ohne Einsatz des InterSystems Enterprise Cache Protocol.

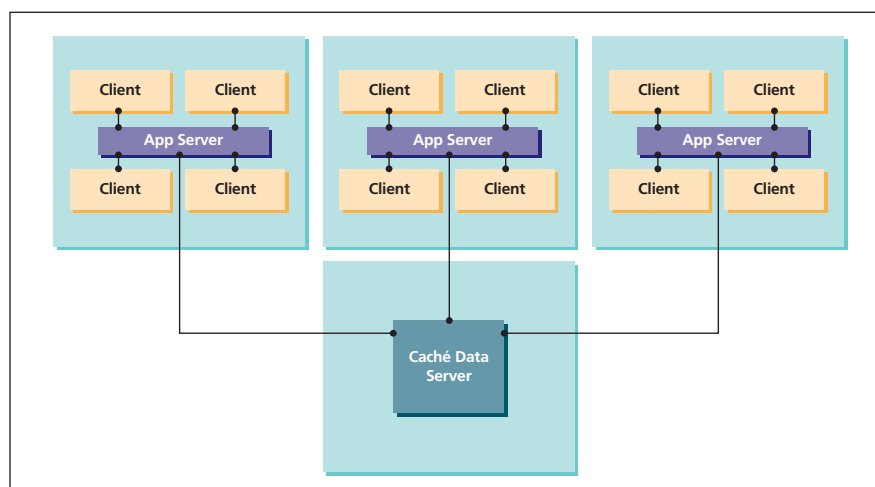


Abb. 2b: Speicherkohärenz in einem Caché-basierenden System.

Den Cache füllen

In vielen verteilten Cache-Applikationen kann das Laden der Anfangswerte eine langatmige Angelegenheit sein. Das kann einerseits an der schieren Menge der Daten liegen und/oder an der Zeit, die es braucht, die Daten aus dem relationalen Speicher in die von der Applikation verlangten objektorientierten Strukturen zu mappen. Für einige datenintensive Applikationen braucht es mehr Zeit, den In-Memory-Cache zu füllen, als für die anschließenden Kalkulationen.

Mit Caché ist das anders. Cachés außergewöhnliche SQL-Fähigkeiten erlauben es, Daten sehr einfach von relationalen primären Datenquellen zu beziehen. Wobei Caché als persistente Datenbank auch zugleich die primäre Datenquelle sein kann. In diesem Fall bestünde nicht einmal die Notwendigkeit, Anfangswerte zu laden. Lokale Caches laden die benötigten Daten automatisch, sobald eine Anforderung eintrifft.

Eine andere Überlegung betrifft die Frage, wie viele Rechner damit beschäftigt sind, die Caches zu füllen. Kommt Caché zum Einsatz, hält ein kleiner Prozentsatz aller Computer in einem Grid die Daten vor. Um dieses Grid mit Daten zu versorgen, bedarf es lediglich eines Zugriffs auf die ECP-Datenserver, die im Hintergrund bestückt werden können, während die übrigen Rechner für andere Aufgaben zur Verfügung stehen. Wenn die Applikationsserver online kommen, wird ihr Cache im laufenden Betrieb automatisch durch die Anfragen gefüllt.

Im Gegensatz dazu läuft der Ladevorgang der meisten In-Memory-Produkte derart ab, dass die Daten partitioniert und dem verteilten Cache zugewiesen werden, so dass alle oder nahezu alle Daten zumindest im Speicher eines Rechners vorhanden sind. Als Ergebnis ist es oftmals unmöglich, die Anfangswerte nur mit Hilfe einiger Rechner zu laden und die übrigen erst bei Bedarf dazuzuholen.

Fazit

Der wichtigste Grund für den Einsatz von In-Memory-Datenbanken ist ihre Geschwindigkeit. Aber obwohl sie schnell sind, leiden In-Memory-Datenbanken oft unter schlechter Skalierbarkeit, fehlender SQL-Unterstützung, hohen Hardwareanforderungen und dem Risiko, Daten bei Störungen zu verlieren.

Caché ist die einzige persistente Datenbank, die eine zu In-Memory-Datenbanken vergleichbare Performanz bietet. Caché unterstützt zudem extrem große Datenvolumen, bietet nahtlosen Datenzugriff per SQL und auf Objekte, ermöglicht verteilte Systeme mit Hunderten von Rechnern und ist absolut zuverlässig.

Dies alles macht Caché zu einer attraktiven Alternative für Applikationen, die mit höchster Geschwindigkeit große Mengen an Daten verarbeiten müssen.

Über InterSystems

InterSystems ist ein weltweit führendes Unternehmen für Softwaretechnologien. Hauptsitz ist Cambridge, USA. Zudem verfügt InterSystems über weltweit 23 Niederlassungen. InterSystems bietet innovative Produkte, die eine schnelle Entwicklung, Inbetriebnahme und Integration unternehmensweiter Anwendungen ermöglichen. **InterSystems Caché®** ist eine hochperformante Objektdatenbank, die Anwendungen beschleunigt und besser skalierbar macht. **InterSystems Ensemble®** ist eine schnelle Integrations- und Entwicklungsplattform, die Anwendungen um neue Funktionen erweitert und miteinander verbindet. **InterSystems HealthShare™** ist eine Plattform, die eine zeitnahe Implementierung elektronischer Patientenakten für den regionalen oder nationalen Austausch von Gesundheitsdaten ermöglicht. **InterSystems DeepSee™** ist eine Software, mit der Echtzeit-Business-Intelligence-Funktionen direkt in unternehmenskritische Anwendungen eingebettet werden können, um bessere Entscheidungsgrundlagen für das Tagesgeschäft zu ermöglichen.

Weitere Informationen finden Sie unter www.intersystems.de.

Deutschland

InterSystems GmbH
Hilpertstr. 20a
D-64295 Darmstadt
Tel.: +49.6151.1747-0
Fax: +49.6151.1747-11
www.InterSystems.de

Schweiz

InterSystems B.V.
In der Luberzen 42
CH-8902 Urdorf
Tel.: +41.43.455.7711
Fax: +41.43.455.7722
www.InterSystems.ch

INTERSYSTEMS