Oracle™9i Real Application Clusters (RAC) のマーケティング メッセージに明示されていない事実

2002 年 10 月

要約 :

この文書では、Oracle9i Real Application Cluster（RAC）**の主な機能と、それらを実装するための条件について解説します。また、**Oracle **がマーケティング メッセージを通じて主張しているさまざまな機能について、事実に合致する点とそうでない点を独自に調査した結果で示します。**

Microsoft

このドキュメントに記載されている情報は、このドキュメントの発行時点におけるマイクロソフトの見解を反映したものです。変化する市場状況に対応する必要があるため、このドキュメントは、記載された内容の実現に関するマイクロソフトの確約とはみなされないものとします。また、発行以降に発表される情報の正確性に関して、マイクロソフトはいかなる保証もいたしません。

このドキュメントに記載された内容は情報提供のみを目的としており、マイクロソフトは本書に記載されている情報について明示的にも暗黙的にも一切の保証をいたしません。

お客様ご自身の責任において、適用されるすべての著作権関連法規に従ったご使用を願います。このドキュメントのいかなる部分も、米国 Microsoft Corporation の書面による許諾を受けることなく、その目的を問わず、どのような形態であっても、複製または譲渡することは禁じられています。ここでいう形態とは、複写や記録など、電子的な、または物理的なすべての手段を含みます。ただしこれは、著作権法上のお客様の権利を制限するものではありません。

マイクロソフトは、このドキュメントに記載されている内容に関し、特許、特許申請、商標、著作権、またはその他の無体財産権を有する場合があります。別途マイクロソフトのライセンス契約上に明示の規定のない限り、このドキュメントはこれらの特許、商標、著作権、またはその他の無体財産権に関する権利をお客様に許諾するものではありません。

Windows® および Windows NT® は米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

記載されている会社名、製品名には、各社の商標のものもあります。

目次

概要

Oracle 社はこれまで、同社の Real Application Cluster（RAC）テクノロジについて、今日の企業において求められるスケーラビリティと高可用性の要件を満たす総合ソリューションであるとしてきました。しかし、RAC の公開から 2 年が経過した現在、以下の問題が指摘されています。

- **RAC は、これまでの Oracle 製品と同様に、従来からの Windows ユーザーにとって所有コストが高くつきます。Oracle が主張する省コスト効果というのは、主に従来の Unix プラットフォームから Intel ベースのプラットフォームに移行する際に得られる、ハードウェアに関する理由によるものです。**

システムの実際の展開状況を考えれば、システムに高いパフォーマンスと信頼性を要求するユーザーにとって最も効率的な総所有コストを実現するのは、依然として SQL Server です。Oracle9i RAC の導入によって期待できる省コスト効果は、ユーザーが高価な UNIX システムから Intel ベースの安価なシステムに移行する場合に限ってのことであり、ハードウェア上の理由のみによるものです。既に Intel ベースのサーバーを使用しているユーザーが RAC に移行しても、コストの低減は期待できないでしょう。実際には、移行に際して以下のコストが生じるため、RAC によって総所有コストが増加する可能性も十分にあります。

  - サーバーやネットワーキング ハードウェアの追加
  - ストレージ ソリューションの購入 (付属しているストレージは、RAC に対応していません)
  - Oracle データベース ライセンスの追加 (データベースや必要なオプションに対するライセンスは、ノードごとに必要になります)
  - オペレータや運用管理者の訓練
  - 移行や証明に必要なサービスの導入

  これに対して SQL Server 2000 は、総所有コストが最も低いだけでなく、パフォーマンスにおけるコスト効率においても優れています。また、インテリジェントで動的な自動リソース管理機能により、安心して使用でき、管理コストも低減されます。

- **RAC のスケーラビリティは、アプリケーションのトランザクション量がそれほど多くない場合にのみ実現するものです (どの業界標準ベンチマークにおいても、主導的な立場を得てはいません)**

  Oracle は、約 2 年前の製品発表以来、RAC のスケーラビリティについてさまざまな主張をしていますが、これまでのところ、TPC-C や TPC-W などの業界標準ベンチマークにおいても、SAP などの LOB アプリケーション ベンチマークにおいても、いまだトップの地位を実現しておらず、主張内容の具体的な証明はなされていません。いくつかのベンチマークでは (結果の信頼度はそれぞれの分析力によるところですが)、Oracle がマーケティング関連のさまざまなプレゼンテーションや文書で主張してきた内容とは程遠い結果も出ています。つまり、現在のところ、RAC は Oracle が断言している利点をまだ実現できていません。

  これに対して SQL Server は、TPC-C、TPC-W、SAP、Onyx など、さまざまな業界ベンチマークでトップの地位を一定期間維持することにより、そのスケーラビリティを証明しています。また、これらのベンチマークにおいて、Oracle は毎回の結果にあまり変化が見られないのに対し、SQL Server は今後も成績を伸ばし続けることが予想されます。

- **RAC は、導入するだけで自動的に高可用性が得られるものではありません。ほとんどのアプリケーションでは、トランザクションのフェールオーバーは自動化されないでしょう。**

  RAC で実現される可用性のレベルは、ここ数年の間一般に利用されている他の技術と同程度のものです。最近では他のベンダでも同様の手法が使用されており、RAC で独自に採用されている手法や高度なテクノロジはありません。確かに、場合によっては RAC でプロセスを簡素化できることもあります (たとえば、クラスタにノードを追加または削除するプロセスはかなり簡素化されます)。しかし、画期的と言えるような新しいテクノロジは一切導入されておらず、高可用性のレベルを高めるものにもなっていません。また、他のサーバーにトランザクションを再送信することは可能なので、トランザクションが完全に行えなくなることはありませんが、自動的な再接続は実行されず、現在出回っているほとんどのシステム (TAF 機能が採用されていないシステム) では、トランザクション状態データも失われます。

この文書では、Oracle が RAC に関して主張している主な性能である、スケーラビリティ、可用性、および総所有コストについて詳細に分析しながら、RAC の実際の動作の仕組みを説明すると共に、Oracle のマーケティング メッセージからは読み取れない事実を明らかにしていきます。

はじめに

この文書では、Oracle9i の Real Application Cluster (RAC) テクノロジと、その主な機能、および長所や欠点についての、ハイレベルな分析結果を提供します。また、RAC のスケーラビリティ、可用性、および総所有コストに関して Oracle が発表しているさまざまな点についての検証も行います。この文書の主な目的は、Oracle のマーケティング メッセージの内容をそのまま受け取るのではなく、筆者が調査、テスト、および測定した結果に基づき、RAC に関するさまざまな事実を読者に提供することです。

この文書は、RAC についての包括的なテクニカル ホワイトペーパーではありません。また、RAC やその関連テクノロジについての詳細な情報をすべて記載したものでもありません。なお、必要に応じて、他のテクノロジについての説明も含まれています。

対象読者

この文書は、エンタープライズ データベース管理システムについて知識のある方を対象としています。ビジネスや技術に関する意思決定を行う際に、マーケティング メッセージではなく事実に基づいて適切な判断を下すための資料にしていただくことを想定しています。

Oracle9i Real Application Clusters の概要

Oracle は、Oracle 9i データベース製品のリリース時に Real Application Clusters (RAC) を発表し、スケーラビリティと信頼性の両方を実現する企業ソリューションとして同テクノロジを位置付けました。基本的には、RAC は、9i 以前のバージョンの Oracle で提供されていた Oracle Parallel Server (OPS) というスケールアウト クラスタリング テクノロジのアップデート バージョンです。

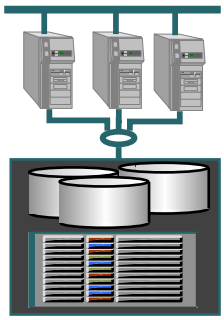


図 1:共有データ

Oracle9i RAC

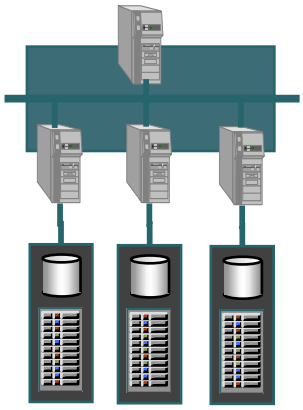


図 2:連合データ

SQL Server 2000

上の図 1 は、RAC のアーキテクチャを大まかに示したものです。基本的に、RAC は、共有ディスク/共有キャッシュ型のアーキテクチャに基づいた、制限のあるスケールアウト システムです。この種のシステムでは、単一のデータベース内に実際のデータのコピーが 1 つだけ存在し、それを 1 つ以上のデータベース サーバーによって処理します。すべてのサーバーは、リソース、ロック、アクセスなどを管理するためのさまざまなメカニズムに従いながら、同じコピー データに対して処理を実行します。

一方、図 2 は、SQL Server 2000 におけるスケールアウトの実装方法を示したものです。複数の独立したデータベースを 1 つのグループに連合させ、ユーザーからは単一のデータベースとして利用できるようにしますが、実際には 2 つ以上の物理的なサーバーやデータベースを配置することができます。本来、この概念を説明するには、上記の説明よりもはるかに詳細な説明が必要になりますが、この文書の目的は SQL Server 2000 におけるスケールアウトの実装方法を説明することではないので、ここではこれ以上の説明は控えます。詳細については、http://www.microsoft.com/japan/msdn/library/ja/createdb/cm_8_des_06_17zr.asp を参照してください。

RAC の仕組み

前のセクションでは、RAC のアーキテクチャについて大まかに説明しました。次に、RAC がどのような仕組みで機能するかについて説明します。

RAC では同じ Oracle9i データベース エンジンを使用していますが、RAC に固有のキー コンポーネントがいくつかあります。次に示すのは、RAC 固有のコンポーネントです。各コンポーネントの機能については、このセクションの後の方で説明します。

- クラスタ マネージャ
- グローバル キャッシュ サービス
- グローバル エンキュー サービス
- クラスタ インターコネクトとプロセス コミュニケーション (ノード間)
- データベースの静止機能
- 共有ディスク サブシステム

下の図 3 は、2 つのノードを持つ典型的な RAC 構成を示したものです。構成を拡張してノードを追加する場合、追加される各ノードは、既存のすべてのノードおよび共有ディスク サブシステムに接続される必要があります。すべてのノードをつなぐ Interconnect は、ハブまたはスイッチを経由して接続された、各ノードの標準の NIC (ネットワーク インターフェイス カード) を使用して実装されます。共有ディスクとの接続には、HBA (ホスト バス アダプタ) または NIC (ネットワーク インターフェイス カード) を使用します。どちらを使用するかは、SAN (ストレージ エリア ネットワーク) と NAS (ネットワーク接続ストレージ) のどちらのテクノロジを使用しているかによって決まります。

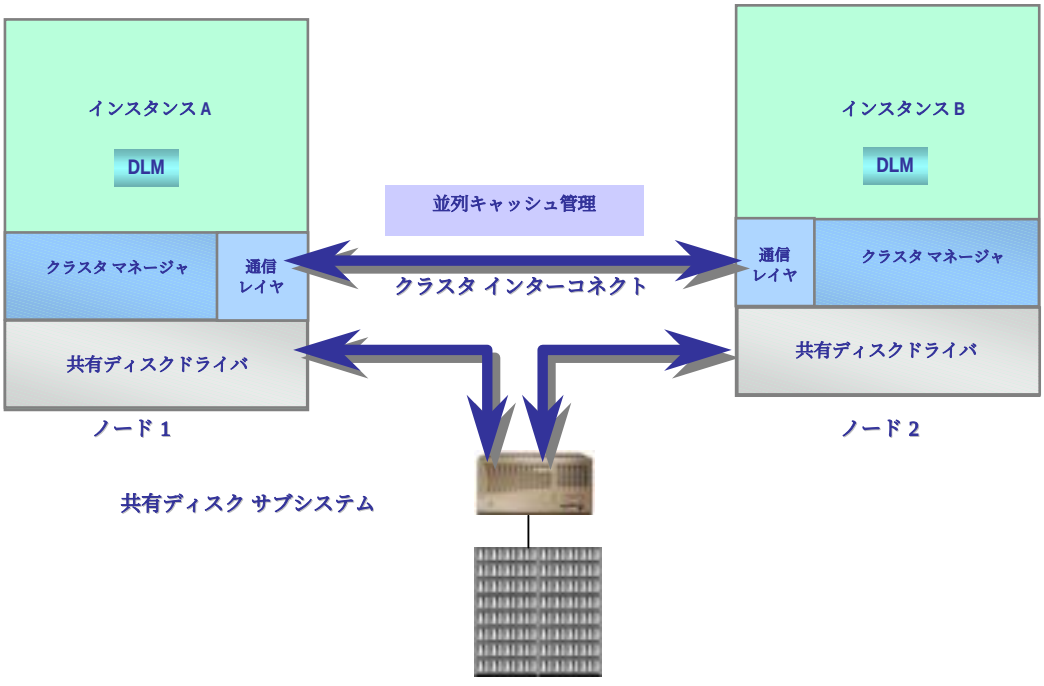


図 3: 2 ノード構成の RAC システム例

システム内に送信されたトランザクションは、手動か、またはクラスタ エイリアス (Compaq クラスタ サービスの 1 機能) 経由で、任意のアクティブなノードに送られます。クラスタ エイリアスを使用すると、トランザクションは利用可能なノードに自動的に転送されます。各インスタンスのバッファ キャッシュ間における、データブロックのステータスと転送の管理は、グローバル キャッシュ サービスによって行われます。なお、このコンポーネントは、グローバル リソース ディレクトリ内のリソース情報を参照できるように、バッファ キャッシュ マネージャと統合されています。グローバル リソース ディレクトリは、リソース (全ノード間での調整が必要なすべてのデータ ブロックなど) のステータス情報を管理するもので、すべてのインスタンスに配布されます。

特定の列 (1 つまたは複数) を要求するトランザクションが発生すると、要求を受けたノードはまずローカル キャッシュをチェックし、要求された列がキャッシュされているかどうかを確認します。ローカルにキャッシュされている場合は、要求された列が呼び出し元に返され、トランザクションは終了します。ローカルにキャッシュされていない場合、そのノードは、クラスタ内のその他いずれかのノードに目的の列がキャッシュされていないかどうかをチェックします。目的の列がいずれかのキャッシュ内に見つかった場合は、そのキャッシュ ブロックを最初のノードに送るための一連のプロセスが開始され、その後、結果がユーザーに返されます。これにより、呼び出し元のノードが目的のブロックを取得するために数回の I/O 動作を実行しなければならなかった OPS での問題点が一部解消されています。ただし、要求された列がどのノードにもキャッシュされていない場合は、通常の読み取り動作が実行されます。

ディスク I/O が実行されるのは、必要なデータがどのキャッシュにも含まれていない場合と、更新トランザクションで、ディスクへの書き込みを必要とする コミット処理が実行される場合だけです。

複数の Oracle インスタンスがデータベース内の同じ列にアクセスする場合、起こりえる状況は以下の 4 とおりになります。なお、以下の例は、説明を単純化するために、2 つのノード (ノード 1 とノード 2) を持つクラスタを想定したものとなっています。

Read/read — ノード 2 のユーザーが少し前に読み取ったブロックを、ノード 1 のユーザーが読み取ろうとする場合です。読み取り要求は、クラスタ データベース内のいずれかのキャッシュを使用して処理されます。アクセスの優先順位は、ローカル キャッシュ、リモート キャッシュ、最後にディスク I/O となります。

Read/write — ノード 2 のユーザーが少し前に更新したデータを、ノード 1 のユーザーが読み取ろうとする場合です。この "read/write" の場合と、ノード 1 のユーザーがブロックを更新する "write/write" の場合においては、インスタンス間での同期化が必要になります。これは、読み取られるブロックに "一貫性読み取り" 用データを使用したり (read/write の場合)、更新されるブロックでデータの整合性を保ったり (write/write の場合) する必要があるためです。どちらの場合も、更新されたデータ ブロックを保持するノードが、クラスタの高速な Interconnect を介して要求元のノードにブロックを送ります。それにより、読み取りのための非効率なディスク I/O が回避されます。また、この動作は、完全なリカバリ機能のサポートの下で実行されます。

Write/read — ノード 2 のユーザーが少し前に読み取ったデータを、ノード 1 のユーザーが更新しようとする場合です。更新処理では通常、関連するブロックがまずメモリ内に読み取られ、その後、更新されたブロックがディスクに書き込まれます。この例では、リモート インスタンス (ノード 2) によって既に読み取られているキャッシュ済みブロックを、ノード 1 が更新することになります。要求されたブロックはノード 2 のキャッシュからノード 1 のキャッシュに送られるので、読み取りのためのディスク I/O が回避され、パフォーマンスが向上します。

Write/write — ノード 2 のユーザーが少し前に更新したブロックを、ノード 1 のユーザーが更新しようとする場合です。上の read/write の説明を参照してください。

ノード間でのメッセージングの効率を左右する主な要素は、次の 3 つです。

- 同期化シーケンスごとに必要となるメッセージの数
- 同期化の頻度 (頻度が低いほど効率が上がる)
- ノード間通信の待ち時間と速度

同期化に必要なメッセージの数

ノード間の同期化に必要となるメッセージの数は、クラスタ データベースに実装される グローバル エンキュー サービス（GES）と複雑に関連します。GES は、どのデータベース リソースがどのノードにキャッシュされているかを追跡します。GES はまた、ユーザー プロセスによって実行されるデータ操作の同期をとるために、リソースの割り当てと割り当て解除を管理します。

また、キャッシュ間での高速なブロック転送は、グローバル キャッシュ サービス（GCS）によって管理されます。GCS は、最大でも 2 つのノード間メッセージと 1 つのノード内メッセージのみを使用することにより、高速転送を実現します。

ユーザー プロセスによって特定のデータが要求されると、メモリ内で グローバル リソース ディレクトリ（GRD）に対する検索が実行されます。関連するブロックがローカルまたはリモートのノードにキャッシュされていれば、目的のデータはすばやく返されます。なお、要求の処理方法は、状況に応じて次のように分岐します。

- データがローカルまたはリモートのキャッシュにない場合は、通常のディスク I/O が実行されます。それにより、処理が遅くなります。
- データがローカル キャッシュにある場合は、メッセージを一切生成することなく、即座にデータにアクセスできます。
- データがリモート キャッシュにある場合は、通常、次の処理が実行されます。
  - ユーザー プロセスが、リモート ノード上の GCS プロセスに対するノード間メッセージを生成します。
  - 対応する GCS プロセスとインスタンス プロセスが、メモリ内の GRD を更新し、要求元のユーザー プロセスにブロックを送ります。場合によっては、さらに別のインスタンスへのメッセージが送られることもあります。

それぞれの同期処理は最低限の通信によって実行され、それにより高速な応答が実現します。しかし、数千のユーザー プロセスがシステムにアクセスする可能性のある e ビジネス アプリケーション環境の場合、この方式で適応することは可能なのでしょうか。これが可能になるかどうかの鍵は、Oracle のノード間リソース モデルにあります。

ノード間メッセージの待ち時間

Cache Fusion が有効に機能するには、待ち時間の少ない通信プロトコルが必要です。したがって、RAC の機能性や有用性は、クラスタの Interconnect やその他の関連コンポーネントのパフォーマンスとキャパシティに大きく依存していると言えます。

まとめ

RAC は、プロセスの負荷を複数のサーバーに分散することによって機能するものですが、結局、それらのサーバーはすべて、同じストレージ システム上に存在する単一のデータベースに対して動作する仕組みになっています。また、キャッシュやデータの整合性が保たれるかどうかは、クラスタの Interconnect に依存します。これらのコンポーネントの中に 1 つでも欠点や障害を抱えたものが存在すれば、その問題はシステム全体に影響します。

以下のセクションでは、スケーラビリティ、可用性、および総所有コストに関する Oracle の主張に深く踏み込んで、Oracle のマーケティング メッセージには明示されていない事実について説明します。

Oracle の主張 :"RAC はスケーラブルである"

Oracle は、RAC について、画期的なスケーラビリティを提供するテクノロジであるとしています。ユーザーは必要に応じていつでも簡単にノードを追加でき、事実上無限のスケーラビリティが実現するとしています。しかし、発表後約 2 年が経過した現在、RAC の評価は、スケーラビリティを判定する各業界標準ベンチマーク (TPC-C、TPC-H、TPC-W など) においても、主要な業種別アプリケーション ベンチマーク (SAP R/3 Sales and Distribution、SAP R/3 Retail、J.D. Edwards OneWorld、Pivotal eRelationship など) においても、ほとんど伸びていません。

Oracle は、TPC-C のクラスタ ベンチマークと SAP-SD ベンチマークを発表しましたが、いずれも現在の世界記録 (多くは SQL Server が保持) には及ばず、それに近いと言えるものでもありませんでした。実際、ベンチマークにおける Oracle の最高結果はすべて、RAC ではなく、単一のサーバー システム上で動作する Oracle9i に基づくものです。

事実 :アーキテクチャがスケーラブルでない

RAC が Oracle の主張する性能を果たしていない理由の 1 つは、RAC のアーキテクチャそのものにあります。OPS で障害となっていたディスク I/O プロセスが排除されてはいるものの、RAC は、問題を根本的に解決しているわけではなく、問題を、許容限度を少し上げているだけです。RAC の仕組みに関するセクションで既に説明したように、RAC のデザインでは、クラスタの Interconnect が障害になることは明白です。このことは何らかのかたちで表面化するはずであり、このような方法でスケーラビリティの問題に対処しても、RAC テクノロジがその限界に達するのは時間の問題です。

高ボリュームの OLTP 環境などのように、各ノードが非常に多くのトランザクションを処理しようとする環境では、ブロック要求を処理するために Interconnect を通るトラフィックと、それに関連するすべてのプロセス間通信の量が、ノードの NIC (Network Interface Card) や、Interconnect 用のスイッチやハブの許容量を超えてしまいます。また、Interconnect の待ち時間は、SMP (Symmetric Multi-Processing) システム バスの待ち時間を大きく上回ることが頻繁にあります。より高性能の NIC やスイッチを使用したとしても、許容限度が少し高くなるだけで、問題そのものが解決するわけではありません。問題の根本的な原因は RAC の構造にあります。ストレージや Interconnect の中を多量のリソースが移動するため、最終的には移動のためのすきまがなくなってしまいます。OpenWorld 2001 と Oracle International User Group conference 2001 での発表内容によると、RAC の構成は、1 ノードあたり 4 CPU とし、クラスタのノード数を 4 個から 8 個にするのが最適であるとされていますが、一番の理由は、おそらく上記のような構造上の問題だと考えられます。つまり、ノードあたりの CPU を増やしたり、クラスタ内のノード数を増やしたりすると、通信や要求の量が大幅に増え、アーキテクチャの処理限度を超えてしまうということでしょう。

次の図は、ノード数が増えていくと、各ノードの生成するトランザクションが Interconnect の許容限度を超えてしまうことをわかりやすく示したものです。またこの図からは、ノード数の多いシステムの複雑さや、その維持に要するコストや労力についても理解できます。この図は、Oracle が最近発表した TPC-C ベンチマークのエグゼクティブ サマリーに含まれていたもので、ベンチマークに使用された RAC システムの実際のアーキテクチャを示すものです。

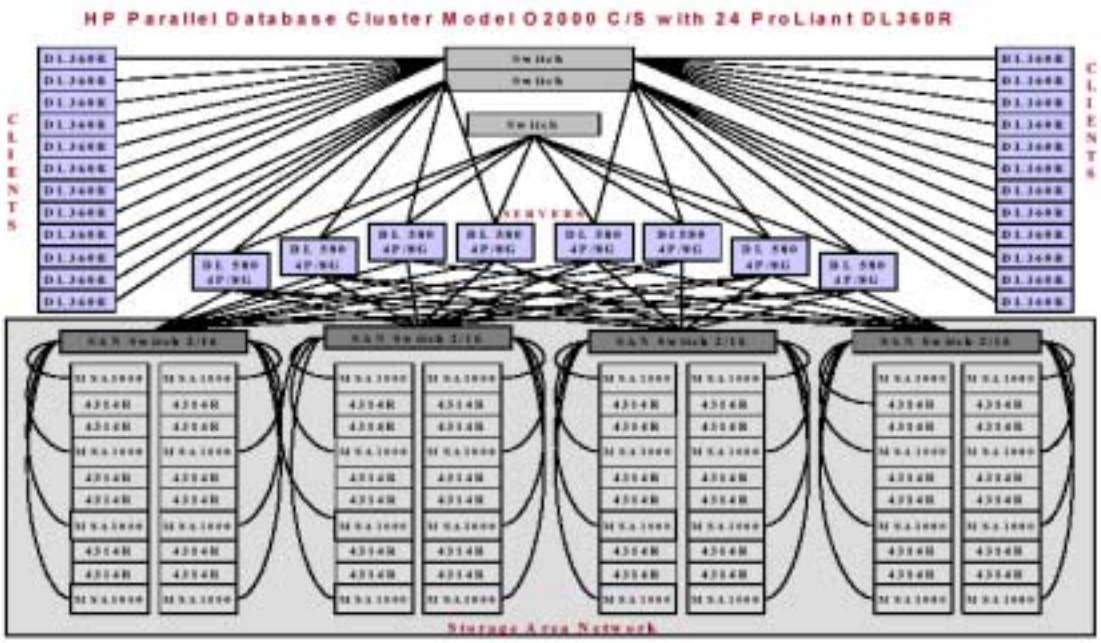


図 4 Oracle 9i RAC TPC-C **ベンチマーク システム アーキテクチャ**

HP ProLiant DL580-0200032P c/s
システムの総コスト :US$2,533,095; TPC-C Throughput:137,260.89tpmC;
コスト パフォーマンス :US$18.46; **使用可能日** :2002 **年** 9 **月** 6 **日**

ベンチマークのエグゼクティブ サマリーの図 :

www.tpc.org/results/individual_results/HP/HP_PDCO2000_ES.pdf

2002 **年** 9 **月** 15 **日の結果**

発信元www.tpc.org

まとめ

Oracle は、約 2 年前の製品発表以来、RAC のスケーラビリティについてさまざまな主張をしていますが、これまでのところ、TPC-C や TPC-W などの業界標準ベンチマークにおいても、SAP SD などの LOB アプリケーション ベンチマークにおいても、いまだ世界記録のパフォーマンスを実現しておらず、主張内容の具体的な証明はなされていません。いくつかのベンチマークでは (結果の信頼度はそれぞれの分析力によるところですが)、Oracle がマーケティング関連のさまざまなプレゼンテーションや文書で主張してきた内容とは程遠い結果も出ており、一般的に主導的な立場を確立するには至っていません。つまり、現在のところ、RAC は Oracle が断言している利点をまだ実現できていません。

これに対して SQL Server 2000 は、TPC-C、TPC-W、SAP、Onyx など、さまざまな業界ベンチマークでトップの地位を一定期間維持しており、スケーラビリティを証明しています。また、これらのベンチマークにおいて、Oracle は毎回の結果にあまり変化が見られないのに対し、SQL Server は今後も成績を伸ばし続けることが予想されます。さらに、スケールアウト ベンチマークに関して言えば、SQL Server がノード追加におけるスケーラビリティで結果を出している (16、24、および 32 ノード) のに対し、Oracle RAC は同様の結果を出しておらず、RAC のアーキテクチャそのもののスケーラビリティに疑問が生じています。

Oracle の主張 :"RAC は高可用性を実現する"

Oracle 社では、システムのダウンタイムを回避する RAC の機能や、高可用性に関する RAC のその他のさまざまな機能について主張しています。しかし、マーケティング メッセージの裏に隠れた事実を調べると、これらの機能がそれほど魅力的なものではないことが判明します。

事実 :ダウンタイムを完全に回避することはできない

Oracle 社では、RAC を使用することによって、メンテナンスやハードウェア障害によるダウンタイムを回避するシステムを構築できるとしています。しかし残念ながら、この主張は部分的に真実であるとしか言えません。

RAC では、セットアップや構成を適切に行い、すべてのハードウェア コンポーネントに十分な冗長性を持たせさえすれば、ハードウェア障害に対するフォールト トレランスが提供されます。しかし、パッチを適用する必要がある場合 (Oracle を含むすべてのデータベースにおいて珍しいことではありません) には、通常、Oracle 内のデータ ディクショナリを更新する必要があります。この更新作業を実行するにはシステムを制限モードにする必要があるため、作業が完了するまでエンドユーザーはデータベース内のデータにアクセスできません。したがって、システムがダウンするわけではありませんが、エンドユーザーへのサービス提供は事実上停止します。データ入力をしているエンドユーザーや、e コマース サイト上でトランザクションを実行しようとしているエンドユーザーにとっては、システムがオフラインになるのと変わりありません。

オペレーティング システム レベルで言えば、ローリング アップグレードを実行することは確かに可能です。これにより、パッチやサービス パックを一度に 1 ノードずつ適用していくことができます。パッチやサービス パックを適用した後にノードを再起動する必要がありますが、残りのノードがアクティブなので、エンドユーザーへのサービスは中断されることなく続行します。しかし、この機能は数年前からほとんどの主要なデータベース ベンダによって既に提供されており、現在では特に魅力的な機能ではありません。現在の水準から見て画期的と言える機能ではありません。

事実 :トランザクションのフェールオーバーは自動的には実行されない

RAC の可用性に関する Oracle 主張について注意すべきもう 1 つの点は、トランザクションやクエリのフェールオーバーが自動的には実行されないという点です。自動フェールオーバーを実装するには、Oracle の TAF (Transparent Application Failover) を利用するアプリケーションを構築する必要があります。その際には、アプリケーションが接続先のサーバーの状態を追跡できるようにするために、OCI (Oracle Call Interface) が必要となります。自動フェールオーバーを実装した環境では、現在接続されているサーバーに障害が起きると、アプリケーションによってクエリが再送信されるか、またはクエリが他のノードにリダイレクトされます。 しかし、この機能を使用すると、トランザクションの状態が常に格納され続けるため、アプリケーションのオーバーヘッドが非常に高くなります。

つまり、TAF を適切に使用してアプリケーションを開発しなければ、完全に自動化されたフェールオーバーを利用することはできません。また、ノードに障害が起きたときに実行中であったトランザクションやクエリを再送信する機能も利用できません。TAF やそれに類似する機能を利用したアプリケーションの開発は特に新しい手法ではなく、SQL Server を含むエンタープライズ データベースでも一般的に利用されています。したがって、これもやはり画期的な機能とは言えません。

まとめ

RAC で実現される可用性のレベルは、ここ数年の間一般に利用されている他の技術と同程度のものです。最近では他のベンダでも同様の手法が使用されており、RAC で独自に採用されている手法や高度なテクノロジはありません。確かに、場合によっては RAC でプロセスを簡素化できることもあります (たとえば、クラスタにノードを追加または削除するプロセスは OPS に比べてかなり簡素化されます)。しかし、RAC の導入にはかなりの追加コストが必要となるにもかかわらず、画期的と言えるような新しいテクノロジは一切導入されておらず、高可用性のレベルを高めるものにもなっていません。

また、他のサーバーにトランザクションを再送信することは可能ですので、トランザクションが完全に行えなくなることはありませんが、自動的な再接続は実行されず、現在出回っているほとんどのシステム (TAF 機能が採用されていないシステム) では、トランザクション状態データも失われてしまいます。

Oracle の主張 :"RAC によって総保有コスト (TCO) を削減できる"

RAC に関して Oracle が最も強く宣伝している利点の 1 つは、業界の他のデータベースに比べ、大幅に総保有コスト（TCO）を削減できるということです。大幅な節減を実現できるケースもいくらかありますが、ほんの一部の顧客にしかそれを実感できないと考えられます。ほとんどの場合、顧客は TCO の変化に気がつかないどころか、多くの場合、TCO が実際には増加します。

TCO についての Oracle の主張

Oracle 社は、RAC を実装する際に、スケーラビリティと高可用性の対策として Oracle9i データベースを導入することにより TCO を大幅に削減できると主張しています。同社によると、RAC は Linux を稼動している低コストの Intel™ サーバーに導入できるため、ハードウェアのコストやオペレーティング システムのコストを削減できます。

さらに、他のデータベースに関して、ダウンタイムについての所説とダウンタイムにかかるコストなど、データベースの管理コストがどのくらいかかるかについても主張しています。この節減論をそのまま受け取るには、Oracle データベースのダウンタイムがゼロで、他のすべてのデータベースのダウンタイムが長いことが前提となります。幸いなことにほとんどの顧客は、この節減論の欠陥を見通しており、設計、展開、および管理に優れているすべてのシステムが高可用性を提供できることも、Oracle データベースを含むすべてのデータベースでダウンタイムによる影響が発生することも理解しています。インターネット ベースの主要なオークション サイトや E コマース サイトなど、多数のデータベース管理者を擁する大規模な企業でさえ、データベースによる深刻なダウンタイムを経験しています。

事実 :RAC によるコスト増大の可能性

Oracle 社が主張する節減を実現できるのは、従来の UNIX や SUN、IBM などのメインフレーム プラットフォームを稼動している顧客だけです。この節減は、ハードウェア自体とそれに関連するサービスから得られるものに過ぎず、Oracle データベースのソフトウェアとサービスのコストは、減少しません。ユーザーは、関連する Oracle ライセンスを購入する必要があるうえ、メンテナンス、サポート、コンサルティング、およびトレーニングにかかる費用を払う必要があります。RAC オプションで使用する CPU やノードあたりにかかる追加料金、およびデータベース ライセンスを取得する必要があるノードの数の増加により、従来以上に Oracle に支払うことになる可能性もあります。RAC は、一般的な Oracle の契約やサイト ライセンスには含まれておらず、追加料金がかかるライセンスの特別オプションに含まれています。 また、クラスタ システムの配置に伴い、システム管理の複雑さが増すことになりますが、これについては、Oracle のメッセージで一切触れていません。

一部では、以下のような理由により、RAC が高可用性に対するコスト効率の優れたソリューションであると信じられています。

- RAC では、ノード障害が発生するまでサーバーを待機させるのではなく、常時ユーザーにサービスを提供する複数のアクティブなサーバーを使用することにより、コストを分散する。
- すべてのサーバーは同じデータベースのユーザーにサービスを提供するので、サーバー間で負荷のバランスがとれ、待機状態のリソースがなくなる。
- ユーザーによる負荷がサーバー間でほとんど平均的に分散されるので、各サーバーのリソースは半分しか必要ない。したがって、高い容量を持つサーバーに投資しないでも、フェールオーバーがサポートされる。

このような誤った認識は危険です。なぜなら、実際には十分なレベルの可用性とスケーラビリティを提供していないシステムを配置しているにもかかわらず、誤った安心感を与えてしまうことがあるからです。ここでユーザーが理解しなくてはならないのは、どのような高可用性のシステムにおいても、複数ノードで 1 つのクラスタを構成してフェールオーバー サポートを提供している場合、ノードが 1 つしか機能していないという障害発生時を想定したうえで、クラスタ内の各ノードがクラスタの全負荷をサポートできる必要があるということです。したがって、クラスタ全体の負荷すべてを処理するのに十分なリソースが各ノードに構成されている必要があります。ノードの総数が 4 つ以下のシステムでは、ベンダに関係なく、高可用性シナリオで同じことが言えます。ノードの総数が 4 つを超えると、複数ノード障害のリスクは統計的には減少します。各ノードでサポートされる冗長性レベルを削減できますが、完全に取り除くことはできません。

たとえば、8 CPU と 16GB RAM を搭載した単一サーバー環境でキャッシュ トランザクション システムをサポートする必要があり、このシステムに対して 2 つのノードで構成されるクラスタ システムを導入する場合、クラスタ内の各ノードが、8 CPU と 16GB RAM で構成される必要があります。この構成でない場合、ノード障害が発生したときに、存続するノードが増加する負荷に対応できず、その結果、トランザクションが拒否されてしまいます。さらに、システムでは対応できなくても、一部のリソースが追加のトランザクション要求の対応に使用されているため、既にサービスを受けている作業負荷にもパフォーマンスの影響がでることがあります。いずれにしても、一部の顧客がシステムを利用できない状態になります。単一ノードの SMP システムから RAC ベースのシステムに移行しても、顧客のハードウェア コストは削減されないうえ、ソフトウェアや管理のコストが増加する可能性が高いことは明白です。

これは、クラスタの使用を避ける必要があるという意味ではありません。高可用性を実現するにはクラスタが適しています。ここで理解すべきことは、RAC を使用したクラスタによって高可用性を実現するこのシナリオでは、ハードウェア コストを削減できないということと、ソフトウェアや管理にかかるコストが増加する可能性が高いということです。

以下の図と説明で、さらに詳細に示します。

単一サーバー、最小限の高可用性機能

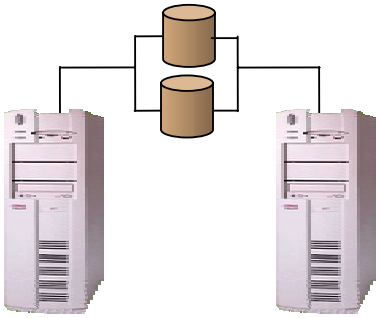
– ACME ERP システムを稼動

– 10,000 人のユーザーを同時にサポート

– システム要件: 8 CPU と 16GB RAM

図 5

高可用性フェールオーバー クラスタ (SQL サーバーで使用)



– 同一の ACME ERP システム

– 10,000 人のユーザーを同時にサポート

– システム要件: 8 CPU と 16GB RAM

– フェールオーバー クラスタリング機能と冗長性のあるストレージ機

– 各ノードは 8 CPU と 16 GB RAM で構成され、

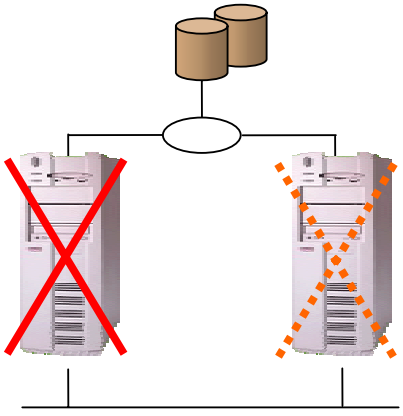
ノード障害発生時に、存続するノードが

全体の負荷に問題なく対応して実行

図 6

RAC 構成についての誤った想定

Oracle 社の説明によると、前例に挙げたシステムに類似するシステムでは、10,000 人のユーザーが 4 CPU で構成される 2 つのノード (ノードあたり 8GB RAM) に均等に分散されており、一方のノードに障害が発生した場合も、高可用性ソリュ-ションを提供できます。これは、両方のノードが稼動しているときは、作業負荷が両方のノードに分散していることを意味します。しかし、前述のように、これは、高可用性クラスタの誤った展開です。ノード障害が発生すると、障害が発生したノードを使用していたすべてのユーザーが存続するノードに移動し、(アプリケーションによって異なりますが) 手動または自動で処理を再開することを試みます。その場合、存続するノードは 5,000 人のユーザーだけに対応できるように設計されているため、10,000 人のユーザーからの要求に対応することができません。その結果、半数のユーザーはトランザクションを実行できません。さらに、一部のシステム リソースがサービスを受けることができない 5,000 人のユーザーの管理に使用されるため、トランザクションを実行できる残りのユーザーにも、パフォーマンスの影響が明らかにでます。



4-CPU
8GB RAM
-FAILURE-

4-CPU
8GB RAM
-OVERLOAD-

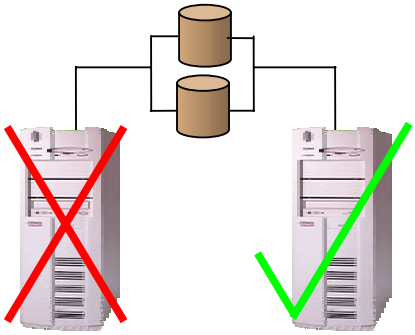
– 同一の ACME ERP システム
– 10,000 人のユーザーを同時にサポート
– システム要件: 8 CPU と 16GB RAM
– フェールオーバー クラスタリング機能と冗長性のあるストレージ機能
– 各ノードに 4 CPU と 8GB RAM

– 障害発生時には、存続するノードに
10,000 人のユーザーからの要求が殺到し、その負荷をサポート
できなくなります。
したがって、約 50% のユーザーはサービスを受けられません。

図 7. 1 つのノードに障害が発生し、システムは 4 CPU、8 GB の RAM だけで稼動しています。この状態では 10,000 人のユーザーによる負荷をサポートできません。

RAC を含む、高可用性クラスタの正しい展開

10,000 人のユーザーが、ノードあたり 8 CPU と 16GM RAM で構成される 2 つのノードに均等に分散される



8-CPU
16GB RAM

8-CPU
16GB

– 同一の ACME ERP システム
– 10,000 人のユーザーを同時にサポート
– システム要件: 8 CPU と 16GB RAM
– フェールオーバー クラスタリング機能と冗長性のあるストレージ機能
– 各ノードは 8 CPU と 16 GB RAM で構成され、
– ノード障害発生時には、存続するノードが
全体の負荷に問題なく対応して実行

図 8

1 つのノードに障害が発生しても、まだ 8 CPU と 16GM RAM を利用できるため、10,000 人のユーザーをサポートすることができます。この例では、負荷についての概念だけを説明していますが、推論はすべての環境について妥当であり、すべての環境に適用されます。

真の高可用性を必要とするシステムにとってこれがなぜ重要であるか (Oracle 社による説明では省略されていますが) をいったん理解すると、コストが削減しない仕組みだけでなく、ノード数 (Oracle 社の提案するように RAC を使用した場合) と作業負荷の増加によってコストが増える可能性さえあることがすぐにわかります。さらに、実際の展開において、4 CPU と 8GB RAM を搭載した単一のマシンが 5,000 人のユーザーをサポートできる場合、同一のハードウェア構成による 2 つのノードで構成される RAC の配置では 10,000 人のユーザーをサポートできません。2 つ目のノードを追加しても、処理能力は 85% 程度しか向上しません。また、クラスタ内のノード数が増えるにつれ、ノード毎の処理能力の向上率が著しく減少します。これらについては、Oracle 社によって明確に示されています。

したがって、RAC による経費削減についての Oracle 社の主張は、興味は引くものの根拠にかける主張か、マーケティングのための宣伝でしかありません。実際には、真の高可用性を備えたシステムを実現するにあたって、RAC を導入しても経費を削減することはできません。削減できると主張するような営業担当者には用心する必要があります。

結論

Oracle 社のマーケティング上の主張にもかかわらず、上記の一般的な展開シナリオでは、SQL Server と Oracle の両方によって実装できる マイクロソフト フェールオーバー クラスタリングと比較して、高可用性のために RAC を配置することによってコストを削減できないことが明確に示されています。さらに、RAC では、フェールオーバー クラスタと比較して、追加のコンポーネント (Interconnect など) が必要であるため、ハードウェア、ソフトウェア、および管理にかかるコストが増加する可能性があります。このソリューションは、自動化されたフェールオーバーを提供することによって動作しますが、主張されているようなコスト削減につながらないことは確かです。むしろ、総保有コスト (TCO) が増加する可能性があります。

事実 :RAC によるライセンス コストの増大

Oracle によるソリューションと マイクロソフトによるソリューションのコスト比較例

基本的な機能が同等である、様々な構成の推定小売価格を以下の表に示します。推定小売価格は、Oracle **社、**SUN **社、**Dell **社の各** Web **サイトに公開されている価格を基に作成しています。またマイクロソフト社については推定小売価格を基に作成しています。ソフトウェアのメンテナンス料金、税金、送料、その他の料金が適用される可能性があります。**

項目	**SUN™ 上で Oracle9i を稼動 8 CPU、1 つのノード**	**Dell Linux 又は Redhat Linux 上で Oracle9i RAC を稼動 (4 CPU、2 つのノード)**	**Windows 2000 Advanced Server 上で SQL Server 2000 を稼動 8 CPU、1 つのノード**
データベース ライセンス			
• Enterprise Edition	¥40,000,000	¥40,000,000	¥13,968,800
• OLAP **およびデータ マイニング**	¥20,000,000[1]	¥20,000,000[1]	- **含む** -
• RAC **オプション**	N/A	¥20,000,000	N/A
• **ネットワーク暗号化、**PKI**、およびシングル サインオン サポート**	¥10,000,000[2]	¥10,000,000[2]	- **含む** -
• **パフォーマンスの監視ツールおよびチューニング ツール**	¥3,000,000[3]	¥3,000,000[3]	- **含む** -
• **診断ツール**	¥3,000,000[4]	¥3,000,000[4]	- **含む** -
データベース コスト	**¥76,000,000**	**¥96,000,000**	**¥13,968,800**

表 1. **推定コストは、各社** Web **サイトに公開されている価格および推定小売価格に基づいており、通知なしに変更されることがあります。**

1. 8 CPU **に対して、**CPU **あたり** ¥2,500,000 **の** OLAP **オプションが必要**
2. 8 CPU **に対して、**CPU **あたり** ¥1,250,000 **の高度セキュリティ オプションが必要**
3. 8 CPU **に対して、**CPU **あたり** ¥375,000 **の診断パックが必要**
4. 8 CPU **に対して、**CPU **あたり** ¥375,000 **のチューニング パックが必要**

ハードウェアに対するコスト節減は、UNIX **システムから** Intel **ベースのシステムに移行することで実現できます。しかし、この節減分は、単にハードウェアのコストの削減によるものです。データベースは、単一サーバー上で稼動している場合はコストが一定に保たれますが、**RAC **オプションを使用するとコストが増加します。**

まとめ

システムの実際の展開状況を考えれば、システムに高いパフォーマンスと信頼性を要求するユーザーにとって最も効率的な総所有コストを実現するのは、依然として SQL Server です。Oracle9i RAC の導入によって期待できる省コスト効果は、ユーザーが高価な UNIX システムから Intel ベースの安価なシステムに移行する場合に限ってのことであり、ハードウェア上の理由のみによるものです。既に Intel ベースのサーバーを使用しているユーザーが RAC に移行しても、コストの低減は期待できないでしょう。実際には、RAC の導入によって、総所有コストが増加する可能性も十分にあります。これは、スケーラビリティや高可用性に対する大きな利点は提供されないうえに、次のような追加の購入が必要になるためです。

- サーバーやネットワーキング ハードウェアの追加
- ストレージ ソリューションの購入 (付属しているストレージは、2 つ以上のノードの RAC に対応していません)
- ファイバ接続ソリューション (2 つ以上のノードがある場合に必要です)
- Oracle データベース ライセンスの追加 (ノードごとにライセンスが必要です)
- RAC ライセンス
- オペレータと管理者のトレーニング
- 移行や証明に必要なサービスの導入

上記には、インフラストラクチャの要件や、データベース管理者、システム管理者、およびネットワーク管理者に対する追加の管理タスクは考慮されていません。

これに対して SQL Server 2000 は、総所有コストが最も低いだけでなく、パフォーマンスにおけるコスト効率においても優れています。インテリジェントで動的な自動リソース管理機能により、安心して使用でき、管理コストも低減されます。

結論

RAC によって、パフォーマンスに関する OPS 特有のさまざまな問題を回避し、スケーラビリティをわずかに向上させることが実現されましたが、いまだ Oracle 社の主張するような企業向けのスケーラビリティと可用性は提供されていません。さらに、RAC の導入と管理にかかる実際のコストは、Oracle 社が主張するような低コストからはかけ離れています。RAC によって OPS の有効性が実現されたことは、Oracle 社による大きな前進ですが、設計上スケーラビリティに制限があるアーキテクチャでは、これ以上はあまり期待できません。

結論 RAC は Oracle 社の重要なリリースですが、Oracle 社が主張する利点を実際にすべて実現するものではありません。RAC では、同等またはそれ以上のパフォーマンスを提供する SMP システムに比べ、実装や管理により多くのコストがかかる可能性があります。SQL サーバーは、業界標準のほとんどのベンチマークにおいて依然として主導的な立場にあり、総所有コストも最も低くなります。さらに、管理のし易さや、動的で自動化されたリソース管理においても、Oracle は SQL サーバーに何年も遅れています。

付録 I:リソース リンク

トランザクション処理性能評議会 (TPC)

www.tpc.org

SQL Server ベンチマーク

http://www.microsoft.com/japan/sql/evaluation/compare/benchmarks.asp

圧倒的な「使いやすさ」により、主要な競合製品に比べ、47% の大幅な TCO 削減を実現!!

http://www.microsoft.com/japan/sql/choice/tco/

SQL Server Business Intelligence

http://www.microsoft.com/japan/solutions/BI/