Oracle GRID Center

富士通 SPARC Enterprise
Oracle9*i* Database から Oracle Database 11g への移行

～ 旧システムとのパフォーマンス比較と Oracle Real Application Testing の活用 ～

Creation Date: December 1, 2008
Last Update: July 1, 2009
Version: 1.1

ORACLE

FUJITSU
THE POSSIBILITIES ARE INFINITE

目次

1. はじめに

日本オラクル株式会社（以降、日本オラクル）と富士通株式会社（以降、富士通）は1989 年に OEM 契約を締結して以来､お客様に安心してご利用いただけるソリューションを提供するため、システム構築、共同検証、導入後のサポートなど、様々なアライアンス活動を行ってまいりました。

日本オラクルは 2006 年 11 月、企業の IT システム基盤の最適化を実現する次世代のビジネス・ソリューションを構築するため、先鋭の技術を集結した「Oracle GRID Center（オラクル・グリッド・センター)」（http://www.oracle.co.jp/solutions/grid_center/index.html）を開設しました。富士通は Oracle GRID Center 開設に賛同し、富士通のもつサーバ、ストレージ製品を使用して Oracle GRID Center にて共同で技術検証を行っております。

このたび、両社は富士通の最新の UNIX サーバ SPARC Enterprise と最新バージョンの Oracle Database へ移行することのメリット、移行時に発生する課題への対策として Oracle Real Application Testing の有効性を SPARC Enterprise M3000 と Oracle Database 11g を用いて Oracle GRID Center にて検証しました。その成果をここに報告致します。

2. 製品紹介／機能概要

本検証にて使用した製品および機能について説明します。

2.1. 富士通 SPARC Enterprise M3000

今日の企業ビジネスにおいて、IT システムの果たす役割はますます重要となっています。企業が扱うデータ量は増加し、Web サーバ、アプリケーションサーバ、データベースサーバなど、企業内のサーバ用途は大きく広がっています。一方で、サーバの処理性能の不足やサーバ台数増による運用コストの増大などが IT システムの課題となっており、コスト削減や効率化の面から、サーバ統合や最適化の必要性も高まっています。

「SPARC Enterprise」では、メインフレーム並みの高信頼性を持ったミッションクリティカル業務に最適な SPARC Enterprise M9000, M8000, M5000, M4000,と、Web フロント業務やアプリケーションサーバなどに最適な高いスループット性能を持った SPARC Enterprise T5440, T5240, T5220, T5140, T5120, T2000, T1000 という 2 種類のラインナップを提供してきました。

しかしながら、中堅規模のお客様においては、基幹サーバとしてエントリークラスの UNIX サーバを導入するケースも多く、データベースやバッチ業務など様々な基幹業務に適用できるマシンへの要望が多く寄せられました。

- **ミッドレンジ、ハイエンドサーバの高いミッションクリティカル機能を継承**

UNIX サーバ「SPARC Enterprise M3000」は、このようなお客様の声を反映し、基幹システムに求められる高性能と高信頼性を兼ね備え、エントリーモデルでありながらフロント業務だけでなく総合的に使うことができます。

ミッションクリティカルのラインナップである「SPARC Enterprise M3000」は、「SPARC Enterprise M4000」以上のモデルに搭載されてきた「SPARC64 Ⅶ」を搭載し、最大で 4 コア／8 スレッドのマルチスレッド環境を実現。メモリは最大 64GB 搭載可能で、SAS は 1 ポート、PCI Express は 4 スロット標準装備するなど、わずか 2U のスペースに業務に必要な機能をすべて実装。エントリーモデルにおける最高クラスの性能を発揮し、データベースサーバやアプリケーションサーバなどの幅広い業務に適用可能となっています。

- **基幹業務にも適用できる高い信頼性**

「SPARC Enterprise M3000」は、「M4000」から「M9000」の高信頼性を踏襲。LSI レベル、ユニットレベル、システムレベルとシステム全体で信頼性を積み上げ、確保しています。

たとえば、LSI レベルでは、1CPU につき約 3400 個の故障抽出用チェッカーを備え、ハードウェアによるメモリパトロールを強化するなど、データインテグリティの徹底追求をしています。

特に、キャッシュメモリに関してはプロセッサを構成する回路のなかで最も故障が起きやすい部分のため、重点的にデータ保護機能を装備。2 次キャッシュメモリはすべて ECC で保護され、1bit エラーが多発すると、キャッシュ way 単位、CPU コア単位で動的に段階的に縮退します。そのため、1CPU であっても、万一の際でもシステムの継続運用が可能であり、さらに性能劣化を最小限に抑えることができます。

また、ユニットレベルでは、ディスク・FAN・電源ユニットなどの冗長化や活性交換、動的縮退を備え、状態監視、故障通知機能を持たせるなど、耐故障性を追求。システムレベルでは、クラスタシステムサポートや、ストレージ／ネットワークの冗長化などにより、業務無停止の実現を図っています。

- **低消費電力、静音、省スペース化を目指したエコロジーサーバ**

「SPARC Enterprise M3000」は、富士通が独自に定める環境配慮型製品である、スーパーグリーン製品です。コンパクトな 2U(ユニット)サイズで、PRIMEPOWER450(4U)と比較し、50％の省スペースと軽量化を実現。消費電力は最大 470W (100V) と 57％も削減。性能の向上と合わせて、CO2 排出量を年間約 65％削減できます。しかも、一般的なサーバ設置場所の環境温度 25℃の音圧レベルで 47dB という静音設計で、他社 4 コアサーバと比較しても、消費電力値や騒音値において、最も優れたエコロジーサーバとなっています。

「SPARC Enterprise M3000」では、省電力化を進めるため、冷却の取り組みにおいて、エアダクトを採用し、CPU、メモリなど、発熱の大きい部分を集中的に冷却。筐体内を、本体のファンと電源ファンでの冷却という２つの冷却グループに分割して冷却することで、冷却効率を向上させています。冷却ファンの回転数は 9 段階で制御することができ、PRIMEPOWER250/450 と比較すると、音圧レベルを 3dB 低減するなど、数々の工夫により、省電力化や静音性を実現しています。

さらに「SPARC Enterprise M3000」は、Solaris コンテナと呼ばれる仮想化技術を標準で装備。エントリーモデルでありながらサーバ統合により資源を集約することが可能となっており、システムの効率化を実現できます。

このように、「SPARC Enterprise M3000」の高性能、高信頼性、エコロジーという優れた特長を活かすことで、お客様のシステムの最適化に貢献し、ビジネス機会の拡大を企業にもたらします。

【SPARC Enterprise M3000 の特長】

- SPARC/Solaris エントリークラス最高のプロセッサコア性能を実現
- ミッドレンジクラスの高信頼技術をエントリークラスに継承
- 省電力/省スペースなど「Green Policy Innovation」の具現化

2.2. ストレージシステム ETERNUS（エターナス）

富士通は、SAN 対応ディスクアレイとして、2.72PB（ペタバイト）の世界最大の容量を実現したエンタープライズディスクアレイ「ETERNUS8000」と、コストパフォーマンスに優れたミッドレンジディスアレイ「ETERNUS4000」、省スペース・省電力・静音設計なエントリーディスクアレイ「ETERNUS2000」を提供、加えて ETERNUS2000 の後継として拡張性やグリーン IT への取り組みをより強化した新エントリーディスクアレイ「ETERNUS DX60/DX80」を提供するなど、幅広い製品ラインナップであらゆるニーズに応えます。

部品レベルからシステムレベルまで徹底した信頼性と優れた拡張性を備え、SAN を利用したストレージ統合を実現します。また、高速コピー機能やデータ暗号化機能をサポートし、セキュリティ強化やデータ保護が可能です。さらに、グローバルな環境対策に則した製品で、システム要件に応じた省電力運用にも対応します。

ETERNUS ディスクアレイは、企業の重要なデータを確実かつ効率的に保管し、投資効率の大幅な向上を可能とするストレージソリューションを提供します。

- **ETERNUS2000 モデル 200**

エントリーディスクアレイ「ETERNUS2000 モデル 200」は、最薄約 9cm(2U)より導入可能なラックマウント型のディスクアレイです。部品点数の削減などで従来の ETERNUS4000 モデル 80 より消費電力を約 40%削減(当社比)すると共に、42dB(デシ

ベル、29℃以下の場合)という優れた静音性を実現しています。

ミッドレンジクラス同等の高信頼設計を採用し、主要コンポーネントの二重化に加え、ディスク故障の予防交換機能によりデータの冗長性を常に確保します。また、「アドバンスト・コピー機能※1」をサポートし、オンライン中の高速バックアップが可能です。

エントリークラスながら最大 72TB(テラバイト)の記憶容量を備え、コストパフォーマンスに優れたデータのバックアップや日々増加するデータの格納を実現します。

【ETERNUS2000 モデル 200 の特長】

- 省スペース・省電力・静音設計なエントリーディスクアレイ
- ミッドレンジクラス同等の高信頼設計を採用
- 高速バックアップを実現する「アドバンスト・コピー機能」をサポート
- 最大 72TB の優れた拡張性

なお、新エントリーディスクアレイ「ETERNUS DX80」は、記憶容量を最大 120TB まで拡張すると共に、ファンの回転制御を最適化し、ETERNUS2000 より消費電力を 8%(当社比)、騒音レベルを最大 6dB (25～30℃の場合)削減しています。

2.3. Oracle Database 11g

Oracle Database は、商用リレーショナル・データベース管理システムのリーダーとして常に最先端の技術を搭載し、世界中の企業や政府機関などにおける情報管理の課題を解決してきました。多種プラットフォームへの移植性、データの信頼性を保証する読取り一貫性、制限のない完全な行レベルロック、拡張性と高可用性を両立するデータベース・クラスタである Oracle Real Application Clusters、多様なデータ型／XML サポートなどは、Oracle Database の基本的なアーキテクチャであると同時に圧倒的な技術的優位性のベースとなっています。

Oracle Database 11g は、最先端技術の実装による進化に加え、お客様のご要望を積極的に取り入れ、様々な IT 課題を解決するための機能をご提供する「Real Customer Release」、すなわちお客様のバリューを第一に考えた Oracle Database の最新リリースです。以降、特長について具体的に説明します。

1 オプション、ソフトウェア「ETERNUS SF AdvancedCopy Manager」が必要

- **運用管理コストを削減しながらシステム障害を極小化**

企業の IT コストの 8 割を占めるといわれる「運用管理・維持コストの削減」という課題に対して、運用管理コストを削減しつつシステム障害を極小化するためのソリューションをご提供することが可能です。

Oracle Real Application Testing はより少ない工数・期間で、より精度の高い事前検証の実施を可能にするため、データベースへの変更を最小のリスクで迅速に実装することが可能です。

GUI ベースのデータベース管理ツールである Oracle Enterprise Manager は、システム全体の稼動状況やパフォーマンス情報を一目で確認できる統合的なビューを提供します。また、基本的なデータベース管理作業を行うことができるだけでなく、障害が発生した場合の迅速な診断／復旧を支援する障害診断インフラストラクチャ、パフォーマンス情報を自動収集し、データベースの自動診断／チューニングを可能にする自己管理データベース機能なども Oracle Enterprise Manager を使用して操作を行うことができるため、運用管理コストを大幅に削減しながらシステム障害を極小化し、コスト効率の高い迅速なシステム管理を実現することができます。

- **IT インフラストラクチャの共有により低コストでより高い可用性を実現**

多くの企業内の IT インフラはアプリケーションごとに分散され、管理されています。これによりインフラの管理が複雑になり、かつ IT リソースの効率的な活用が難しくなるため、多くの企業において課題となっている「高い IT の運用管理・維持コスト」の根本原因となっています。また、IT インフラが分散されているとシステム全体を見てサービスレベルを維持するための設計を行うことも困難であるため、十分な可用性を担保するための構成や運用管理を実現することも難しくなります。

Oracle Real Application Clusters により、高可用性とハイ・パフォーマンス、スケーラビリティを全て実現したデータベース・クラスタが実現すると共に、ワークロード管理機能によってグリッド・コンピューティングによる効率的なリソース活用が実現します。

Automatic Storage Management は、ストレージ層における仮想化／リソースの有効活用を実現し、ストレージ管理の負荷を軽減します。

Oracle Data Guard は、災害対策用のスタンバイ・データベースを提供する機能ですが、従来、平常時は待機させておくしかなかったスタンバイ・データベースを有効活用するための機能 (Oracle Active Data Guard)が新たに実装されており、これまでと比べ、より低コストで高い可用性を実現することが可能となっています。

- **大規模データを低コストかつ高パフォーマンスで管理／情報活用の促進**

多くの企業では、システムで取扱うデータの多様化や法的規制による長期保存の必要性、幅広い観点での分析ニーズの増加に伴い、データ量の肥大化とそれに伴うストレージ容量の増大、ひいてはストレージ・コストの増加が課題となっています。

Oracle Advanced Compression はデータベースに格納されたあらゆるデータを圧縮し、パフォーマンスを向上させる画期的な機能です。

Oracle Partitioning は、大規模データの管理／検索におけるパフォーマンスを向上させるのに不可欠な機能です。

データの多様化という観点では、ドキュメントや画像、地図情報など様々な種類の非構造化データをパフォーマンス良く取扱うことのできる Oracle SecureFiles という全く新しいアーキテクチャで設計された機能を搭載しています。これらの機能によって、データウェアハウス、オンライン・トランザクション・システム問わず、大容量データを低コストで管理することができます。また、あらゆる企業データの情報活用を促進します。

- **企業コンプライアンスとリスク管理を強化するデータベース・セキュリティ**

今日の IT システムにおいては、コンプライアンスやリスク管理の観点から、これまで以上に堅牢なセキュリティが求められています。特に情報資産を格納するデータベースのセキュリティ対策は必要不可欠です。

Oracle Database Vault により、データベース管理者に管理権限が一極集中するリスクを回避することができ、また、ビジネスニーズに応じた柔軟なアクセス制御が可能になります。

機密性の高いデータに関しては、データを自動的に暗号化して格納する Transparent Data Encryption が提供されています。データベース監査についても、柔軟かつ効率的に監査ログが取得できるよう実装されています。

Oracle Database は、ISO／IEC 15408 (Common Criteria) 、FIPS 140-2 などの各種国際基準の認証を取得しているだけでなく、RDBMS 製品において ISO／IEC 15408 (EAL4) をはじめて取得しました。

様々な課題を解決する Oracle Database 11g のメリットは、あらゆる規模のシステムで享受することができます。中小規模のシステムでは、高いパフォーマンス、障害対応、豊富な開発環境、そして自動管理データベースなどが IT システム基盤強化に大きく寄与します。また、大規模システムにおいては、運用管理、可用性、情報管理、データウェアハウス向けの機能などが大幅に強化されており、グリッド・コンピューティングによる IT システム基盤を提供することで顧客の競争力強化に大きく貢献します。

2.4. Real Application Testing Option

今日の企業は、IT システム基盤の変更を実施するために、ハードウェアとソフトウェアに対して大きな投資を行う必要があります。たとえば、Oracle Enterprise Linux などの、低コストのコンピューティング・プラットフォームにデータベースを移行する構想を持ったデータ・センターがあるとします。従来の環境だと、こうした移行には、本番アプリケーションをテストするために、Web サーバー、アプリケーション・サーバ、データベースなどのアプリケーション・スタック全体を構成する現行の各ハードウェアと同じハードウェアに投資する必要があります。このため、データ・センターのインフラ変更を評価および実装するには、かなりのコストがかかります。しかも、こうした広範なテストを実施したとしても、最終的に本番システムで変更を行うと、予期せぬ問題が発生する可能性があります。こうした問題が起こるのは、テスト用にシミュレートされたワークロードが、本番環境で発生するワークロードを、正確にまたは完全には反映していないためです。このため、データ・センター管理者は、新しいテクノロジーを採用して、変化の激しい競争圧力にビジネスを適合させることに積極的ではありません。

Oracle Database 11g では、Enterprise Edition のオプション製品「Oracle Real Application Testing」として 2 つの新しい機能、Database Replay と SQL Performance Analyzer を実装することで、こうした問題に対する最適なソリューションをご提供することが可能になりました。

Oracle Real Application Testing に関する詳細な情報は、Oracle Technology Network の「Oracle Database の管理性および Oracle Real Application Testing」ページをご覧下さい。

http://www.oracle.com/technology/global/jp/products/manageability/database/index.html

2.4.1. Database Replay

Database Replay を使用すると、データベース管理者とシステム管理者は、テスト環境において、オンライン・ユーザーやバッチ処理によるワークロードなど、実際の本番環境で発生するワークロードを、忠実で正確かつ現実的に再現することができます。

Database Replay では、同時実行性、依存性、タイミングなどを含んだ本番システムからのデータベース全体の負荷を取得して、本番システムのワークロードを

テスト・システム上で再現することで、システムの変更を現実的にテストできます。これは、スクリプトのセットなどでは決して実現できません。Database Replay を使用すると、データベース管理者とシステム管理者は次のテストを実行できます。

- データベース・アップグレード、パッチ、パラメータ、スキーマの変更など
- シングル・インスタンスから RAC や ASM への変換といった構成の変更
- ストレージ、ネットワーク、インターコネクトの変更
- オペレーティング・システム、ハードウェアの変更、パッチ、アップグレード、パラメータの変更

以降、メリットについて具体的に説明します。

テスト環境構築コストの低減

Database Replay を導入することで、データベース管理者は、システムの変更をテストするためのテスト環境を自由に構築できるようになります。これまでのように、アプリケーション・インフラ全体を重複して持つ必要はありません。Database Replay では、中間層、すなわち Web サーバー層の重複に伴う設定上のオーバーヘッドは発生しません。データベース管理者およびシステム管理者は、変更が本番環境のシナリオを使用して真にテストおよび評価されていることを認識しているため、データ・センターのインフラ・コンポ―ネントを迅速にテストおよびアップグレードできます。

迅速な準備

Database Replay のもう 1 つの利点は、データベース管理者が何ヶ月もかけて、アプリケーションの機能に関する知識を取得して開発用テスト・スクリプトを用意するという必要がない点です。数回マウスをクリックするだけで、本番環境全体のワークロードが容易に再現され、システム変更のテストおよび導入が可能になります。これにより、従来何ヶ月もかかっていたテスト・サイクルが数日または数週間に短縮され、結果として大幅なコスト削減が実現します。

2.4.2. SQL Performance Analyzer

SQL 実行計画は、SQL 文を物理的に実行するためのアクセス・パスを意味しますが、これを変更すると、アプリケーションのパフォーマンスと可用性に重大な影響を与える可能性があります。結果として、データベース管理者は、システムの変更によってパフォーマンスの低下した SQL の特定と修正に多大な時間を費やすことになるのです。SQL Performance Analyzer (SPA) を使用すると、システム環

境の変更によって引き起こされる SQL の実行パフォーマンスの低下を予測し、未然に防ぐことができます。

SPA では、環境の変更前と変更後で SQL を一つずつ実行することで、環境の変更が SQL の実行計画と統計情報に及ぼす影響を、SQL 単位で詳細に提供します。また、効率の低下した SQL だけでなく、システムの変更によってもたらされた、ワークロードに対するパフォーマンスの改善を概説するレポートも生成します。効率の低下した SQL については、適切な実行計画の詳細とそれらを調整するときの推奨事項が提示されます。

SPA は、既存の SQL Tuning Set (STS) と SQL Tuning Advisor(STA)、もしくは SQL 計画の管理(SQL Plan Management [SPM])の各機能と、密接に連携して動作します。システム変更が（数十万の SQL のような）非常に大きな SQL ワークロードに及ぼす影響を査定する作業は、これまで手動で行っていたため、大変時間のかかる作業でした。SPA を使用すれば、こうした作業を完全に自動化し簡素化できます。

データベース管理者は、STA を使用して、効率の低下した SQL をテスト環境で修正し、新しいプランを生成できます。こうして生成した計画を SQL 実行計画管理にベースラインとして供給し、本番システムにエクスポートして戻します。このように、SPA を使用することで、企業は、本番環境に対するシステム変更が実際にプラスの効果をもたらすことを、これまでより低いコストで極めて高い信頼度で確認できます。

SPA を使用して分析できる、共通のシステム変更の例を以下に示します。

- データベース・アップグレード、パッチ、初期化パラメータの変更など
- オペレーティング・システム、ハードウェア、またはデータベースに対する構成変更
- 新規索引の追加、パーティション化、マテリアライズド・ビューといったスキーマの変更
- オプティマイザ統計情報の収集
- SQL チューニング　（SQL プロファイルの作成など）

正しいソリューション、Database Replay と SPA を選択することで、データベース管理者は、システム変更を効率的に吸収し、管理することができます。Database Replay は、同時並行処理を考慮したスループット・テストを目的に設計されています。一方 SPA は、SQL 単体テストの実施と、そこで発見されたパフォーマンス劣化を SPM や SQL プロファイル、アウトラインを使って修正することを目的としています。これら 2 つの機能は相互補完するソリューションであり、両ソリューションを使用して変更のインパクトを評価することを強く推奨しています。Oracle

Database 11g Real Application Testing は、データベース管理者が、ビジネスに必要不可欠なあらゆる変更を、より少ないリスクで容易に管理および実行できるようにします。

3. 移行の必要性

古いバージョンのデータベースを古いハードウェア上の OS で利用し続ける場合には、次のようなリスクがあります。

- 稼働しているハードウェア能力の限界や業務の拡大による処理性能不足
- 度重なる機器追加などによる運用コストの高騰や、サーバルームの消費電力増大
- 十分なサポートが受けにくくなる、保守リスクの増大
- 必要な技術情報の入手が困難

現在安定稼動しているシステムをアップグレードすることに対するリスクのみを考えがちですが、後々直面するであろうリスクも十分に認識した上で、アップグレードを検討することが必要です。

さらに、システムのライフサイクルにおいて、ハードウェアの保守期限および OS のサポート期限は定期的にやってきます。OS をバージョンアップした際に今まで使用していた Oracle が対応していないなど、システム構築時から期間が開いてしまうほど、システムを移行するための手順が複雑化したり、互換がとれなくなってくる可能性があるため、システムの移行がより困難になる恐れがあります。

このような課題解決のためには、最新の SPARC/Solaris サーバと最新の Solaris OS、そして最新バージョンの Oracle で構築したシステムへの移行が最適です。

3.1. 移行のメリット

現在ご利用されている Solaris サーバを最新の SPARC/Solaris サーバに移行することにより、ハードウェアおよび OS の保守期限切れの解消、マルチコア/マルチスレッド対応 CPU の採用や PCI Express/SAS など最新 I/O インターフェースの採用によるシステム性能向上による安定性を確保することができます。

また、SPARC Enterprise M3000 は、従来機種に比べ、システム性能だけでなく環境性能も大きく向上しており、SPARC Enterprise M3000 に移行していただくことで、大幅な性能向上と電力コストの削減が可能です。

本検証では、PRIMEPOWER 250 から SPARC Enterprise M3000 へのリプレースに併せて Oracle9i Database を Oracle Database 11g へアップグレードすることにより、OLTP アプリケーションにおいてスループットは約 3.5 倍に向上、レスポンスは約 1/6 に改善しています。

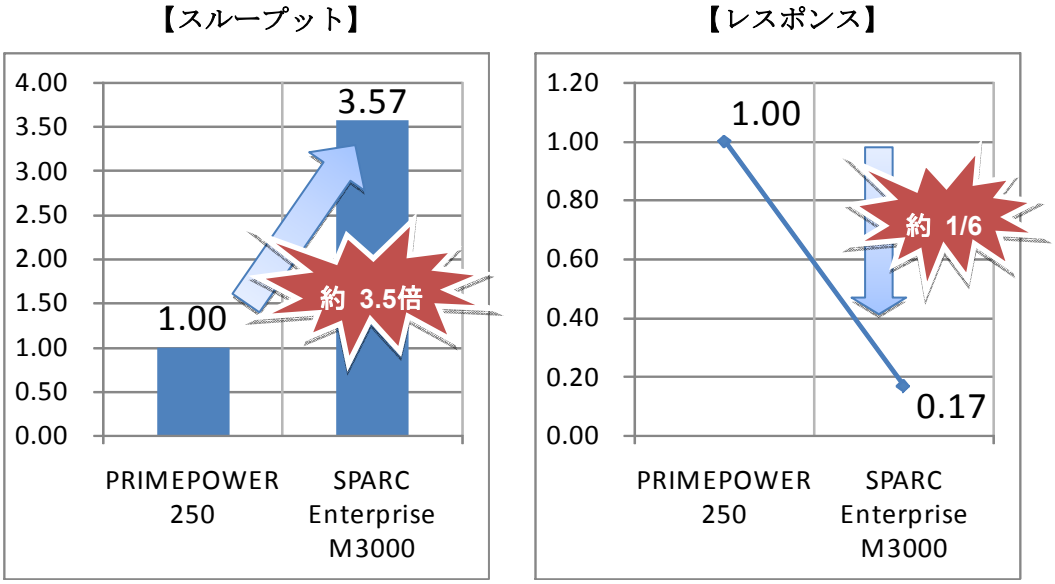


図 3-1 PRIMEPOWER250 と SPARC Enterprise M3000 の性能比較

次に、現在ご利用されている Oracle9i Database を Oracle Database 11g へ移行することによって、企業の IT システムにとってどのようなメリットがあるのかを考えてみます。まず、データベースを最新にアップグレードすることによって、様々な最新技術を活用することが可能です。例えば、データベースのパフォーマンスが悪く、その原因となっているボトルネックを診断する必要がある場合、Oracle9i Database では、ある程度高いスキルと経験を持ったデータベース技術者が自身で様々な、パフォーマンスを診断するために必要な情報を収集し、それらの情報を基に自身で問題点と改善策を導き出す必要がありました。Oracle Database 11g であれば自己診断や強化されたチューニング・アドバイスの機能を活用することができます。セキュリティに関しても、Oracle9i Database では、データの暗号化を実装するためにアプリケーション・コードを書換える必要がありました。Oracle Database 11g では、強化されたデータ暗号化の機能を活用して、既存のアプリケーションに手を入れずにセキュアで堅牢なデータベース環境を構築することが可能です。その他にも「2. 製品紹介／機能概要」にて述べた様々な最新テクノロジーによるメリットを享受することが可能です。

また、古いバージョンのデータベースで運用を続けていくことにもリスクが伴います。古いバージョンを利用し続けることによって、十分なサポートが受けにくくなる、メンテナンス・コストが高騰する、必要な技術情報の入手が困難になってくる、など、後々直面するであろうリスクも十分にご認識頂いた上で、アップグレードをご検討して頂くことを強く推奨します。

最後に、システム構築時から期間が開いてしまえばしまうほど、データベース移

行のための手順が複雑化したり、より互換性がなくなってくる可能性があるため、アップグレードがより困難になる恐れがあります。

3.2. 移行に際しての課題

移行対象とする旧システムのハードウェア、OS、ミドルウェア、ユーザデータおよびユーザアプリケーションそれぞれについて移行する際に考慮しなければならない課題があります。

既存資産（周辺機器やドライバ）を移行先のシステムで利用する場合には、移行先のシステムで新しく使用する予定のハードウェア、OS、ミドルウェアとの組み合わせを確認する必要があります。

また、Solaris 10 は旧バージョンとのバイナリ互換を保証していますが、Solaris OS のエンハンスに伴うバイナリ以外の変更項目（廃止されたコマンドや運用方法）、ミドルウェアのバージョンアップに伴うバージョン間の非互換を確認する必要があります。

Oracle9i Database から Oracle Database 11g への移行に際しては、考慮しなければならない課題もあります。Oracle Database の異なるバージョン間における互換性が、現在ご利用のアプリケーションに影響する場合があります。これは、新しい予約語が追加される、初期化パラメータやデータ・ディクショナリが変更される、などが主な要因ですが、特に最も大きな問題となり得る変更は、問合せオプティマイザの変更です。Oracle9i Database までは、Rule Base Optimizer (RBO) がサポートされていましたが、Oracle Database 10g R1 以降でサポートされるのは Cost Base Optimizer (CBO) のみであり、RBO はサポートされません。RBO は、SQL が実行されるための最適な実行計画を、使用可能なアクセスパスを順序づけるランキングに基づいて作成する問合せオプティマイザで、Oracle6 以前に生まれたレガシーのオプティマイザですが、開発者にとってその考え方が理解しやすい反面、データの変動に追随できない、データ量や偏りが考慮されないといった大きなデメリットがあります。RBO の存在は、お客様にとって Oracle7.3 以降に登場した全ての問い合わせ処理テクノロジーを活用する上で、また、Oracle Database の問合せ処理エンジンにおける重要な機能強化を行う上での妨げとなります。RBO を排除することによりパフォーマンスやデータベース・エンジンにおける問合せ処理コンポーネントの信頼性の改善が可能になります。

すなわち前述のとおり、Oracle9i Database から Oracle Database 11g への移行に際

しては、Cost Base Optimizer (CBO) への切替えが不可欠となります。CBO では、事前に収集されるべき正確な統計情報に基づいてデータ量の変動や偏りを考慮したコストを見積もり、ほとんどの SQL が最適なアクセスパスを選択し実行されます。ただし、特定のケースにおいては元の RBO のアクセスパスを維持することが望ましい場合があり、事前に十分なテストを行わないまま CBO への切替えを実施してしまうことよって、サービス開始後に、重要なパフォーマンス問題を引き起こす可能性があります。

こういった問題を防ぐためには、事前に最適な方法を用いて十分なテストを行い、RBO から CBO への切替えにおいて、データベース・パフォーマンスの劣化がないことを確認することが非常に重要です。

表 3-1 RBO と CBO

	ルールベース・オプティマイザ (RBO)	コストベース・オプティマイザ (CBO)
概要	使用可能なアクセスパスを順序付けるランキングに基づいて実行計画を作成（OLTP 向き）	統計情報に基づきコストを見積り、最もコストの低い実行計画を作成（OLTP、DSS 共に有効）
メリット	• 開発者にとって RBO の考え方は理解しやすい • SQL 実行計画の変動がほとんど起こらない	• データの変動に追随できる • 機能強化の恩恵を受けられる • データの偏りや量に基づいて実行計画を作成できる
デメリット	• データの変動に追随できない • データの偏りや量は考慮されない • 機能強化の恩恵を全く受けられない • Oracle Database 10g 以降のバージョンではサポートされない	• 統計情報の取得が必要 • 統計情報の再収集によって性能が変化する可能性がある

3.3. 移行に際してのアプリケーションテストとパフォーマンステスト

新しいシステムで互換性やパフォーマンスに問題がないことを事前に確認するためにアプリケーションテスト、パフォーマンステストを計画、実施して移行自体の失敗や移行後のリスクを低減します。

互換性については、次のようなアプリケーションテストを計画、実施します。

- **最小テスト**

現行のシステムからアプリケーションの全部または一部を新しいシステム（あるいはテスト環境）に移動し、新機能を使用可能にしないでテストを実行します。最小テストでは、潜在的問題は検出されませんが、少なくともアプリケーションの起動または呼出しに関する問題を確認します。

- **機能テスト**

  すべてのデータベース、ネットワークおよびアプリケーション・コンポーネントのテストが含まれ、システムの新機能と既存機能をアップグレードした後にテストを実行します。機能テストでは、システムの個々のコンポーネントがアップグレード前と同様に機能し、新機能が正常に動作することを確認します。

- **統合テスト**

  システムのコンポーネントの相互作用をテストします。機能テストでは、次のような点を確認します。

  - Pro*C/C++、JDBC、ODBC アプリケーション等が、新しいソフトウェアで問題が無いかの確認
  - データ型やデータ・ディクショナリのデータの変更などによってフロントエンド・アプリケーションに対してどの様な影響があるのかの確認
  - アプリケーションが SQL*Net、Net8 または Oracle Net Services を使用して接続されている場合の接続の確認

- **パフォーマンス・テスト**

  パフォーマンスについては、次のようなパフォーマンステストを計画、実施します。また、テストの際には、新しいシステムでの性能要件に従い現行のシステムのパフォーマンス情報※2を事前に取得します。この作業により、新しいシステムの性能要件が妥当かどうか、パフォーマンスが劣化した場合には相違点を比較しすることで問題点を発見することができます。

- **単体性能の確認**

  単体性能の確認では、業務の最小単位（たとえば SQL 単体）で実行し、主にレスポンスに着目します。単体性能が新しいシステムでの性能要件を達成した後に多重性能の確認を行います。もし、性能要件を達成しなかった場合や逆にパフォーマンスが劣化した場合には、ボトルネックの分析やチューニング作業が必要となります。

- **多重性能の確認**

---

2 業務（SQL）のレスポンスやスループット、OS 統計、STATSPACK、SQL トレースなどがあります。

多重性能の確認では、業務を構成する複数の要素から、実際のシステムと同様の負荷を発生させてシステム全体の性能に着目します。このとき単体性能が出ていないシステムに対して多重性能確認を行っても性能は良くなりません。また、単体性能確認では発生しない次のような問題が発生します。

- 高い CPU 使用率、CPU 待ち
- ディスク使用率、ディスク待ち
- ファイルやデータベースの排他制御（ロック）
- プロセス間通信によるプロセスの待ち

このように、新システムへの移行に際しては移行対象となる現行のシステムに対する互換性の確認（該当の有無、非互換の影響度や対策)、アプリケーションテストやパフォーマンステストを計画・実施する必要があり、必要となる工数やコストの増大、各種作業にともなうシステムトラブルの発生などのリスクが心配されました。

本検証では､アプリケーションテスト､多重性能の確認方法として Database Replay、単体性能の確認として SQL Performance Analyzer を利用しすることでリスクを低減し、より容易に移行することができるか検証を行いました。

4. 検証環境

本検証はデータベースクライアント 1 台、データベースサーバ 2 台、ストレージ 1 台の構成で検証を実施しました。各検証の SQL 等の処理自体はデータベースサーバ上で実行しています。

4.1. システム構成

図 4-1 にシステム構成の概略を示します。クライアントとデータベースサーバ間は 1000Base-T、それぞれのデータベースサーバとストレージ間は FC スイッチを介して 2 本のファイバチャネルを使用して接続しています。

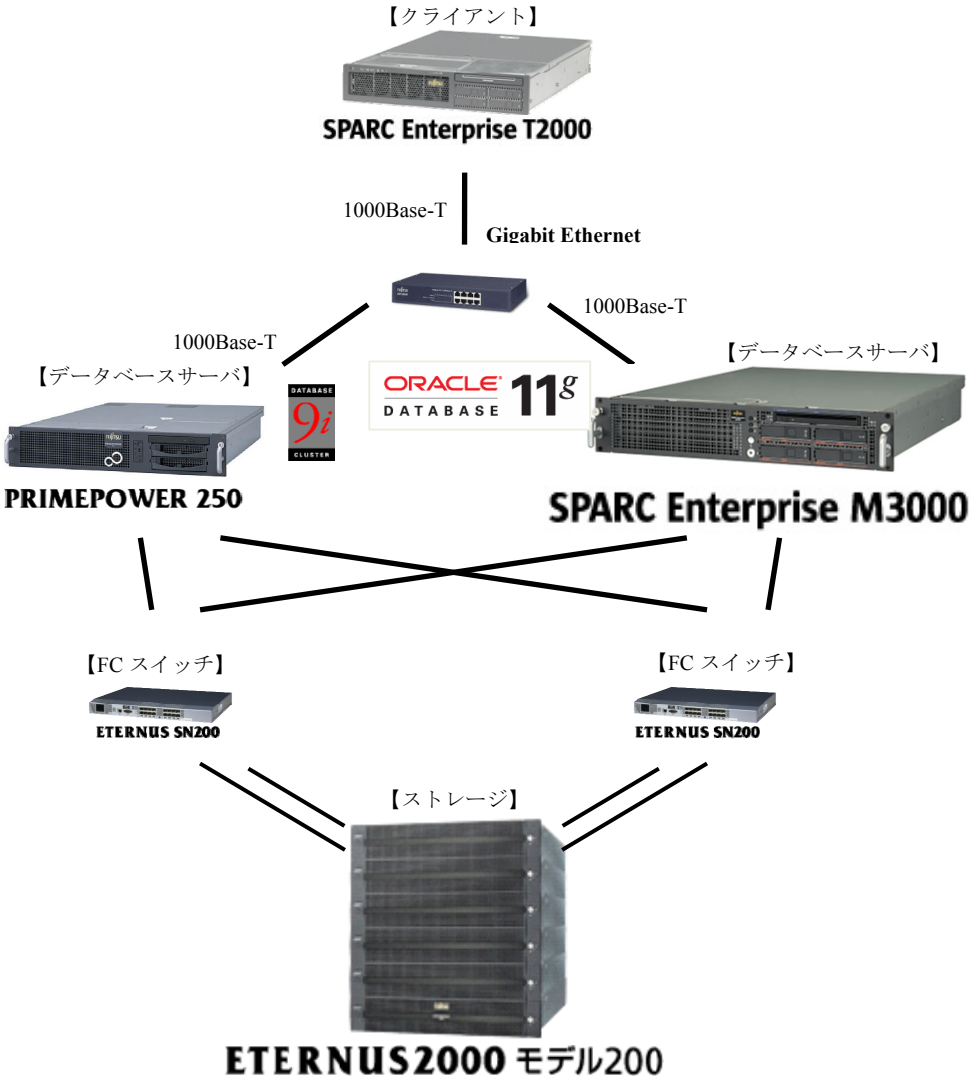


図 4-1 システム構成概略

各機器のスペック、使用したソフトウェア等を示します。

4.1.1. データベースサーバ（移行元）

ハードウェア

モデル	富士通 PRIMEPOWER 250
CPU	SPARC64 Ⅴ 1.1GHz/1MB キャッシュ×2
メモリ	10GB
内蔵 HDD	73GB×1

ソフトウェア

OS	Solaris 9 OS
データベース	Oracle9i Database （9.2.0.8）

4.1.2. データベースサーバ（移行先）

ハードウェア

モデル	富士通 SPARC Enterprise M3000
CPU	SPARC64 Ⅶ 2.52GHz/5MB キャッシュ 1CPU/4 コア/8 スレッド
メモリ	12GB
内蔵 HDD	146GB SAS Disk×2

ソフトウェア

OS	Solaris 10 OS
データベース	Oracle Database 11g （11.1.0.7） ※一部 11.1.0.6 で測定

4.1.3. ストレージ

モデル	富士通 ETERNUS 2000 モデル 200
ディスクドライブ	146GB (15,000rpm) × 32 本

4.1.4. クライアント

ハードウェア

モデル	富士通 SPARC Enterprise T2000
CPU	Ultra SPARC T1 1.2GHz/3MB キャッシュ 1CPU/8 コア/32 スレッド
メモリ	8GB
内蔵 HDD	73GB （10,000rpm） SAS Disk×2

ソフトウェア

OS	Solaris 10 OS
データベースクライアント	Oracle Database 11g （11.1.0.7） ※一部 11.1.0.6 で測定

4.2. ストレージ構成

本検証で用いたストレージの構成について説明します。

図 4-2のストレージ構成図に示すように、32 本のディスクをRAID 1+0（4+4）×4で構成しています。

1 つの RAID グループを 2 つの LUN-V に切り、PRIMEPOWER 250 用と SPARC Enterprise M3000 用にしています。また、各 LUN-V に対して、ファイルシステムを作成し、マウントポイント「/oradata1」、「/oradata2」、「/oradata3」、「/oradata4」でマウントしています。

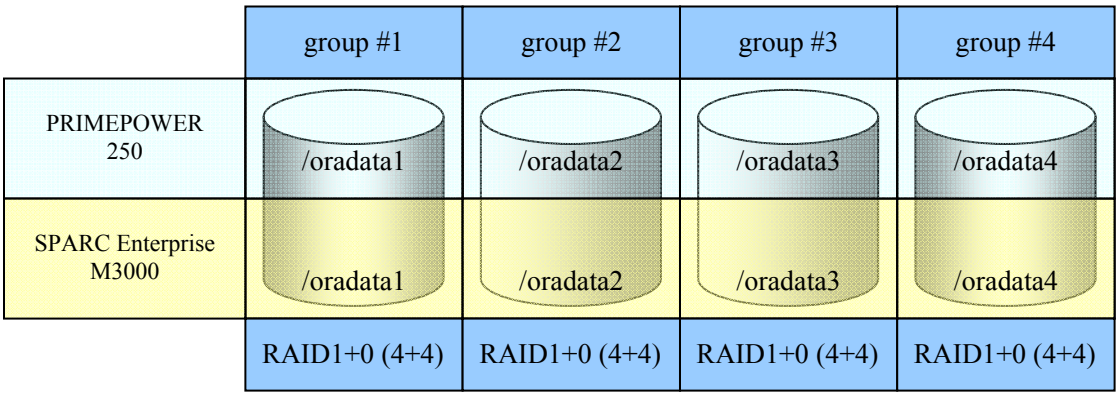


図 4-2 ストレージ構成図

各マウントポイントの用途は表 4-1の通りです。なお、PRIMEPOWER 250 とSPARC Enterprise M3000 で同一の構成です。

表 4-1 ディスクの構成

マウントポイント	用途
/oradata1	オンライントランザクション更新用表領域
/oradata2	オンライントランザクション更新用表領域
/oradata3	・Database Replay 用ディレクトリオブジェクト ・エクスポート、インポートのダンプファイル格納
/oradata4	SYSTEM、SYSAUX、UNDO 表領域、USER 表領域、 一時表領域、REDO ログ、制御ファイル

本検証のオンライントランザクションで更新される表領域のデータファイルは、その他のデータファイル（SYSTEM 表領域、REDO ログ、制御ファイルなど）と RAID グループを分けています。

また、ディレクトリオブジェクトや、エクスポート/インポートのダンプファイルも同様に別の RAID グループへ格納しています。

5. 移行シナリオ

本章では、Oracle Database の移行における全体のフローや必要なポイント、及び、本検証でのフォーカス・エリアについて記述します。

5.1. 移行全体のフロー解説

Oracle Database の移行全体のフローとして、アップグレードの準備、アップグレード処理のテスト、テスト DB の作成と動作確認、本番 DB のバックアップ・アップグレード、本番データベースの調整等のステップが必要になります。また、各ステップには詳細手順として、Step 1 のような新機能の理解やアップグレードパスの決定、アップグレード方法の選択といったサブタスクがあります。(図 5-1 参照)

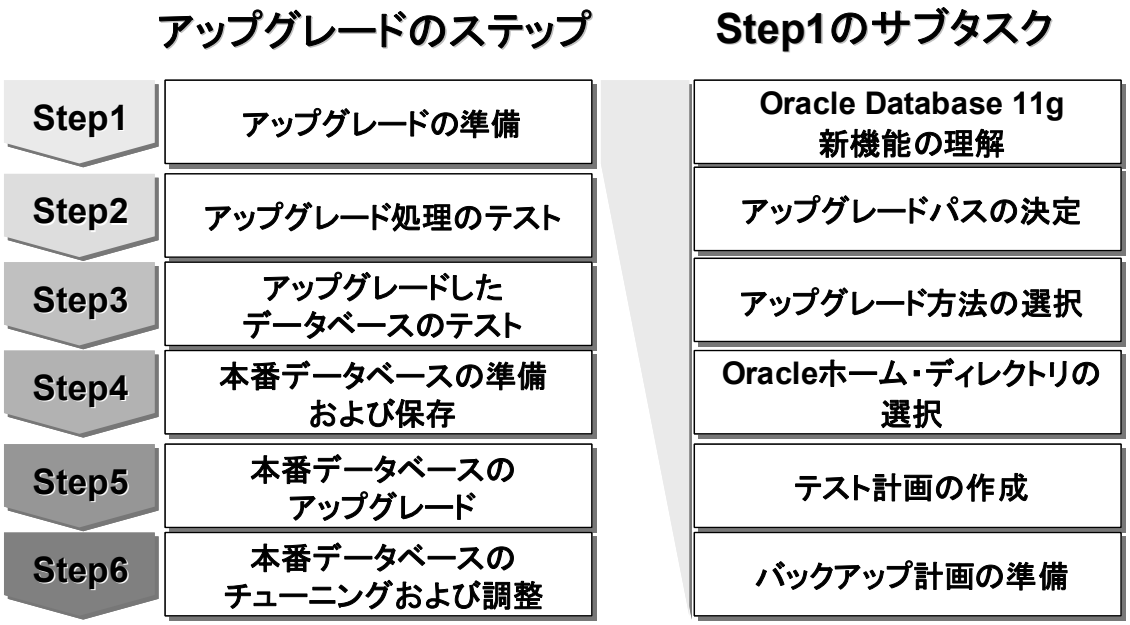


図 5-1 アップグレードのステップ

アップグレードパスについて

Oracle Database 11g へ直接アップグレードできるバージョンは、9.2.0.4 以上、10.1.0.2 以上、10.2.0.1 以上となります。それより以前のバージョンについては、上記バージョンへのアップグレードを経由してから、Oracle Database 11g へのアップグレードを実施します。 (図 5-2 参照)

(例)

- 8.0.6 → 9.2.0.8 → 11.1
- 8.1.7.4 → 10.2.0.x → 11.1
- 9.0.1.4 → 10.2.0.x → 11.1

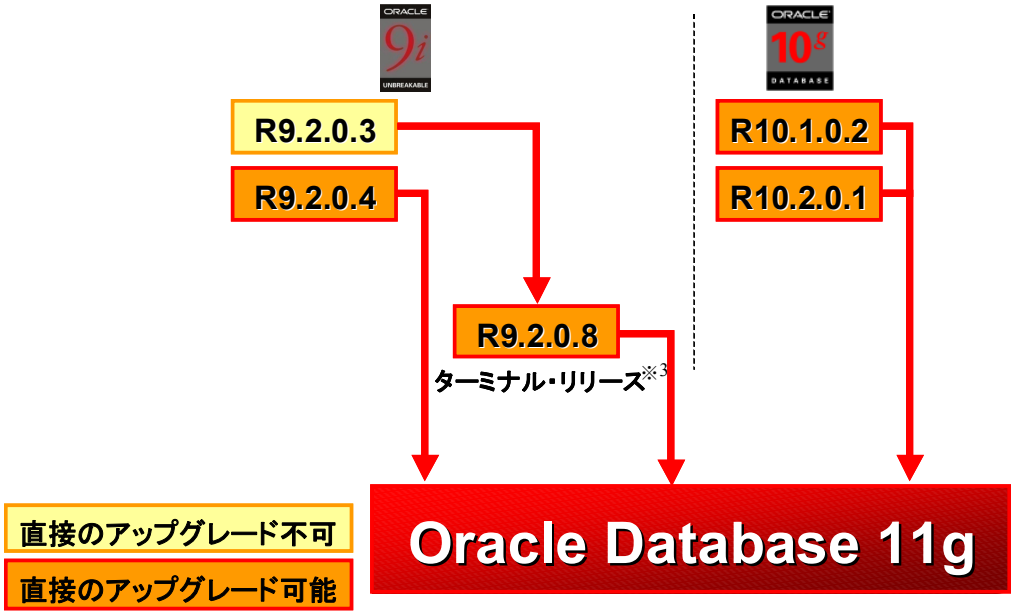


図 5-2 アップグレードパス

アップグレード方法について

データベースをアップグレード方法はいくつかあり、ハードウェア・OS の移行やダウンタイム、万が一失敗した場合の復旧等の要件によって使い分けることが出来ます。

- データベース全体のアップグレード
  - Database Upgrade Assistant (DBUA)
  - アップグレード・スクリプト
- データの移行
  - export/import
  - 表領域移行 (Transportable Tablespace [TTS])
  - アンロード/アップロード
  - DB Link 経由でのデータ・コピー

[3] 9.2.0.8 は、Oracle 9i Database R2 の最終リリースとなります。

アップグレード方法の詳細については、マニュアル「Oracle Database アップグレード・ガイド 11g リリース 1（11.1）E05758-01」、及び MetaLink Note:419550.1 をご参照ください。

表 5-1 各アップグレード方法のメリット・デメリット

方法	メリット	デメリット
DBUA	・ GUI ベースで簡単に処理を実行可能 ・ DB サイズに関係なく比較的高速 ・ 移行に必要なディスク領域は少量	・ DB 全体のみの移行 ・ アップグレード可能なバージョンが限定される
アップグレードユーティリティ	・ DB サイズ関係なく比較的高速 ・ 移行に必要なディスク領域は少量	・ DB 全体のみの移行 ・ アップグレード可能なバージョンが限定される
Export/ Import	・ Oracle のバージョンに依存しない移行 ・ 異なる ハードウェア, OS への移行が可能	・ ダンプファイル用の大容量ディスクが必要 ・ 容量に比例して時間がかかる
TTS	・ メタデータ以外は、ファイルのコピーでよいため高速 ・ 異なる ハードウェア, OS への移行が可能 (Recovery Manager [RMAN]が必要)	・ データファイルのため大容量ディスクが必要 ・ 容量に比例して時間がかかる ・ 上位バージョンへの移行のみ可能 ・ 移行元は Oracle8i 以上であることが必要
Unload/Load	・ 断片化を解消可能 ・ 異なる ハードウェア, OS への移行が可能	・ Unload ファイルのための大容量ディスクが必要 ・ 容量に比例して時間がかかる
DBLink, データ・コピー	・ 断片化を解消可能 ・ 異なる ハードウェア, OS への移行が可能	・ 容量に比例して時間がかかる ・ ネットワーク経由の処理 ・ 操作中は両方のデータベースが同時に使用可能である必要がある

5.2. 本書でのフォーカス・エリア

本書では、Oracle9i Release2 (9.2.0.8)から Oracle Database 11g Release1 (11.1.0.7) へのアップグレードのためのテスト DB の準備とそのテスト DB を使用したパフォーマンス等のテストにフォーカスした内容となっています。

テスト DB の準備については、本番 DB への影響を最小限に抑えるためにオンライン・バックアップから作成する手順をご紹介します。アップグレードに関するテストについては、Real Application Testing (RAT)を使用して本番 DB のワークロード・テストやパフォーマンス・テストを効率的に実行します。

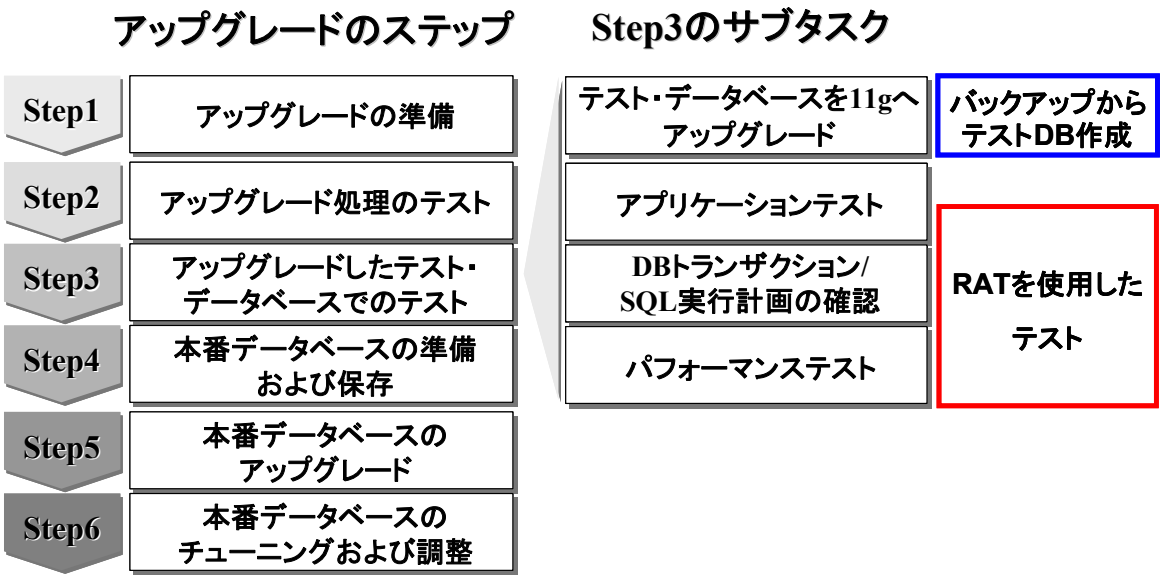


図 5-3 フォーカス・エリア

6. 検証内容／結果

PRIMEPOWER 250 と SPARC Enterprise M3000 の性能比較、Database Replay、SQL Performance Analyzer の検証結果を以下に順を追って示します。

6.1. PRIMEPOWER 250 / SPARC Enterprise M3000 性能比較

本項では、PRIMEPOWER 250＋Oracle9i Database のシステムと SPARC Enterprise M3000＋Oracle Database 11g のシステムでオンライントランザクション処理、バッチ処理の性能比較を検証した結果を示します。

6.1.1. オンライントランザクション処理

検証内容

クライアント（SPARC Enterprise T2000）から在庫管理システムの 5 つの処理（発注処理、支払い残高処理、発注処理状況の確認、配送バッチ処理、売れ筋在庫確認）をシミュレートした OLTP アプリケーションを、さまざまな多重度で実行し、PRIMEPOWER 250 + Oracle9i Database と SPARC Enterprise M3000 + Oracle Database 11g の発注処理の処理性能（CPU 使用率、スループット/レスポンス、ディスク Busy 率）を比較します。

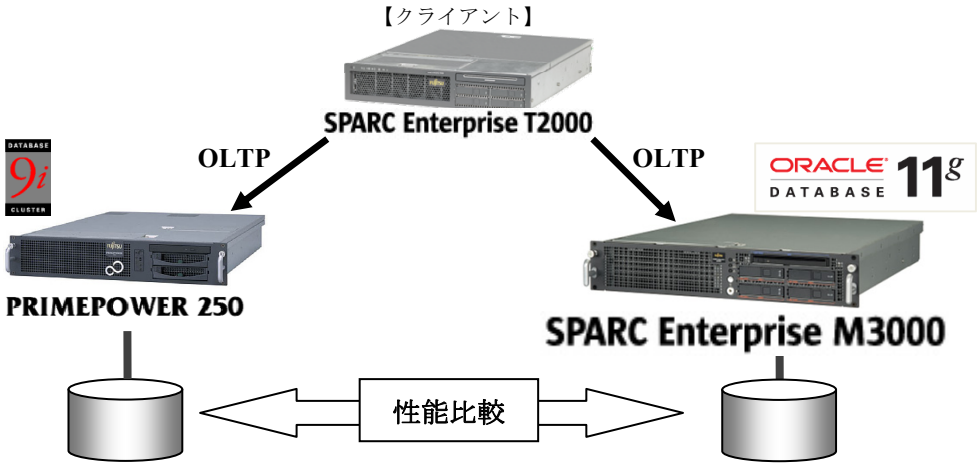


図 6-1 オンライントランザクション処理検証概略図

検証に用いた測定条件は次の通りです。

- DB 規模：85GB
- アプリ多重度：n, n×2,　n×4,　n×10 多重
- アプリ実行時間：35 分（計測時間がアプリ開始 15 分後から 20 分間）

OLTP アプリケーションの表、処理概要、処理内容は次の通りです。

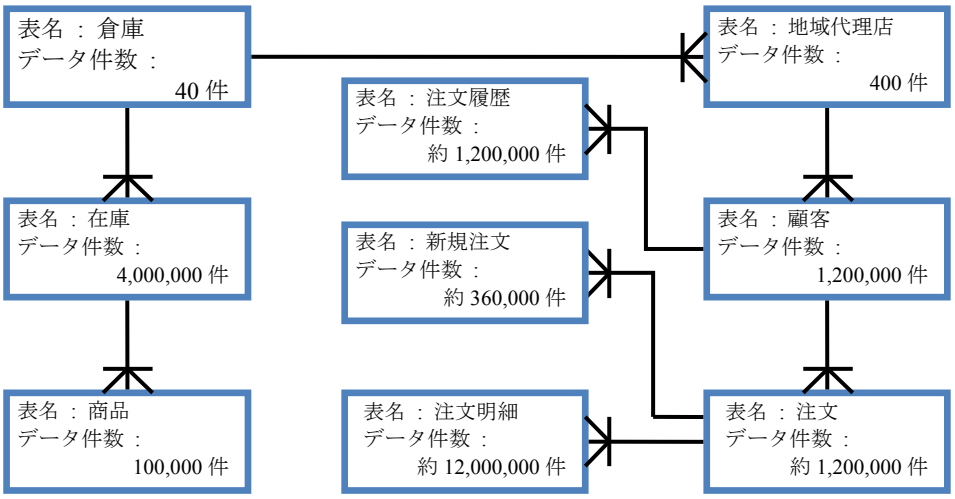


図 6-2 OLTP アプリケーション表

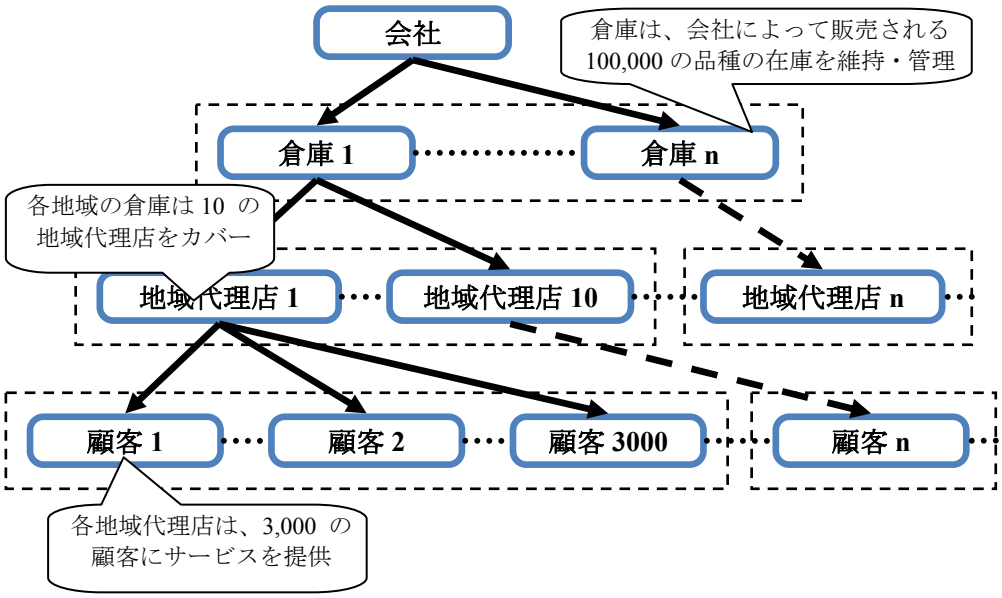


図 6-3 OLTP アプリケーション処理概要

表 6-1 OLTP アプリケーション処理内容

処理内容	負荷	頻度	応答時間	パターン
発注処理	中負荷	高	短い	Read/Write
支払い残高処理	低負荷	高	短い	Read/Write
発注処理状況の確認	中負荷	低	短い	Read Only
配送バッチ処理	中負荷	低	長い	Read/Write
売れ筋在庫確認	高負荷	低	長い	Read Only

検証結果

- スループット

  オンライントランザクション処理におけるスループットを比較すると、すべての多重度において、SPARC Enterprise M3000 + Oracle Database 11g のシステムの方が、PRIMEPOWER 250 + Oracle9i Database のシステムよりもスループットが高く、多重度 n×10 において約 3.5 倍になっています。

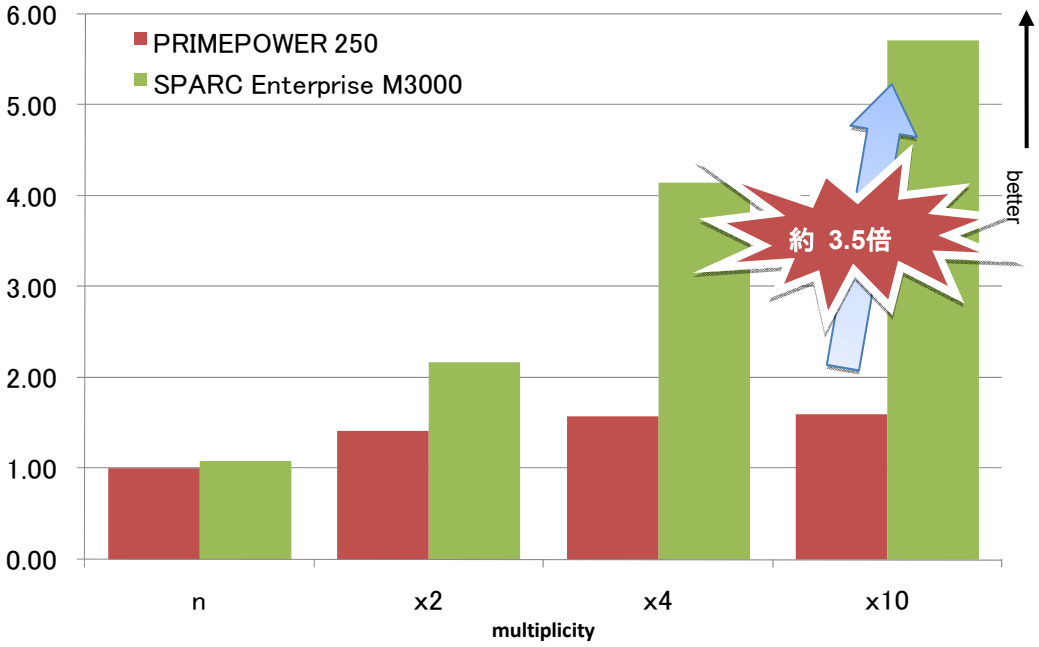


図 6-4 オンライントランザクション処理のスループット

表 6-2 オンライントランザクション処理のスループット量

多重度	PRIMEPOWER 250	SPARC Enterprise M3000
n	1.00	1.08
× 2	1.42	2.17
× 4	1.58	4.15
×10	1.60	5.71

※PRIMEPOWER 250 の n 多重の性能を 1.00 とした場合

- レスポンス

  オンライントランザクション処理におけるレスポンスを比較すると、すべての多重度において、SPARC Enterprise M3000 + Oracle Database 11g のシステムの方が、PRIMEPOWER 250 + Oracle9i Database のシステムよりもレスポンス時間は早く、多重度 n×10 において約 1/6 の時間で処理されています。

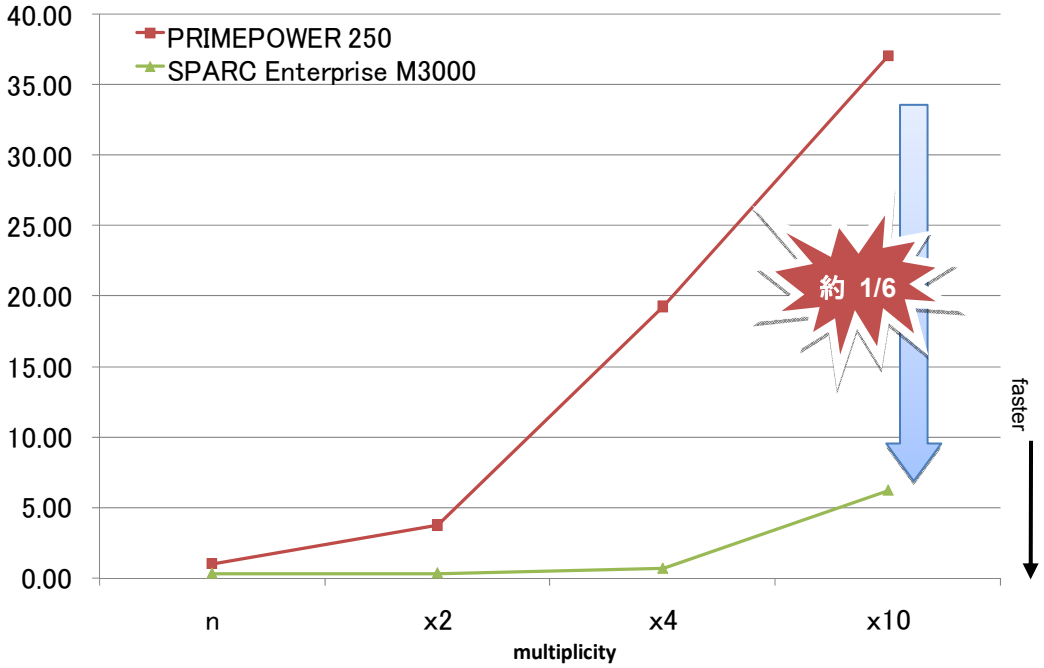


図 6-5 オンライントランザクション処理のレスポンス

表 6-3 オンライントランザクション処理のレスポンス時間

多重度	PRIMEPOWER 250	SPARC Enterprise M3000
n	1.00	0.31
× 2	3.73	0.33
× 4	19.27	0.66
×10	37.08	6.22

※PRIMEPOWER 250 の n 多重の性能を 1.00 とした場合

- CPU 使用率/ディスク Busy 率

  各多重度ごとにおける平均CPU使用率を比較（図 6-6、表 6-4）すると、すべての多重度において、SPARC Enterprise M3000 + Oracle Database 11gのシステムでは、PRIMEPOWER 250 + Oracle9i Databaseのシステムよりも低いCPU使用率でオンライントランザクションが処理されています。

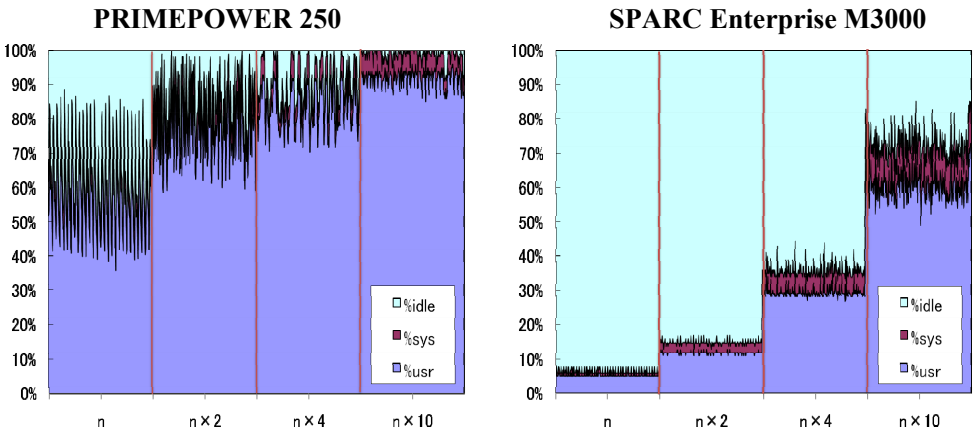


図 6-6 多重度別 CPU 使用率の時間推移

表 6-4 オンライントランザクション処理の平均 CPU 使用率

多重度	PRIMEPOWER 250	SPARC Enterprise M3000
n	59.9%	6.5%
× 2	83.4%	15.1%
× 4	89.7%	35.7%
×10	98.4%	71.7%

また、n×4 多重における REDO 領域のディスク Busy 率を比較（図 6-7）すると、SPARC Enterprise M3000 + Oracle Database 11g のシステムではトランザクションのスループットが高いためディスク Busy 率も高くなっていますが、PRIMEPOWER 250 + Oracle9i Database のシステムでは、CPU がネックになりスループットが低くディスク Busy 率も低くなっていることが分かります。

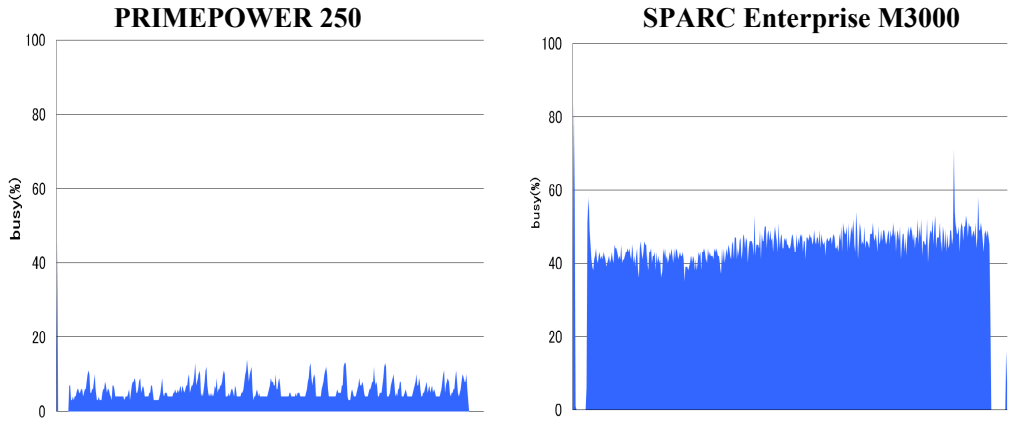


図 6-7 REDO ログ領域のディスク Busy 率の時間推移

6.1.2. バッチ処理

検証内容

SPARC Enterprise M3000 + Oracle Database 11g と PRIMEPOWER 250 + Oracle9i Database で、それぞれバッチアプリケーションを実行し、処理時間を比較します。

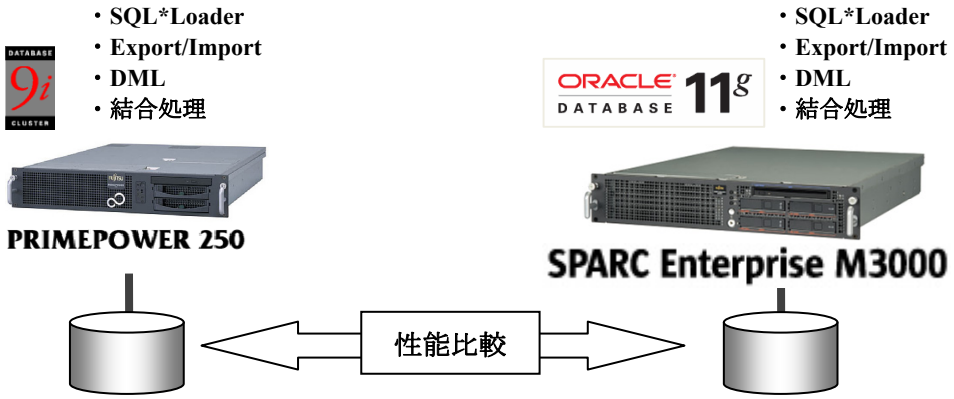


図 6-8 バッチ処理検証概略図

バッチ処理における測定項目は次の通りです。

- エクスポート／インポート

SQL*Loader	TEST_TABLE 以外のテーブルのデータを CSV よりロード
Export※/Import	TEST_TABLE 以外のテーブルのデータを Export によりエクスポート、その後、Import によりロード

- データ操作言語（DML）

Insert	TEST_TABLE に 100 万回の挿入処理
Update	TEST_TABLE の TEST_COMMENT1 列を TEST_NUM 列を条件にした 100 万回の更新処理
Delete	TEST_TABLE の TEST_NUM 列を条件にした 100 万回の削除処理
Fetch	TEST_TABLE の 100 万件のフェッチ処理

- 結合処理

Nested Loop 結合（全表検索）	TRAN_TABLE を駆動表にした MASTER_TABLE との結合処理

※従来の Export ユーティリティは、Oracle Database 11g ではダウングレード用途を除きサポートされません。

バッチ処理対象の表の構造は以下の通りです。

MASTER_TABLE （10,000 件, CSV 1.1MB）

No	項目名	型	長さ	備考
1	M_ITEM1	CHAR	7	PK
2	M_ITEM2	CHAR	10	
3	M_ITEM3	NUMBER	10,2	
4	M_ITEM4	CHAR	7	
5	M_ITEM5	CHAR	10	
6	M_ITEM6	CHAR	7	
		計	100	

TEST_TABLE （1,000,000 件）

No	項目名	型	長さ	備考
1	TEST_NUM	NUMBER	8	PK
2	TEST_ID	NUMBER	2	
3	TEST_DATE	CHAR	8	
4	TEST_COMMENT1	VARCHAR2	50	
5	TEST_COMMENT2	VARCHAR2	50	更新対象
		計	108	

TRAN_TABLE（1,000,000 件, CSV 383MB）

No	項目名	型	長さ	備考
1	T_ITEM1	CHAR	7	PK
2	T_ITEM2	CHAR	10	PK
3	T_ITEM3	CHAR	20	
4	T_ITEM4	VARCHAR2	200	
5	T_ITEM5	NUMBER	10,2	
6	T_ITEM6	CHAR	20	
7	T_ITEM7	VARCHAR2	230	
8	T_ITEM8	DATE	7	
		計	500	

LOG_TABLE （1,000,000 件, CSV 543 MB）

No	項目名	型	長さ	備考
1	L_ITEM1	TIMESTAMP	11	
2	L_ITEM2	CHAR	10	
3	L_ITEM3	CHAR	20	
4	L_ITEM4	CHAR	52	
5	L_ITEM5	VARCHAR2	200	
6	L_ITEM6	VARCHAR2	200	
7	L_ITEM7	VARCHAR2	200	
8	L_ITEM8	DATE	7	
		計	700	

検証結果

SQL*Loader、Export/Import の処理時間を比較すると SPARC Enterprise M3000 + Oracle Database 11g システムの処理時間は、PRIMEPOWER 250 + Oracle9i Database システムに比べ、SQL*Loader では約 1/2、Export、Import では約 2/3 の時間で処理されています。

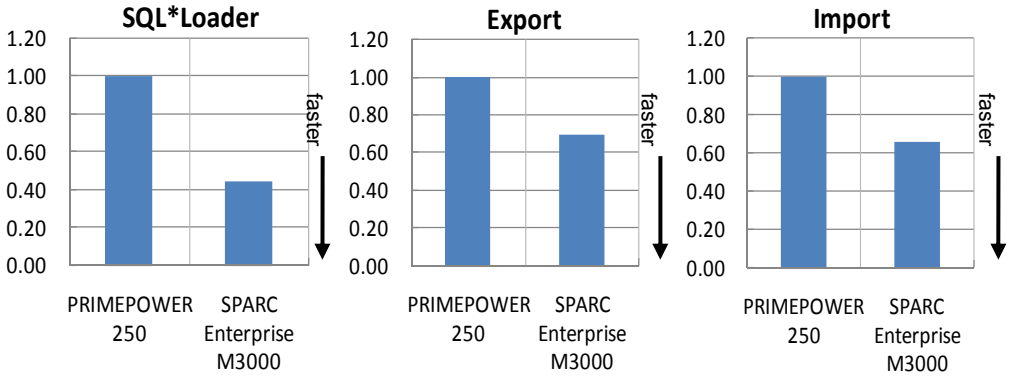


図 6-9 SQL＊Loader, Export/Import の処理時間

表 6-5 SQL*Loader、Export、import の処理時間

	SPARC Enterprise M3000
SQL*Loader	0.44
Export	0.69
Import	0.66

※PRIMEPOWER 250 の性能を 1.00 とした場合

次に、データ操作（DML）と結合処理の処理時間を比較すると、データ操作全体で約 1/3、結合処理では約 1/5 の時間で処理されています。

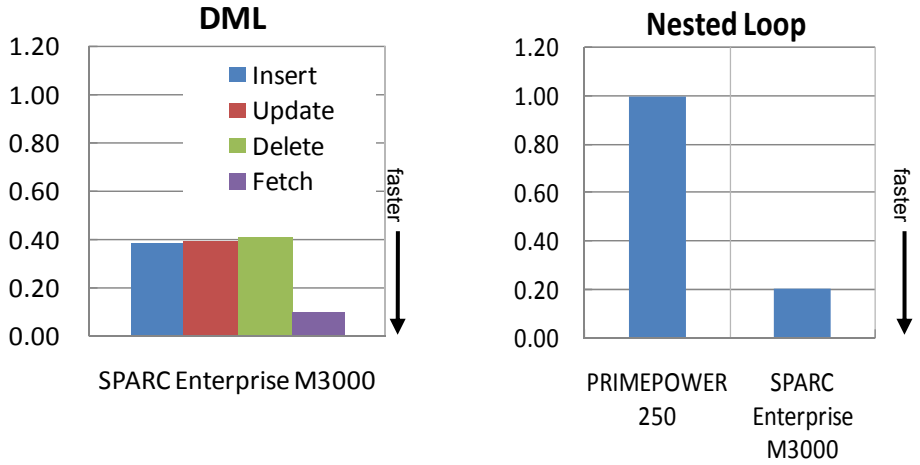


図 6-10 データ操作（DML）、結合処理の処理時間

表 6-6 データ操作（DML）、結合処理の処理時間

	SPARC Enterprise M3000
100 万件 Insert	0.39
100 万件 Update	0.39
100 万件 Delete	0.41
100 万件 Fetch	0.10
合計	0.38
結合処理	0.21

※PRIMEPOWER 250 の性能を 1.00 とした場合

以上より、バッチ処理における、SPARC Enterprise M3000 + Oracle Database 11g システムは PRIMEPOWER 250 + Oracle9i Database システムに比べ、約 2 倍の性能であるという結果が得られました。

6.2. Database Replay

6.2.1. 機能詳細

Database Replay は、本番システムで取得したワークロードを、テスト・システム上でリプレイすることが可能です。

Database Replay は、本番環境でのワークロード取得とテスト環境でのリプレイと分析から成る以下のテスト・ワークフローを適用しています。

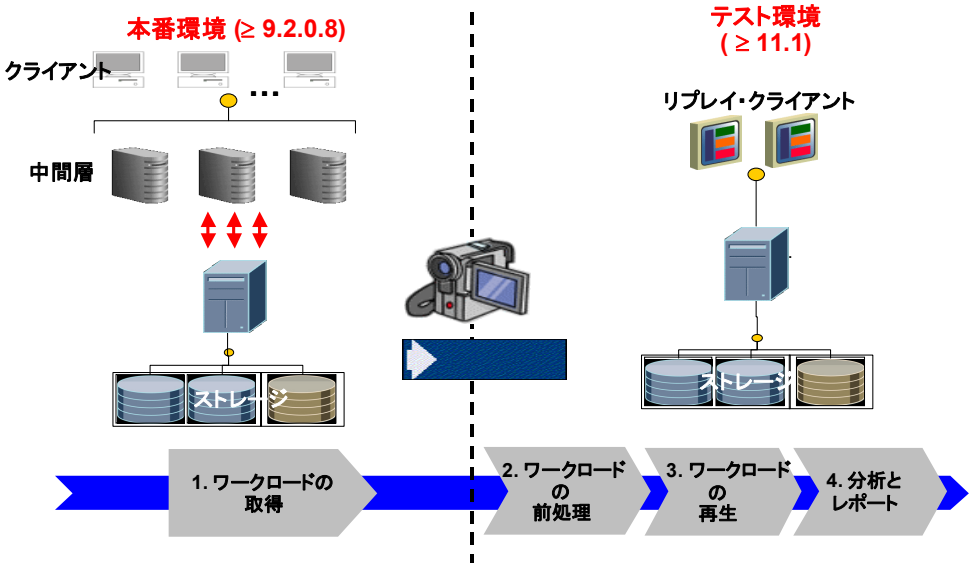


図 6-11 Database Replay 主要処理手順

1. ワークロードの取得

ワークロードの取得を有効にすると、Oracle Database に対するすべての外部クライアントの要求が追跡され、ファイル・システム上のバイナリ・ファイル（キャプチャ・ファイル）に格納されます。オラクル社では、ワークロードの取得を実行する前にデータベース全体のバックアップの取得を推奨しています。ユーザーは、キャプチャ・ファイルの場所およびワークロードの開始と終了の時間を指定します。ワークロードの取得処理中は、外部のデータベース・クライアントからのリクエストに関するすべての情報がキャプチャ・ファイルに書き出されます。

2. ワークロードの前処理

ワークロードを取得したら、キャプチャ・ファイル内の情報を処理する必要があります。この前処理では、キャプチャ・データをリプレイ用のファイル（リ

プレイ・ファイル）に変換し、ワークロードのリプレイに必要なすべてのメタデータを作成します。前処理は、個別の RDBMS バージョン上でリプレイを行うために、全ての取得されたワークロードに対して一度実行されなければなりません。もし、リプレイするデータベースのバージョンが変更される場合、前処理を再度実行する必要があります。前処理は集中的な処理なので、本番システム上で実行すべきではありません。

3. ワークロードの再生（リプレイ）

取得したワークロードの前処理が完了すれば、リプレイ準備が整います。ワークロード･リプレイ･クライアントと呼ばれるクライアント･プログラム（wrc 実行ファイル）は、リプレイ・ファイルを処理して取得元のシステムとまったく同じタイミングおよび同時実行性で、データベースに対する呼出しを発行します。取得したワークロードによっては、ワークロードを正しくリプレイするために複数のリプレイ・クライアントが必要になることもあります。ワークロードに必要なリプレイ･クライアントの数を決定するための見積りユーティリティも用意されています。DML および SQL 問合せを含むワークロード全体がリプレイされるため、テスト・システム内のデータが、取得が開始される直前の本番システム内のデータと完全に一致していることが重要です。これにより、レポートの生成に使用できる信頼性の高い分析が可能となります。

4. 分析とレポート

取得とリプレイの詳細な分析ができるように、広範囲なレポートが用意されています。リプレイ中に発生したエラーはレポートに出力されます。DML や問合せによって返される行の相違も表示されます。取得時とリプレイ時の基本的なパフォーマンスの比較も用意されます。詳細な分析が必要な場合は、Automatic Workload Repository (AWR) レポートによって、取得時とリプレイ時のパフォーマンス統計情報の詳細な比較も可能です。

リプレイ・パラメータ

ワークロードのリプレイは、デフォルトでは取得されたトランザクション実行順序／タイミングを維持する形でリプレイされますが、これらのトランザクション実行順序／タイミングは、リプレイ・パラメータの設定によって変更することが可能です。

どのような目的でのテストを実施したいかによって、特定の要件に適応するために後述のパラメータ設定を変更することができます。

- synchronization パラメータ　[TRUE (Default) / FALSE]

  ワークロードのリプレイ中に同期化を使用するかどうかを指定します。この

パラメータを TRUE に設定すると、取得したワークロードでの COMMIT 順はリプレイ中維持され、すべてのリプレイ・アクションは、すべての依存 COMMIT アクションが完了した後にのみ実行されます。言い換えると、リプレイされるそれぞれのコールが取得された時と同じように動くことが保証され、アプリケーションのテスト（SQL レベルの非互換抽出）に有益なワークロードです。FALSE に設定すると、保証が無く、負荷テストに有益なワークロードです。デフォルト値は TRUE です。

- think_time_auto_correct パラメータ　[TRUE (Default) / FALSE]
  リプレイ時のユーザー・コールの完了が取得時よりも長くかかる場合に、コール間の思考時間を（think_time_scale パラメータに基づいて）修正します。リプレイが取得時のタイミングに追いつけるように、連続したコール間の思考時間は減少されます。
- connect_time_scale パラメータ　[0 ~ 100 (Default)]
  ワークロードの取得を開始した時間から指定された値でセッションが接続する時間までの経過時間をスケール変更し、%値として解釈されます。このパラメータを 100 に設定すると、全てのユーザーはおよそ同じタイミングでデータベースへ接続します。50 に設定すると、2 倍の速度（1/2 のタイミング）で接続することを試み
  ます。
- think_time_scale パラメータ　[0 ~ 100 (Default)]
  同じセッションからの連続する 2 つのユーザー・コール間の経過時間をスケール変更し、%値として解釈されます。このパラメータを 0（ゼロ）に設定すると、リプレイされるリクエスト間の思考時間が無くなります。これは、リクエストが可能な限り高速に RDBMS へ送信されることを意味します。

リプレイ結果の評価方法

次に、リプレイ実行後の実行結果の評価方法について述べます。大まかな流れとしては、データベースへの変更／リプレイの準備が正確に完了しているか、リプレイ（取得）対象のワークロードが妥当であったかを確認します。

ユーザー・コールの成功率は 90%を目標とします。理想的には 100%に近づくのが望ましいですが、これは取得されたワークロードに依存する（サポートされていないコール、バックグラウンド・ワークロード、もしくはデータベースへの変更に起因して発生する問題点などが含まれる可能性があります）ため、何がコールに失敗した原因なのかをレポートから特定します。そして次に、失敗したコールがワークロードにとって許容できるものであるかを判断します。

評価方法の詳細については、下記のフローチャートを参考にして下さい。

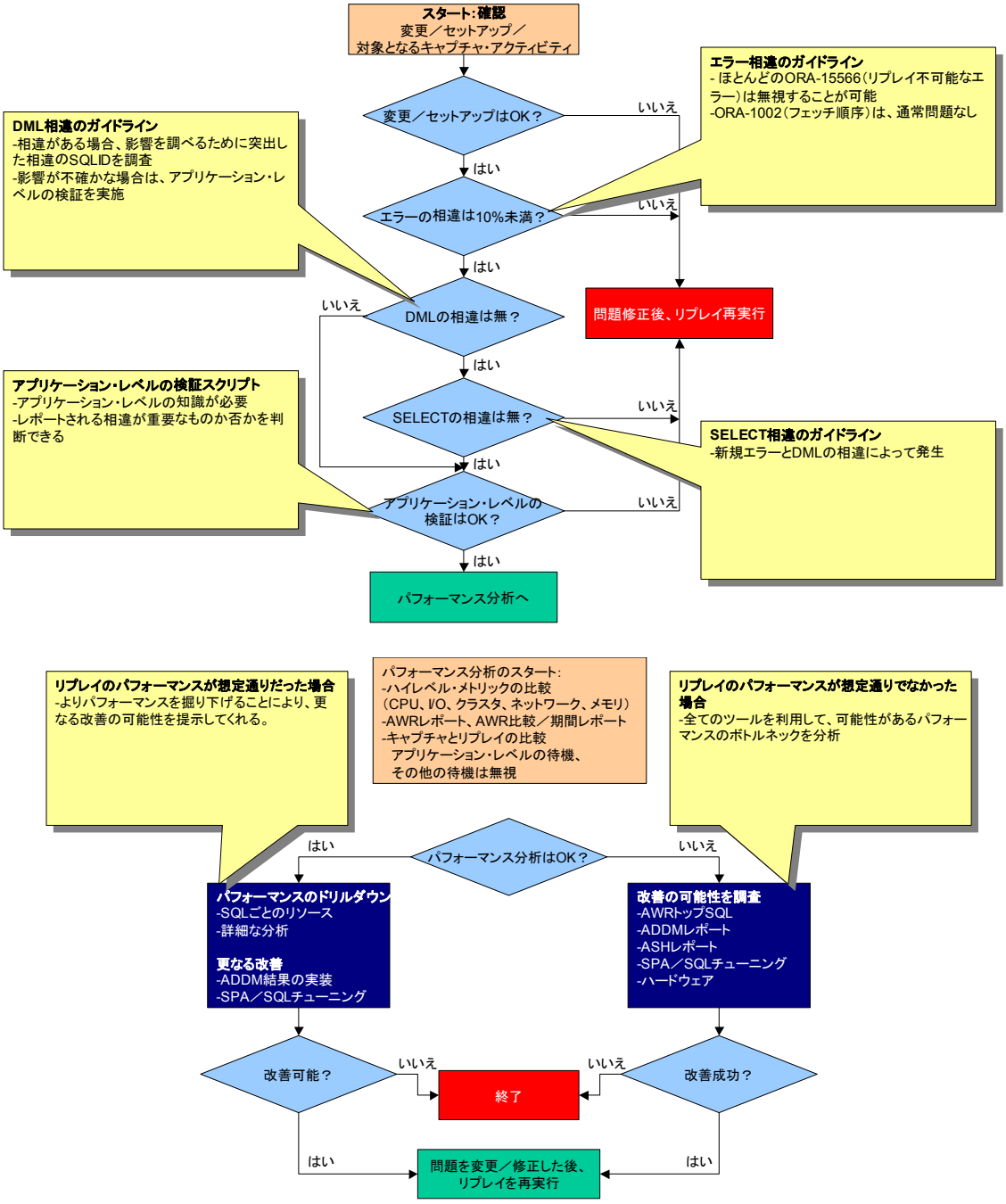


図 6-12 リプレイ結果の評価方法　フローチャート

6.2.2. 前提条件

Database Replay を実行する上で以下の点を考慮する必要があります。

- **データの一致**

  テスト環境では、アプリケーション・データの状態を本番環境と論理的に同一にしておく必要があります。

- **パッチの適用**

  Oracle9i Database でワークロードを取得し、Oracle Database 11g でリプレイをするためには、Oracle9i Database が 9.2.0.8 + One-Off Patch#6973309 である必要があります。

  なお、パッチ適用に関する詳細は MetaLink の Note 560977.1 を参照して下さい。

- **リプレイ・クライアント数の見積もり**

  リプレイの間、取得されたワークロードは、ワークロード・リプレイ・クライアント（以下 wrc）というマルチスレッドプログラムによってテスト・データベースへ発行されます。

  リプレイ・クライアントはワークロードのリプレイに最適なリプレイ・クライアント数、および CPU 数を評価するために wrc を測定モードで実行する必要があります。

  次に、測定モードでワークロード・リプレイ・クライアントを起動した例を示します。

```
$ wrc mode=calibrate replaydir= <WORKLOAD_DIRECTORY 名>

Workload Replay Client: Release 11.1.0.7.0 - Production on 水 11 月 5 15:22:30
2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.

Report for Workload in: /export/home/oracle/replay/REP40_RMAN_9208_2

Recommendation:
Consider using at least 4 clients divided among 1 CPU(s)
You will need at least 150 MB of memory per client process.
If your machine(s) cannot match that number, consider using more clients.
Workload Characteristics:
- max concurrency: 40 sessions
- total number of sessions: 40
Assumptions:
- 1 client process per 50 concurrent sessions
- 4 client process per CPU
- 256 KB of memory cache per concurrent session
```

```
- think time scale = 100
- connect time scale = 100
- synchronization = TRUE
```

※WORKLOAD_DIRECTORY 名　：事前処理済のワークロードのファイルが含まれるディレクトリ

測定モードでの評価の結果、このように 4 つのワークロード・リプレイ・クライアントが推奨されました。リプレイは推奨されたクライアント数をチェックしないので、より多くの、もしくはより少ないクライアントを使用することができるかもしれません。
上記の結果に従ってリプレイを実施します。

6.2.3. Database Replay実行環境の準備方法

手順サマリ

本検証におけるテスト環境の準備手順と Database Replay 実行手順を以下に記述します。(図 6-13 参照)

1. 本番 DB でバックアップを取得
2. 本番 DB でワークロードを取得
3. テスト環境で バックアップから テスト DB (9i R2)を複製
4. テスト DB を 9i R2 から 11g へアップグレード
5. テスト DB （11g）で Database Replay を実行

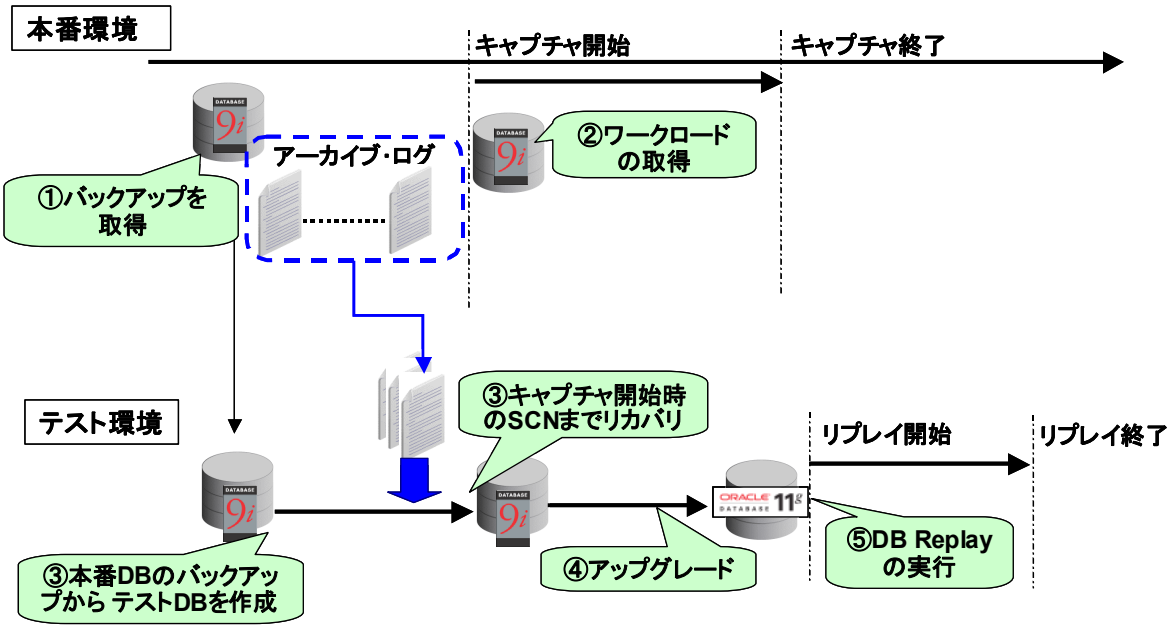


図 6-13 手順サマリ

0. 手順詳細

前提条件:

手順を実施するために以下の前提条件を確認して下さい。

- テスト環境には、同一のプラットフォーム (OS)、同一バージョンの Oracle 環境 (Oracle9i Database R2)を用意
- アーカイブ・ログ・モード

1. 本番DBのバックアップを取得

本番データベースを複製するための準備について、本番データベースを「old」、テスト・データベースを「new」として説明します。

1-1. テスト・データベース用の初期化パラメータ・ファイル init<sid>.ora を作成

(1) 初期化パラメータ・ファイルを編集

旧：db_name = "old"

新：db_name = "new"

初期化パラメータ・ファイルで指定している全てのディレクトリをテスト・データベース用のディレクトリに変更 (control_files, background_dump_dest, user_dump_dest, log_archive_dest_n, etc.)

(2) 修正完了後、初期化パラメータ・ファイルを移行先データベースの $ORACLE_HOME/dbs 配下へ格納

初期化パラメータ・ファイルで指定している全てのディレクトリをテスト・データベース用のディレクトリに変更 (control_files, background_dump_dest, user_dump_dest, log_archive_dest_n, etc.)

1-2. 制御ファイル作成用スクリプトの準備

本番データベースにて下記コマンドを実行

```
SQL> alter database backup controlfile to trace;
```

トレース・ファイルが、本番データベースの user_dump_dest で指定しているディレクトリに作成されます。

1-3. スクリプトの編集

上記 1-2 で作成したスクリプトを編集

(1) 下記の部分を修正

旧

```
CREATE CONTROLFILE REUSE DATABASE "OLD" NORESETLOGS ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 292
LOGFILE
  GROUP 1 '/oradata4/redo01.log'  SIZE 1024M,
  GROUP 2 '/oradata4/redo02.log'  SIZE 1024M,
  GROUP 3 '/oradata4/redo03.log'  SIZE 1024M
-- STANDBY LOGFILE
DATAFILE
  '/oradata4/system01.dbf',
  '/oradata4/sysaux01.dbf',
  '/oradata4/undotbs01.dbf',
  '/oradata4/users01.dbf'
CHARACTER SET JA16EUC;
```

新

```
CREATE CONTROLFILE SET DATABASE "NEW" RESETLOGS ARCHIVELOG
    MAXLOGFILES 16
    MAXLOGMEMBERS 3
    MAXDATAFILES 100
    MAXINSTANCES 8
    MAXLOGHISTORY 292
LOGFILE
  GROUP 1 '/oradata4/redo01.log'  SIZE 1024M,
  GROUP 2 '/oradata4/redo02.log'  SIZE 1024M,
  GROUP 3 '/oradata4/redo03.log'  SIZE 1024M
-- STANDBY LOGFILE
DATAFILE
  '/oradata4/system01.dbf',
  '/oradata4/sysaux01.dbf',
  '/oradata4/undotbs01.dbf',
  '/oradata4/users01.dbf'
CHARACTER SET JA16EUC;
```

- LOGFILE や DATAFILE のファイルをテスト・データベース用のディレクトリに変更
- CREATE CONTROLFILE 文以外(トレース・ファイルのヘッダ部分や

RECOVER 文、ALTER 文)は全て削除

(2) 修正後、テスト環境へスクリプトを格納（任意の場所）

1-4. 本番データベース上の全データ・ファイルのホット・バックアップを取得

(1) 本番データベースを構成するデータ・ファイルを確認

```
SQL> select file_name,tablespace_name from dba_data_files;
```

(2) (1)で確認した全ての表領域をバックアップモードに設定

```
SQL> alter tablespace <TABLESPACE_NAME> begin backup;
```

(3) バックアップ対象の表領域がバックアップモードであることを確認

```
SQL> select tablespace_name,file_id,v$backup.status
2>   from dba_data_files,v$backup
3>   where dba_data_files.file_id = v$backup.file#;
```

(4) OS のコピーコマンドなどにより、バックアップを取得

```
$ cp <FILE_NAME> <BACKUP_FILE_NAME>
```

(5) 上記 (2)で設定したバックアップモードを解除

```
SQL> alter tablespace <TABLESPACE_NAME> end backup
```

取得したオンライン・バックアップをテスト・データベース用のディレクトリに移動

制御ファイルやオンライン REDO ログファイルをコピーする必要はありません。

ヒント：例えば Data Pump、ロジカル／フィジカル・スタンバイなど、キャプチャ時と論理的に同じ時点へリストアするための別のメカニズムを使用することも可能です。

2. 本番データベースでのワークロードの取得

本番データベースでのワークロードの取得方法について説明します。

2-1. ディレクトリ・オブジェクトの作成

本番データベースでディレクトリ・オブジェクトを作成

```
SQL> create directory ディレクトリ・オブジェクト名
2>   as 'ディレクトリ・オブジェクトのパス'
```

2-2. ワークロードを取得する際のフィルタを設定

設定例

```
SQL> begin
2>   DBMS_WORKLOAD_CAPTURE.ADD_FILTER (
3>   fname         => 'TPCCONLY',
4>   fattribute      => 'USER',
5>   fvalue         => 'TPCC'
6>   );
7>   end;
8>   /
```

2-3. ワークロードの取得を開始

```
SQL> begin
2>   DBMS_WORKLOAD_CAPTURE.START_CAPTURE (
3>   name          => 'ワークロード取得名',
4>   dir            => 'ディレクトリ・オブジェクト名',
5>   duration       => NULL,
6>   default_action  => 'EXCLUDE',
7>   auto_unrestrict => TRUE
8>   );
9>   end;
10>  /
```

2-4. ワークロードの取得を終了

```
SQL> exec DBMS_WORKLOAD_CAPTURE.FINISH_CAPTURE ();
```

3. テスト・データベースの作成

3-1. 本番データベースのカレントオンライン REDO ログをアーカイブ

```
SQL> alter system archive log current;
```

3-2. アーカイブ・REDO ログのコピー

本番 DB データベースのアーカイブ REDO ログをテスト・データベースのアーカイブ・ログの格納先（手順 1-1 で指定した、初期化パラメータの「log_archive_dest_n」）へコピー

3-3. テスト・データベースの作成

移行先マシンで環境変数 ORACLE_SID を移行先データベースの SID に変更

(1) SQL*Plus を sys ユーザもしくは internal ユーザにて起動しインスタンスを NOMOUNT で起動

```
SQL> startup nomount;
```

(2) 手順 1-3 で作成した CREATE CONTROLFILE のスクリプトを実行

(3) 本番データベースでワークロードの取得開始時の SCN を確認

> DBA_WORKLOAD_CAPTURES 静的データ・ディクショナリ・ビューを検索する、もしくは手順 2-1 で作成したディレクトリ・オブジェクト内にある「wcr_xx.text」や「wcr_xx.html」から取得開始時の SCN を確認可能

(4) RMAN で本番データベースのワークロード取得開始直前まで不完全リカバリを実施し、テスト・データベースを作成

```
$ rman target /
RMAN> recover database until scn 上記(3)で確認した SCN 番号
RMAN> alter database open resetlogs;
```

(5) 一時表領域の作成

作成例

```
SQL> ALTER TABLESPACE TEMP ADD TEMPFILE
2>    '/oradata4/tpc1_9i/temp01.dbf'
3>    SIZE 1024M REUSE AUTOEXTEND OFF;
```

4. テスト DB のアップグレード

テスト・データベースを 9.2.0.8 から 11.1.0.7 へアップグレードします

4-1. テスト環境に Oracle Database 11g をインストール

テスト環境に Oracle Database 11gR1 をインストール

インストール方法については、該当のプラットフォームのインストレーション・ガイドをご参照下さい。

4-2. データベースのアップグレード
テスト・データベースを 9i から 11g へアップグレード
- 本検証では、DBUA を使用

Oracle Database 11g R1（11.1）へのアップグレードではタイムゾーン・ファイルのバージョンが 4 以上である必要があります。タイムゾーン・ファイルをバージョン 4 に入れ替えるためには「patch5632264 － TZ V4 file」を適用します。patch5632264 の詳細、適用方法については「KROWN#123651 米国における 2007 年夏時間変更への対応に関する個別パッチ適用方法（DB, Client, JVM）をご参照下さい

4-3. Patch Set Release (PSR)11.1.0.7 の適用とアップグレード
Oracle Database 11g に PSR11.1.0.7 を適用し、データベースをアップグレード
します。適用方法については PSR 内の README を確認して下さい。

5. テスト DB で Database Replay を実行
5-1. ワークロードの前処理
5-2. ワークロードのリプレイ
5-3. 分析とレポート

Enterprise Manager を使用したワークロードの前処理、ワークロードのリプレイ、分析とレポートの手順については、マニュアル「Oracle® Database Real Application Testing User's Guide 11g Release 1 (11.1) E12253-01」をご参照ください。

6.2.4. ワークロード取得時の負荷

検証内容

ワークロードの取得を開始すると、既存のサーバー・プロセスがワークロードを取得するために動作するため、オーバヘッドが発生します。取得する場合としない場合の OLTP モデルの発注処理の処理性能（スループットとレスポンス、CPU 使用率、ワークロード出力デバイスのディスク I/O）を比較することで取得時の業務への影響を検証します。今回の検証環境では、データファイルとキャプチャ・ファイルの出力先を別に設定し、完全に I/O を分離しています。

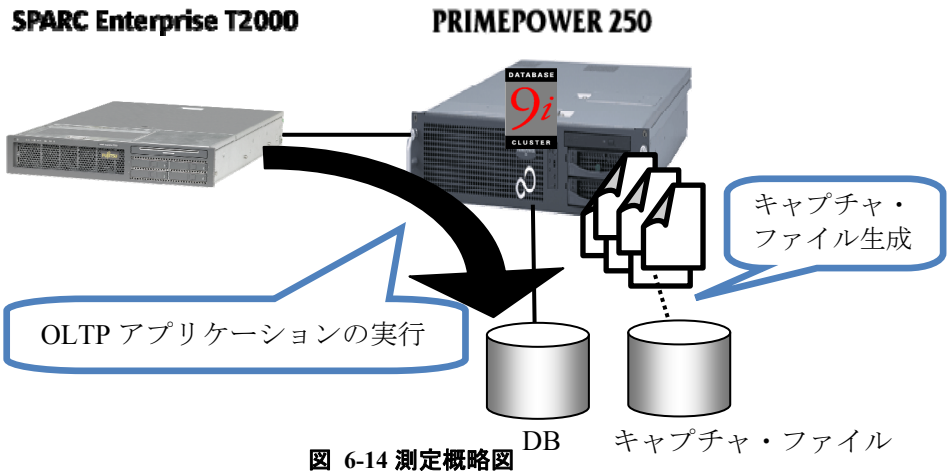


図 6-14 測定概略図

検証結果

- **スループット/レスポンス**

図 6-15 に取得する場合としない場合のスループットとレスポンスを示します。

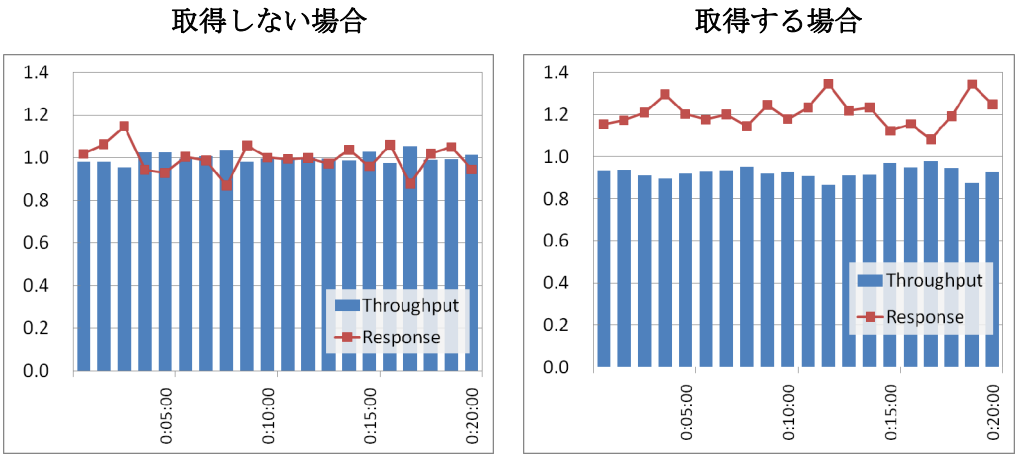


図 6-15 取得しない場合と取得する場合のスループットとレスポンス

表 6-7 平均スループットと平均レスポンス

ワークロードの取得	平均スループット	平均レスポンス
しない	1.00	1.00
する	0.93	1.21

※取得しない場合の性能を 1.00 とした場合

ワークロードを取得しない場合に比べ、取得する場合は、スループットは約7%の減少で留まりましたが、レスポンスについては、20%ほど増加しています。これは、もともとのレスポンスが 100 ミリ秒未満なので取得による影響が大きくなったと想定されます。

- **CPU 使用率**

図 6-16 に CPU 使用率の時間推移を示します。
ワークロードを取得しない場合の CPU 使用率は平均 83.4%、取得する場合は平均 85.7%になりました。

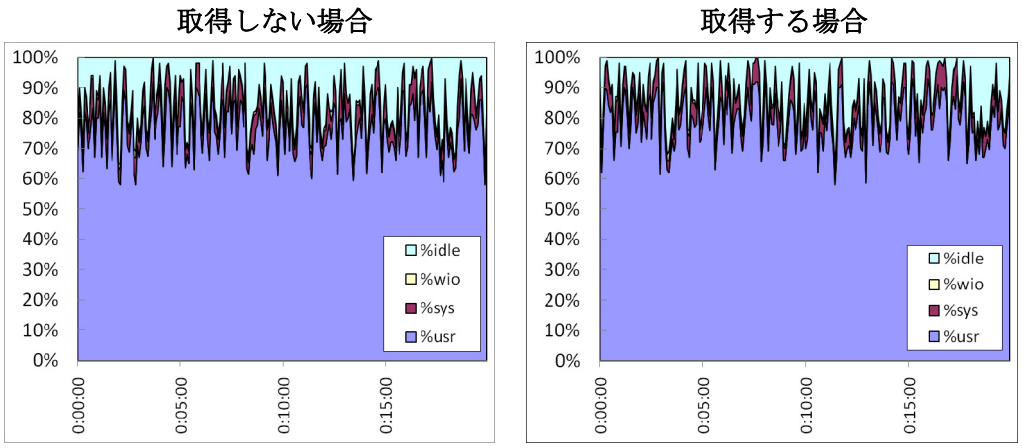


図 6-16 取得しない場合と取得する場合の CPU 使用率

表 6-8 計測時間中の平均 CPU 使用率

ワークロードの取得	平均 CPU 使用率（sys+user）
しない	83.4%
する	85.7%

ワークロードを取得しない場合に比べ、取得する場合の CPU 使用率は 2.7%の増加で留まっており、CPU 使用率への影響は、ほぼ見られません。

- **ワークロード出力デバイスのディスク I/O**

図 6-17 にワークロード出力デバイスのディスク I/O とディスク Busy 率を比較した結果を示します。

ワークロードを取得する場合、ワークロード出力デバイスへの平均書込量は、約 230KB/s、ディスク Busy 率もほぼ 0%であり、ごく微量であることがわかります。

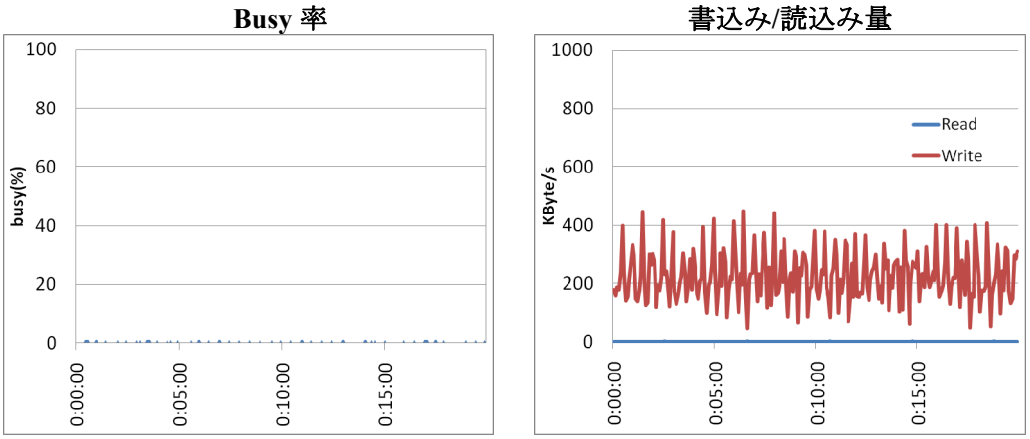


図 6-17 ワークロード出力デバイスのディスク I/O

これらの結果からワークロード取得における CPU 使用率やワークロード出力デバイスのディスク I/O のオーバヘッドは小さいと考えられます。しかしながら、取得対象とするトランザクションの内容により傾向は異なるので注意が必要です。

6.2.5. リプレイ・パラメタの効果

検証内容

リプレイ・パラメータの think_time_scale は、ユーザー・コール間の経過時間のスケールを変更することができます。取得時とは、異なるスケールに設定することで SQL の発行間隔を変更し、システムに与える負荷を制御することができます。

think_time_scale を 100 と 50 に設定した時のリプレイについて比較（図 6-18）します。100 を設定した場合には、取得時と同じ間隔で SQL が発行されますが、50 を設定した場合には、取得時の半分の間隔で SQL が発行されます。このように think_time_scale を小さくすることで、短い間隔で SQL が実行されシステムに与える負荷が高くなります。

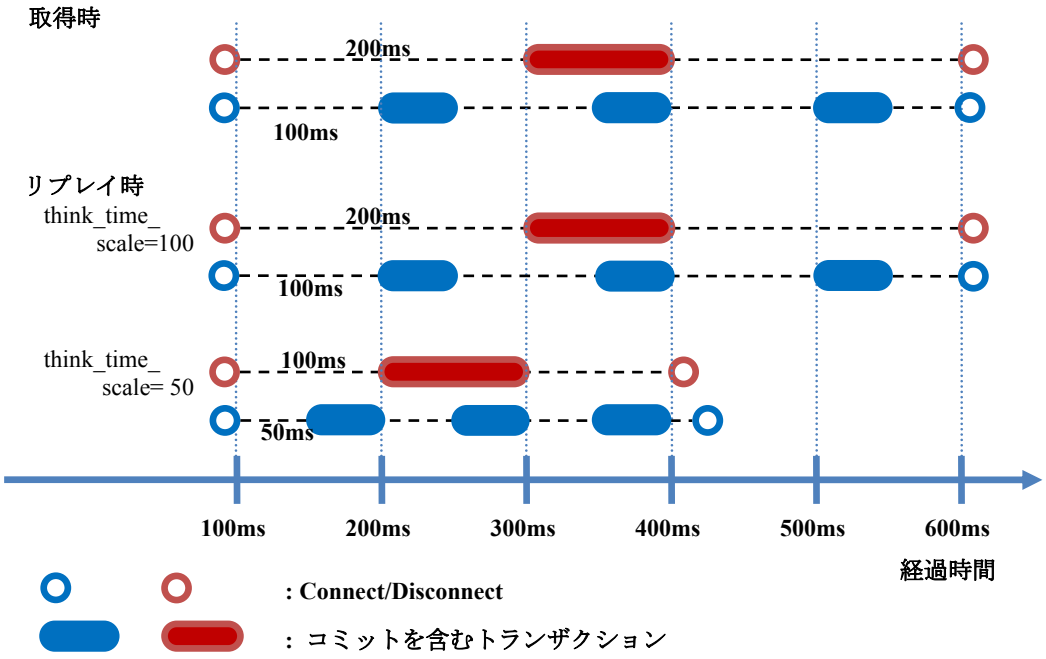


図 6-18 think_time_scale パラメータの効果

検証結果

- **CPU 使用率**

図 6-19 に think_time_scale を 100 に設定した場合と 50 に設定した場合の CPU 使用率の時間推移を示します。

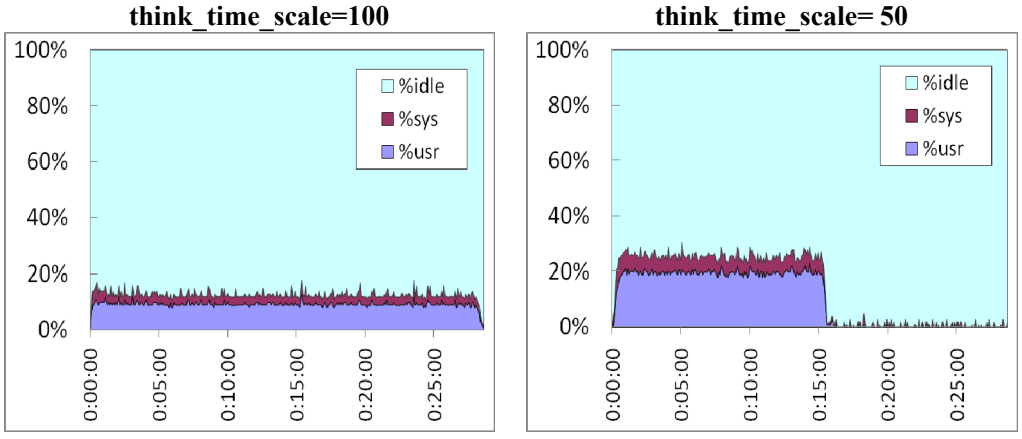


図 6-19 パラメータ値による CPU 使用率の比較

think_time_scale を 100 に設定した場合の平均 CPU 使用率は約 12%で、処理全体のリプレイ時間は約 28 分であったのに対し、think_time_scale を 50 に設定した場合の平均 CPU 使用率は約 23%で、処理全体のリプレイ時間は約 16 分にな

りました。

think_time_scale を 50 に設定することによって半分の間隔で SQL が発行されることにより、リプレイ時間は短く、CPU 使用率は増加することが確認できます。

- **network 転送量、ディスク Busy 率**

図 6-20 は network 転送量を、図 6-21 はディスクの Busy 率を示しています。

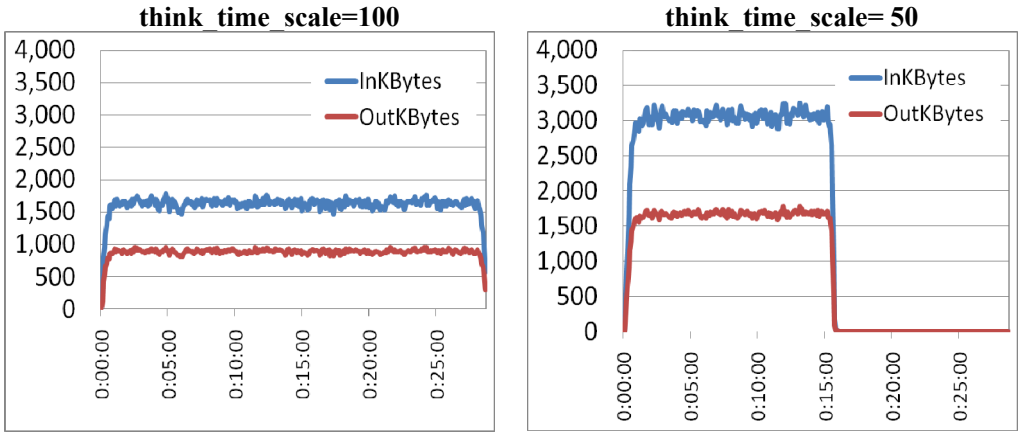


図 6-20 パラメータ値によるネットワーク転送量の比較

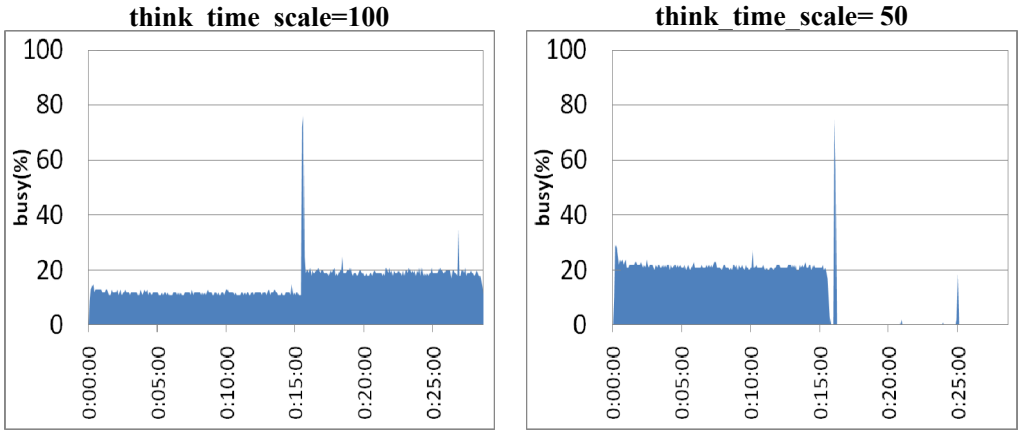


図 6-21 パラメータ値によるディスク Busy 率の比較

図 6-20、図 6-21 も CPU 使用率と同様に、半分の間隔で SQL が発行されことにより増加することが確認できます。

この結果から、OLTP モデルのようなユーザー思考時間を含むトランザクションの場合には、think_time_scale パラメータを設定することで SQL の発行間隔を変更し、システムに与える負荷を制御することができます。

6.2.6. ワークロードのリプレイ精度

検証内容

ワークロードを取得したシステムと同等なシステムでワークロードをリプレイした場合、データやエラーの違いが発生しないことが望まれますが、わずかな違いは問題ありません。許容される違いの量は主観的なものですが、一般的に違いが少なければ、リプレイ精度が高いと言えます。リプレイ精度を確認するためにリプレイ後に重要な表をチェックすること、ワークロードが完了した後に重要なアプリケーション・データが反映されているのをチェックすることを推奨します。例えば、記入済の注文書データが入った表をチェック対象とします。

以降では同一のハードウェアで取得、リプレイを行った際の synchronization パラメータの影響について検証します。

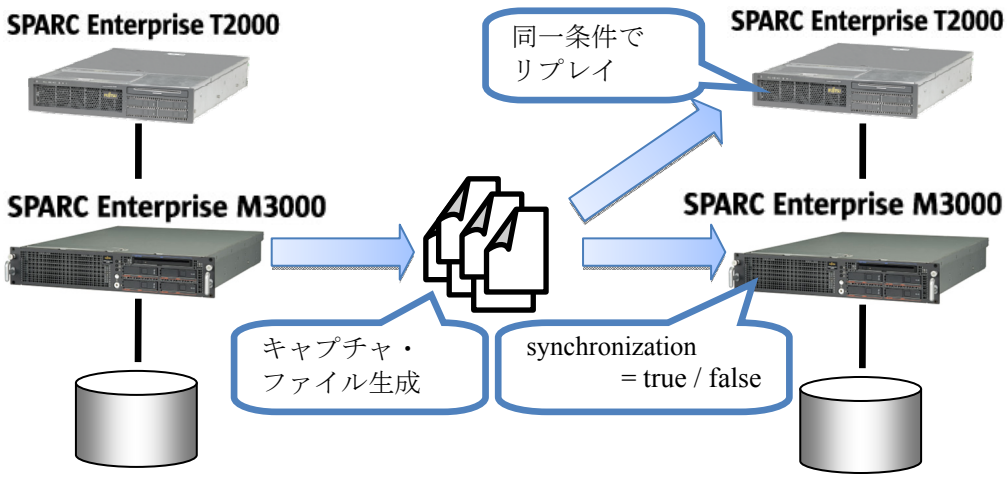


図 6-22 検証概略図

検証結果

- **エラー／データの相違**

synchronization パラメータは、取得したワークロードの COMMIT の順序を保持するかどうかを制御（図 6-23）します。true の場合、リプレイされるトランザクションは、依存するトランザクションがコミットされた後に実行されます。

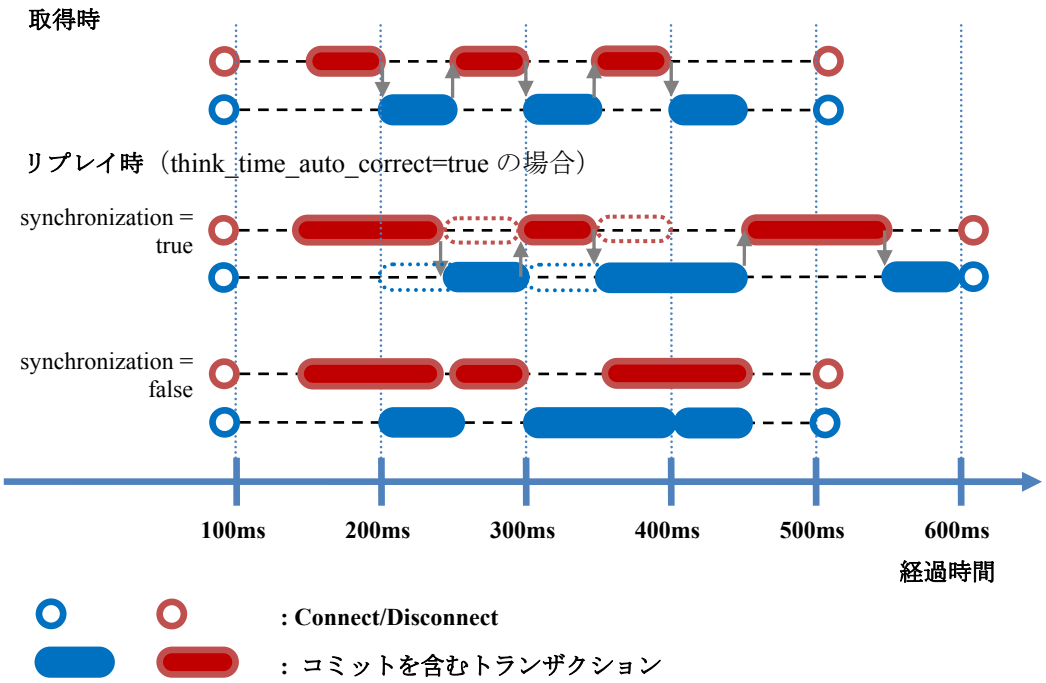


図 6-23 synchronization パラメータの効果

以降では、COMMIT 順を保持した場合としない場合の結果を比較します。

図 6-24にコミット順序を保持した場合としない場合での測定結果を示します。

synchronization=true

Errors Mutated During Replay	0	0.00
DMLs with Different Number of Rows Modified	0	0.00
SELECTs with Different Number of Rows Fetched	0	0.00

Synchronization=false

Errors Mutated During Replay	0	0.00
DMLs with Different Number of Rows Modified	1	0.00
SELECTs with Different Number of Rows Fetched	27	0.00

図 6-24 レポート出力結果（データの相違部分）

コミット順序を保持した場合、ワークロードの取得時と同様、リプレイ時に発生したデータの相違は 0 件になりました。
それに比べ、コミット順序を保持しない場合は、データの相違が 28 件発生しました。

この結果から、コミット順序を保持しないことで、トランザクションの順序を無視して実行するため、DML および SELECT 文についてデータの相異が発生することがわかりました。

- **リプレイ精度改善パッチの効果**

OLTPモデルのトランザクションをCOMMIT順を保持（synchronization=true）してリプレイした時の結果が図 6-25です。リプレイ時の経過時間が取得時の経過時間より遅延する事象が見られましたが、このリプレイ精度の問題を解決するためにオラクル社からOracle Database 11g（11.1.0.7）に対する個別パッチ(One-Off Patch#7240198)が提供されました。
図 6-25、図 6-26、図 6-27にこの個別パッチの適用前と適用後の結果を比較します。

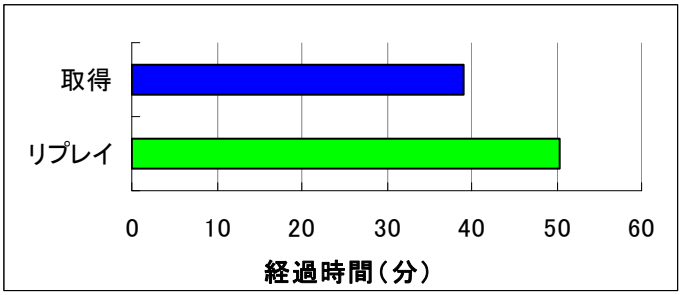


図 6-25 パッチ適用前の取得とリプレイの時間

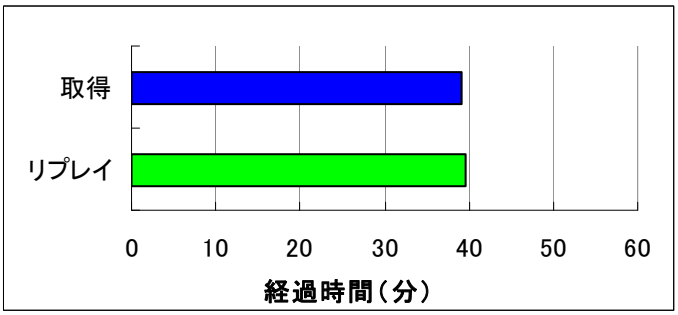


図 6-26 パッチ適用後の取得とリプレイの時間

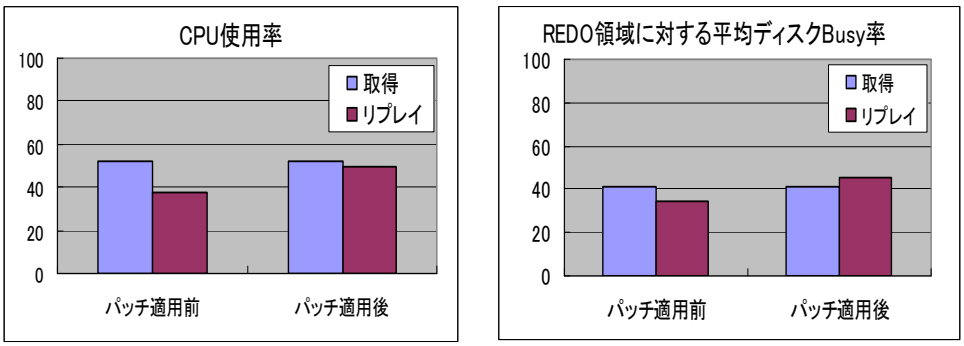


図 6-27 パッチ適用前後の CPU 使用率とディスク Busy 率

リプレイ処理時間を比較（図 6-25、図 6-26）すると、パッチを適用することによってリプレイ処理時間が取得時間とほぼ同等になり、この時のCPU使用率とディスクBusy率も、取得時とほぼ同等になっていることがわかります。

以上の結果から、この個別パッチを適用することの必要性が証明されました。

これにより、リプレイ時の経過時間の精度が改善され、取得とリプレイ処理時間がほぼ同等になります。もし、Database Replay を新／旧システム構成間のパフォーマンス比較に使用することを期待しているのであれば、この個別パッチは非常に重要です。

6.2.7. まとめ

データベース・リプレイ検証結果について、以下にまとめます。

- 取得時のオーバヘッドは小さく、本番環境のパフォーマンスへの影響はほとんど無視できるレベルです。
- OLTP モデルのトランザクションの場合、think_time_scale を設定することによって、ユーザー・コール間の経過時間のスケールを変更し、異なる負荷でリプレイすることができます。
- Synchronization パラメータを true に設定することによって、取得時との「エラーの相異」、「データの相異」を確認することができます。これにより、移行先のシステム評価に Database Replay を使用することで SQL レベルのアプリケーションの動作、非互換確認ができます。
- Synchronization パラメータを false にすると、COMMIT 順が保持されないため「データの相違」が発生する可能性があり、アプリケーションの負荷測定を目的とした場合のみ false を設定します。
- 個別パッチを適用することにより synchronization を true にした場合のリプレイ精度が改善され、取得とリプレイ処理時間がほぼ同等になります。

6.3. SQL Performance Analyzer

SQL Performance Analyzer（以降 SPA）検証では、Oracle9i Database で SQL ワークロードを取得し、Oracle Database 11g で SPA を実施する手順や、SQL Tuning Advisor（以降 STA）による SQL のチューニング、SQL 計画の管理機能による実行計画の固定化について検証しました。

6.3.1. 機能詳細

SPA は、環境の変更前と変更後で SQL を実行することで、環境の変更が SQL の実行計画と統計情報に及ぼす影響を、いくつかの測定基準に沿ってより詳細に分析／レポートすることができます。

SPA を使用するには、次の 5 つの手順を実行する必要があります。

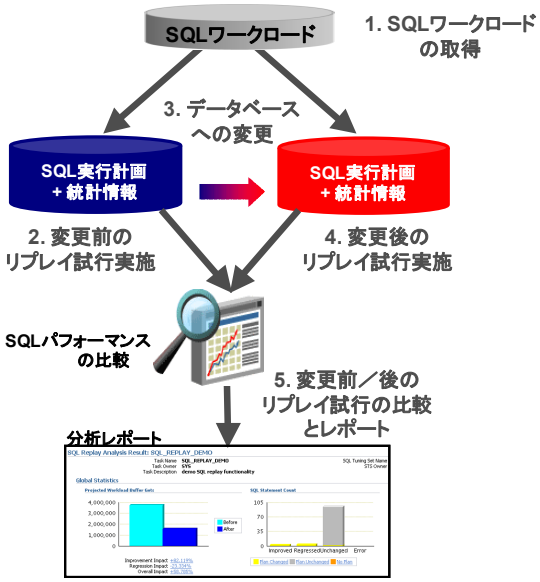


図 6-28 SQL Performance Analyzer 主要処理手順

1. **SQL ワークロードの取得**

SPA で分析する SQL ワークロードを取得します。Oracle Database には、カーソル・キャッシュや Automatic Workload Repository (AWR)などの複数のソースから SQL ワークロードを取得して、SQL Tuning Set (STS)に格納する方法が提供されます。この処理は、通常本番システム上で実行されます。その後、STS をテスト・システムに転送し、そこで SPA 分析を実行します。

2. **変更前のリプレイ試行実施**

STS に対して SPA を実行することで、変更前のワークロードのパフォーマンスを計測します。

3. データベースへの変更

データベースのアップグレードやデータベース・オプティマイザ統計のリフレッシュなどの変更を行います。

4. 変更後のリプレイ試行実施

再度、STS に対して SPA を実行することで、変更後のワークロードのパフォーマンスを計測します。

5. 変更前／後のリプレイ試行の比較とレポート

STS の 2 回の実行結果を比較して、効率が向上した SQL、効率が低下した SQL、変化のなかった SQL を特定します。

SPAによってSQLのパフォーマンス劣化が検出されると、ユーザーは、SQL Tuning Advisor (STA)[4]またはSQL 計画ベースライン（Oracle Database 11gで新規に導入されたプラン・スタビリティ機能）を使用して、そのSQLを修正することができます。

SPA では、いくつかのメトリックを用いることで多角的に、データベースへの変更実施前／実施後の SQL パフォーマンスを比較／分析することができます。
使用することのできるメトリックは下記のとおりです。

表 6-9 SQL Performance Analyzer 比較メトリック

メトリック名	メトリックの意味	再現性	包括性
解析時間	SQL 解析で使用される CPU 時間	有	無
経過時間	経過時間	無	有
CPU 時間	SQL 実行で使用される CPU 時間	有	有
バッファ読取り	論理 I/O	有	有
ディスク読取り	物理 I/O 読取り	無	有
ダイレクト書込み	ダイレクト・パス書込み	無	有
オプティマイザ・コスト	コスト見積もり	有	無

比較メトリックの中には、環境（バッファ・キャッシュの状態など）に大きく依存するため、繰り返しテストを行った際に必ずしも同じ結果が繰り返されると

[4] SQL Tuning Adviser を使用するためには、Tuning Pack ライセンスが必要です。Tuning Pack ライセンスは、Diagnostic Pack ライセンスが前提条件となっています。

は限らないものがあります。このようなメトリックを上図では、再現性「無」と記します。また、SQL パフォーマンスの一つの特質を完全に表す比較メトリックについては、包括性「有」と記します。

再現性「有」で SQL のパフォーマンスを取得できる統計は「CPU 時間」です。ただし、これは I/O を考慮したメトリックでなく、Oracle9i での SQL 実行の CPU 時間が SQL トレース取得の影響を受けるので、「CPU 時間」と「バッファ読取り」両方を使用して分析を実施することを強く推奨します。

6.3.2. 前提条件

SPA を実行する上で、以下の点を考慮する必要があります。

- **データの一致**

  テスト環境ではアプリケーション・データの状態を本番環境とできるだけ論理的に同一にしておく方が良いです。

- **パッチの適用**

  Oracle9i Database で SQL ワークロードを取得し、Oracle Database 11g で SPA を実施するためには、Oracle Database 11g が 11.1.0.6 + One-Off Patch #6865809、もしくは 11.1.0.7 以降である必要があります。Oracle9i Database にパッチは必要ありません。

  なお、パッチ適用に関する詳細は MetaLink の Note 560977.1 を参照して下さい。

6.3.3. SPA実行環境の準備方法

本項では、本番環境（Oracle9i Database）で SQL ワークロードを取得し、テスト環境（Oracle Database 11g）で STS を作成するまでの準備方法を示します。

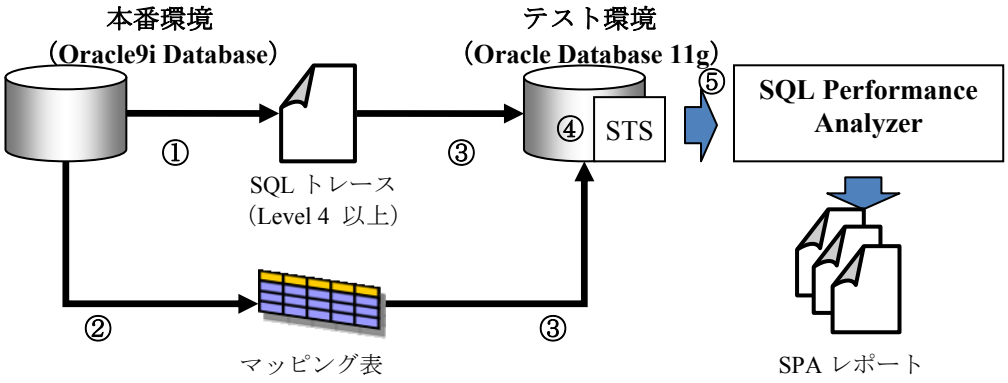


図 6-29 SQL Performance Analyzer の概略図

① 本番環境（Oracle9i Database）の SQL ワークロードを取得するために、SQL トレースを有効にし、SQL トレース・ファイルを取得します。
SQL トレースの有効化は、以下の SQL で設定します。

```
SQL> alter session set events '10046 trace name context forever, level 4';
```

SQL トレースは、以下の SQL で設定を解除します。

```
SQL> alter session set events '10046 trace name context off';
```

② SQL トレース・ファイルでは、オブジェクトやユーザーは、オブジェクト ID で識別されているため、テスト環境（Oracle Database 11g）で SQL トレースを STS へ変換するためには、オブジェクト ID とオブジェクト名のマッピング情報が必要となります。そのため、本番環境（Oracle9i Database）でマッピング表を作成します。マッピング表は以下の SQL で作成します。

```
SQL> connect system/password
SQL> create table <マッピング表名> as
2>     select object_id id, owner, substr(object_name, 1, 30) name
3>     from dba_objects where object_type not in
4>       ('CONSUMER GROUP','EVALUATION CONTEXT',
5>        'JAVA DATA','JAVA RESOURCE','LIBRARY',
6>        'LOB','OPERATOR','PACKAGE','PACKAGE BODY',
7>        'PROCEDURE','QUEUE','RESOURCE PLAN',
8>        'TRIGGER','TYPE','TYPE BODY')
9>     union all
10>    select user_id id, username owner, null name from dba_users;
```

本番環境（Oracle9i Database）でマッピング表を作成後、エクスポートします。

③ SQL トレース・ファイルを、テスト環境（Oracle Database 11g）の任意のディレクトリにコピーし、そのディレクトリに対して、ディレクトリ・オブジェクトを作成します。

```
SQL> create or replace directory <ディレクトリ・オブジェクト名> as
2>   '< SQL Trace file を格納したディレクトリのパス名>';
```

上記②で作成したマッピング表をテスト環境（Oracle Database 11g）へインポートします。※今回は、SYSTEM スキーマへインポートしています。

④ 本番環境（Oracle9i Database）で取得した SQL トレース・ファイルから STS を作成します。

```
SQL> connect / as sysdba
SQL> declare
2>     mycur dbms_sqltune.sqlset_cursor;
3>   begin
4>     dbms_sqltune.create_sqlset ('<STS 名>');
5>     open mycur for
6>       select value (P)
7>       from table (dbms_sqltune.select_sql_trace (
8>         directory => '<ディレクトリオブジェクト名>',
9>         file_name => '%ora%',
10>        mapping_table_name => '<マッピング表名>',
11>        mapping_table_owner => 'SYSTEM',
12>        select_mode => dbms_sqltune.SINGLE_EXECUTION)) P;
13>    dbms_sqltune.load_sqlset ('<STS 名>', mycur);
14>    close mycur;
15>  end;
16>  /
```

⑤ テスト環境（Oracle Database 11g）上で作成した STS を使用し、テスト環境上で SPA を実行することにより、本番環境（Oracle9i Database）とテスト環境の SQL ワークロードについて分析することができます。

6.3.4. SPAを使った分析作業

検証内容

SPA検証を実施するにあたり、本検証では、パターンを 4 つ準備しました。各パターンごとの概要を表 6-10に示します。

表 6-10 検証パターン概要

パターン	概要
1	LIST_1 表と MASTER_1 表を結合し、データを取得。
2	LIST_1_2 表、MASTER_1_2_1 表、MASTER_1_2_2 表、MASTER_1_2_3 表を結合した副問合せ結果のデータを取得。
3	LIST_4 表、MASTER_4_1 表、MASTER_4_2 表、MASTER_4_3 表結合し、行数を取得。
4	SALES 表について、WHERE 句の条件に当てはまるすべてのデータを取得。 ※ただし、SALES 表に対しての統計情報は 124 レコード時に取得されたもの（実際は 1837810 レコード）であり、古い状態。

これら全 4 パターンについて、Oracle9i Database のルールベースオプティマイザから、Oracle Database 11g のコストベースオプティマイザへ移行した場合の実行計画の変化、パフォーマンス比較を SPA を使用して分析しました。

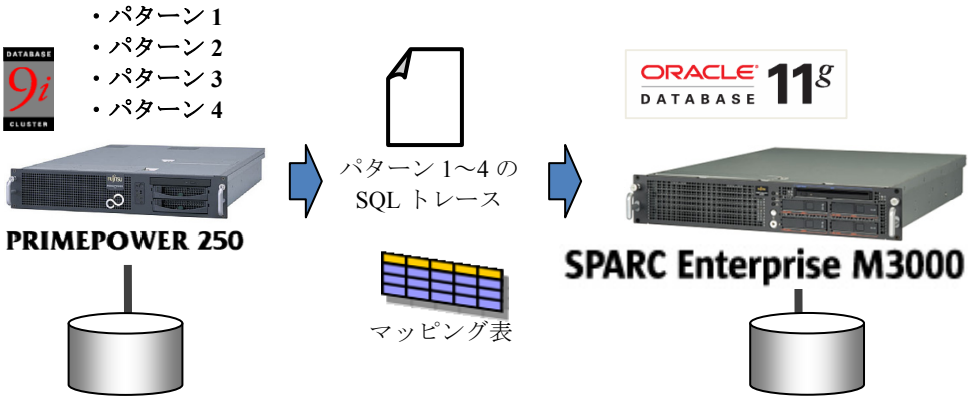


図 6-30 検証概略図

パターンごとに用意した表とSQLは図 6-31～図 6-34の通りです。

パターン 1

表名：LIST_1 データ件数：10000000件
列名
M_UID
L_UID
LAST_DATE
COMMENT1
COMMENT2

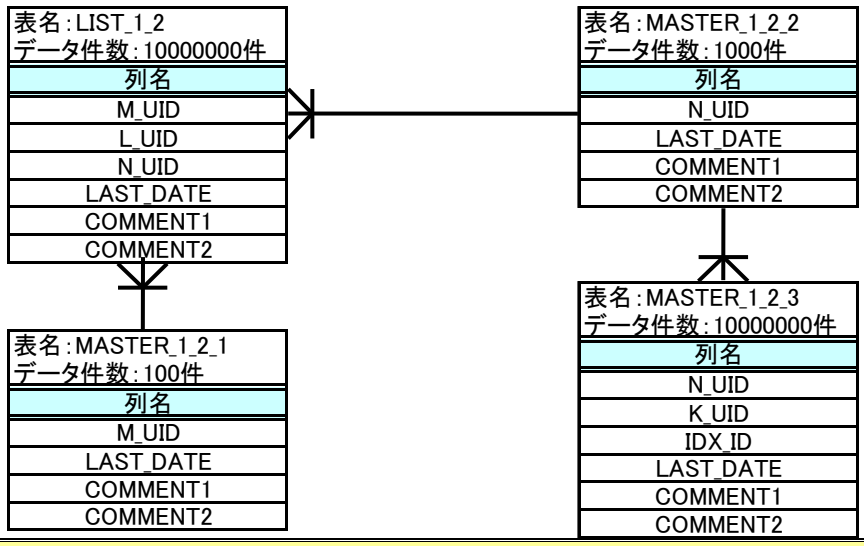
表名：MASTER_1 データ件数：100件
列名
M_UID
LAST_DATE
COMMENT1
COMMENT2

SQL

```
SELECT /* PROBLEM_SQL 1_1 */ T1.COMMENT1, T2.COMMENT1, COUNT(*)
FROM LIST_1 T1, MASTER_1 T2
WHERE T1.M_UID = T2.M_UIDAND T1.LAST_DATE > SYSDATE -31
GROUP BY T1.COMMENT1, T2.COMMENT1;
```

図 6-31 パターン 1

パターン 2

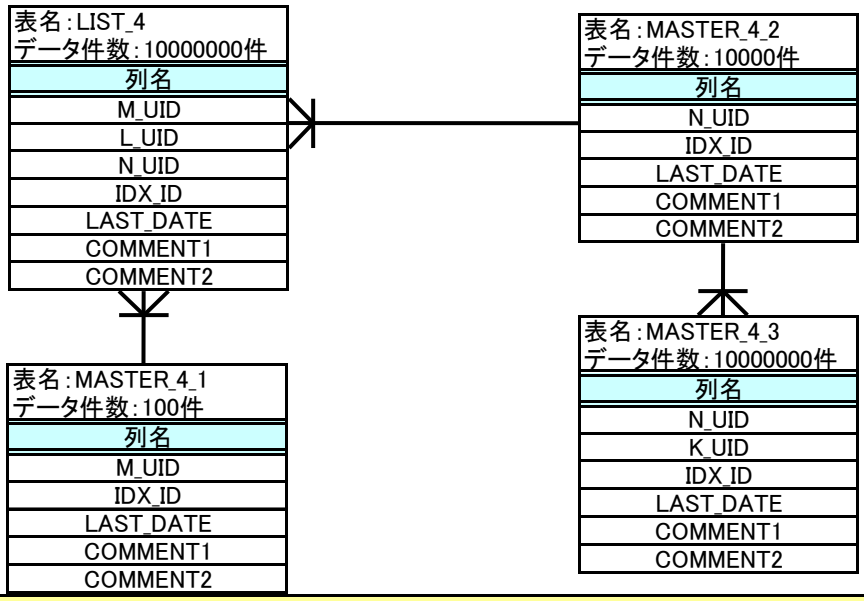


SQL

```
SELECT /* PROBLEM_SQL 1_2 */ * FROM (
  SELECT T2.L_UID, T2.M_UID, T2.N_UID, T4.K_UID,
      T4.IDX_ID, T1.LAST_DATE, T2.COMMENT1, T3.COMMENT2
  FROM MASTER_1_2_1 T1, LIST_1_2 T2, MASTER_1_2_2 T3,
      MASTER_1_2_3 T4
  WHERE T1.M_UID = T2.M_UID AND T2.N_UID = T3.N_UID
      AND T3.N_UID = T4.N_UID AND T4.IDX_ID = 1
  ORDER BY T2.M_UID, T2.N_UID, T2.L_UID)
WHERE ROWNUM < 11;
```

図 6-32 パターン 2

パターン 3

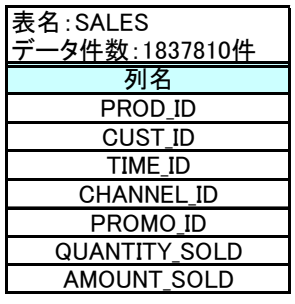


SQL

```
WITH
  SAMPLE4 AS (
    SELECT /* PROBLEM_SQL 4 */ T4.L_UID, T4.M_UID, T4.N_UID,
      T1.K_UID, T1.IDX_ID, T4.LAST_DATE, T3.COMMENT1, T2.COMMENT2
    FROM MASTER_4_3 T1, MASTER_4_2 T2, MASTER_4_1 T3, LIST_4 T4
    WHERE T3.M_UID = T4.M_UID AND T2.N_UID = T4.N_UID
    AND T1.N_UID = T2.N_UID AND T4.IDX_ID BETWEEN 101 AND 200
    ORDER BY T4.M_UID, T4.N_UID, T4.L_UID
) SELECT COUNT(*) FROM SAMPLE4;
```

図 6-33 パターン 3

パターン 4



SQL

```
SELECT * FROM SCOTT.SALES WHERE TIME_ID < '1998-01-10';
```

図 6-34 パターン 4

検証結果（パターン 1）

パターン 1 での比較レポート（図 6-35）を確認すると、バッファ読取りは改善していますが、経過時間、CPU時間が劣化していることがわかります。次に実行計画を比較するとOracle9i DatabaseではNESTED LOOPが選択されているのに対し、Oracle Database 11gでは、HASH JOINが選択されおり実行計画が変化しています。

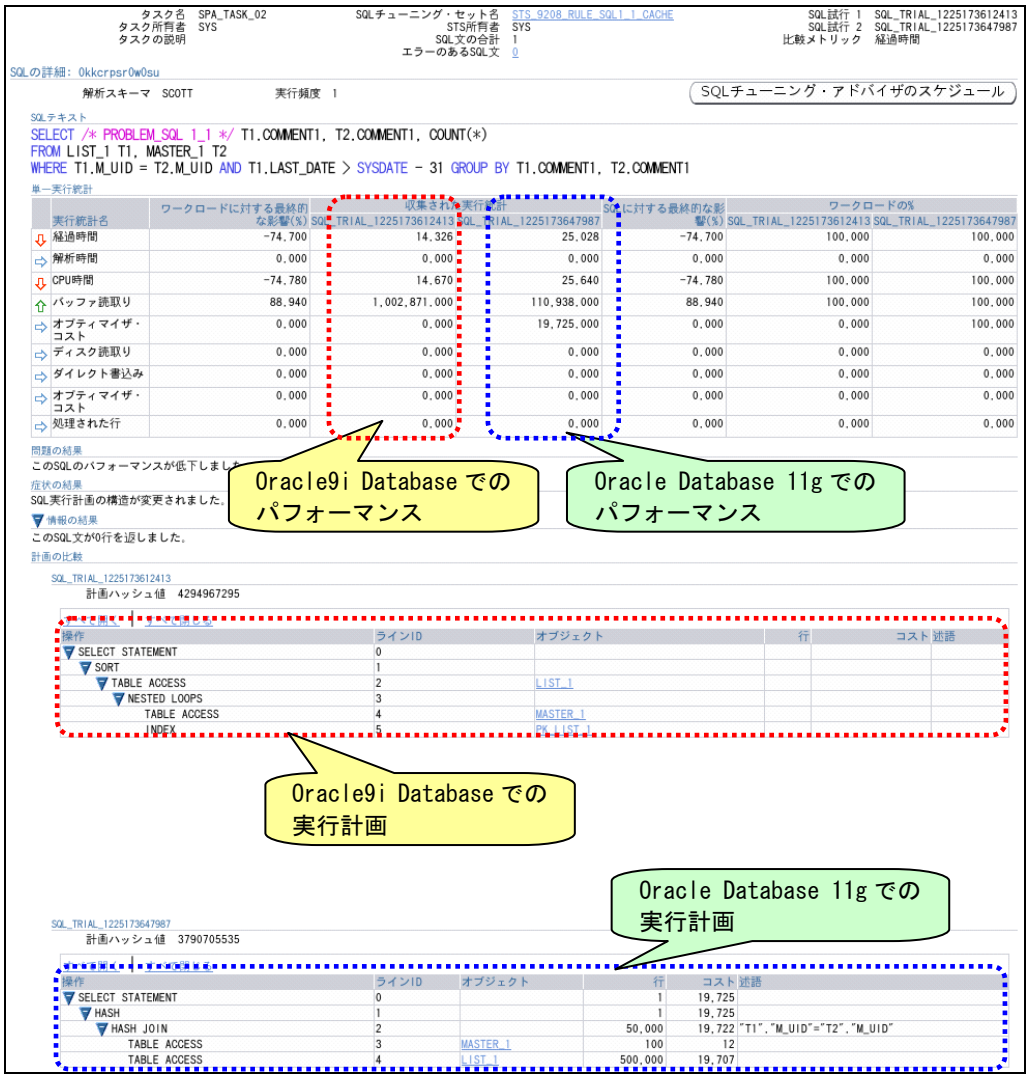


図 6-35 パターン 1 の比較レポート

表 6-11 パターン 1 の経過時間

Oracle9i Database	Oracle Database 11g
0:00:14.32	0:00:25.02

検証結果（パターン 2）

パターン 2 での比較レポート（図 6-36）を確認すると、経過時間、CPU時間、バッファ読取りが改善していることがわかります。次に実行計画を比較するとパターン1と同様にOracle9i DatabaseではNESTED LOOPが選択されているのに対し、Oracle Database 11gでは、HASH JOINが選択されおり実行計画が変化しています。

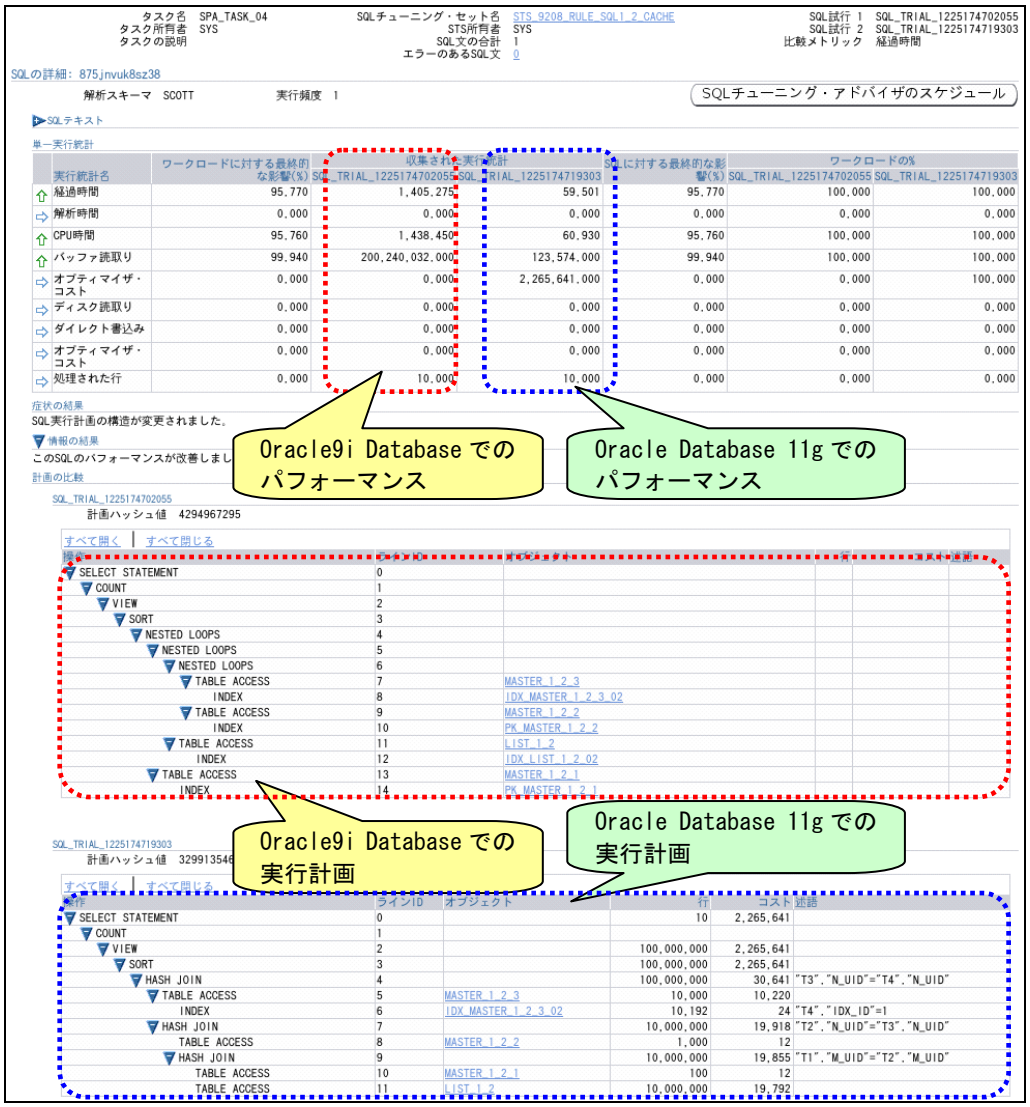


図 6-36 パターン 2 の比較レポート

表 6-12 パターン 2 の経過時間

Oracle9i Database	Oracle Database 11g
0:23:25.28	0:00:59.50

検証結果（パターン 3）

パターン 3 での比較レポート（図 6-37）を確認すると、経過時間、CPU時間、バッファ読取り、ディスク読取りが改善していることがわかります。次に実行計画を比較するとOracle9i Databaseでは、NESTED LOOPが選択されているのに対し、Oracle Database 11gでは、HASH JOIN、MERGE JOINが選択されおり実行計画が変化しています。

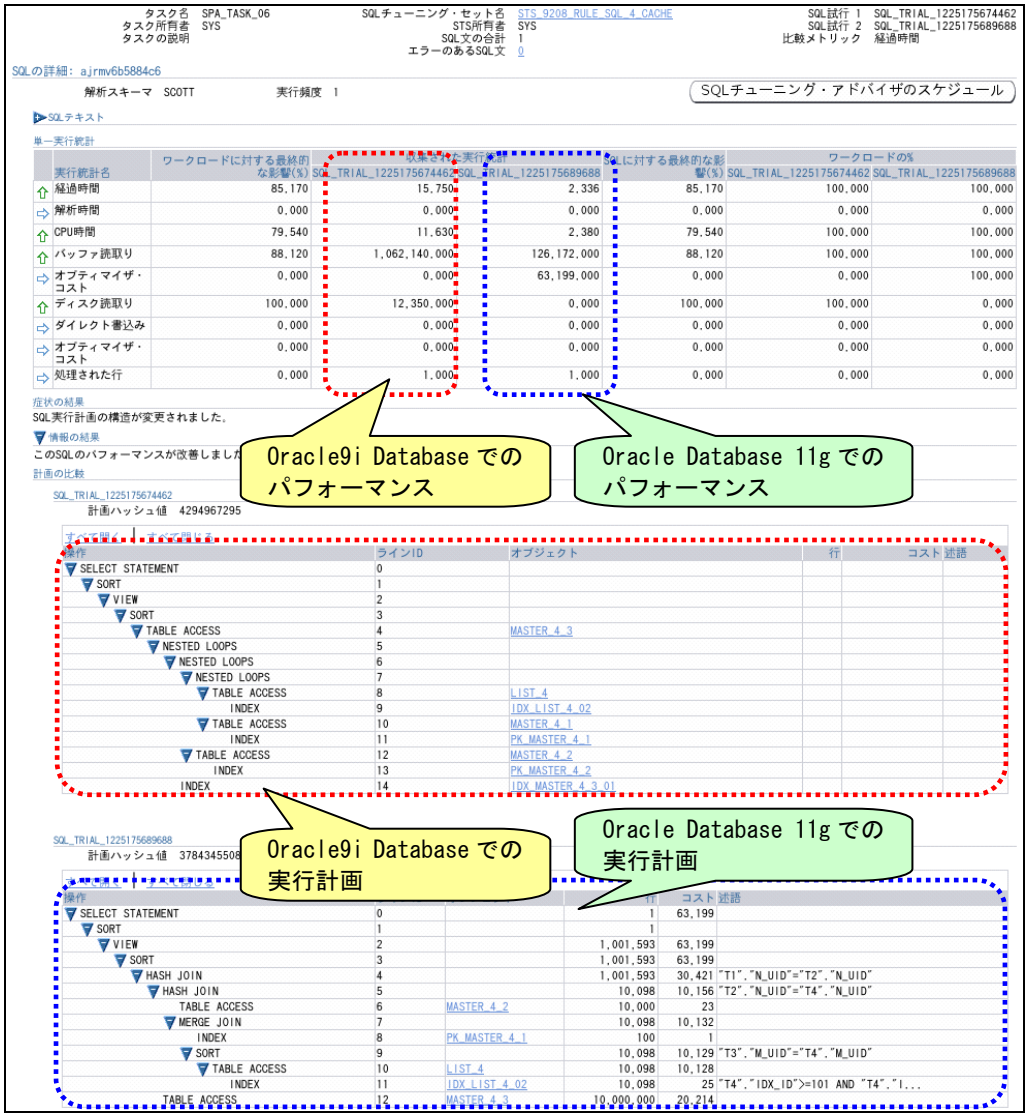


図 6-37 パターン 3 の比較レポート

表 6-13 パターン 3 の経過時間

Oracle9i Database	Oracle Database 11g
0:00:15.75	0:00:02.34

検証結果（パターン 4）

パターン 4 での比較レポート（図 6-38）を確認すると、経過時間、CPU時間、バッファ読取り、ディスク読取りがが劣化していることがわかります。次に実行計画を比較するとOracle9i Databaseでは、インデックス・スキャンが選択されているのに対し、Oracle Database 11gでは、フル・スキャンが選択されおり実行計画が変化しています。

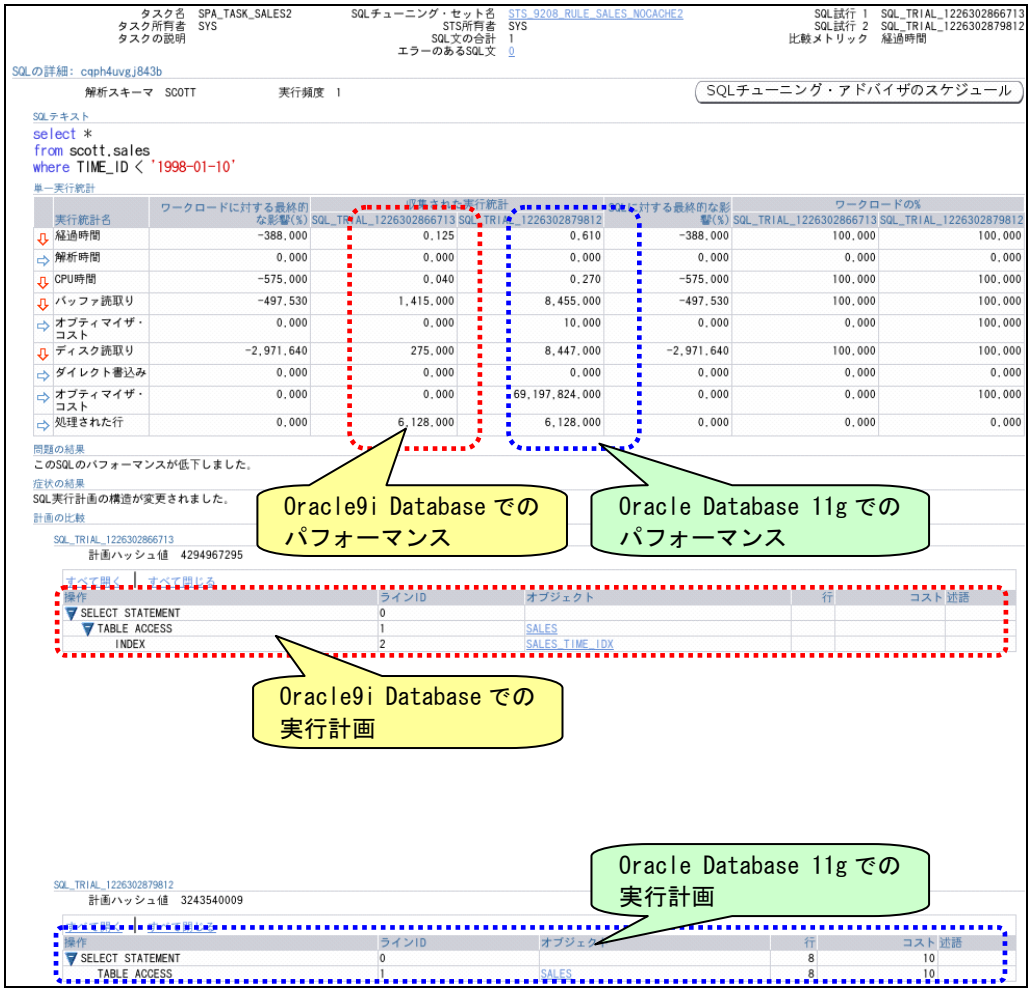


図 6-38 パターン 4 の比較レポート

表 6-14 パターン 4 の経過時間

Oracle9i Database	Oracle Database 11g
0:00:00.13	0:00:00.61

以上より、パターン 2 とパターン 3 については、Oracle9i Database から Oracle Database 11g へアップグレードしたことにより、パフォーマンスが向上しています。特にパターン 2 では、最適な実行計画が選択された結果、移行前と比較すると約 23 倍も性能が向上しています。しかし、パターン 1 とパターン 4 については、ハードウェア性能が向上しているにも関わらず、移行することによりパフォーマンスが劣化しています。

次項では、性能の劣化した SQL に対する対処方法の紹介および、検証結果を示します。

6.3.5. SQLのチューニング

SPA の結果レポートからの STA の実行

SPA 実施後、パフォーマンス劣化が見られる場合は、レポート画面に「SQL チューニング・アドバイザの実行」ボタンが表示されます。このボタンから STA によるパフォーマンスチューニングが可能です。

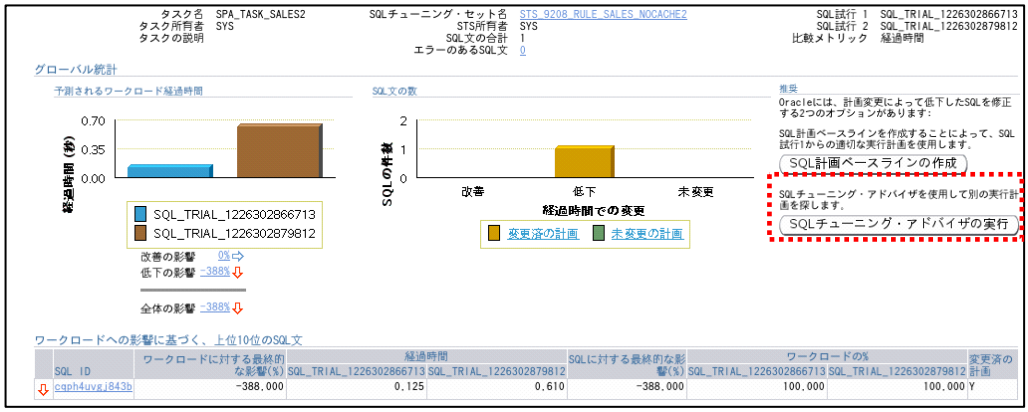


図 6-39 STA の実行

以下では、SPA の結果、パフォーマンスの劣化が見られたパターン 4 について STA によるチューニングを実施した結果を示します。

検証結果

STA を実行した結果、SALES テーブルと索引に対するオプティマイザの統計情報が古いということと、より適している可能性のある実行計画が見つかりました。より適したものを選んで実装します。

この中から、表に対する統計を取得する項目にチェックを入れ実装しました。

図 6-40 STA のアドバイス

実装によって統計が再収集されます。この状態で、再度 SPA の該当タスクにて試行を行い、比較することが可能です。

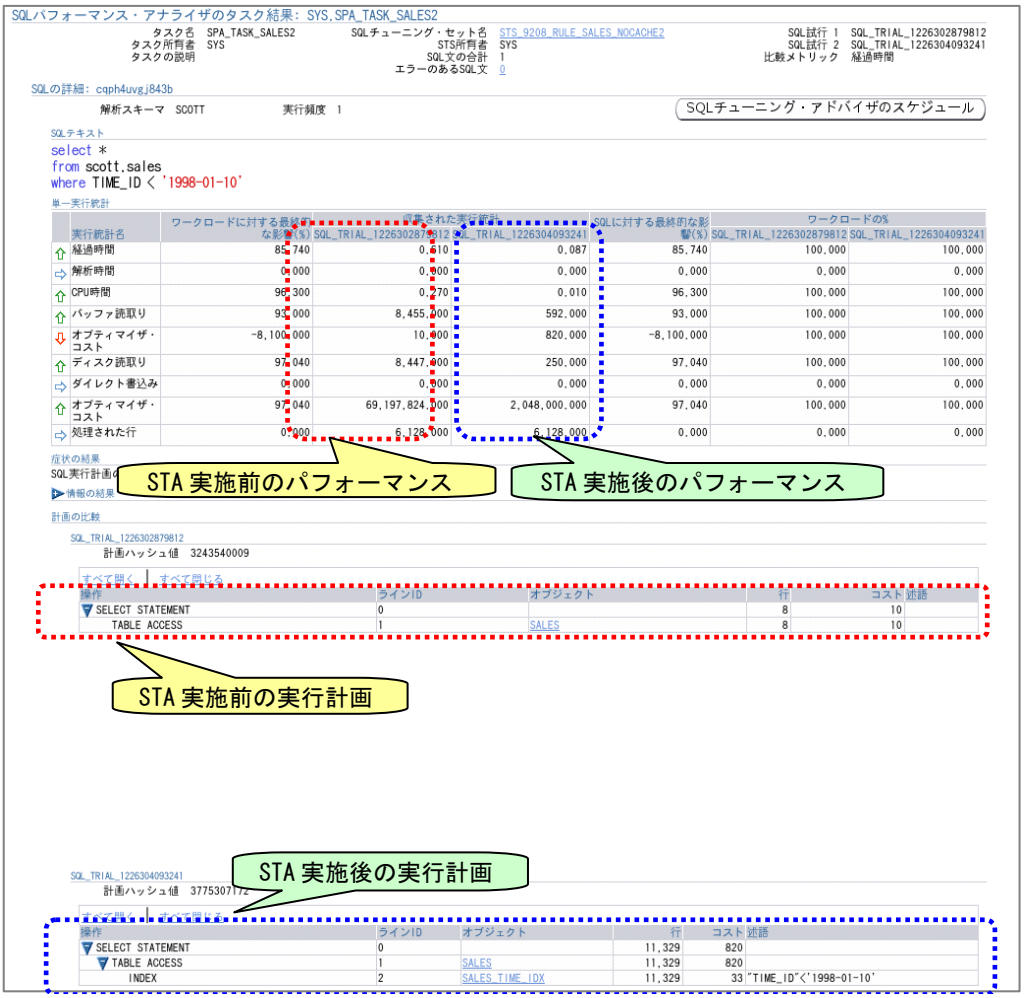


図 6-41 STA 実施前後の SPA 比較レポート

パターン 4 の SQL を STA 実施前と実施後で比較した結果、実施前ではフル・スキャンが選択されていたのに対し、実施後はインデックス・スキャンを選択しています。これにより、経過時間や、CPU 時間などパフォーマンスが改善されたことがわかります。

6.3.6. 実行計画の固定化

実行計画の固定化が必要なケース

現行システムで十分なパフォーマンスがあり、新システムへのアップグレード時に行計画が変化によるパフォーマンス劣化のリスクを低減させたい場合、ヒント句やプラン・スタビリティを使用してアップグレード前に対象とするストアド・アウトラインを取得し、新システムへ移行することで実行計画の変化を防ぐことができました。

しかし、ヒント句やプラン・スタビリティによって実行計画を固定してしまうと他に最適な実行計画がある場合や新しい索引が有効になった場合などでも常に古い実行計画を使用し続けてしまいます。

Oracle Database 11g では、予期しない実行計画の変化によるパフォーマンス劣化を防止するだけではなく、実行計画の変化を管理するために SQL 計画の管理が導入されました。アウトラインを SQL 計画ベースラインとすることで現行システムの実行計画の維持、新システム上でのパフォーマンス比較が行えます。

以降では、実行計画の変化によりパフォーマンスが劣化したパターン 1 について、アウトラインによる実行計画の移行と SQL 計画の管理の検証結果を示します。

検証手順および結果

①～③を現行システム（Oracle9i Database）で行います。

① プラン・スタビリティのアウトラインを作成するスキーマへ「CREATE ANY OUTLINE」システム権限を付与します。

```
SQL> connect / as sysdba
SQL> GRANT CREATE ANY OUTLINE TO SCOTT;
```

② 特定のセッション[※5]で実行された、すべてのSQLをアウトラインに格納します。ここでは、パターン 1 のSQLを格納します。

```
SQL> ALTER SESSION SET create_stored_outlines=OUTLN_920;
SQL> SELECT /* PROBLEM_SQL 1_1 */ T1.COMMENT1, T2.COMMENT1,
2>       COUNT(*)FROM LIST_1 T1, MASTER_1 T2
3>       WHERE T1.M_UID = T2.M_UID
4>       AND T1.LAST_DATE > SYSDATE - 31
5>       GROUP BY T1.COMMENT1, T2.COMMENT1;
SQL> ALTER SESSION SET create_stored_outlines=false;
```

[5] アウトラインは、システム全体、または特定の SQL のみを格納することもできます。

格納したアウトラインは以下の SQL で確認できます。

```
SQL> SELECT category, version, sql_text FROM USER_OUTLINES;
```

③ エクスポート･ユーティリティによりアウトライン（OL$表、OL$HINTS 表、OL$NODES 表）をエクスポートします。

```
$ exp outln/outln file=outln_920.dmp tables='OL$' 'OL$HINTS'
'OL$NODES' log=exp.log
```

現行システム（Oracle Database 11g）での作業は以上です。④より新システム（Oracle Database 11g）での作業となります。

④ 新システム（Oracle Database 11g）で、インポート・ユーティリティによりアウトライン（OL$表、OL$HINTS 表、OL$NODES 表）をインポートします。

```
$ imp outln/outln file=outln_920.dmp ignore=Y tables='OL$'
'OL$HINTS' 'OL$NODES' log=imp.log
```

⑤ 以前のリリースで生成されたアウトラインをインポートした場合は、DBMS_OUTLN.UPDATE_SIGNATURES によりアウトラインを更新します。

```
SQL> connect outln/outln
SQL> execure DBMS_OUTLN.UPDATE_SIGNATURES;
```

更新後の USER_OUTLINES 表を確認した結果を以下に示します。

```
SQL> connect scott/tiger
SQL> SELECT category, used, version, sql_text FROM USER_OUTLINES;

CATEGORY    USED    VERSION    SQL_TEXT
--------    ----    -------    --------------------------
OUTLN_920   UNUSED  9.2.0.8.0  SELECT /* PROBLEM_SQL 1_1,
```

⑥ インポートしたアウトラインを使用して、SQL を実行します。
なお、アウトラインを適切に機能させるには、以下のパラメータの設定を移行前、移行後で統一する必要があります。

- QUERY_REWRITE_ENABLED
- STAR_TRANSFORMATION_ENABLED
- OPTIMIZER_FEATURES_ENABLE

```
SQL> ALTER SESSION SET query_rewrite_enabled='TRUE';
SQL> ALTER SESSION SET star_transformation_enabled='FALSE';
SQL> ALTER SESSION SET optimizer_features_enable='9.2.0';
SQL> ALTER SESSION SET use_stored_outlines=OUTLN_920;
SQL> SELECT /* PROBLEM_SQL 1_1 */ T1.COMMENT1, T2.COMMENT1,
2>       COUNT(*)FROM LIST_1 T1, MASTER_1 T2
3>       WHERE T1.M_UID = T2.M_UID
4>       AND T1.LAST_DATE > SYSDATE -31
5>       GROUP BY T1.COMMENT1, T2.COMMENT1;

SQL> ALTER SESSION SET use_stored_outlines=false;
```

アウトラインを使用して SQL を実行すると、USER_OUTLINES の「USED」列が”UNUSED”から”USED”へ変わり、アウトラインが使用されたことが確認できます。

⑦ Enterprise Manager(以降 EM)を使用し、SQL 計画ベースラインのロードで、カーソル・キャッシュのアウトラインを利用した実行計画を SQL 管理ベースへ格納します。

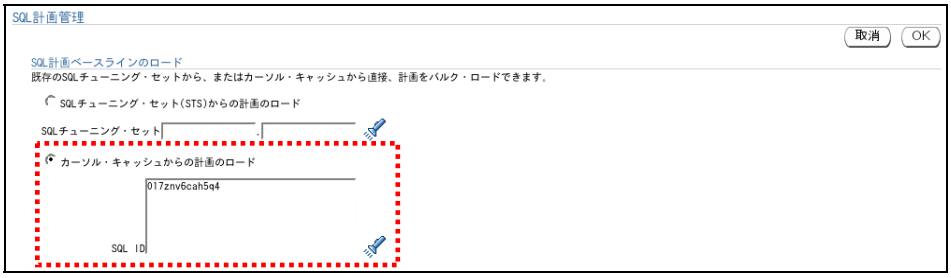


図 6-42 SQL 計画ベースラインのロード

⑧ 作成された SQL 計画ベースラインは自動承認され、「確定済」ステータスが「YES」となり、以降の SQL 実行時に、オプティマイザで生成された新しい実行計画より優先してこの実行計画が選択される状態となります。

図 6-43 SQL 計画の管理

⑨ 格納した実行計画は上記⑧の SQL テキストのリンクより確認することができます。

本検証では以下の通り、「NESTED LOOP」結合を使用しており、現行システム（Oracle9i Database）と同じ実行計画であることがわかります。

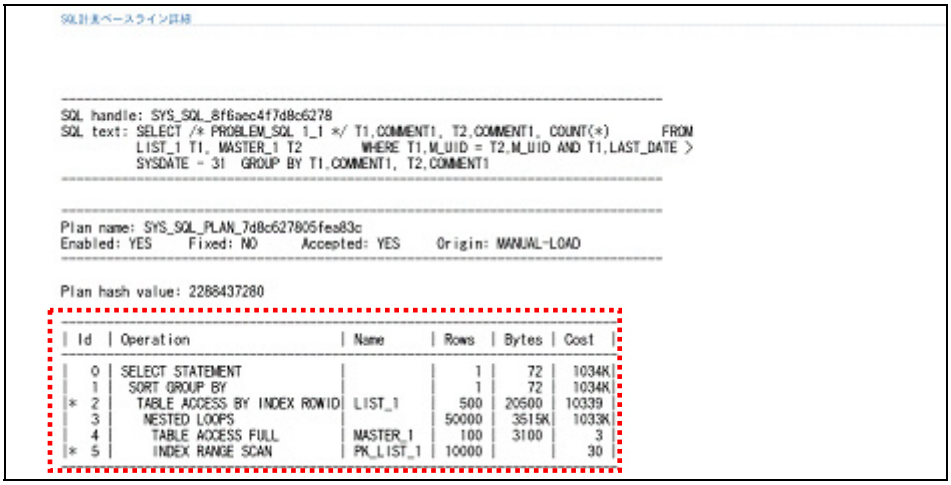


図 6-44 SQL 計画ベースラインの詳細

⑩ SQL を実行した際に SQL 計画ベースラインの実行計画とは異なる実行計画がオプティマイザにより生成された場合、新しく作成された実行計画が SQL 計画ベースラインとして SQL 管理ベースに格納されます。
なお、SQL 管理ベースへ格納された新しい SQL 計画ベースラインはまだ承認されていないため、「確定済」ステータスは「NO」の状態となっています。

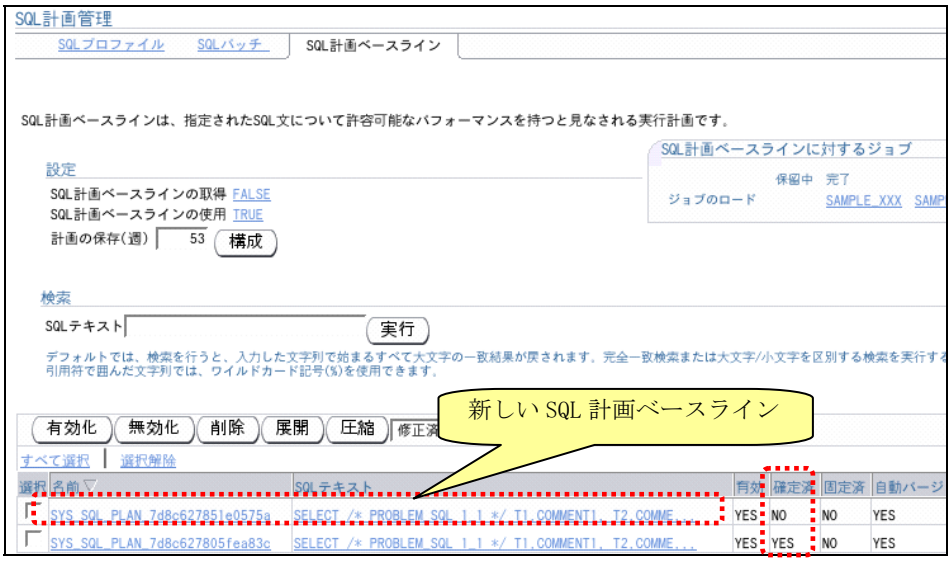


図 6-45 新しい SQL 計画ベースライン

⑪ この SQL 計画ベースラインの実行計画を比較すると、新しく格納された SQL 計画ベースラインでは、HASH JOIN 結合が選択されています。

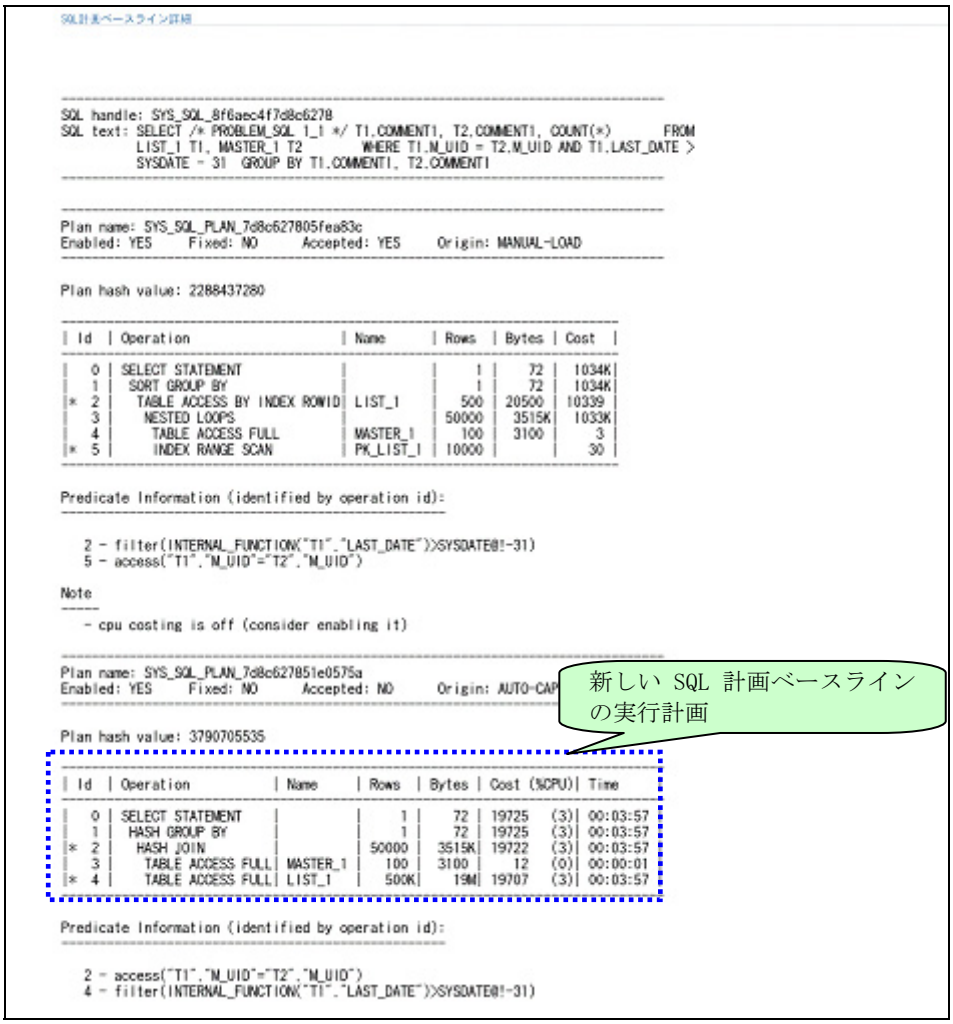


図 6-46 SQL 計画ベースラインの詳細（新しい実行計画）

⑫ 上記⑩の 2 つの SQL 計画ベースラインについて、現在有効である SQL 計画ベースラインから新しい SQL 計画ベースラインに変更した時の改善率を検証します。

検証は SQL 計画管理画面の「展開」ボタンより実施します。

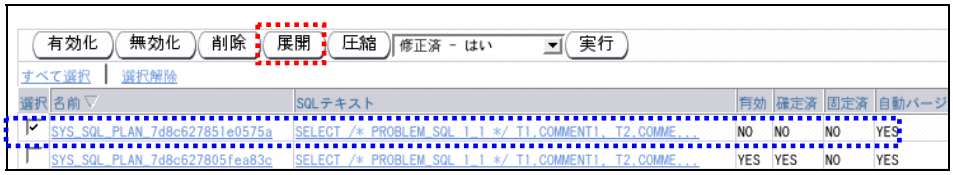
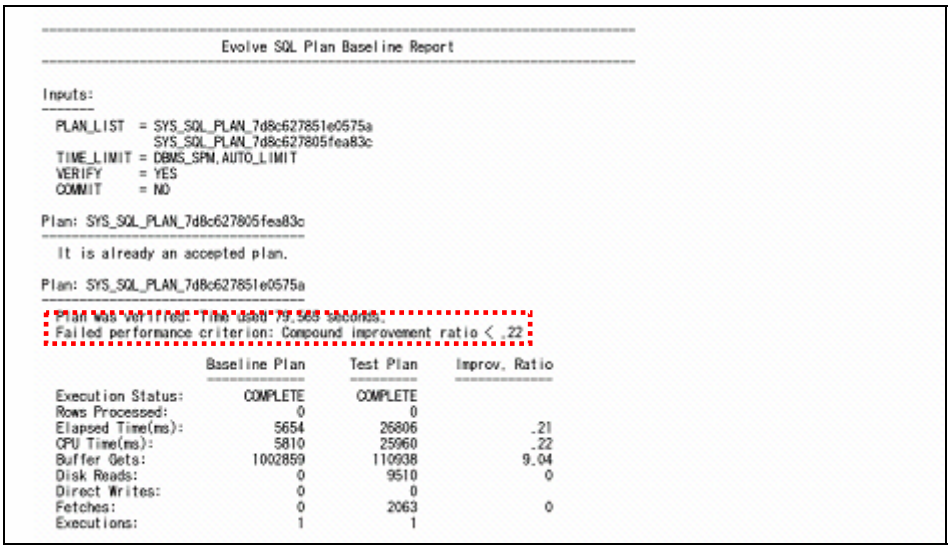


図 6-47 SQL 計画ベースラインの展開

⑬ 検証の結果、新しい SQL 計画ベースラインは、現在有効である SQL 計画ベースラインに比べ、改善率は 0.22 未満となっており、「Failed performance criterion」と記載されていることから、承認できない（変更を推奨しない）と判断されました。



```
                    Evolve SQL Plan Baseline Report

Inputs:
-------
  PLAN_LIST  = SYS_SQL_PLAN_7d8c627851e0575a
               SYS_SQL_PLAN_7d8c627805fea83c
  TIME_LIMIT = DBMS_SPM.AUTO_LIMIT
  VERIFY     = YES
  COMMIT     = NO

Plan: SYS_SQL_PLAN_7d8c627805fea83c
-----------------------------------
  It is already an accepted plan.

Plan: SYS_SQL_PLAN_7d8c627851e0575a
-----------------------------------
  Plan was verified: Time used: 77.565 seconds.
  Failed performance criterion: Compound improvement ratio < .22

                      Baseline Plan     Test Plan     Improv. Ratio
                      -------------     ---------     -------------
  Execution Status:        COMPLETE      COMPLETE
  Rows Processed:                 0             0
  Elapsed Time(ms):            5654         26806              .21
  CPU Time(ms):                5810         25960              .22
  Buffer Gets:              1002859        110938             9.04
  Disk Reads:                     0          9510                0
  Direct Writes:                  0             0
  Fetches:                        0          2063                0
  Executions:                     1             1
```

その他の比較方法

SQL 計画の管理で比較する他、該当の SQL を格納した STS を作成し、SPA により SQL 計画ベースラインを有効、無効にした場合の試行を比較することでも、パフォーマンスを比較できます。

図 6-48は、1 回目の試行を新しく作成された実行計画のSQL計画ベースラインのみ有効、2 回目の試行をアウトラインの実行計画から作成されたSQL計画ベースラインのみ有効にした状態でSPAを実施し、経過時間の比較メトリックで比較した結果です。この結果からも、アウトラインの実行計画から作成されたSQL計画ベースラインを使用することが良いことが判断できます。

このように、SPA を使用した比較では、視覚的によりわかりやすい結果を得ることができます。

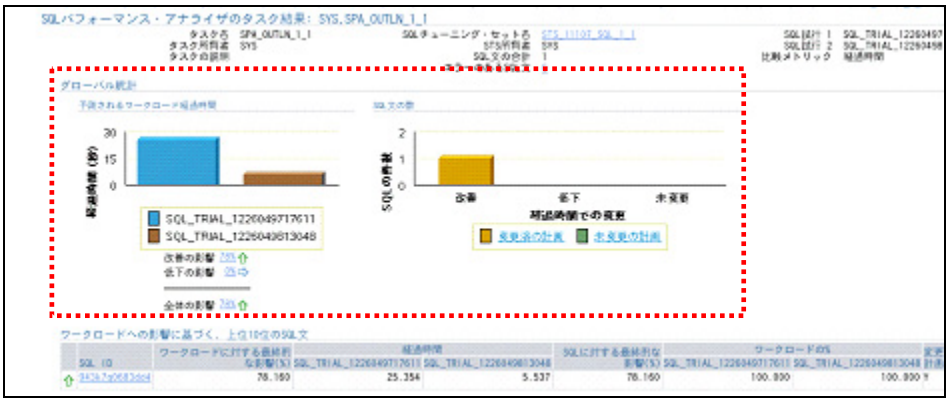


図 6-48 SPA による比較

6.3.7. まとめ

データベースのアップグレード時には、オプティマイザの動作変更やシステム I/O の特性などにより実行計画が変化する可能性があります。特に、Oracle Database 10g 以降ではコストベース・オプティマイザのみがサポートされおり、Oracle9i Database 以前でサポートされているルールベース・オプティマイザからコストベース・オプティマイザに変更する場合には注意が必要となります。

コストベース・オプティマイザでは、ほとんどの場合、最適な実行計画が選択されますが、今回の検証結果のように最適ではない実行計画が選択されるためアップグレード前の実行計画を維持した方が良い場合があります。

Oracle Database 11g では、SPA を使用することによりアップグレード前・後のパフォーマンスを SQL 単位で比較することができます。

また、SPA を実施後、パフォーマンスの劣化が見られる SQL に対して、STA や SQL 計画の管理を組み合わせることで、チューニングも容易に行える他、アウトラインを SQL 計画ベースラインへ格納し、固定化するか検討する際に、SPA を使用して比較できるなど、さまざまな機能と連携させることが可能です。

7. 考察／結論

本検証では、旧システムからの移行として PRIMEPOWER 250 から、最新の SPARC/Solaris サーバである SPARC Enterprise M3000 への移行および Oracle9i Database から Oracle Database 11g へのアップグレードに関する検証を行いました。

パフォーマンスの比較では、SPARC Enterprise M3000 と Oracle Database 11g のシステムは、PRIMEPOWER 250 と Oracle9i Database のシステムに比べて、オンライントランザクション処理性能が 約 3.5 倍、バッチ処理性能においても約 2.0 倍に向上するという最新の SPARC/Solaris サーバと Oracle Database 11g に移行することのメリットが確認できました。

しかし、旧システムからの移行では、互換性の問題や性能劣化をはじめとする様々なリスクが存在します。これらのリスクを低減させるためには、アプリケーション・テスト、パフォーマンス・テストが重要になります。

この課題に対し、Oracle Real Application Testing （RAT）を使用することによる効果を確認しました。Database Replay は、実際に実行されているワークロードを取得、移行先の環境でリプレイすることで互換性の確認やシステム全体の負荷を確認できます。特に、ユーザー・コール間の経過時間のスケールを変更できるので将来を見越した負荷をかけることができ有益だと考えられます。

SPA は、これまでほぼ手作業であった個々の SQL の性能比較を Oracle Enterprise Manager より比較レポートを容易に作成できることは、性能比較を行う上での効率化が見込めます。また、万一性能劣化が発生した場合においても SQL Turning Advisor や SQL 計画管理機能を使用することで SQL のチューニングや実行計画の固定化が行うことができます。

Oracle9i Database から Oracle Database 11g へ移行する際に RAT を使用することでアプリケーション・テスト、パフォーマンス・テストを効率的に実施し、移行時のリスクや移行後のリスクを低減させることができます。

本書を現行システムから新システムへ移行する際に活用していただければ幸いです。

8. Appendix

8.1. 移行後のReal Application Testing活用

Oracle Database 11g へのアップグレードが完了したお客様は、継続してデータベース環境の変更に伴うテストを実施するために、Database Replay および SQL Performance Analyzer をご活用頂くことが可能です。

データベース環境の変更とは、具体的に以下のようなものを指します。

- Oracle Database 11g 新機能の利用
- データベース・アップグレード、パッチ、パラメータ、スキーマの変更など
- シングル・インスタンスから RAC や ASM への変換といった構成の変更
- ストレージ、ネットワーク、インターコネクトの変更
- オペレーティング・システム、ハードウェアの変更、パッチ、アップグレード、パラメータの変更
- オプティマイザ統計情報の収集 - SQL プロファイルの作成などの SQL チューニング操作
- データベース初期化パラメータの変更

また､既存データベース環境に新規業務アプリケーションを追加した際のパフォーマンスへの影響を評価する目的でも、ご利用頂くことができます。そして、上記のようなテスト結果を履歴で管理することも可能です。

8.2. ハードウェア・リソースの有効活用

データベースをアップグレードする際に最新のハードウェアにリプレースすることで様々なメリットを享受することができます。図 8-1 は、PRIMEPOWER 250 から SPARC Enterprise M3000 に移行した場合のワークロードを Database Replay により再現した際の CPU 使用率を表しており、システム負荷が低減されハードウェア・リソースに余裕があることが確認できます。

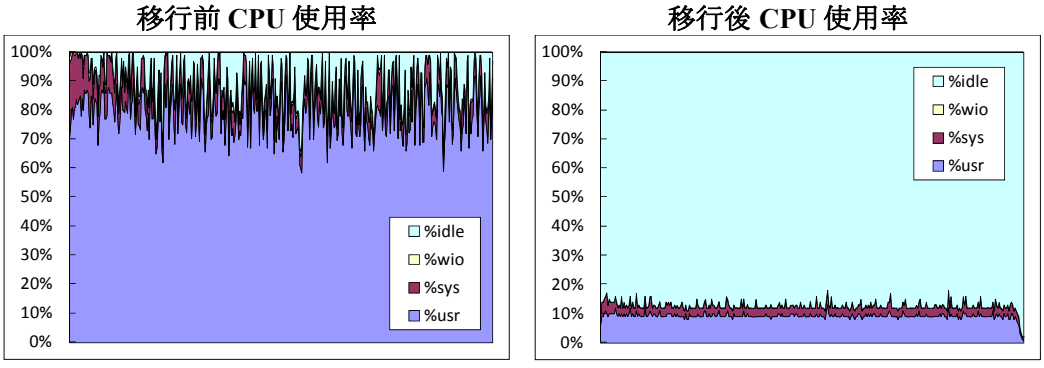
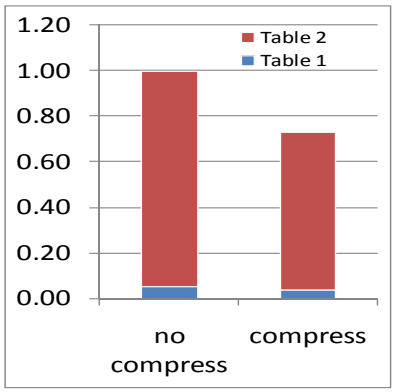


図 8-1 移行前と移行後の CPU 使用率

このようなハードウェア・リソースは、将来の業務量・データ量の増加への備えでもありますが、セキュリティや圧縮といった新機能に用いることができます。

Oracle Database 11g の新機能の一つとしてEnterprise Editionのオプション機能であるOracle Advanced Compression※6があります。Oracle Database 10g Release2 まではDWH向けとしてデータ・セグメント圧縮機能が提供されましたが、Oracle Database 11gでは、通常のDML文でも圧縮されるよう機能拡張が行われました。



Oracle Advanced Compressionにより、ワークロードで使用している表のうち 2 つの表を圧縮した結果が図 8-2になります。

圧縮の対象とした表は、約 30%程度圧縮されることが確認できます。

図 8-2 圧縮の効果

Oracle Advanced Compression では、通常の DML 文でも圧縮が行われますのでシステムへ与える負荷を確認する必要があります。

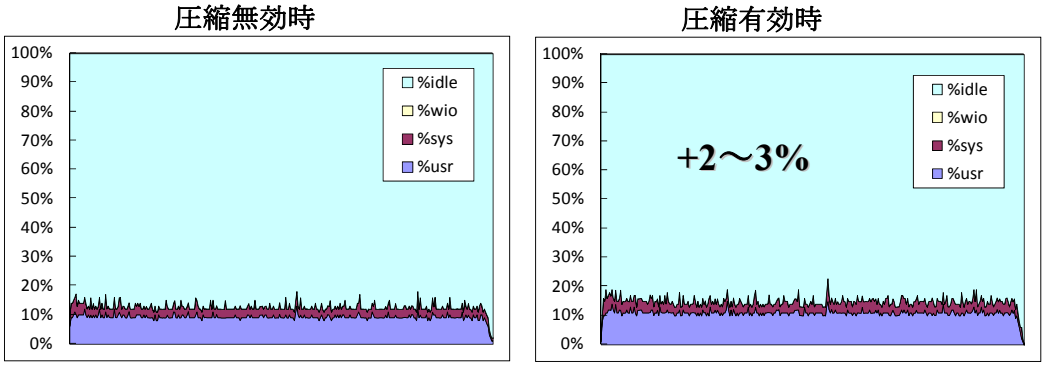


図 8-3 CPU 使用率の時間推移

右のグラフが圧縮を無効にした状態で Database Replay によりリプレイした結果、左のグラフが圧縮を有効にした状態でリプレイした結果です。圧縮を有効することで CPU の使用率は高くなりますが、2～3%の増加に止まることが確認できます。

次に圧縮した表に対するInsert/Update処理のCPU使用時間が表 8-1となります。圧縮を有効にするとInsert / Update処理の双方に影響がありますが、特にUpdate処理に影

6 詳細は「富士通 SPARC Enterprise による Oracle Database 11g データ・ウェアハウス検証」を参照

URL http://primeserver.fujitsu.com/sparcenterprise/documents/data/

響を与えることが表 8-1から確認することができます。

表 8-1 Insert / Update 処理の CPU 使用時間

圧縮	Insert 処理	Update 処理
無効	1.0	1.0
有効	1.1	5.6

※圧縮無効時の CPU 使用時間を 1.0 とした場合

このように、Real Application Testing（Database Replay）を活用頂くことで、システム変更や新機能を追加した際にシステム全体や個々の SQL の処理に与える影響を確認することが可能です。

8.3. Oracle Database 10g Release2 へアップグレードするお客様へ

本書では、Oracle 9i Database から Oracle Database 11g への移行について取り上げてきました。ただ、現在使用しているサードパーティ・ビジネス・アプリケーションが最新の Oracle Database 11g をサポートしていない、などの理由により、既存の Oracle 9i Database システムから Oracle Database 10g R2 への移行をご検討されているお客様もいらっしゃいます。そうしたお客様をサポートするために、ここではこのアップグレードによる SQL パフォーマンスへの影響をテストするための方法を明確にします。

Oracle 9i DatabaseからOracle Database 10g R2[7]へのアップグレードにおけるSQLパフォーマンスへの影響をテストできるよう、SPAが機能拡張されました。

具体的には下図のように、6 つの主要な処理手順があります。

[7] 詳細はホワイト・ペーパー 「Oracle Real Application Testing : SQL Performance Analyzer を使用した Oracle9i Database から Oracle Database 10g Release 2 へのアップグレードによる SQL パフォーマンスへの影響のテスト」を参照 (URL: http://www.oracle.com/technology/global/jp/products/manageability/database/doc/owp_spa_9i.pdf).

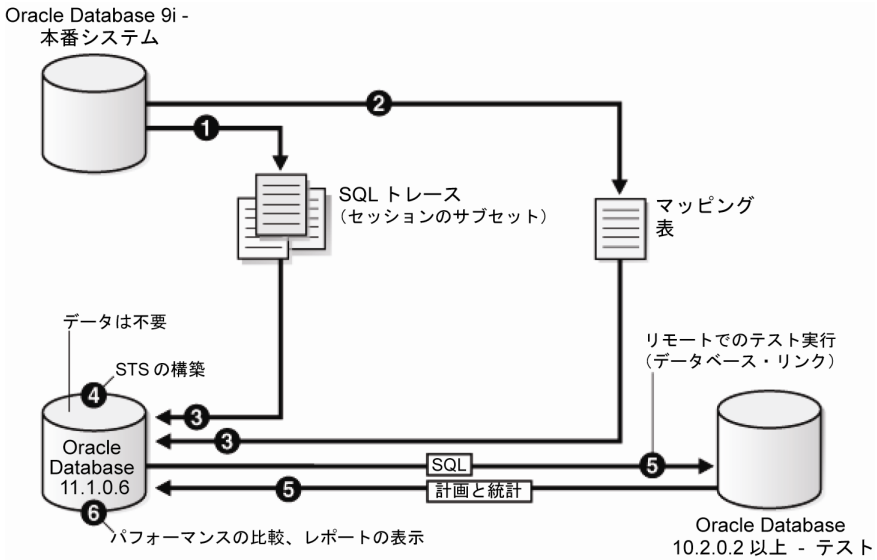


図 8-4 9i から 10g R2 へのアップグレードにおける SQL パフォーマンス・テスト

1. Oracle9i Database 本番システムの信頼できる SQL ワークロードを一連の SQL トレース・ファイルに取り込みます。
2. Oracle9i Database 本番システムの SQL トレース・ファイルの情報をデコードするオブジェクト・マッピング表を作成します。
3. SPA テストの中心となる Oracle Database 11g および Oracle Database 10.2 のテスト・データベース・システムのペアを構築します。手順 1 と 2 で生成された SQL トレース・ファイルとマッピング表が Oracle Database 11g テスト・システムにコピーされます。
4. Oracle Database 11g テスト・システムで、SQL トレース・ファイルとマッピング表から SQL Tuning Set (STS) を作成します。STS は、一連の SQL を実行コンテキストおよびパフォーマンス・データで表すデータベース・オブジェクトです。SPA の主要な入力として使用できます。
5. Oracle Database 11g テスト・システムから、SPA を使用してデータベース・リンク経由で Oracle Database 10.2 システムに接続し、SQL ワークロードを 1 つの文ずつ実行して、Oracle Database 11g データベースの中にすべてのパフォーマンス・データを収集します。
6. 最後に、収集した 2 つのセットのパフォーマンス・データを SPA を使用して分析し、パフォーマンスが低下した SQL をチューニングして、期待するレスポンスタイムが得られるようになるまで提案された修正のテストを繰り返します。

次に、パフォーマンスが低下した SQL のチューニングについて、その具体的なサイクルを下記に示します。

1. SPA 比較レポートより、性能が劣化した SQL に対して SQL Tuning Advisor

を実行します。

2. SQL Tuning Advisor より有益な推奨が得られた場合、推奨の実装を検討します。
3. 推奨を実装し、実装後のリプレイ試行を作成し SQL を実行します。
4. 実装前後の性能を比較します。
5. SQL Tuning Advisor より有益な推奨が得られなかった場合、Oracle9i Database 本番システム上で対象 SQL のストアド・アウトラインを作成します。
6. 作成したストアド・アウトラインを Oracle Database 10g R2 のテスト・システムへ移行し、実行計画を固定します。

Oracle Real Application Testing は、何らかのやむを得ない理由によって Oracle9i Database を Oracle Database 10g R2 にまでしかバージョンアップできない場合であっても、事前に十分なテストを実施できる環境を提供することが可能です。

日本オラクル株式会社
〒107-0061
東京都港区北青山 2-5-8
オラクル青山センター

富士通株式会社
〒105-7123
東京都港区東新橋1-5-2
汐留シティセンター