IBExpert White Paper

Firebird-Datenbankreplikation: Zuverlässige Synchronisation nahezu in Echtzeit für unternehmenskritische Systeme

Holger Klemt, August 2026

Moderne Anwendungen stellen zunehmend Anforderungen an Datenbanken, die über mehrere Standorte, Server und Umgebungen hinweg kontinuierlich verfügbar sein müssen. Ob Hochverfügbarkeit, die Migration von einer Desktop-Anwendung zu einer webbasierten Lösung, verteilte Datenbanken oder die kontinuierliche Synchronisierung zwischen verschiedenen Rechenzentren – eine zuverlässige Datenbankreplikation ist dabei ein entscheidender Bestandteil.

Unsere Firebird-Replikationslösung wurde genau für solche Szenarien entwickelt. Sie kombiniert transaktionssichere Replikation, Synchronisierung nahezu in Echtzeit, One-to-Many-Replikation und Unterstützung für große Firebird-Datenbanken – und kann in bestehende Produktionsumgebungen integriert werden, ohne die Datenbank offline nehmen zu müssen.

Entwickelt für Firebird

Das Replikationsframework unterstützt Firebird 2.5 und neuere Versionen, einschließlich Firebird 4, Firebird 5 und aktuellen Firebird 6 Datenbanken.

Ein wesentlicher Vorteil besteht darin, dass die Replikation direkt auf Datenbankebene mithilfe von aktiven Triggern und einem Transaktionsprotokoll umgesetzt wird. Änderungen an der Quelldatenbank werden als Transaktionen erfasst und können anschließend an eine oder mehrere Ziel-Datenbanken übertragen werden.

Diese Architektur stellt sicher, dass Änderungen nicht verloren gehen, wenn beispielsweise eine Netzwerkverbindung oder der Replikationsprozess vorübergehend nicht verfügbar ist.

Replikation ohne Ausfallzeit der Produktionsdatenbank

Eine der wichtigsten Anforderungen bei Migrations- oder Hochverfügbarkeitsprojekten besteht darin, die Replikation aktivieren zu können, ohne den laufenden Betrieb der Anwendung zu unterbrechen.

Unser Replikationsverfahren ist genau auf dieses Szenario ausgelegt.

Sobald die Replikation initialisiert wurde, beginnen die erforderlichen Trigger damit, INSERT-, UPDATE- und DELETE-Operationen in einem dedizierten Transaktionsprotokoll zu erfassen. Die Produktionsdatenbank läuft weiterhin einwandfrei.

Die initiale Synchronisierung erfolgt anschließend durch die Erstellung eines Backups der Master-Datenbank und dessen Wiederherstellung auf dem Zielserver. Auch hierfür muss die Produktionsdatenbank nicht offline genommen werden.

Nachdem die Ziel-Datenbank initialisiert wurde, überträgt der Replikationsprozess alle Transaktionen, die während der Erstellung des Backups entstanden sind, und setzt anschließend die kontinuierliche Synchronisierung mit allen neuen Transaktionen fort.

Transaktionssichere Replikation

Zuverlässigkeit steht im Mittelpunkt der Architektur.

Angenommen, die Master-Datenbank hat 100 Transaktionen gesammelt, die noch repliziert werden müssen. Fällt die Netzwerkverbindung aus, nachdem nur ein Teil der Daten übertragen wurde, gehen die verbleibenden Transaktionen nicht verloren.

Sie bleiben im Transaktionsprotokoll gespeichert und werden beim nächsten Replikationsvorgang erneut berücksichtigt.

Der Replikationsprozess ist daher nicht auf eine dauerhaft verfügbare Netzwerkverbindung angewiesen. Ein Zielserver kann sogar über mehrere Tage oder Wochen nicht verfügbar sein, während die Master-Datenbank weiterhin alle für die spätere Synchronisierung erforderlichen Transaktionen erfasst.

Sobald das Zielsystem wieder erreichbar ist, wird die Replikation an der Stelle fortgesetzt, an der sie unterbrochen wurde.

One-to-Many-Replikation

Die Architektur unterstützt auch die One-to-Many-Replikation.

Eine einzelne Master-Datenbank kann ihre Daten an mehrere Zielserver replizieren. Dabei wird jedes Zielsystem unabhängig verwaltet und überwacht.

Beispielsweise:

Master-Server → Slave 1
Master-Server → Slave 2
Master-Server → Slave 3

Ist beispielsweise Slave 2 vorübergehend nicht verfügbar, kann die Replikation zu Slave 1 und Slave 3 normal weiterlaufen. Die für Slave 2 erforderlichen Transaktionen bleiben auf dem Master verfügbar, bis dieses Zielsystem wieder erreichbar ist.

Dadurch eignet sich die Architektur besonders für verteilte Systeme, bei denen einzelne Standorte über unzuverlässige Netzwerkverbindungen verfügen oder Server zeitweise nicht verfügbar sind.

Multi-Master-Replikation

Abhängig von der Struktur der Datenbank kann die Technologie auch eine Multi-Master-Replikation unterstützen.

In einer solchen Konfiguration können mehrere Server als Master-Datenbanken arbeiten und Änderungen untereinander austauschen.

Multi-Master-Umgebungen erfordern jedoch eine sorgfältige Betrachtung des Datenbankdesigns, insbesondere bei der Generierung von Primärschlüsseln. Automatisch generierte Schlüssel müssen beispielsweise so gestaltet sein, dass unabhängig auf verschiedenen Master-Servern erzeugte Datensätze nicht zu Schlüsselkonflikten führen.

Falls erforderlich, können Datenbank-Metadaten und Anwendungslogik entsprechend angepasst werden, um eine solche Architektur zu unterstützen.

Synchronisierung nahezu in Echtzeit

Der Replikationsprozess ist für einen Betrieb nahezu in Echtzeit ausgelegt.

In einer typischen Umgebung werden Transaktionen kontinuierlich über einen Push/Pull-Prozess übertragen. Die tatsächliche Latenz hängt dabei von der Leistungsfähigkeit der Server, dem Transaktionsvolumen und den Netzwerkbedingungen ab.

In einer großen verteilten Installation waren Änderungen an entfernten Standorten typischerweise innerhalb von weniger als 10 Sekunden verfügbar. Selbst bei hoher Systemlast lag die Verzögerung in der Regel bei etwa einer Minute.

Auch Standorte mit langsameren oder zeitweise unzuverlässigen Internetverbindungen konnten weiterhin lokal arbeiten. Sobald die Verbindung wiederhergestellt war, wurden die aufgelaufenen Transaktionen automatisch synchronisiert.

Bewährt bei großen und verteilten Firebird-Installationen

Die Technologie wurde bereits in anspruchsvollen Produktivumgebungen mit großen Datenbanken und geografisch verteilten Systemen eingesetzt.

Ein besonders umfangreiches Projekt umfasste:

- 9 zentrale Server
- 6 verschiedene Rechenzentren und Städte
- Rund 225 Firebird-Datenbanken an einzelnen Standorten
- Etwa 500 GB für die größte zentrale Datenbank
- Rund 2 Milliarden Datensätze in der größten Datenbank
- Etwa 5–6 Millionen INSERT-, UPDATE- und DELETE-Operationen pro Tag
- Synchronisierung nahezu in Echtzeit zwischen den zentralen Systemen und den einzelnen Standorten

Die einzelnen Datenbanken an den entfernten Standorten enthielten jeweils Teilmengen der zentralen Daten und übertrugen lokale Änderungen zurück an die zentrale Infrastruktur.

Auch bei unzuverlässigen Netzwerkverbindungen an einzelnen Standorten stellte die Replikationsarchitektur sicher, dass Transaktionen solange verfügbar blieben, bis sie erfolgreich übertragen werden konnten.

Umgang mit sehr großen Datenbanken und BLOB-Daten

Große Firebird-Datenbanken können erhebliche Mengen an BLOB-Daten enthalten, insbesondere in Anwendungen aus Bereichen wie Gesundheitswesen, Dokumentenmanagement, Bildverarbeitung oder Archivierung.

In einer anderen Installation erreichte die gesamte Datenbankumgebung einschließlich Transaktions-protokollen und jährlicher BLOB-Datenbanken eine Größe von rund 7 TB.

Bei sehr großen Datenmengen kann es sinnvoll sein, historische Daten in separate Datenbanken auszulagern.

Beispielsweise können jährliche Transaktions- oder BLOB-Datenbanken nach Abschluss eines Geschäftsjahres geschlossen und schreibgeschützt werden. Sie bleiben weiterhin über die Hauptdatenbank zugänglich, vergrößern jedoch nicht mehr die aktive Produktionsdatenbank.

Dieser Ansatz kann zukünftige Backup- und Restore-Vorgänge erheblich vereinfachen und beschleunigen.

Kein Single Point of Failure im Replikationsprozess

Ein vorübergehender Ausfall eines Zielservers oder einer Netzwerkverbindung führt nicht dazu, dass die Master-Datenbank keine Änderungen mehr erfassen kann.

Dies ist insbesondere bei verteilten Installationen von großer Bedeutung.

Ein entfernter Server kann neu gestartet werden, die Netzwerkverbindung verlieren oder über einen längeren Zeitraum nicht verfügbar sein. Sobald er wieder erreichbar ist, kann der Replikationsprozess mit den Transaktionen fortgesetzt werden, die während seiner Abwesenheit im Transaktionsprotokoll gespeichert wurden.

Das Ergebnis ist eine Replikationsarchitektur, die auf zuverlässiger nachträglicher Synchronisierung ohne Datenverlust basiert und nicht voraussetzt, dass alle Server und Netzwerkverbindungen jederzeit verfügbar sind.

Die Datenbank-Metadaten spielen eine entscheidende Rolle

Eine zuverlässige Replikation beginnt mit einem geeigneten Datenbankdesign.

Idealerweise verfügt jede Tabelle, die repliziert werden soll, über einen Primärschlüssel oder zumindest über einen geeigneten eindeutigen Index.

Auch Primärschlüssel sollten möglichst kompakt sein. Extrem komplexe Primärschlüssel, die sich über eine große Anzahl von Spalten erstrecken, können die Replikation und die Entwicklung der Anwendung unnötig komplizieren.

Tabellen ohne Primärschlüssel benötigen gegebenenfalls einen alternativen Mechanismus zur eindeutigen Identifizierung von Datensätzen. Abhängig von der Struktur der Datenbank kann hierfür eine Technologie auf Basis von Firebirds RDB$DB_KEY in Betracht gezogen werden.

Bei komplexen Datenbankumgebungen können die Metadaten bereits vor Beginn der Implementierung analysiert werden, um die geeignete Replikationsstrategie zu bestimmen.

Kontinuierliche Replikation mit geringem Wartungsaufwand

Nach der Implementierung arbeitet das Replikationssystem weitgehend im Hintergrund.

In vielen Installationen ist keine regelmäßige manuelle Wartung erforderlich.

Dennoch kann es vorkommen, dass Datenbankadministratoren umfangreiche Änderungen an den Metadaten oder der Datenbankstruktur durchführen. In solchen Fällen kann es erforderlich sein, die Replikation neu zu initialisieren oder anzupassen.

Für diese Situationen kann technische Unterstützung über ein Prepaid-Support- und Hotline-System bereitgestellt werden. Administratoren können dadurch bei Bedarf Unterstützung anfordern, ohne einen permanenten Wartungsvertrag ausschließlich für gelegentliche Replikationsvorgänge abschließen zu müssen.

Implementierung und Migration

Jedes Replikationsprojekt ist unterschiedlich. Die Größe einer Datenbank allein reicht daher nicht aus, um den tatsächlichen Implementierungsaufwand zu bestimmen.

Vor Beginn eines Projekts empfehlen wir daher eine Analyse der folgenden Punkte:

- Datenbank-Metadaten
- Primärschlüssel und Indizes
- Datenbankstatistiken
- Anzahl und Größe der Datenbanken
- Transaktionsvolumen
- Verwendung von BLOB-Daten
- Netzwerkarchitektur
- Anzahl der Replikationsziele
- Gewünschte Replikationsrichtung
- Anforderungen an Verfügbarkeit und Wiederherstellung

Für eine erste technische Bewertung genügt häufig bereits ein Metadata-Only-Backup. Dadurch kann die Struktur der Datenbank analysiert werden, ohne dass sensible Kundendaten bereitgestellt werden müssen.

Die Implementierung und Schulung können vollständig remote durchgeführt werden – auch bei Kunden in weit entfernten Zeitzonen.

Quellcode und Anpassungsmöglichkeiten

Das Replikationsframework kann zusammen mit seinem Quellcode bereitgestellt werden. Dadurch haben Kunden die Möglichkeit, die Funktionsweise nachzuvollziehen, Anpassungen vorzunehmen und die Lösung in ihre bestehende Systemumgebung zu integrieren.

Dies ist insbesondere dann von Vorteil, wenn die Replikation Teil eines größeren Migrationsprojekts ist, beispielsweise bei der Umstellung einer bestehenden Desktop-Anwendung auf eine webbasierte Architektur, während das bestehende Produktivsystem weiterhin verfügbar bleiben soll.

Eine praktische Grundlage für die Systemmigration

Datenbankreplikation ist besonders wertvoll, wenn ein Unternehmen eine bestehende Anwendung modernisieren möchte, ohne längere Ausfallzeiten in Kauf nehmen zu müssen.

Beispielsweise kann ein Krankenhausinformationssystem (KIS) zusammen mit einer zugehörigen Bilddatenbank kontinuierlich verfügbar bleiben müssen, während die bestehende Desktop-Anwendung schrittweise auf eine moderne Webanwendung migriert wird.

Eine transaktionsbasierte Replikationsschicht kann dabei als Brücke zwischen der bestehenden und der neuen Systemumgebung dienen. Beide Systeme können parallel betrieben werden, während die Daten kontinuierlich synchronisiert werden.

Dadurch lassen sich Migrationsrisiken erheblich verringern und ein kontrollierter Übergang anstelle einer disruptiven „Big-Bang“-Migration ermöglichen.

Eine Lösung, die auf langjähriger Praxiserfahrung basiert

Die hier beschriebene Architektur wurde nicht nur für theoretische Szenarien entwickelt. Sie wird seit mehr als einem Jahrzehnt in anspruchsvollen Produktivumgebungen mit geografisch verteilten Firebird-Datenbanken, großen Transaktionsvolumina, unzuverlässigen Netzwerkverbindungen und erheblichen Datenmengen eingesetzt.

Die Kombination aus Transaktionsprotokollierung, triggerbasierter Erfassung von Änderungen, Push/Pull-Replikation und unabhängiger Verwaltung der einzelnen Replikationsziele bietet eine robuste Grundlage für hochverfügbare und verteilte Firebird-Systeme.

Für Unternehmen, die eine Firebird-Migration, ein Hochverfügbarkeitsprojekt, eine verteilte Datenbankarchitektur oder eine Synchronisierung nahezu in Echtzeit planen, ist eine technische Analyse der bestehenden Datenbankumgebung der erste Schritt.

Auf Grundlage der Datenbank-Metadaten und aktueller Datenbankstatistiken können anschließend eine realistische Implementierungsplanung, eine geeignete Replikationsarchitektur und ein projektspezifisches Angebot erstellt werden.