

IBM solidDB

IBM solidDB, solidDB Cache for DB2, solidDB Cache for IDS の概要

超高速

solidDB は、データをディスクではなく常時、メイン・メモリー上に保持するため、超高速に動作します。アプリケーションからは、標準インターフェース (ODBC、JDBC、SQL) 経由でこの機能を活用できます。

究極の可用性

solidDB は、常に 2 セットのデータを同期させて保ち、究極の可用性を提供します。システムに障害が発生しても、1 秒以内に、アプリケーションから solidDB へのアクセスを回復することができます。また、データが失われることもありません。

低コスト

solidDB は展開も管理も容易、アプリケーションに直接、組み込むことも可能、基本的に無人運用も可能であることから、低い総所有コストを実現できます。

IBM solidDB 製品ファミリー

solidDB Cache for DB2



solidDB Cache for IDS



solidDB (スタンドアローン)



IBM solidDB は、リレーショナル・インメモリー・データベースで、従来のデータベースに対し最大で 10 倍という超高速をたたき出すことができます。アプリケーションからのアクセスには一般的な SQL 言語が使えます。これで、マイクロ秒単位の応答時間、1 秒あたり数万単位のトランザクションというスループットが得られます。また、別インスタンスの solidDB へ 1 秒以下の短時間でフェイルオーバーする能力も持ち、究極のデータ可用性も提供します。

IBM DB2 や IDS のキャッシュとして使う場合も、スタンドアロンの記録用データベースとして使う場合も、パフォーマンス・クリティカルなデータを超高速に提供することができます。

DB2 データや IDS データへのアクセスを加速する

solidDB のインメモリー・データベースを IBM DB2 や IDS とシームレスに統合するコネクタが新しく登場しました。この solidDB Cache を利用すると、DB2 や IDS に格納されているデータの一部を solidDB のパワフルなインメモリー・データベースにキャッシュし、アクセス速度を高めることができます。この機能が真価を発揮するのは、数多くのユーザーが同時にアクセスすることが多いパフォーマンス・クリティカルなデータです。このようなデータは、チケット発行システム、予約システム、オンライン・ゲーム、イベント処理とアラート発報、電子商取引、SaaS (Software as a Service) アプリケーション・プラットフォームな



どでよく使われるタイプです。DB2 や IDS のデータを solidDB へキャッシングし、カスタマー・サービスや株式トレーディングなどのアプリケーションで発生することが多いピーク・ワークロードの処理に対応することもできます。

ディスクベースの従来型データベースには巨大なデータベースが格納できる、幅広いワークロードに対応できる柔軟性を持つといった利点があるのに対し、solidDB のインメモリー・データベースには、メイン・メモリーに展開できるデータであれば超高速なアクセスができる、特殊なワークロードに最適化されているという利点があります。つまり、幅広い対応力を持つ DB2 や IDS に超高速な solidDB を組み合わせれば、両方の利点を生かすことができるのです。

図1:IBM solidDB を DB2 あるいは IDS のキャッシュとして使用

超高速

インメモリー・データベース技術によって超高速を実現

インメモリー・データベースとは、超高速で応答時間を一定範囲におさえなければならないリアルタイム・アプリケーションの要求に応えるものです。その名前から想像されるように、インメモリー・データベースとは、ディスク上ではなく、すべてをメイン・メモリー上に展開しているため、データ・アクセスの速度が一桁、向上します。

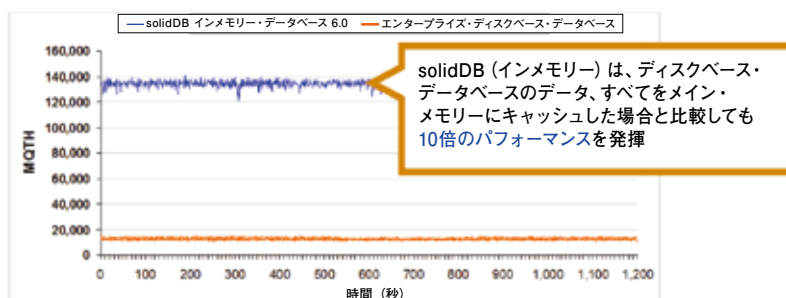


図2:ディスクベースの従来型データベースと超高速 solidDB との比較

solidDB のインメモリー・データベースは、ディスクベース・データベースですべてのデータをメイン・メモリー上にキャッシュした場合よりもさらに速く、高いトランザクション・スループットを実現するとともに、応答時間が短く、かつその変動が少なくなります。

図 2 は、Telecom One ベンチマークの結果です。1 秒あたりのトランザクション数で測定した MQTH (Maximum Qualified Throughput) を示す縦軸から、solidDB インメモリー・データベースと従来型のディスクベース・データベースのパフォーマンスを比較すると、solidDB のほうが 10 倍のスループットを実現できることがわかります。

solidDB のようなインメモリー・データベースは、超高速でアクセスできなければならないデータのすべてを常にメイン・メモリー上に置くことを前提としています。そのため、メイン・メモリー上にデータを格納し、そのデータを検索・処理するために特別に設計されたデータ構造とアクセス方法、および高効率の同時実行制御メカニズムを備えています。この結果、ディスク・ベースの従来型データベースに対してパフォーマンスの観点から2つのメリットが生まれます。まず第1に、solidDB では、ディスクからメイン・メモリーへデータ・ブロックを移動する必要がありません。何故ならアプリケーションから要求されたデータは、すべて、すでにメイン・メモリー上に存在するからです。第2に、solidDB は、ディスク・ベースのデータベースですべてのデータをメイン・メモリー上のバッファーにプールした場合よりもさらに速く動作します。これは、solidDB のデータ構造とアクセス方法がメイン・メモリーのアクセスに最適化されているからです。

solidDB はまた、64 ビット・コンピューターで使用可能となる膨大なメモリー・サイズを活用できるように設計されています。さらに、マルチコア・アーキテクチャーやマルチプロセッサ・アーキテクチャーが提供するプロセッシング・スケーラビリティにも対応しています。また、そのアクセス方法は効率的な同時実行制御メカニズムを実装できるものとなっており、トランザクションの完全性を保ちつつ、マイクロ秒単位の応答時間で数多くのトランザクションを並行処理することができます。こうして、IBM DB2 や IDS のキャッシュとして使う場合も、スタンドアロンの記録用データベースとして使う場合も、パフォーマンス・クリティカルなデータを超高速に提供することができるのです。

超高速を実現しつつ、データの永続性と復元性を確保

solidDB はすべてのデータをメイン・メモリー上でアクセスできる状態に保つことが前提となっていますが、同時に、更新データをディスクに書き込み、サーバーに障害が発生してもデータを復元できるようにもしています。この機能を実現しているのが、solidDB のチェックポイント機構とトランザクション・ロギング機構です。各チェックポイントでは、コミットしたトランザクションをメイン・メモリーからディスク上のデータベース・ファイルにコピーします。これで、チェックポイント間でサーバーに障害が発生しても、ディスク上に一貫したデータのスナップショットが存在することになります。チェックポイントとチェックポイントとの中間では、コミットしたトランザクションをトランザクション・ログに書き込みます。システムがクラッシュしたら、このトランザクション・ログとロールフォワード回復機能を使って、最新のチェックポイントのあとにコミットしたトランザクションを solidDB が自動的に回復し、最新のコミット状態とするわけです。solidDB のトランザクション・ロギング機構は、パフォーマンスと耐久性のバランスを最適化できるようにさまざまな構成オプションが使えるようになっています。ストリクト・ロギングとすると、トランザクションをコミットした直後、同期的にロギングが行われます。リラックス・ロギングとすると、トランザクション・ログに遅延書き込みを行う非同期動作となります。なお、solidDB にはスナップショットで整合性を確保するチェックポイント機能があるため、トランザクション・ロギングをオフとすることもできます。最新のチェックポイントまで回復できればいい場合には、こうすれば速度をさらに高めることが可能です。

DB2 や IDS で超高速を実現する

ディスク・ベースの従来型データベースには巨大なデータベースが格納できる、幅広いワークロードに対応できる柔軟性を持つといった利点があるのに対し、solidDB のインメモリー・データベースには、メイン・メモリーに展開できるデータであれば超高速なアクセスができる、特殊なワークロードに最適化されているという利点があります。つまり、幅広い対応力を持つ DB2 や IDS に超高速な solidDB を組み合わせれば、両方の利点を生かすことができるのです。

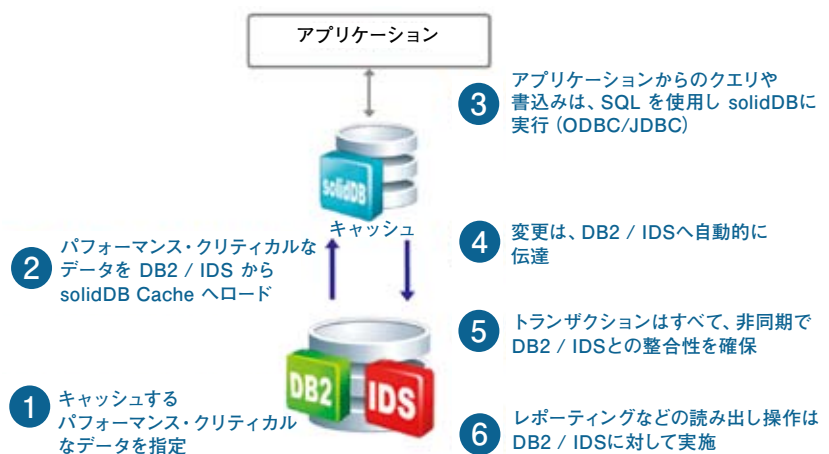


図3:solidDB Cache による DB2 あるいは IDS との統合

solidDB を DB2 や IDS のキャッシュとして使う場合には、まず、高速アクセスのメリットが得られるデータベース・テーブルを特定します。solidDB でキャッシュするテーブルは、テーブルのサイズとワークロード要件に応じてすべての場合も、一部の場合もあります。数百ものテーブルを持つオンライン電子商取引用データベースがあったとすると、パフォーマンス・クリティカルなテーブル、10 個だけをキャッシュするということもできます。たとえば、セッション管理、ショッピングカート管理、それからユーザーのウィッシュ・リストなどパーソナライズ・コンテンツの動的生成に利用するテーブルなどが考えられるでしょう。このテーブルを DB2 や IDS のスキーマで指定すると、solidDB コネクタがデータを DB2 や IDS からロードし、アプリケーションから超高速でアクセスできるようにします。これで、アプリケーションからこの 10 個のテーブルのデータに対するアクセスは、クエリーも変更も SQL を用いて solidDB で行えるわけです。リレーショナルデータベースに期待されるトランザクションの整合性は solidDB が処理します。solidDB では、すべての変更を、チェックポイント機能とトランザクション・ロギングでローカルに保持します。その上で、solidDB から DB2 や IDS へと変更処理を伝達します。このような形で、solidDB と DB2 あるいは IDS とのコミュニケーションに障害が発生しても、トランザクションが失われなくなっているわけです。なお、パフォーマンス・クリティカルなアプリケーションから solidDB にキャッシュされたテーブルへ直接、超高速にアクセスする際、同時に、アプリケーションから直接、DB2 や IDS のデータベースにアクセスすることも可能です。

solidDB Cache for DB2 や solidDB Cache for IDS はリードオンリー・キャッシュとすることもできます。この場合、データの更新は DB2 あるいは IDS から行い、solidDB Cache は更新されたデータをリロードしてアプリケーションからのクエリーに備えます。

また、1 台のサーバーのメイン・メモリー上に収まらないほどテーブルが大きい場合、solidDB Cache for DB2 や solidDB Cache for IDS はテーブルを複数の solidDB Cache インスタンスに分割し、複数サーバーにスケールアウトすることができます。DB2 あるいは IDS のクライアント・テーブルが 100 万件ものクライアント・データを持つ巨大なものであった場合、solidDB Cache は、100 万件をまとめて solidDB Cache インスタンスとせず、たとえば 25 万件のデータを持つインスタンス、4 つに分割することができます。このテーブル分割機能は、キャッシュ対象のパフォーマンス・クリティカルなデータが数十ギガバイトもの大きさがある場合に役立ちます。80 ギガバイトものメイン・メモリーを持つ 64 プロセッサ構成のサーバーを 1 台、用意しなくても、一般的なサーバーを複数台、用意して、複数の solidDB Cache インスタンスを利用すればいいからです。データを複数の solidDB Cache インスタンスに分割すると同時に、読み出し操作の自動ロードバランシングと 1 秒以下のフェイルオーバーを提供するホットスタンバイ構成の solidDB とすれば、パフォーマンスのさらなる向上とパフォーマンス・クリティカルなデータの高可用性を両立させることができます。

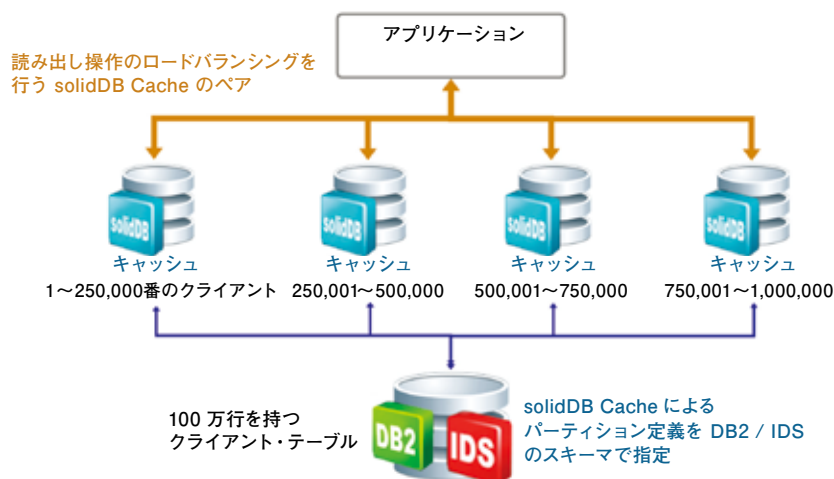


図4: 複数インスタンスの solidDB Cache により、複数のサーバーを用いて DB2/IDS のパフォーマンスを高めることができます。

スタンドアローンの solidDB によって超高速を実現する

solidDB は、スタンドアローンのインメモリー・データベースとしても使えます。この構成では記録用データベースとして機能し、データベース・サーバーとして使う形でもアプリケーションに直接リンクした形でも、前述のように超高速で動作します。

この構成では、インメモリー・データベースほど超高速なアクセスを必要としないデータ用に、solidDB 自体がディスク・ベースのテーブルを持つことができます。solidDB からディスク上のテーブルを管理する場合、そのパフォーマンスは従来のディスク・ベース・データベースと同等となります。アプリケーションからのアクセスはインメモリー・テーブルでもディスク・ベースのテーブルでも透過的で、同じトランザクションとなります。

究極の可用性

システムに障害が発生した場合、IBM solidDB は、1 秒以内に回復することが可能で、パフォーマンス・クリティカルなアプリケーションが要求する究極のデータ可用性を提供します。2 ノードのホットスタンバイ構成となっており、常に 2 セットのデータを同期させて持つようになっています。システムに障害が発生すると、1 秒以内に solidDB ノード間でフェイルオーバーが行われ、究極のデータ可用性を提供するのです。単一障害点はありません。このため、不測の停止が発生することはありません。このアーキテクチャーには、そのほかにもメリットがあります。ローリング・アップグレードやレポーティング、バックアップなどのメンテナンスを行う場合、solidDB のホットスタンバイ・ノードを使えば、solidDB のプライマリー・ノードをオフラインにする必要がないのです。

システムに障害が発生した場合や solidDB ノード間の「スイッチオーバー」を計画した場合などには、1 秒以下の時間で solidDB のセカンダリー・サーバーが処理を引き継ぎます。データが失われることはありません。フェイルオーバーはアプリケーションから透過的に行われ、データベース・コネクションとセッション・アトリビュートは solidDB の ODBC ドライバーと JDBC ドライバーが維持します。

高可用性についてはシステム、セッション、トランザクションという 3 レベルの構成オプションが用意されており、solidDB のプライマリー・サーバーとセカンダリー・サーバーの同期方法を指定することができます。このため、スループットと耐久性、回復時間がかつてないほど柔軟にバランスさせることができます。

このホットスタンバイ構成を solidDB Cache for DB2 や solidDB Cache for IDS に適用すれば、solidDB Cache で HADR (High Availability and Disaster Recovery) 環境を実現することができます。

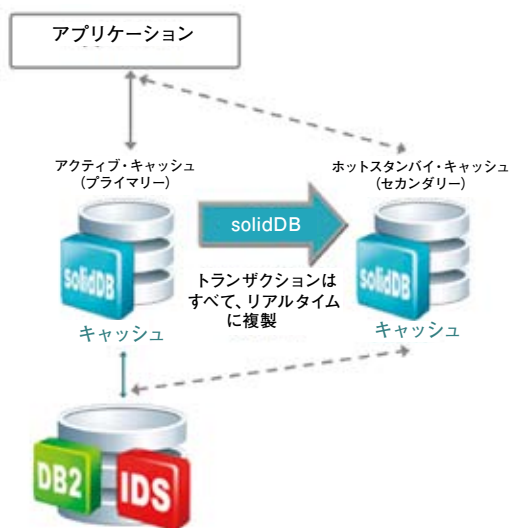


図5:HADR(High Availability and Disaster Recovery)構成とした DB2/IDS

ホットスタンバイ構成:

- 複製とロギングの設定が構成可能で、パフォーマンス、MTTR、耐久性のトレードオフを最適化することができます。
- いずれかの solidDB Cache インスタンスが利用できなくなった場合、1 秒以内にフェイルオーバーが行われます。
- 透過的な接続性:
 - アプリケーションからは、ひとつの論理的コネクションで solidDB Cache のプライマリー・インスタンスとセカンダリー・インスタンスにアクセスできます。
 - フェイルオーバーやスイッチオーバーが行われた場合でも、アプリケーション側のコネクション・ハンドルやセッション・アトリビュートを変える必要はありません。
- 読み出し操作について、透過的なロードバランシングが行われます。

究極の可用性と超高速の組み合わせ

IBM DB2 や IDS のキャッシュとして使う場合も、スタンドアローンの記録用データベースとして使う場合も、ホットスタンバイ構成の solidDB は、solidDB のプライマリー・インスタンスとホットスタンバイ・インスタンスに読み出し操作の負荷をアプリケーションに対して透過的に分散させ、パフォーマンスをさらに高めることができます。このロードバランシングを活用するため、アプリケーション側では、ひとつの論理的コネクションで solidDB Cache のプライマリー・インスタンスとセカンダリー・インスタンスにアクセス可能な solidDB の ODBC ドライバーあるいは JDBC ドライバーを使用します。このとき、書き込みトランザクションはプライマリー solidDB ノードへと自動的に接続されるのに対し、読み出しトランザクションはホットスタンバイ側 solidDB ノードにのみ接続する形にもプライマリー・インスタンスとホットスタンバイ・インスタンスの間でロードバランシングを行う形にもすることができます。そのためのコードをアプリケーション側で書く必要はありません。ロードバランシングはパフォーマンスを最大で 100%、高めることができますし、ホットスタンバイ構成は障害発生時に 1 秒以内でフェイルオーバーを実現し、solidDB ノードの可用性を高めることができます。

solidDB のホットスタンバイ構成では、トランザクション・ロギングの最適化によるパフォーマンスの向上が可能です。solidDB の耐久性は適応力が高いため、リラックス・ロギングとして、耐久性を犠牲にすることなくプライマリー solidDB インスタンスのパフォーマンスを高めることができます。耐久性は、ホットスタンバイ・インスタンスとネットワーク経由で同期を取ることによって確保されるからです。どちらかのインスタンスに障害が発生した場合は、データの耐久性を確保するため、自動的に同期ロギングへ移行します。

データの安全性、アプリケーション・スループット、回復時間を柔軟にバランス

solidDB には、プライマリー・データベース・サーバーとセカンダリー・データベース・サーバーの同期方法について、高可用性を確保するさまざまな構成オプションが用意されています。しかも、この構成をランタイムにアプリケーションからクエリーし、セッションやトランザクションごとに変更することができます。データの安全性、アプリケーション・スループット、回復時間をかつてないほどの柔軟性でバランスさせることが可能となったのです。

同期ホットスタンバイ構成

絶対にトランザクションを失ってはいけないケースでは、同期複製構成とするべきです。この場合、セカンダリー・データベースから応答が返ってきたからプライマリーに対するトランザクションのコミットが行われます。同期複製となる設定は、以下のように 3 種類あります。

- セカンダリー・データベースは、プライマリー・データベースからリクエストを受けると同時に、コミット・レコードの応答を返します。
- セカンダリー・データベースは、プライマリー・データベースから受けとったトランザクション・リクエストを処理した後、ログに書き込む前に、コミット・レコードの応答を返します。
- セカンダリー・データベースは、セカンダリー・データベース側でトランザクション・ログを完全にコミットしてから、コミット・レコードの応答を返します。

どの設定を選ぶかで、総合的な回復時間とアプリケーションのパフォーマンスが変化します。

回復時間が最も短くなるのは、プライマリー・データベースにコミットされたトランザクションがすべて、セカンダリー・データベースにもコミットされるケースです。この場合、プライマリー・データベースに障害が発生しても、回復のためにセカンダリー・データベース側で行うべき作業はなく、一瞬でプライマリー・データベースを引き継ぐことができます。

非同期ホットスタンバイ構成

非同期複製構成にすると、アプリケーションが発行した Commit コマンドは、プライマリー・サーバーでトランザクションがコミットされると同時に返されます。この構成ではセカンダリー・データベース・サーバーからの応答をアプリケーションが待つ必要がないため、プライマリー・データベース・サーバー側の待ち時間がのびることがありません。

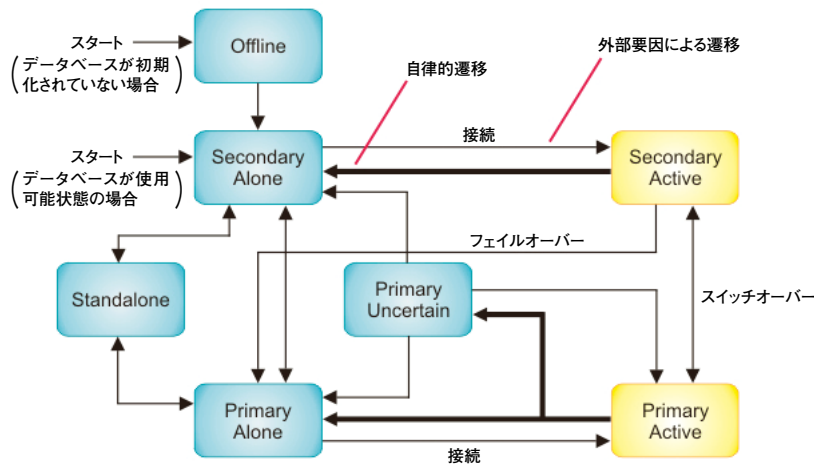


図6:高可用性のステートマシンが組み込まれているため、高可用性(HA)マネージャーとシンプルに統合することができます。

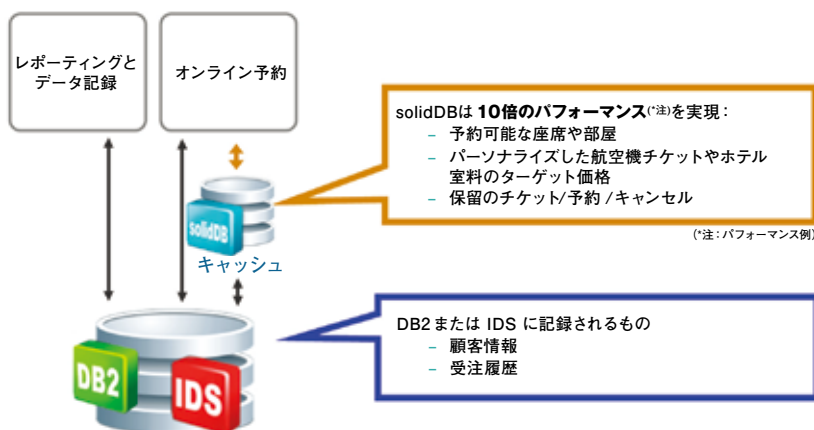
さらに、solidDB はアクティブ、スタンバイ、あるいはフェイルオーバーで他の状態になっているなど、サーバーの両方についてホットスタンバイ状態を把握しています。把握した状態はアプリケーションに公開されており、アプリケーションからクエリーすることもその状態を変更することもできます。高可用性マネージャー(HA マネージャー)は、この情報を使って solidDB インスタンスの状態をモニタリングし、必要に応じてフェイルオーバー・プロセスを指揮します。

低コスト

IBM solidDB には超高速性と究極の可用性という特長があるため、データを常に利用できる状態に保ち、計画停止や不測の停止に伴うコストを削減したいというビジネス側の要件に対応することができます。さらに、アプリケーションからの制御ができること、基本的に無人運用ができることから、すばやい展開と管理コストの削減が可能で、総所有コストをさらに引き下げることができます。ハードウェアについてもベスト・オブ・ブリードなものだけでなく、コモディティなものでも使えますから、高い実績を持ち、費用対効果が高いソリューションを選ぶことができます。

IBM solidDB Cache for DB2, solidDB Cache for IDS の使用例

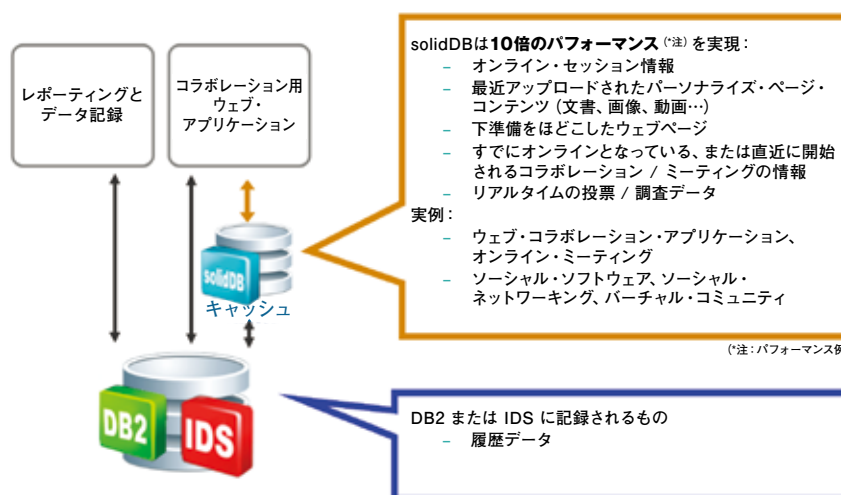
航空会社や観光業界の予約システム



パフォーマンス・クリティカルなデータへの高速アクセスに大きな価値が存在する例のひとつが、航空会社やホテル、チケット販売などの予約システムです。

大量のチケットを電話とオンラインで販売するケースを考えてみましょう。コンサートやイベントでは、いい席が取り合いになります。ローリングストーンズの最終コンサートが土曜日、朝 10 時に発売されたりしたら、チケットの販売代理店にはすさまじいワークロードがかかるはずです。世界各地の販売があるので、世界的にすさまじいワークロードとなるでしょう。このような人気チケットが発売された場合、ベンダーは需要の集中に対応し切れないことがよくあります。その結果、ウェブページがなかなかロードされなかったり、座席の予約に長い時間がかかったりします。トランザクションのタイムアウトで購入手続きを完了できないというもよくある話です。このような場合、チケットの販売に関するデータを solidDB にキャッシュしておけば、従来よりも大きな需要に短い応答時間で対応することが可能になります。

Web 2.0/コラボレーション・アプリケーション

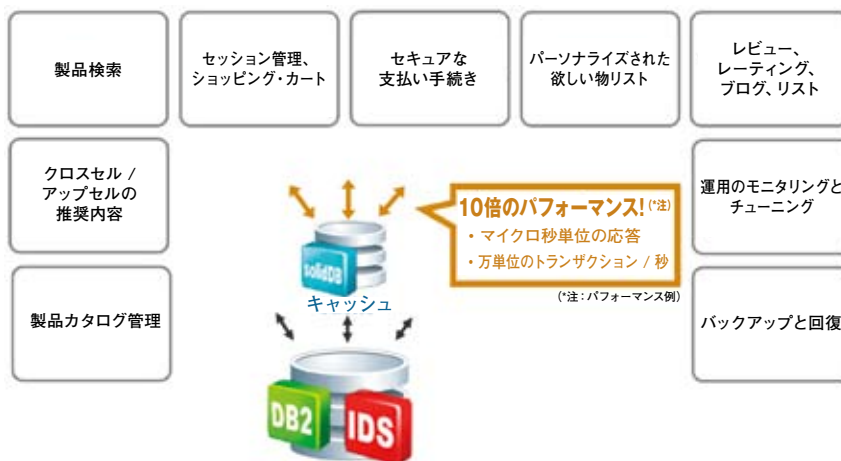


ウェブ・コラボレーションのプロバイダーは、数百ものオンライン・ミーティングを同時にホスティングします。そのお客様の中には、数万人もの参加者を集めてコミュニケーション・ミーティングを開く大企業もあります。このようなミーティングの多くは、朝 9 時から、あるいは四半期末など、一定のパターンで開催されます。ログイン情報や参加者へのアンケート、チャット・ログ、その他のミーティング・イベントなど、一部のデータを solidDB にキャッシングしておけば、優れたユーザー体験を提供しつつ、大人数のミーティングをサポートするスケラビリティを得ることができます。

ソーシャル・ネットワークが急成長していますが、こちらも、たくさんのユーザーが同時にログインするなど、ワークロードにピークが発生する場合があります。このような場合、友だちリストや最近のコメントなど、ユーザーのログイン ID に関連する動的データへは solidDB を使ってすばやくアクセスし、グループ・メンバーシップやプロフィール情報などの静的データはバックエンド・データベース側に置くといった利用が考えられます。

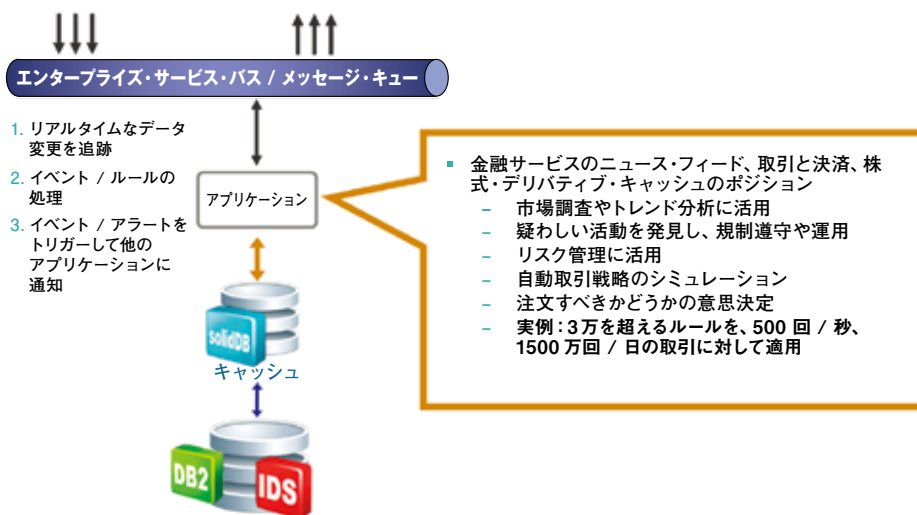
電子商取引アプリケーション

お客様と直接やりとりすることになる小売店の商品販売用電子商取引アプリケーションも、パフォーマンス・クリティカルなデータをインメモリー・データベースにキャッシングすると大きなメリットが得られる一例です。電子商取引では、製品カタログの管理、関連商品の販売につながる製品の追跡と推奨、顧客ごとの欲しい物リストの維持管理、製品検索など、さまざまなアプリケーションがインフラストラクチャーとして必要になります。



大規模な電子商取引サイトには、驚くほど多くのユーザーが同時にアクセスすることがあるため、トランザクション・スループットのスケーラビリティに十分な余裕を持たせておく必要があります。ファン待望の DVD やビデオ・ゲームが発売されたときなど、電子商取引インフラストラクチャーは、ページの閲覧、レビューの表示、ショッピング・カートの処理、精算などを同時並行に処理することになります。大規模電子商取引サイトを実現するためには、多種多様なテクノロジーが必要です。中でもデータ管理が重要な役割を果たします。なぜなら、アップセリング(グレードの高い製品への誘導など)やクロスセリング(関連商品の販売促進)ができるためには、データベースに格納された動的データにすばやくアクセスできる必要があるからです。画像や動画といった静的コンテンツをキャッシングしてパフォーマンスを高めるソリューションは市場に数多く存在しますが、いずれも、電子商取引アプリケーションで発生するさまざまな動的リレーショナル・データには対応できません。solidDB 製品ファミリーなら、データへのアクセスに要する時間を短縮し、ダイナミックなウェブページをすばやく表示することができます。こうして、ユーザー体験が改善されるとともに、ウェブ店舗への来訪者からあがる収益を増やすことが可能になります。

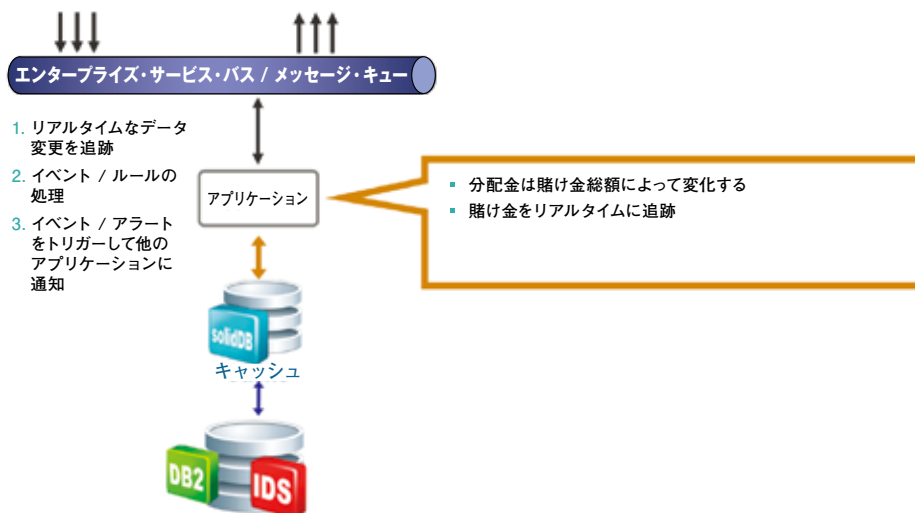
金融サービス・アプリケーション



金融サービスでは、株式売買、リスク管理など、金融サービス・プロバイダーや消費者にとって重要なさまざまな側面において、リアルタイム・アラートやイベントをトリガーするアプリケーションが数多く使われています。

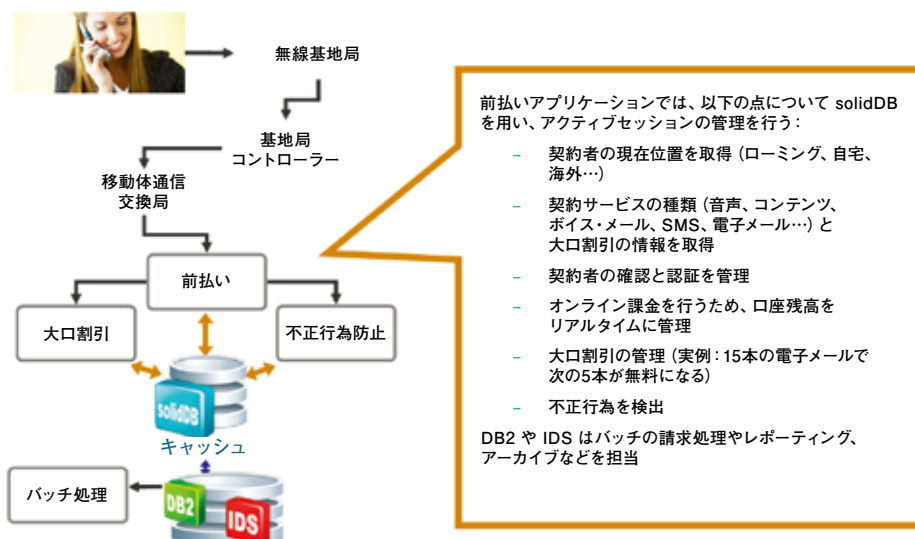
この分野では、一般に、WebSphere などのエンタープライズ・サービス・バス (ESB) を用いてリアルタイムなデータの変更を追跡し、あらかじめ定められたルールに従い、適切な通知を他のアプリケーションに出す形となっています。トランザクションをリアルタイムに行うアプリケーションから solidDB を使い、ESB から受けとる中間データを記憶するようにすれば、プリセット・ルールの処理を超高速に行えるようになります。

オンライン・ゲーム



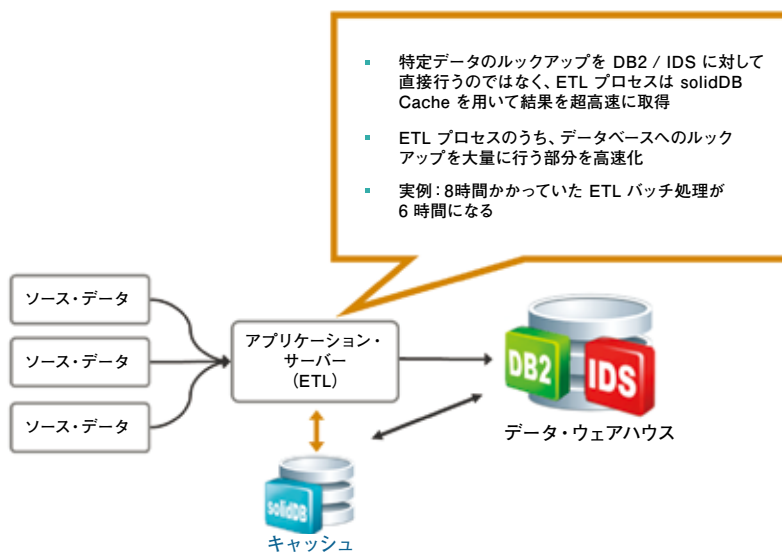
solidDB を使うと、データの変更をアプリケーションからリアルタイムに追跡することができます。この機能は、ゲームなどのアプリケーションで特に重要な意味を持ちます。ゲームでは参加者数と賭け金総額からオッズをリアルタイムに算出するからです。データをインメモリーに置いてアクセスできれば、不審な賭け方をリアルタイムに検出するなど、リスク管理上のメリットも得られます。得られた結果は DB2 や IDS に保存し、経理やアーカイブに使用します。

オンライン課金処理の実装



通信サービスで前払い、後払い、あるいは都度払いの処理をリアルタイムに行うオンライン課金のアプリケーションの場合も、solidDB に大きな価値があります。オンライン課金を処理するためには、クレジット残高、割引、料金、ロイヤルティ・ポイントなど、さまざまなデータをチェックした上で、サービスへのアクセスをリアルタイムに認証する必要があります。この場合、複数のネットワーク要素から solidDB を使って情報をリアルタイムに集めれば、事業者は、使用量(パケット数/バイト数、イベント数、時間など)に応じた料金を算出したり、契約者の位置、時間帯、コンテンツ、イベントの種類などに応じたポリシーによる調整を行ったりすることができます。このような処理を行ってから、サービスを提供したり超高速アクセスを許可したりするわけです。更新されたデータがあれば、DB2 や IDS へ伝達し、バッチ処理による請求書作成、レポート生成、アーカイブなどに備えます。

データをデータウェアハウスへロードする



ETL という言葉があります。これは、複数の業務システム (CRM、ERP、受注管理など) からデータを抽出し、変換して、データウェアハウスにロードするプロセスを指しています。これはバッチ・プロセスで、通常、データウェアハウスに保存されているレファレンス・テーブルに対し、数百万回ものデータ・ロックアップを行う必要があります。このテーブルを solidDB にキャッシュすれば、データへのアクセス時間がミリ秒からマイクロ秒に短縮されるので、ETL プロセスのパフォーマンスを何桁も高めることができます。

IBM solidDB の主な機能

堅牢なインメモリー・リレーショナル・データベース

- ・ SQL-92 エントリー・レベルをサポート。SQL-98 と SQL 2003 も一部機能をサポート。
- ・ ACID フルサポート
- ・ 複数の solidDB テーブル・タイプをサポート
 - パーシスタント・インメモリー・テーブル
 - トランジェント・インメモリー・テーブル
 - テンポラリー・インメモリー・テーブル
 - ディスク・ベース・テーブル
- ・ トランザクション分離レベルを設定可能な同時実行制御
- ・ スナップショットで整合性が確保されるチェックポイント機能、トランザクション・ロギング、自動ロールフォワード回復
- ・ 優れたコストベースのクエリー最適化
- ・ マルチプロセッサ・アーキテクチャーやマ

ルチコア・アーキテクチャーを活用するマルチスレッド対応データベース・エンジン

- ・ テーブル、ビュー、インデックス、シーケンス
- ・ 主キー、外部キーなどを含め、完全性制約条件を動的に制御
- ・ SQLによるストアド・プロシージャー、トリガー、ユーザーが構成可能なイベントによりプログラミングが可能
- ・ ユーザー特権、ロール特権によってセキュリティを構成可能
- ・ さまざまな言語の文字を表示できる Unicodeをサポート
- ・ 診断ツールとパフォーマンス・モニタリング・ツール
- ・ データのバルクロードやエクスポートが行えるツール

・ オンライン・バックアップ、ネットワーク・バックアップ、ステータス・レポートなどの管理タスクを起動できるスケジューラーをビルトイン

IBM DB2、IDS との統合

- ・ solidDB Cache コネクター
 - DB2 や IDS からデータを solidDB Cache へロード可能
 - 変更点を solidDB Cache から DB2 や IDS へ伝達可能
 - 複数の solidDB Cache インスタンスへのスケールアウトをサポート
- ・ DB2 for Linux, Unix and Windows をサポート
 - DB2 Enterprise Server Edition 9.5
 - DB2 Enterprise Server Edition 9.1



- ・ DB2 for z/OS をサポート
 - DB2 9 for z/OS
 - DB2 for z/OS バージョン 8
- ・ Informix Dynamic Server をサポート
 - IDS 11.5 Enterprise Edition
 - IDS 11.1 Enterprise Edition

高可用性とロードバランシング

複製とロギングの設定が構成可能なホットスタンバイ構成

- ・ アプリケーションから、ひとつの論理的コネクションでプライマリー・インスタンスとセカンダリー・インスタンスの両方にアクセスできる透過的な接続性、および solidDB のプライマリー・インスタンスとホットスタンバイ・インスタンスに読み出し操作の負荷を分散させる構成可能なロードバランシングを実現
- ・ solidDB インスタンスの状況をモニタリングし、フェイルオーバーを行う高可用性コントローラーを実現
- ・ 動的クエリーとプログラムからの構成が可能な高可用性ステート・マシンを内蔵しており、サードパーティ製高可用性マネージャーとシンプルな統合が可能

優れた複製機能

- ・ データの整合性を確保するため、solidDB インスタンス間で、パブリック/サブスクライブによる双方向性複製機能を搭載
- ・ データベース全体の複製、テーブル単位の複製、行単位の複製、列単位の複製をサポート

組込が可能

- ・ ネットワーク経由でアプリケーションがアクセスできるように、サーバー・プロセスとして展開することが可能
- ・ solidDB と同じサーバー上に展開された Java アプリケーションや C/C++ アプリケーションとリンクさせてパフォーマンスを高めるとともに、組込展開として solidDB を見えなくすることが可能

オープンな標準準拠のインターフェース

- ・ ODBC 3.51 準拠のドライバー
- ・ JDBC 2.0 準拠の Type 4 ドライバー

- ・ テーブルへの直接アクセスが可能な Solid SA C 言語クライアント・ライブラリー

solidDB、solidDB Cache for DB2、solidDB Cache for IDS がサポートするプラットフォーム

- ・ Linux :
 - Red Hat Enterprise Linux (RHEL) 4 および 5
 - SUSE Linux Enterprise Server (SLES) 9 および 10
- ・ Microsoft Windows :
 - 32 ビット版と 64 ビット版(x64) の Windows Server 2003 Edition、Standard Server Edition、Enterprise Server Edition、Datacenter Edition
 - 32 ビット版と 64 ビット版(x64) の Windows XP Professional Edition
 - 32 ビット版と 64 ビット版(x64) の Windows Vista Business Edition、Enterprise Edition、Ultimate Edition
- ・ IBM AIX :
 - AIX 5L™ V5.3 for POWER5/POWER6
- ・ HP-UX :
 - 64 ビット HP Integrity サーバー用 HP-UX 11i v2 (Itanium ベースのシステム)
- ・ Sun Solaris
 - UltraSPARC サーバー用および x86 サーバー用の Solaris 10

ハードウェア要件

- ・ 32 ビット版と 64 ビット版(x64) の Windows と Linux : Intel® プロセッサあるいは AMD プロセッサを搭載し、オペレーティング・システムとしてサポートする Windows あるいは Linux が稼働するシステム (x86 システムおよび x64 システム)。
- ・ POWER5™ 以降のプロセッサを搭載した AIX® 64 ビット・システムが必要です。
- ・ Solaris : UltraSPARC プロセッサあるいは x86 プロセッサを搭載した 32 ビットあるいは 64 ビットのシステムが必要です。
- ・ HP-UX : Itanium を搭載した HP Integrity シリーズのシステムが必要です。

詳細情報

IBM solidDBの詳細については、以下のサイトをご覧ください。

ibm.com/jp/software/data/soliddb/ IBM製品、詳細情報については、IBMホームページ <http://www.ibm.com/jp> をご利用ください。

お問い合わせは、IBMビジネス・パートナー、製品販売店、弊社営業担当員または、ダイヤルIBM (☎0120-04-1992)へ。
受付時間:月～金 9:00～18:00 (祝日、12/30～1/3を除く)
携帯電話でおかけのお客様は、下記の電話番号をご利用ください。
ダイヤルIBM 03-6220-8002 (この場合通話料はお客様のご負担となります。)

© Copyright IBM Corporation 2008

日本アイ・ビー・エム株式会社
〒106-8711 東京都港区六本木 3-2-12

Produced in Japan
July 2008
All Rights Reserved

DB2、IBM、IBM ロゴおよび solidDBは、International Business Machines Corporation の米国およびその他の国における商標または登録商標です。

Intel は、Intel Corporation の米国およびその他の国における商標または登録商標です。

Java およびすべての Java 関連の商標およびロゴは Sun Microsystems, Inc. の米国およびその他の国における商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

他の会社名、および製品名はそれぞれ各社の登録商標または商標です。

本書において、IBM製品、またはサービスについて言及または説明する場合があります。しかし、このことは、IBMが営業を行っているすべての国においてこのような製品、またはサービスが利用可能であることを必ずしも示すものではありません。また、内容は、予告なしに変更する場合があります。

このカタログの情報は、2008年7月現在のものです。