Deployment Best Practices for Oracle Database with Dell EMC PowerMax

2020 年 2 月

H17390.4

VMAX と PowerMax のエンジニアリングに関するホワイト ペーパー

要約

Dell EMC PowerMax ストレージ システムは、ハイパフォーマンスの NVMe フラッシュ ストレージ向けに設計および最適化されており、使いやすく、信頼性、可用性、セキュリティ、汎用性に優れています。このホワイト ペーパーでは、PowerMax ストレージ システムに Oracle データベースを導入するメリットとベスト プラクティスについて説明します。

目次

概要

PowerMax ファミリーは、アプリケーションに提供される NVMe のパフォーマンス メリットを最大限に引き出すために、特別に作成されました。NVMe は、Non-Volatile Memory（NVM）ストレージ メディアに効率的にアクセスするための PCI Express（PCIe）インターフェイスを定義する規格です。PowerMax NVM メディアには、NAND ベースのフラッシュ ストレージと、インテル Optane などのデュアル ポート ストレージ クラス メモリー（SCM）ドライブ テクノロジーの両方が含まれています。

PowerMaxOS の 2019 年第 3 四半期リリースでは、NAND ドライブと SCM ドライブの両方が使用されている場合のデータ配置用に、SCM ドライブとともに機械学習（ML）アルゴリズムが PowerMax で新たにサポートされるようになりました。このアップデートでは、サーバーとストレージ間のデータ アクセスを最適化するエンドツーエンドの FC-NVMe（NVMe over Fiber-Channel Fabrics）と、ポートあたりの接続速度を向上できる 32 Gb フロントエンド モジュールも導入されています。

さらに、PowerMax ファミリーでは、エンタープライズ アプリケーションが必要とする 99.9999%の可用性、暗号化、レプリケーション、データ削減、大規模統合などのあらゆる機能を継続して提供しており、現在はマイクロ秒という測定値の I/O レイテンシーを実現しています。

このホワイト ペーパーでは、PowerMax ストレージ システムに Oracle データベースを導入するメリットとベスト プラクティスについて説明します。

対象者

このホワイト ペーパーは、PowerMax ストレージ システムを使用した Oracle データベースの実装、管理、維持を担当するデータベースとシステムの管理者、ストレージ管理者、およびシステム設計者を対象としています。読者の方は、Oracle と PowerMax ファミリーにある程度精通し、ストレージ管理におけるデータベースの可用性、パフォーマンス、および容易さの向上に関心があるものと想定しています。

Oracle データベースが PowerMax にもたらす主なメリット

Oracle データベースの導入がメリットとなる、PowerMax の主な機能の概要を次に示します。

パフォーマンス

一般的なパフォーマンス メリット

- **PowerMax キャッシュが究極の読み書きパフォーマンスを実現**：PowerMax ストレージ システムは、最大 16 TB の生の DRAM ベース キャッシュをサポートします。PowerMax キャッシュの一部はシステム メタデータに使用されますが、大部分はアプリケーションの読み取りおよび書き込み操作で究極のパフォーマンスをサポートするために使用されます。
- **32 Gb のフロントエンド モジュール**：PowerMaxOS の 2019 年第 3 四半期リリースでは、ポートあたり 32 Gb のフロントエンド接続が PowerMax でサポートされるようになりました。これにより、アプリケーションの読み取りおよび書き込み操作でポートあたりの速度が向上します。これらのモジュールは、FC と FC-NVMe の両方の接続をサポートしています。
- **エンドツーエンドの FC-NVMe**：PowerMaxOS Q3 2019 リリースでは、PowerMax の NVMe バックエンド接続が拡張され、サーバーとストレージ間でエンドツーエンドの NVMe over Fabrics がサポートされるようになりました。これにより、FC と比較してプロトコルのさらなる最適化が可能になります。
- **ホスト I/O の上限値とサービス レベル**：一部のお客様は、非本番ワークロードやマルチテナント設計など（チャージバックやサービス プロバイダー用など）のためのパフォーマンス制限設

定機能の活用を選択しています。PowerMax のホスト I/O の上限値機能によって、特定のストレージ グループ（SG）の IOPS または帯域幅に制限を設けることができます。同様に、サービス レベル（SL）を使用して、SG のパフォーマンス目標を設定することができます。

詳細については、PowerMax のサービス レベルを参照してください。

最適化された書き込み

- **書き込み用の永続キャッシュ**：PowerMax キャッシュは、電源障害が発生した場合に書き込みとヴォールト用にミラーリングされるため、永続的であると見なされます。その結果、すべてのアプリケーションの書き込みは、キャッシュに登録されるとすぐにサーバーに認識され[1]、書き込みのレイテンシーが非常に低くなります。
- **書き込みの保持**：データベースの書き込みでは、短時間で同じブロック（または隣接するブロック）が複数回アップデートされる傾向があります。PowerMax の書き込み保持機能により、キャッシュ（DRAM）で複数のアップデートが可能になり、NVMe フラッシュ メディアの最新アップデートが定期的に保持されます。つまり、不要な書き込みを回避することによって、メディアの保存性を向上し、ストレージ リソースの使用率を改善しています。
- **書き込み統合**：PowerMax ストレージ システムは、キャッシュされたデータを NVMe フラッシュメ ディアに書き込む場合、アプリケーションの書き込みよりも大きなサイズの I/O への書き込みを集約して最適化し、不要な I/O 操作を排除できます。

最適化された読み取り

- **FlashBoost**：PowerMax キャッシュによって提供されるデータベース読み取り I/O は、すでにとても高速です。しかし、データがキャッシュに存在しない場合（つまり「読み取りミス」の場合）、PowerMax ストレージ システムでは、データをバック エンド（NVMe フラッシュ メディア）からフロント エンド（サーバー）へ送信してから、後で発生する可能性のある読み取り用としてキャッシュに配置することによってデータ転送を高速化します。

データ削減

データ削減機能には次のものが含まれます。

- **シン デバイス**：PowerMax ストレージ デバイスはすべてデフォルトでシンとして作成されます。そのため、ここにストレージ容量が割り当てられるのは、アプリケーションによる書き込みが行われた場合のみです。
- **圧縮と重複排除**：PowerMax の Adaptive Compression Engine（ACE）は、インライン ストレージ圧縮と重複排除を提供します。ハードウェア モジュールは、圧縮と重複排除の両方をサポートして、アクティビティ ベースの圧縮（ABC）やきめ細かなデータ パッキングなどの他のソフトウェア機能とともに、ハイ パフォーマンスを実現します。

  詳細については、PowerMax の圧縮と重複排除を参照してください。
- **ASM オンライン ストレージの再利用**：Oracle ASM フィルター ドライバー（AFD）によって、ASM ディスク グループがオンライン ストレージを再利用できるように宣言することができます。ASM 内で大規模なデータ セットを削除した場合（たとえば、従来のデータベースを削除した場合）、PowerMax ストレージ システムでは、ASM ディスク グループがオンラインのままでもストレージ システム内の削除された容量を解放できます。

  詳細については、付録 II.Oracle ASM オンライン ストレージの再利用を参照してください。

[1]ただし、リモート PowerMax キャッシュ登録のための書き込みを行わないと I/O が発信元ホストに認識されない同期レプリケーションを除きます。

ローカル レプリケーション

PowerMax SnapVX ソフトウェアを使用すると、各ストレージ グループから最大 256 個のローカル スナップショットを作成してソース データを保護することができます。これらのスナップショットはいつでもリストアすることができ、最大 1,024 個のターゲットにリンクさせることができます。スナップショットとリンクしたターゲットからは、スナップショットのデータに直接アクセスすることができます。SnapVX では、ポイントインタイム保護のほか、テスト環境の作成、バックアップ/リカバリー イメージなどといった目的のためにデータベース コピーを瞬時に作成（またはリストア）します。

SnapVX のスナップショットには、次のような特徴があります。

- **整合性がある**：すべてのスナップショットは、ネイティブに「ストレージコンシステント」（再開可能なデータベース レプリカ）です。Oracle のバックアップ/リカバリーのベスト プラクティスに従うことで、スナップショットは「アプリケーション コンシステント」（リカバリー可能なデータベース レプリカ）になり、データベースのロールフォワード リカバリーが可能になります[2]。
- **保護されている**：すべてのスナップショットが保護されています。スナップショットは、（たとえば、パッチ テスト中に）テストが成功するまで何度でもリストアできます。また、後で別のサーバーからマウントしたターゲット デバイスにスナップショットをリンクさせることもできます。ターゲット デバイスを変更しても、元のスナップショットのデータには影響しません。
- **名前が付けられている**：すべてのスナップショットの作成時に、ユーザー フレンドリーな名前が付けられます。同じ名前が使用されている場合は、管理を容易にするために新しい世代のスナップショットが作成されます。
- **自動的に期限が切れる**：オプションで、スナップショットに自動有効期限の日付と時刻を設定して、スナップショットを自動的に終了させることができます。
- **安全に保護できる**：必要に応じて、スナップショットを安全に保護することができます。安全に保護されたスナップショットは、有効期限が切れる前に削除することはできません。
- **アドホックにできる、またはスケジュール設定できる**：スナップショットは、いつでもアドホックにすることができます。また、Unisphere を使用してスケジュール設定することもできます。

SnapVX の詳細については、ホワイト ペーパー『Oracle Database Backup, Recovery, and Replications Best Practices with VMAX All Flash Storage』を参照してください。

リモート レプリケーション

PowerMax SRDF では、同期および非同期モードのほか、カスケード、Star、および Metro（Oracle extended RAC と連動するアクティブ/アクティブ機能）トポロジーを含む、幅広いレプリケーション モードとトポロジーを提供します。

SRDF の詳細については、ホワイト ペーパー『Oracle Database Backup, Recovery, and Replications Best Practices with VMAX All Flash Storage』を参照してください。

データ保護

データ保護機能には次のものが含まれます。

- **内部 T10-DIF**：T10-DIF（データの整合性フィールド）、別名 T10-PI（保護情報）は、SCSI ブロックを 512 バイトから 520 バイトに変更し、CRC およびブロック アドレスなどの保護情報を 8 バイト追加するデータ保護標準です。内部的には、PowerMax システム内

[2]Oracle 12c 以降、SnapVX スナップショットは、Oracle がホットバックアップ モードのときに作成しなくても、データベースのリカバリーに使用できます。これにより、SnapVX を使用したより積極的なバックアップ/リカバリー戦略が実現します。

のデータはすべて、フロントエンド モジュール、キャッシュ、バックエンド、およびフラッシュ ストレージ間を移動するときに T10-DIF で保護されます。PowerMax の T10-DIF 保護には、データの破損を防ぐ、ローカルおよびリモートのレプリケーションが含まれています。

- **外部 T10-DIF**：サポートされている構成では、PowerMax ストレージ システムによって T10-DIF 保護をデータベース サーバーとバック エンドに拡張することができます。参加レイヤーは、すべての読み取りおよび書き込み I/O をリアルタイムで検証し、破損をブロックします。外部 T10-DIF は、Oracle ASMlib、Red Hat Linux などによって実装することができます。サポートされる構成の全リストについては、Dell EMC サポート マトリックスを参照してください。
- **PowerProtect Storage Direct**：PowerProtect Storage Direct（以前の名称は ProtectPoint）は、PowerMax ストレージ システムと Data Domain バックアップ ストレージ アプライアンス間の統合によって、大規模なデータベースのバックアップを数秒で実行できるようにします。Data Domain では、バックアップをカタログ化し、圧縮、重複排除、および任意のリモート レプリケーションを追加することができます。統合システムでは、リストア時間も最適化されます。
- **D@RE**：静止データ暗号化（D@RE）によって、ストレージ システム内の透過的なデータ暗号化が実現しています。PowerMax ストレージ システムでは、特殊なハードウェア モジュールを使用してパフォーマンス ペナルティを回避しています。

PowerMax 製品の概要

Dell EMC PowerMax ファミリーは、次の図に示すように、2 つのモデルで構成されています。

- **PowerMax 2000**：20U の設置面積で、効率性と最大限の柔軟性を提供するように設計
- **PowerMax 8000**：大規模なスケールとパフォーマンスを実現するように設計されており、すべてがフロアタイル 2 枚の設置面積に収まるように設計

図 1.　PowerMax 2000 と PowerMax 8000

いずれの PowerMax ストレージ システムにも、その基盤に信頼性の高い Dynamic Virtual Matrix アーキテクチャと、NVMe プラットフォーム向けに書き換えられた PowerMaxOS 5978 と呼ばれる新しいバージョンの HYPERMAX OS 管理ソフトウェアが搭載されています。PowerMaxOS は、PowerMax ストレージ システムと従来の VMAX All Flash システムの両方でアップグレードとしてネイティブに実行することができます。PowerMax ストレージ システムは、エンタープライズ データ センターのストレージ容量とパフォーマンス要件を満たすことを具体的な目的としています。

PowerMax のアーキテクチャ

PowerMax は、次の図に示すように、PowerMax ブリックと呼ばれるモジュール型ビルディング ブロックで構成されています。モジュール型ブリック アーキテクチャによって、複雑さが軽減され、システムの構成と導入が容易になっています。

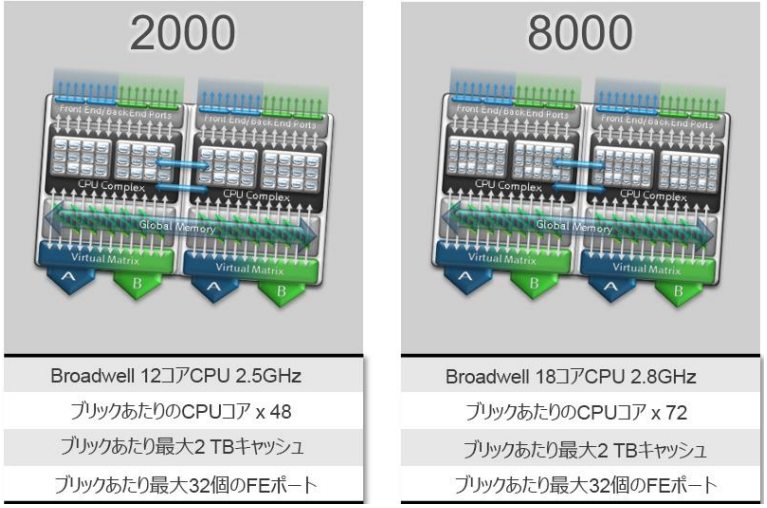

図 2. PowerMax 2000 と PowerMax 8000 のブリック

最初の PowerMax ブリックには、エンジン 1 個、ダイレクター2 台、システム電源（SPS）2 台、24 スロットの 2.5 インチ NVMe ドライブ アレイ エンクロージャ（DAE）2 台が含まれます。PowerMax 2000 には、RAID 構成に応じて、11 TBu または 13 TBu[3]の初期容量が搭載されています。PowerMax 8000 には、オープン システム用に 53 TBu の初期容量が搭載されています。

ブリックの概念によって、PowerMax ストレージ システムのスケール アップとスケール アウトが可能です。お客様は、フラッシュ容量パックを追加することでスケールアップできます。PowerMax 8000 ストレージ システムの各フラッシュ容量パックには、13 TBu の使用可能なストレージがあり、PowerMax 2000 ストレージ システムには、使用可能なストレージが 11 TBu または 13 TBu あります（RAID 保護タイプによって異なります）。

PowerMax ストレージ システムは、単一システム内において、PowerMax 2000 ストレージ システムに最大 2 個のブリックを統合するか、PowerMax 8000 ストレージ システムに最大 8 個のブリックを統合することによってスケール アウトし、接続、処理能力、および直線的な拡張性を全面的に共有することができます。

[3] TBu：テラバイト単位の有効容量。有効容量とは、使用中の RAID タイプの RAID 効率を考慮してストレージ システムで提供される物理容量を指します。

PowerMax のアーキテクチャと機能の詳細については、次の資料を参照してください。

- ホワイト ペーパー『Dell EMC PowerMax Family Overview』
- データ シート『Dell EMC PowerMax ファミリー』
- スペック シート『Dell EMC PowerMax ファミリー』

PowerMax の圧縮と重複排除

PowerMax Adaptive Compression Engine（ACE）

PowerMax ストレージ システムには、パフォーマンスを犠牲にすることなく、最適なデータ削減を実現するための戦略が採用されています。PowerMax Adaptive Compression Engine（ACE）には、次のコンポーネントが組み合わされています。

- **ハードウェア アクセラレーション**：各 PowerMax エンジンは、データの圧縮と解凍を処理する 2 個のハードウェア圧縮モジュール（ダイレクターごとに 1 個）で構成されます。また、これらのハードウェア モジュールでは、重複排除を可能にするハッシュ ID を生成することができ、この ID は VMAX All Flash アレイで使用されるモジュールよりも強力なものです。
- **最適化されたデータ配置**：アプリケーションのデータは、1:1（128 KB のプール）から最大 16:1（8 KB のプール）までの圧縮率（CR）を提供するさまざまな圧縮プールに保存され、最適なパフォーマンスを実現するために PowerMax のバック エンド全体に分散されます。プールは、必要に応じて動的に追加または削除されます。
- **アクティビティ ベースの圧縮（ABC）**：通常は、最新のデータが最もアクティブであり、「アクセス スキュー」を作成しています。ABC では、このスキューに依存し、頻繁にアクセスされるデータ エクステントの圧縮と解凍を常に回避します。ABC 機能によって、システムに割り当てられたすべてのデータ エクステントで、最もビジーな 20%がマークされ、圧縮ワークフローをスキップできるようになります。ストレージ グループで圧縮が有効になっている場合でも、非常にアクティブなデータ エクステントは圧縮されないままです。データ エクステントが比較的アクティブでなくなると、そのエクステントは自動で圧縮され、その一方で新しくアクティブになったエクステントが、（十分な空きストレージ容量が利用可能である限り）「最もビジーな」20%の一部になります。
- **きめ細かなデータ パッキング**：PowerMax によってデータが圧縮されると、各 128 KB トラックは 4 個の 32 KB バッファーに分割されます。すべてのバッファは並列で圧縮されます。4 個のバッファの合計が最終的な圧縮サイズとなり、データがどの圧縮プールに割り当てられるかが決定されます。このプロセスにはゼロ再利用機能が含まれており、すべてゼロのバッファと実際のデータではないバッファのアロケーションが回避されます。サイズの小さい書き込みまたは読み取りでは、4 個のバッファすべてではなく、必要なバッファだけが参加します。
- **Extended Data Compression（EDC）**：すでに圧縮されているデータが 30 日以上放置されている場合は、自動でより強力な追加圧縮を行い、ストレージの効率性を向上させます。

また、次の点に注目できます。

- 圧縮はストレージ グループ レベルで有効または無効となるため、管理が容易になります。一般に、ほとんどのデータベースにストレージ圧縮のメリットがあります。データベースが完全に暗号化されている場合や、ストレージ グループに継続的に上書きされるデータ（Oracle REDO ログなど）が含まれている場合は、圧縮を有効にしないことを選択できます。
- 圧縮を有効にすると、すべての新しい書き込みにインライン圧縮のメリットがあります。圧縮を有効にしたときにストレージ グループにすでにデータが含まれている場合は、優先度の低いバックグラウンド圧縮が実行されます（アプリケーション I/O にはより高い優先度が設定されます）。

PowerMax の重複排除

PowerMax ストレージ システムでは、より強力なハードウェア圧縮モジュールが用意されているだけでなく、データ重複排除（重複除外）も可能です。PowerMax の重複排除は、圧縮が有効または無効になっていると、自動で有効または無効になります（圧縮と重複排除は個別に管理できません）。

PowerMax の重複排除は、128 KB の粒度で動作します。Oracle ASM アロケーション ユニット（AU）の粒度が 1 MB 以上であるため、PowerMax の重複排除は、ASM ディスク グループに存在する Oracle データベースとうまく連動します。すべての新しい ASM エクステントは 1 MB（またはそれ以上）のオフセットで配列されます。これにより、PowerMax ストレージ システムでは、配列不良の懸念なく、データが固有のものであるかを容易に判別することができます。このホワイト ペーパーで後述するように、PowerMax ストレージ システムでは、ASM に存在する Oracle データベースに対して 100%の重複排除というメリットを実現しています。

PowerMax のデータ削減の詳細については、

- 『Dell EMC PowerMax：データ削減に関するテクニカル ホワイト ペーパー』を参照してください。

PowerMax と FC-NVMe

PowerMaxOS Q3 2019 リリースでは、エンドツーエンドの NVMe over FC Fabrics、つまり FC-NVMe が PowerMax で導入されました。内部的には、PowerMaxOS は、すでに NVMe プロトコルを使用して NAND と SCM の両方のフラッシュ メディアにアクセスしていました。今回のリリースの新機能として、データベース サーバーも FC ファブリック（SAN スイッチ）を介して NVMe プロトコルを使用し、すべての読み取りおよび書き込み I/O 操作の際に PowerMax ストレージ システムにアクセスできるようになりました。

このプロトコルを使用するには、32 Gb PowerMax フロントエンド モジュールが Gen 6 FC スイッチおよび HBA（32 Gb）とともに必要です。このインフラストラクチャにより、FC または FC-NVMe の導入が可能になります。唯一の違いは、サーバーがストレージ デバイスにアクセスするために使用するプロトコルです。PowerMax フロントエンド アダプター ポートは、FC 接続用に設定されている場合に FA として表示され、FC-NVMe 用に設定されている場合に FN として表示されます。

NVMe を使用する理由

NAND SSD フラッシュ ストレージ以降では、また、さらに重要なことに SCM では、ストレージのレイテンシーがミリ秒単位ではなくマイクロ秒単位で測定されるようになりました。また、各 SCM は数十万 IOPS をサポートしています。

NVMe プロトコルは、回転式ドライブ用の従来の SAS および SATA ストレージ アクセス プロトコルに代わるものとして、ゼロから構築されました。NVMe はマルチコア CPU を活用できるよう NUMA 向けに最適化されているため、コアで I/O 送信キューの制御を共有できます。フラッシュ メディアと CPU 間の高速 PCIe アクセスを中心に構築されており、キューの深度を大幅に高めることができるため、同時実行性が向上します。このようなハイパフォーマンス ドライブは、多くの場合、I/O の高密度化（1 GB ストレージあたりの I/O の同時実行性の向上）を実現するため重要です。

FC-NVMe を使用する理由

オンボード サーバーの PCIe NVMe フラッシュ ドライブは新しいものではありませんが、PCIe FC-NVMe プロトコルを使用してサーバーに接続されている、ストレージ システム内の SCM と比べていくつかの欠点があります。まず、SCM などのハイパフォーマンス NVMe フラッシュ ドライブは、NAND SSD よりもコストがかかります。このようなドライブをサーバーに配置するには、コストを正当化できるだけの高いサーバー使用率が必要です。ところが、サーバーがピーク容量で稼働しているのは、毎日、毎週、または毎月のいずれの割合で見てもごく一部にすぎません。それ以外の時間帯のサーバー使用率は低くとどまっています。これにより、コストとリソースが浪費される可能性があります。

SCM ドライブをストレージ システムに配置すると、多くのサーバーが NVMe ファブリックを介してハイパフォーマンス メディアにアクセスできるようになります。どのサーバーのアクティビティがピーク状態であるかに関係なく、メディアが共有され、使用率が常に高くなります。このことは、SCM 投資に付加価値をもたらし、リソースの使用率向上につながります。また、PowerMax のサービス レベルでは、ストレージ グループによる SCM の消費が優先されています。

同様の比較は、HCI（ハイパーコンバージドインフラストラクチャ）とも行うことができます。HCI は、アプリケーションの導入環境にコンピューティング（CPU）、メモリー、ネットワーク、およびストレージ リソースを提供するサーバーのグループに基づいています。オンボード サーバーの SCM フラッシュ ドライブを使用する場合は、HCI の性質上、アプリケーションがクラスター内の任意のノードに配置される可能性があるため、すべてのサーバーで均等に SCM ドライブが必要になる可能性が非常に高くなります。

ワークロードのバランスが取れていない場合、一定時間にビジー状態になるサーバーはごくわずかです。ビジー状態のサーバーは、ローカルで利用可能なサーバーよりも多くの SCM ドライブにアクセスできる場合もあれば（SCM ドライブを利用する他のサーバーは十分に活用されません）、十分にビジーでないために、ローカルにインストールされている 1 台の SCM ドライブでさえも十分に活用できないこともあります。

FC-NVMe を使用することにより、ストレージ システムに配置されているこれらのハイパフォーマンス リソースを、それを必要としているすべてのサーバーで共有できるようになります。これにより、ストレージ システムに接続されているすべてのサーバーが、指定されたサービス レベルに基づいて SCM ドライブを活用できるようになるため、すべての HCI ノードに SCM ドライブを配置するコストが削減され、ドライブの使用率が向上します。

FC-NVMe は比較的新しいため、Linux OS とマルチパス ソフトウェアのサポートはまだ制限されています。詳細については、「FC-NVMe マルチパス オプション」を参照してください。

PowerMax と Oracle のパフォーマンス テスト

このセクションでは、当社のラボで Oracle ワークロードを実行して行ったパフォーマンス テストとその結果について説明します。テスト ケースは、さまざまな状況がデータベースのパフォーマンスにどのように影響するかを示すことを目的としています。

テストに使用した PowerMax ストレージ システムはシングルエンジン（ブリック）システムです。パフォーマンスの数値は、このプラットフォームのピーク パフォーマンスの数値と見なすべきではありません。そうではなく、Oracle データベースのワークロードが比較的小さな構成（単一の PowerMax 8000 ブリック システムと 4 つの 28 コア PowerEdge R740 サーバー）で達成できるパフォーマンス レベルの例を示しています。テストでは、このホワイト ペーパーで説明されているベスト プラクティスを活用して実証することもできました。

テスト環境

ハードウェアとソフトウェアの設定

表 1.には、パフォーマンス テストに使用したハードウェアおよびソフトウェアのコンポーネントを記載しています。

一般に、Oracle データベースのパフォーマンス（FC と FC-NVMe の両方）のテストには、FC-NVMe のオペレーティング システム ベンダーによって新たにサポートされる、SLES 15、RHEL 8.0、および OL 7.7/UEK5 アップデート 2 が使用されました。全体的なパフォーマンスは、さまざまなオペレーティング システム間で非常に類似していました。同じ結果を何度も報告しても意味がないため、このホワイト ペーパーでは、OL 7.7/UEK5u2 に基づく OLTP のテスト結果と、RHEL 8.0 に基づく DSS のテスト結果を示します。一部のテストは高 IOPS に重点を置いているため、FC-NVMe の代わりに FC エミュレーションを

使用しました（それが単一ブリックのベンチマーク テストで重要な理由については、「FC または FC-NVMe プロトコルの選択とコア割り当て」を参照してください）。

テストに使用した PowerMax 8000 ストレージ システムには、このシステムの最小構成である単一ブリック（1 個のエンジン）および 1 TB の未フォーマット キャッシュが存在しました。

Oracle の Grid Infrastructure 19c とデータベースは、4 ノード クラスター（RAC）として構成しました。

SLOB 2.4 のベンチマークを使用して、Oracle OLTP ワークロードを生成しました。SLOB の構成は、96 人のユーザー（およびデータベースのスキーマまたはテーブル）で構成され、合計データセット サイズは 30 GB スケールで、2.8 TB（96 x 30 GB）となっています。+DATA ASM ディスク グループの使用容量は、Oracle システムおよび UNDO テーブルスペースとともに 3.5 TB 近くになりました。パフォーマンス テストは、「lite」の REDO 世代と 30%のアップデート（`slob.conf` パラメーター）を使用して実行しました。

Oracle DSS テスト（大規模 I/O のシーケンシャルな読み取り）には、TPC-H ツールの dbgen ユーティリティーを使用して、1 TB の Lineitem テーブルを作成しました。

表 1.　ハードウェアおよびソフトウェア コンポーネント

カテゴリー	タイプ	数量/サイズ	迅速な
ストレージ システム	PowerMax 8000 ストレージ システム	• ブリック x 1、1 TB の未フォーマット キャッシュ • NVMe NAND SSD x 30 • NVMe SCM x 8	Q3 2019 リリースに基づく PowerMaxOS 5978.444.444
データベースサーバ	Dell R740 x 4	• 各 Dell サーバー：Intel Xeon E5-2690v4（2.6GHz） x 2（合計 28 コア）、128 GB RAM	
オペレーティングシステム（OS）	OL 7.7 と UEK5u2（OLTP テスト）、RHEL 8.0（DSS テスト）		
ホスト バス アダプター（HBA）	Broadcom（Emulex）	各サーバー：デュアル ポート 32 Gb HBA x 2（合計 4 つのイニシエーター ポート）	LPe32002 x 2（サーバー1 台あたり）
Oracle Database	Oracle Database と Grid Infrastructure 19c を ASM と併用	4 ノードの Oracle RAC	Oracle Database と Grid Infrastructure 19.3
ベンチマーク ツール	OLTP と DSS	OLTP：SLOB 2.4 DSS：TPC-H ツール（dbgen）を使用して作成された Lineitem テーブル。	

標準冗長性で設定された+GRID ディスク グループを除くすべてのディスク グループに対し、外部冗長性を使用して ASM ディスク グループを設定しました。+DATA ASM ディスク グループはデータ ファイルを含み、+REDO ASM ディスク グループは REDO ログを含んでいました。REDO ログのストライピングには、粒度の高い ASM ストライピング テンプレート（128 KB）が使用されました。

OLTP テストでは、デバイスごとに 16 個のパス（16 個のフロントエンド ポート）を使用しました。DSS テストでは、デバイスごとに 24 個のパス（24 個のフロントエンド ポート）を使用しました。デバイス 1 台あたりのパスの数とストレージ ポート数に関する考慮事項については、ベスト プラクティスを示したセクション「ストレージ接続」を参照してください。

OLTP パフォーマンス テスト ケース

OLTP テストの概要と結果のサマリー

OLTP テスト ケースの実行には、SLOB 2.4 を使用しました。最初のテスト ケースは、「最悪のシナリオ」を示しています。ここでは、完全な「読み取りミス」ワークロードがシミュレートされています。つまり、ほぼすべての読み取りがストレージ フラッシュ メディアから処理されるため、PowerMax キャッシュが読み取りに役立たない状況です。このテスト ケースは、PowerMax キャッシュが非常に大きな利点をもたらし、他のオールフラッシュ ストレージ システムとの差別化要因となる一方で、PowerMax によってハイ パフォーマンスと低レイテンシーも実現することを示しています。

2 番目のテスト ケースは、データベース全体のごく一部である最近のデータにお客様がアクセスする、より現実的なワークロードを示しています。その結果、データは PowerMax に部分的にキャッシュされ、典型的な読み取りヒットは 60%になります（読み取りのデータの 60%は PowerMax キャッシュにあります）。最初のテスト ケースと同様に、比較的低いレイテンシーを維持しながら、高 IOPS を達成することに重点が置かれました。

3 番目のテスト ケースは 2 番目のテスト ケースと似ていますが、高 IOPS を達成しようとする代わりに、低レイテンシーに重点が置かれました。これを行うために、システムの負荷を軽減する一方で、合理的なレベルの IOPS を維持しました。このテスト ケースの正当化理由は、IOPS の最大化（金融取引、Web カウンター、クレジット カード検証など）ではなく、高速レスポンス タイム（低レイテンシー）に、一部のデータベース ワークロードの重点が置かれていることです。

表 2.　OLTP のパフォーマンス テスト ケースと結果のサマリー（AWR ベースデータ）

テスト ケース	テストの詳細	データ ファイルの IOPS	データ ファイルの読み取り レイテンシー（ミリ秒）	ログ ライターの書き込みレイテンシー（ミリ秒）
1	6%の読み取りヒット、高 IOPS にフォーカス	324,402	0.58	0.43
2	60%の読み取りヒット、高 IOPS にフォーカス	489,944	0.49	0.72
3	60%の読み取りヒット、低レイテンシーにフォーカス	279,972	0.23	0.31

テスト ケース 1OLTP、6%のキャッシュ読み取りヒット、高 IOPS

テスト ケース 1 の目標は、完全な「読み取りミス」ワークロードをテストすると同時に（6%の読み取りヒットを達成）、良好なレイテンシーで可能な限り高い IOPS レベルを達成することでした。2.8 TB のデータベース全体で SLOB を実行しました。PowerMax のキャッシュ アルゴリズムは非常に効率的であり、通常は高い読み取りヒット率を生み出すため、この動作は現実的ではありません。また、データベース ワークロードは、データベース全体のごく一部でしかない最新のデータにアクセスする傾向があります。このテストを実施した理由は、「最も悪い」条件におけるパフォーマンスを示すことでした。

図 3 は、テスト ケース 1 の Oracle AWR IOPS を示しています。レポートによると、テスト中のデータベースのパフォーマンスは、平均して読み取り IOPS が 248,500、書き込み IOPS が 75,902 であり、合計で 324,402 IOPS でした。

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
...							
physical read IO requests	448,082,675	248,464.87	216.69	62,116.22	792.25	61,168.22	63,020.54
physical read bytes	3,670,791,839,744	2,035,478,907.93	1,775,188.62	508,869,726.98	6,491,659.42	501,100,719.26	516,278,364.84
physical read total IO requests	448,145,629	248,499.78	216.72	62,124.95	792.57	61,176.50	63,029.50
physical read total bytes	3,696,308,049,408	2,049,627,820.57	1,787,528.22	512,406,955.14	6,736,613.39	504,356,332.64	520,020,758.13
physical read total multi block requests	24,343	13.50	0.01	3.37	0.26	3.09	3.62
physical reads	448,094,707	248,471.55	216.70	62,117.89	792.44	61,169.52	63,022.26
physical reads cache	448,094,699	248,471.54	216.70	62,117.89	792.44	61,169.52	63,022.26
physical reads cache prefetch	12,043	6.68	0.01	1.67	0.25	1.31	1.89
physical reads direct	8	0.00	0.00	0.00		0.00	0.00
physical reads direct temporary tablespace	1	0.00	0.00	0.00		0.00	0.00
physical write IO requests	132,423,356	73,429.65	64.04	18,357.41	237.44	18,063.19	18,611.67
physical write bytes	1,095,950,278,656	607,711,841.24	529,999.67	151,927,960.31	1,963,937.58	149,496,505.89	154,036,033.95
physical write total IO requests	136,881,740	75,901.85	66.20	18,975.46	247.69	18,669.44	19,239.13
physical write total bytes	1,140,744,526,848	632,550,555.05	551,662.09	158,137,638.76	2,040,630.07	155,617,597.12	160,336,183.80
physical write total multi block requests	314	0.17	0.00	0.04	0.02	0.03	0.06
physical writes	133,782,993	74,183.57	64.70	18,545.89	239.74	18,249.09	18,803.23
...							

図 3. テスト ケース 1AWR IOPS の結果

図 4 は、テスト ケース 1 の Oracle AWR レスポンス タイムを示しています。レポートによると、データ ファイルの読み取りレスポンス タイムは平均して 579.8 マイクロ秒（0.58 ミリ秒）、REDO ログ ライターのレスポンス タイムは 427.2 マイクロ秒（0.43 ミリ秒）でした。

Top Timed Events

	Wait		Event		Wait Time			Summary Avg Wait Time				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg Wait	%DB time	Avg	Min	Max	Std Dev	Cnt
*	User I/O	db file sequential read	446,424,508	0.00	258,824.07	579.77us	94.46	579.86us	570.31us	591.03us	8.98us	4
*		DB CPU			27,947.51		10.20					4
*	System I/O	log file parallel write	2,295,545	0.00	980.75	427.24us	0.36	427.35us	417.56us	444.68us	12.50us	4
*	Cluster	gc cr grant 2-way	9,243,472	0.00	956.23	103.45us	0.35	103.55us	96.03us	106.62us	5.04us	4
*	System I/O	db file parallel write	2,216,219	0.00	280.62	126.62us	0.10	127.50us	105.51us	145.88us	18.02us	4
*	Cluster	gc buffer busy release	346	0.00	241.92	699.19ms	0.09	676.11ms	567.38ms	787.94ms	97.91ms	4
*	User I/O	ASM IO for non-blocking poll	2,779,340	0.00	57.10	20.54us	0.02	20.86us	15.96us	22.75us	3.27us	4
*	Other	ASM file metadata operation	21,916	0.00	13.04	594.91us	0.00	594.34us	571.80us	610.41us	16.33us	4
*	Other	KSV master wait	11,428	47.27	11.05	.97ms	0.00	.97ms	913.53us	1.00ms	38.15us	4
*	System I/O	control file sequential read	37,721	0.00	9.95	263.86us	0.00	263.92us	250.97us	276.72us	13.12us	4

図 4. テスト ケース 1AWR レスポンス タイムの結果

図 5 は、Oracle データ ファイル（data_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。平坦な線は、ワークロードが非常に安定していることを示しています。レポートによると、IOPS は 322,387、読み取りレスポンス タイムは 0.52 ミリ秒、書き込みレスポンス タイムは 0.35 ミリ秒でした。

これらのメトリックは、Oracle AWR で報告されたメトリックと非常によく似ています。ストレージ システムによっては、人為的に見栄えはするものの、アプリケーションのパフォーマンスと一致していない内部メトリックが報告される場合があるため、このことは重要です。Unisphere メトリックは、Oracle AWR メトリックに

類似しているはずです。そうでない場合は、2 つの間の考えられるボトルネック（サーバーのキューイングの問題や、十分な接続の欠如など）を調査してください。

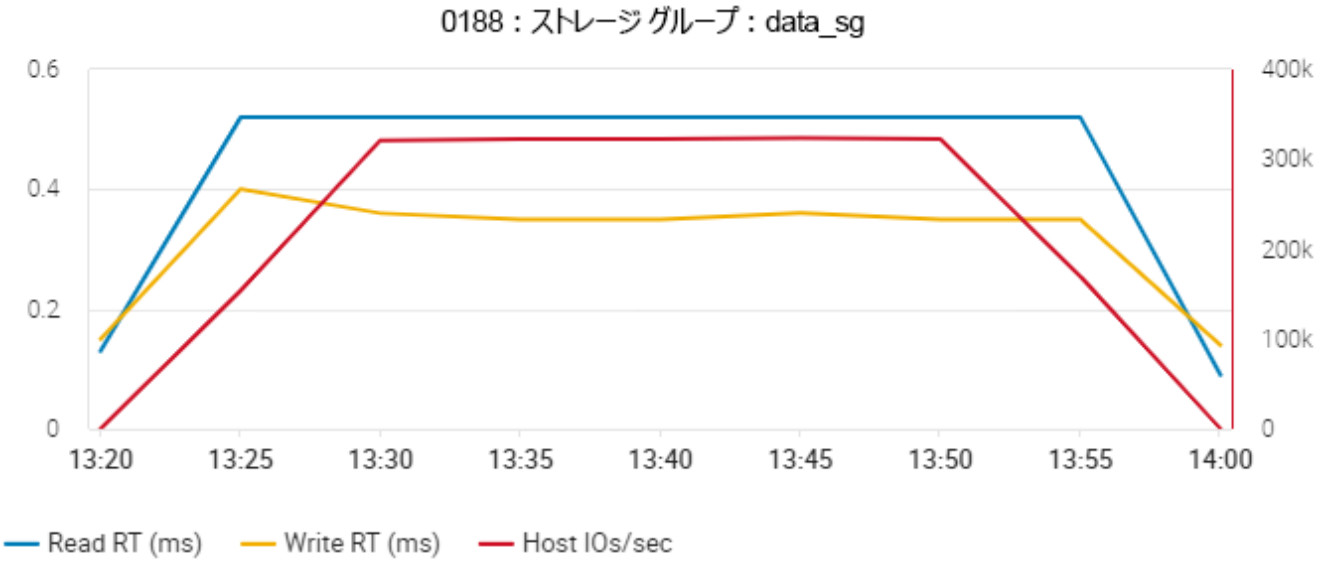

図 5.　テスト ケース 1Unisphere メトリック：Oracle データ ファイルのパフォーマンス

図 6 は、Oracle REDO ログ ファイル（redo_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。このテスト ケースは、100%の書き込みワークロードに対して、0.22 ミリ秒の書き込みレスポンス タイムと 1,485 IOPS を示しています。アーカイブはバックグラウンドで実行されるため、アーカイブ ログがパフォーマンスへの影響を示すとは考えられませんが、テスト中はアーカイブ ログが無効化されました。

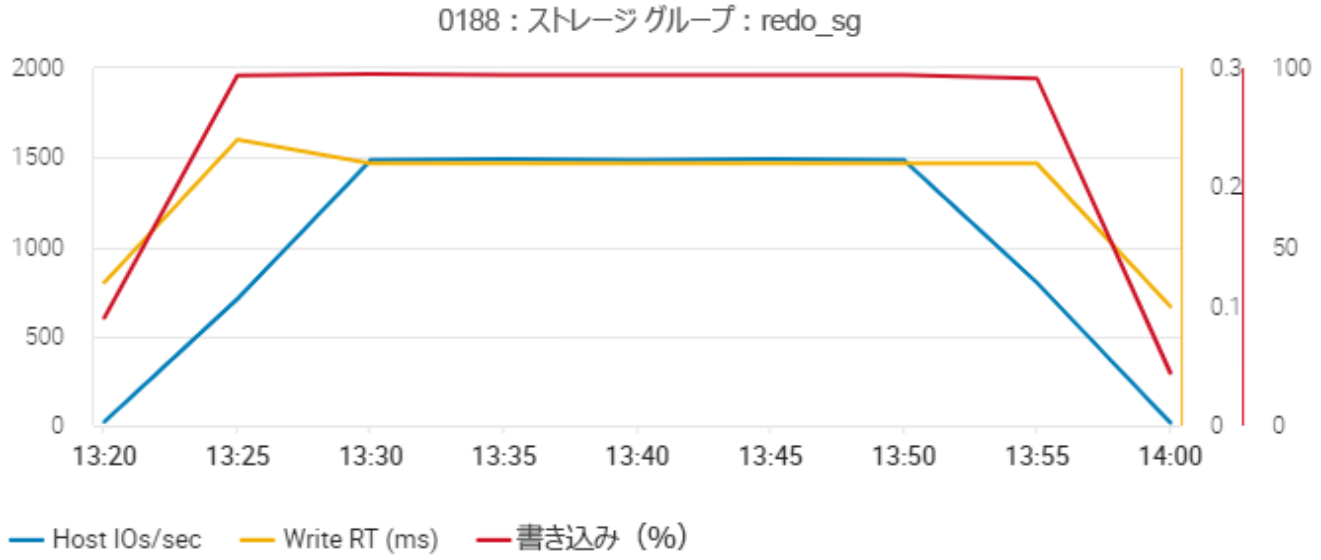

図 6.　テスト ケース 1Unisphere メトリック：Oracle REDO ログのパフォーマンス

要約すると、テスト ケース 1 は、ワークロードのアクティブ データセットが大きすぎるためにキャッシュ アルゴリズム（読み取りミス ワークロード）のメリットがまったくないという非現実的な動作を示しましたが、データ ファイルの読み取りレスポンス タイム 0.58 ミリ秒で 320,000 を超える IOPS という極めて良好なパフォーマンスを達成しました。

テスト ケース 2OLTP、60%のキャッシュ読み取りヒット、高 IOPS

テスト ケース 2 は、より現実的なワークロードを特徴としており、アクティブ データセットは高 IOPS と最小限のレイテンシーという PowerMax キャッシュ（60%のキャッシュ読み取りヒット）のメリットを享受します。データベースのフル サイズよりも小さいアクティブ データセットをシミュレートするため、各データベース ユーザーがフル データの一部にアクセスできるようにする SLOB の「ホットスポット」機能を使用しました。

図 7 は、テスト ケース 2 の Oracle AWR IOPS を示しています。テスト中のデータベースのパフォーマンスは、平均して読み取り IOPS が 373,310、書き込み IOPS が 116,635 であり、合計で 489,945 IOPS でした。

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
...							
physical read IO requests	673,127,417	373,253.23	208.62	93,313.31	1,653.62	91,203.64	95,212.12
physical read bytes	6,236,554,485,760	3,458,206,096.89	1,932,897.14	864,551,524.22	210,802,390.20	747,142,955.39	1,180,484,705.89
physical read total IO requests	673,229,026	373,309.57	208.65	93,327.39	1,659.63	91,213.40	95,237.46
physical read total bytes	6,275,332,006,912	3,479,708,463.35	1,944,915.47	869,927,115.84	211,384,719.82	751,857,893.32	1,186,718,302.17
physical read total multi block requests	5,812,857	3,223.26	1.80	805.81	1,602.09	4.42	3,208.94
physical reads	761,298,139	422,144.29	235.95	105,536.07	25,732.71	91,203.97	144,102.13
physical reads cache	682,654,565	378,536.08	211.58	94,634.02	4,048.89	91,203.97	100,493.92
physical reads cache prefetch	14,459,423	8,017.81	4.48	2,004.45	4,008.20	0.34	8,016.75
physical reads direct	78,643,590	43,608.22	24.37	21,804.11	30,835.67	0.00	43,608.22
physical reads direct temporary tablespace	105	0.06	0.00	0.03	0.04	0.00	0.06
physical write IO requests	203,458,425	112,818.93	63.06	28,204.73	433.35	27,806.47	28,619.78
physical write bytes	1,684,028,514,304	933,804,043.95	521,931.45	233,451,010.99	3,591,914.53	230,163,310.74	236,893,471.64
physical write total IO requests	210,340,555	116,635.12	65.19	29,158.78	450.66	28,717.67	29,596.41
physical write total bytes	1,754,065,664,512	972,640,069.48	543,638.08	243,160,017.37	3,727,048.88	239,736,508.56	246,750,106.53
physical write total multi block requests	252	0.14	0.00	0.03	0.01	0.03	0.05
physical writes	205,569,887	113,989.75	63.71	28,497.44	438.47	28,096.11	28,917.66

図 7.　テスト ケース 2AWR IOPS の結果

図 8 は、テスト ケース 2 の Oracle AWR レスポンス タイムを示しています。データ ファイルの読み取りレスポンス タイムは平均して 490.2 マイクロ秒（0.49 ミリ秒）、REDO ログ ライターのレスポンス タイムは 724.4 マイクロ秒（0.72 ミリ秒）でした。

Top Timed Events

	Wait		Event		Wait Time			Summary Avg Wait Time				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg Wait	%DB time	Avg	Min	Max	Std Dev	Cnt
*	User I/O	db file sequential read	664,091,759	0.00	325,548.94	490.22us	94.14	490.28us	483.69us	497.49us	6.29us	4
*		DB CPU			37,170.34		10.75					4
*	System I/O	log file parallel write	3,523,513	0.00	2,552.46	724.41us	0.74	724.12us	709.89us	741.21us	14.55us	4
*	Cluster	gc buffer busy release	1,738	0.00	1,377.56	792.61ms	0.40	769.47ms	703.72ms	836.49ms	57.64ms	4
*	Cluster	gc cr grant 2-way	8,804,855	0.00	862.15	97.92us	0.25	98.06us	96.22us	100.09us	1.65us	4
*	User I/O	db file scattered read	1,036,180	0.00	639.74	617.40us	0.18	601.30us	531.25us	655.47us	51.97us	4
*	System I/O	db file parallel write	7,124,953	0.00	633.91	88.97us	0.18	89.11us	86.70us	91.53us	1.97us	4
*	User I/O	read by other session	1,218,222	0.00	604.38	496.12us	0.17	496.19us	486.26us	504.68us	7.57us	4
*	User I/O	direct path read	578,371	0.00	313.30	541.69us	0.09	541.69us	541.69us	541.69us		4
*	Cluster	gc cr multi block grant	799,603	0.00	204.72	256.03us	0.06	203.34us	176.92us	256.05us	36.90us	4

図 8.　テスト ケース 2 AWR レスポンス タイムの結果

図 9 は、Oracle データ ファイル（data_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。平坦な線は、ワークロードが非常に安定していることを示しています。このテストでは、0.46 ミリ秒の読み取りレスポンス タイムと 0.68 ミリ秒の書き込みレスポンス タイムによって、481,349 IOPS が生成されました。

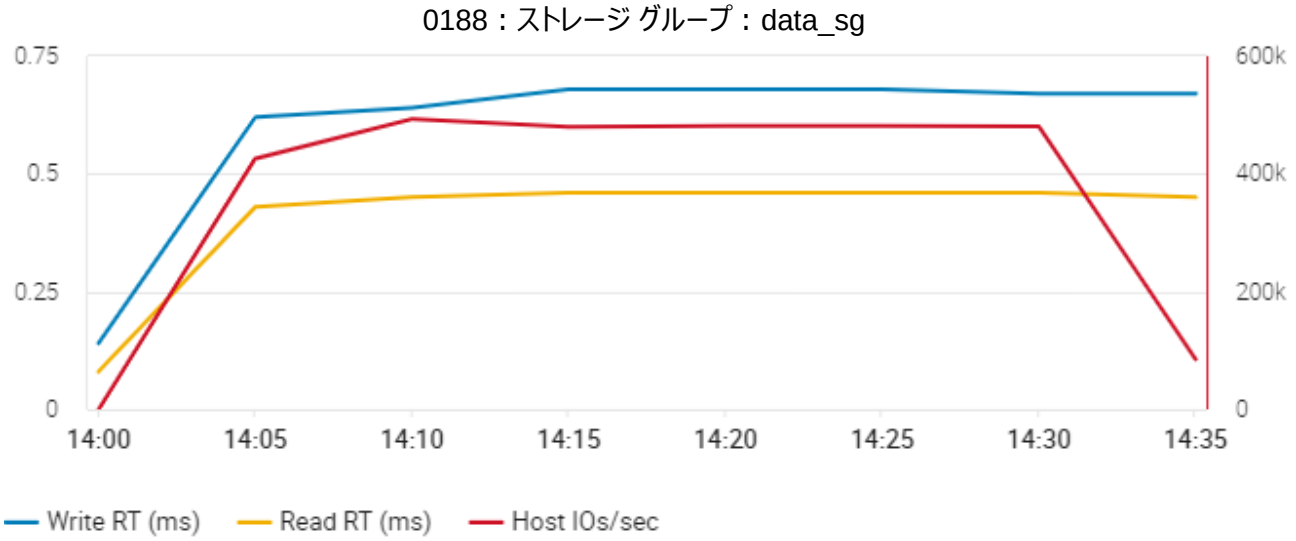

図 9.　テスト ケース 2Unisphere メトリック：Oracle データ ファイルのパフォーマンス

図 10 は、Oracle REDO ログ ファイル（redo_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。このテストでは、0.45 ミリ秒の書き込みレスポンス タイムによって 2,254 IOPS が生成されました。

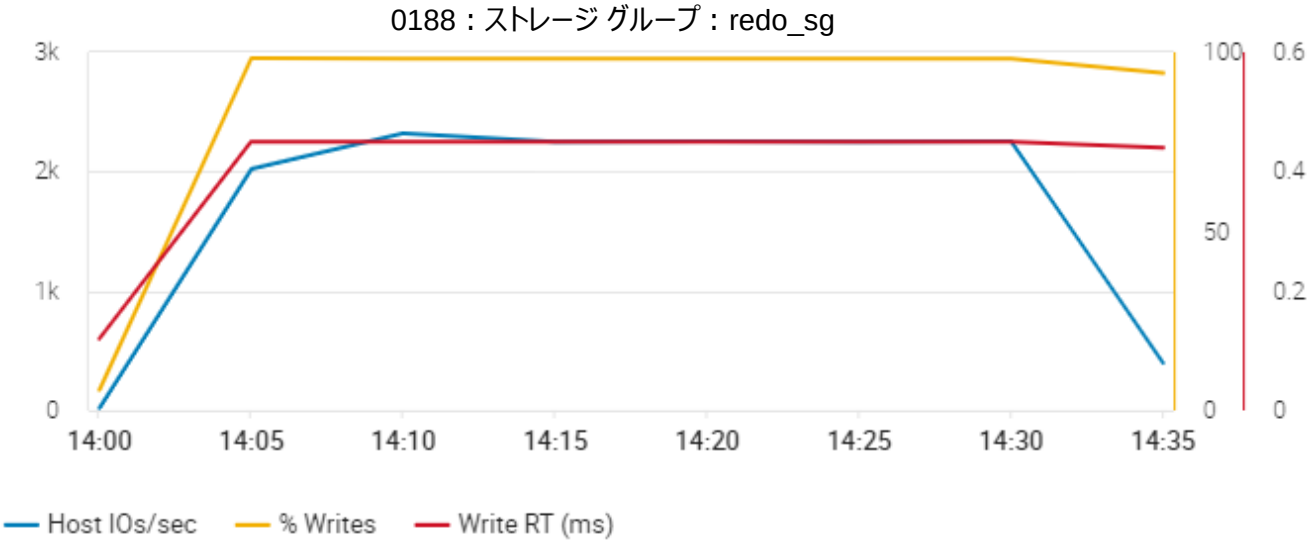

図 10.　テスト ケース 2Unisphere メトリック：Oracle REDO ログのパフォーマンス

要約すると、より現実的なワークロード動作により、Dell のラボ環境では、0.5 ミリ秒未満の読み取りレスポンス タイムで 50 万近くの IOPS を生成することができました。この高トランザクション レートによって、REDO ログの書き込みレイテンシーがわずかに 0.7 ミリ秒まで上昇しましたが、それでもかなり良好です。

テスト ケース 360%のキャッシュ読み取りヒットと低レスポンス タイムの OLTP

テスト ケース 3 では、データベースの読み取りレスポンス タイムを低く保つことに重点を置きました。この目標を達成するため、一部のアプリケーションでは高 IOPS よりも低レイテンシーに重きが置かれることを踏まえ、データベースの負荷を軽減しました。

図 11 は、テスト ケース 3 の Oracle AWR IOPS を示しています。テスト結果によると、平均してデータベースの読み取り IOPS は 212,548、書き込み IOPS は 67,426 であり、合計で 279,973 IOPS でした。

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
…							
physical read IO requests	383,256,402	212,511.17	201.54	53,127.79	326.52	52,672.97	53,435.83
physical read bytes	3,140,162,699,264	1,741,183,292.71	1,651,267.57	435,295,823.18	2,678,782.29	431,564,476.73	437,827,063.79
physical read total IO requests	383,322,056	212,547.57	201.57	53,136.89	326.07	52,682.43	53,443.84
physical read total bytes	3,165,658,295,808	1,755,320,301.36	1,664,674.54	438,830,075.34	2,392,150.69	435,348,495.54	440,597,511.94
physical read total multi block requests	26,255	14.56	0.01	3.64	0.46	2.94	3.92
physical reads	383,320,642	212,546.79	201.57	53,136.70	327.00	52,681.21	53,445.69
physical reads cache	383,317,806	212,545.22	201.57	53,136.30	327.00	52,680.82	53,445.29
physical reads cache prefetch	64,079	35.53	0.03	8.88	1.06	7.83	10.00
physical reads direct	2,836	1.57	0.00	0.39	0.00	0.39	0.40
physical write IO requests	117,759,793	65,296.42	61.92	16,324.11	134.94	16,128.14	16,431.67
physical write bytes	974,915,870,720	540,579,386.00	512,663.55	135,144,846.50	1,134,738.15	133,497,454.09	136,049,344.12
physical write total IO requests	121,600,093	67,425.82	63.94	16,856.46	135.27	16,659.96	16,962.79
physical write total bytes	1,015,920,395,264	563,315,912.68	534,225.95	140,828,978.17	1,177,688.55	139,120,126.89	141,769,609.70
physical write total multi block requests	227	0.13	0.00	0.03	0.01	0.02	0.05
physical writes	119,008,285	65,988.69	62.58	16,497.17	138.52	16,296.08	16,607.59
…							

図 11. テスト ケース 3AWR IOPS の結果

図 12 は、テスト ケース 3 の Oracle AWR レスポンス タイムを示しています。データ ファイルの読み取りレスポンス タイムは平均して 232.0 マイクロ秒（0.23 ミリ秒）、REDO ログ ライターのレスポンス タイムは 308.6 マイクロ秒（0.31 ミリ秒）でした。

Top Timed Events

	Wait		Event		Wait Time			Summary Avg Wait Time				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg Wait	%DB time	Avg	Min	Max	Std Dev	Cnt
*	User I/O	db file sequential read	382,159,894	0.00	88,659.55	232.00us	87.38	232.02us	228.22us	238.25us	4.38us	4
*		DB CPU			24,733.33		24.38					4
*	System I/O	log file parallel write	2,029,282	0.00	626.28	308.62us	0.62	308.59us	300.46us	315.36us	6.14us	4
*	Cluster	gc cr grant 2-way	3,841,528	0.00	460.51	119.88us	0.45	119.83us	116.10us	123.00us	2.89us	4
*	System I/O	db file parallel write	1,442,732	0.00	202.68	140.48us	0.20	153.70us	94.43us	199.86us	45.96us	4
*	Cluster	gc buffer busy release	162	0.00	91.66	565.81ms	0.09	544.16ms	403.88ms	740.73ms	147.01ms	4
*	User I/O	read by other session	223,091	0.00	63.53	284.75us	0.06	283.47us	253.10us	358.93us	50.46us	4
*	User I/O	ASM IO for non-blocking poll	1,976,676	0.00	38.89	19.67us	0.04	22.53us	12.82us	27.12us	6.72us	4
*	Configuration	enq: HW - contention	347	0.00	37.93	109.30ms	0.04	46.37ms	30.00us	149.23ms	69.30ms	4
*	Concurrency	buffer busy waits	2,441	0.00	27.78	11.38ms	0.03	8.34ms	14.46us	32.90ms	16.37ms	4

図 12. テスト ケース 3 AWR レスポンス タイムの結果

図 13 は、Oracle データ ファイル（data_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。平坦な線は、ワークロードが非常に安定していることを示しています。このテストでは、

0.14 秒の読み取りレスポンス タイムと 0.15 ミリ秒の書き込みレスポンス タイムによって、279,116 IOPS が生成されました。

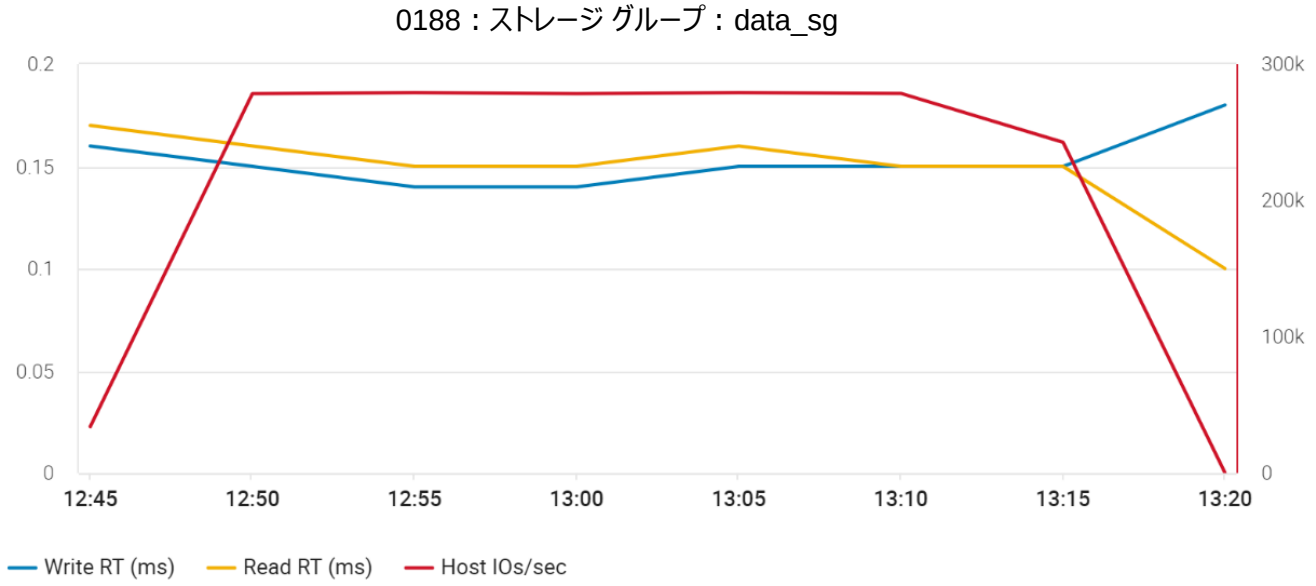

図 13.　テスト ケース 3Unisphere メトリック：Oracle データ ファイルのパフォーマンス

図 14 は、Oracle REDO ログ ファイル（redo_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。 このテストでは、0.13 ミリ秒の書き込みレスポンス タイムによって 1,317 IOPS が生成されました。

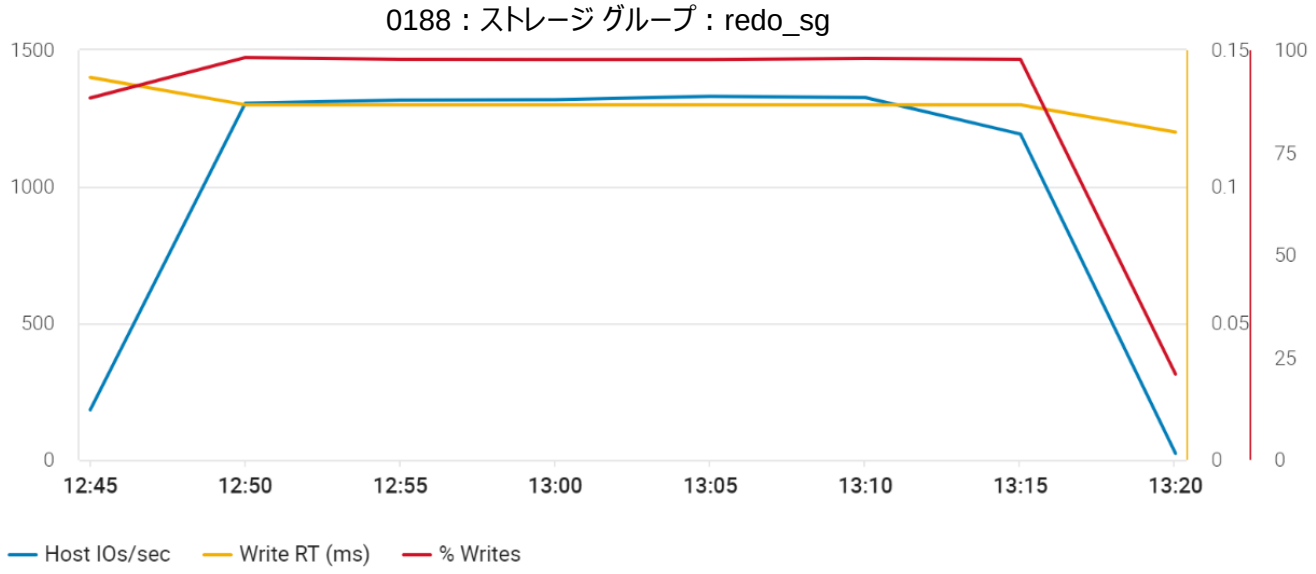

図 14.　テスト ケース 3Unisphere メトリック：Oracle REDO ログのパフォーマンス

要約すると、非常に低いレイテンシーを実現するには、システムの使用率を中程度に保ち、高くなりすぎないようにする必要があります（IOPS が増加すると、レイテンシーも増加します）。テスト ケース 3 では、テスト ケース 2（489,934 IOPS）の約 60%に相当する 279,073 IOPS が生成されました。システムをスケールアップする（サーバーとブリックを追加する）ことで、より高い IOPS を達成し、レイテンシーを低く保つことができます。

DSS パフォーマンス テスト ケース

DSS テストの概要と結果のサマリー

意思決定支援システム（DSS）のテスト ケースは、データ ウェアハウスや分析クエリで使用されるものと同様に、シーケンシャルな読み取りを処理するための PowerMax 機能を示しています。OLTP テストとは異なり、DSS テストでは、IOPS やレイテンシーではなく、帯域幅（GB/秒）に重点が置かれています。帯域幅が広いほど、レポートの実行が速くなります。

ここでは、TPC-H ツールの dbgen ツールキットを使用して、日付によるプライマリー パーティションとセカンダリー ハッシュ パーティションのある Lineitem テーブルに約 1 TB のデータを生成しました。フル テーブル スキャンを強制実行するために、SQL クエリでヒントを使用し、実行計画をレビューして確定しました。各テストが安定した実行状態で 30 分継続するように、遅延のないループでクエリを実行しました。このシナリオでは、同じビジネス データをさまざまな角度から見るため、多数のユーザーによって同様の分析クエリが実行されるデータベースをシミュレートします。

テストでは、データベース マルチブロック読み取り I/O サイズとして 128 KB を使用しました。シーケンシャルな読み取りの場合のマルチブロック読み取り I/O サイズの設定の詳細については、ベスト プラクティスを示したセクション「Oracle シーケンシャル読み取りの I/O サイズ」を参照してください。

2 つの DSS テスト ケースを実行しました。最初のテスト ケースは、スキャンされるデータセットが PowerMax キャッシュよりもはるかに大きくなる大規模データ ウェアハウスに典型的な状況を表しており、読み取りヒットの少ないワークロードになります。

2 番目のテスト ケースは、読み取りヒット ワークロードのメリットを示しています。このテストは、Oracle パーティション プルーニング機能を使用して実行されました。この機能により、データセット全体ではなくデータセットのサブセット内でデータを検索するように SQL クエリが最適化されます。パーティション プルーニングを使用した場合、クエリのデータセットは大幅に小さくなって PowerMax キャッシュに十分に収まり、より高い帯域幅を実現しました。Oracle クエリを最適化することが常に可能であるとは限りませんが、結果が示すとおり、最適化によるメリットがあることは確かです。

次の表にテスト ケースと AWR の結果をまとめています。

表 3. DSS のパフォーマンス テスト ケースと結果のサマリー

テスト ケース	テストの詳細	キャッシュ読み取りヒット率	MBRC	データ ファイルの読み取り（GB/秒）
1	フルテーブル スキャン、パーティション プルーニングなし	31	16（128 KB）	11.8
2	フルテーブル スキャン、パーティション プルーニング使用	100	16（128 KB）	29.7

テスト ケース 1DSS、フルテーブル スキャン、パーティション プルーニングなし

テスト ケース 1 の目的は、クエリの最適化を行わずに、大規模な Lineitem テーブルのフルテーブル スキャンを実行することでした。このテスト ケースは「最悪のシナリオ」を表していますが、分析とデータ ウェアハウス クエリがどのように構築されるかを予測し、Oracle Optimizer がパーティション プルーニングを利用できるかどう

かを予測することは困難です。このことは、ユーザーがビジネス インテリジェンス（BI）ツールを使用して、クエリの実行用に最適化されていない「アドホック」クエリを作成する場合に特に当てはまります。

図 15 は、テスト ケース 1 の Oracle AWR 帯域幅を示しています。レポートによると、データ ファイルの読み取り帯域幅は平均して 12,703,573,103 バイト/秒、つまり 11.8 GB/秒でした。

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
…							
physical read IO requests	240,894,031	97,171.64	1,771,279.64	24,292.91	1,529.77	22,261.04	25,720.01
physical read bytes	31,492,251,025,408	12,703,318,426.61	231,560,669,304.47	3,175,829,606.65	200,111,643.82	2,910,235,122.76	3,362,633,040.40
physical read total IO requests	240,932,570	97,187.18	1,771,563.01	24,296.80	1,529.82	22,264.91	25,724.05
physical read total bytes	31,492,882,382,848	12,703,573,102.95	231,565,311,638.59	3,175,893,275.74	200,112,426.53	2,910,298,528.17	3,362,699,228.70
physical read total multi block requests	238,578,557	96,237.62	1,754,254.10	24,059.41	1,519.81	22,050.07	25,483.15
physical reads	3,844,268,924	1,550,698.05	28,266,683.26	387,674.51	24,427.69	355,253.31	410,477.67
physical reads cache	20,671	8.34	151.99	2.08	1.84	0.47	4.69
physical reads cache prefetch	12,152	4.90	89.35	1.63	1.54	0.31	3.33
physical reads direct	3,844,248,253	1,550,689.71	28,266,531.27	387,672.43	24,429.42	355,248.62	410,476.47
…							

図 15.　テスト ケース 1AWR 帯域幅の結果

図 16 は、Oracle データ ファイル（tpch_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。平坦な線は、ワークロードが非常に安定していることを示しています。レポートによると、Oracle AWR について報告された帯域幅と非常によく似た 12,135 MB/秒（11.85 GB/秒）となっています。また、テストで MBRC に 16 を使用した結果、平均読み取り I/O サイズは 128 KB でした。さらに、読み取りレスポンス タイムは 1.7 ミリ秒であり、これは 128KB の平均読み取り I/O サイズに基づいています。

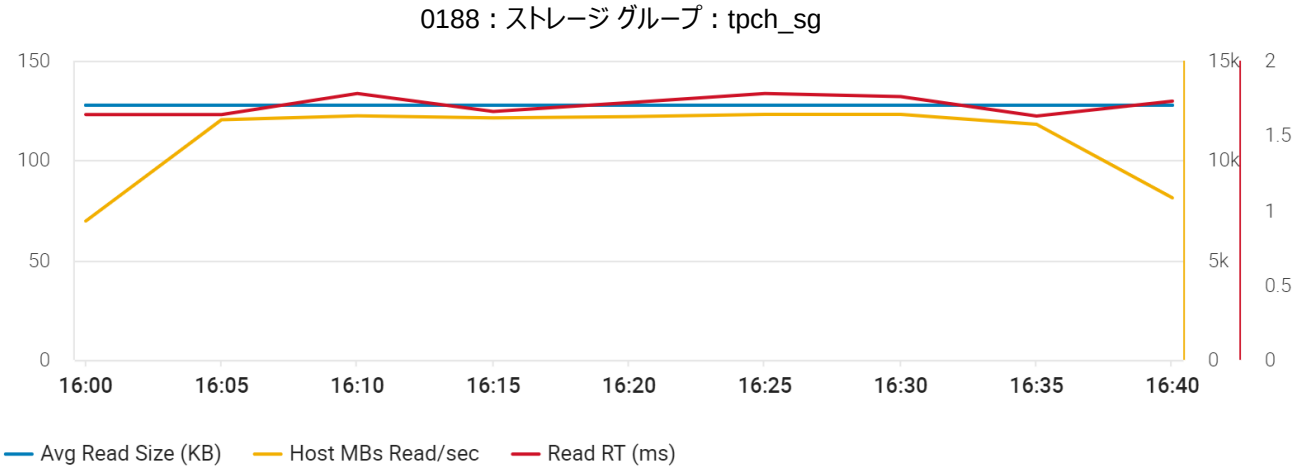

図 16.　テスト ケース 1Unisphere メトリック：Oracle データ ファイルのパフォーマンス

要約すると、テスト ケース 1 は、データセットが大きいために読み取りヒットの少ないワークロード（31%）ですが、単一の PowerMax ブリックを備えた小規模構成を考慮しても、Oracle で 11.8 GB/秒の帯域幅を実現しました。

テスト ケース 2DSS、フルテーブル スキャン、パーティション プルーニング使用

テスト ケース 2 では、大規模 Lineitem テーブルに対して同じフルテーブル スキャンを実行しましたが、今回は、SQL 構文で WHERE 句を使用して、必要なデータをフェッチするためにスキャンする必要の

あるパーティションの数を Oracle が削減できるようにしました（Oracle でパーティション プルーニングと呼ばれる最適化）。クエリはフェッチするデータ量を制限することにより、PowerMax キャッシュのメリットを享受し、完了までの時間が短縮します。

図 17 は、テスト ケース 2 の Oracle AWR 帯域幅を示しています。レポートによると、データ ファイルの読み取り帯域幅は 31,114,539,806 バイト/秒、つまり 29.7 GB/秒でした。

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
…							
physical read IO requests	437,447,745	238,089.05	1,792,818.63	59,522.26	7,005.68	50,474.42	66,536.75
physical read bytes	57,167,123,570,688	31,114,268,114.52	234,291,490,043.80	7,778,567,028.63	916,605,237.69	6,594,339,390.52	8,695,067,021.64
physical read total IO requests	437,477,058	238,105.01	1,792,938.76	59,526.25	7,005.61	50,478.53	66,540.69
physical read total bytes	57,167,622,756,864	31,114,539,805.86	234,293,535,888.79	7,778,634,951.47	916,605,885.49	6,594,406,769.62	8,695,131,573.96
physical read total multi block requests	432,910,419	235,619.53	1,774,223.03	58,904.88	6,980.31	49,870.47	65,837.88
physical reads	6,978,408,639	3,798,128.43	28,600,035.41	949,532.11	111,890.29	804,973.07	1,061,409.55
physical reads cache	6,555	3.57	26.86	0.89	1.57	0.10	3.24
physical reads cache prefetch	3,871	2.11	15.86	1.05	1.49	0.00	2.10
physical reads direct	6,978,402,084	3,798,124.86	28,600,008.54	949,531.22	111,890.54	804,972.95	1,061,409.45
…							

図 17.　テスト ケース 2AWR 帯域幅の結果

図 18 は、Oracle データ ファイル（tpch_sg ストレージ グループ）の Unisphere パフォーマンス メトリックを示しています。平坦な線は、ワークロードが非常に安定していることを示しています。レポートによると、Oracle AWR について報告された帯域幅と非常によく似たデータ読み取り帯域幅 29,832 MB/秒（29.13 GB/秒）となっています。また、テストで MBRC に 16 を使用した結果、平均読み取り I/O サイズは 128 KB でした。さらに、読み取りレスポンス タイムは、I/O サイズが 128 KB の場合でも 0.3 ミリ秒です。 この結果は、データが PowerMax にすでにキャッシュされているためです。

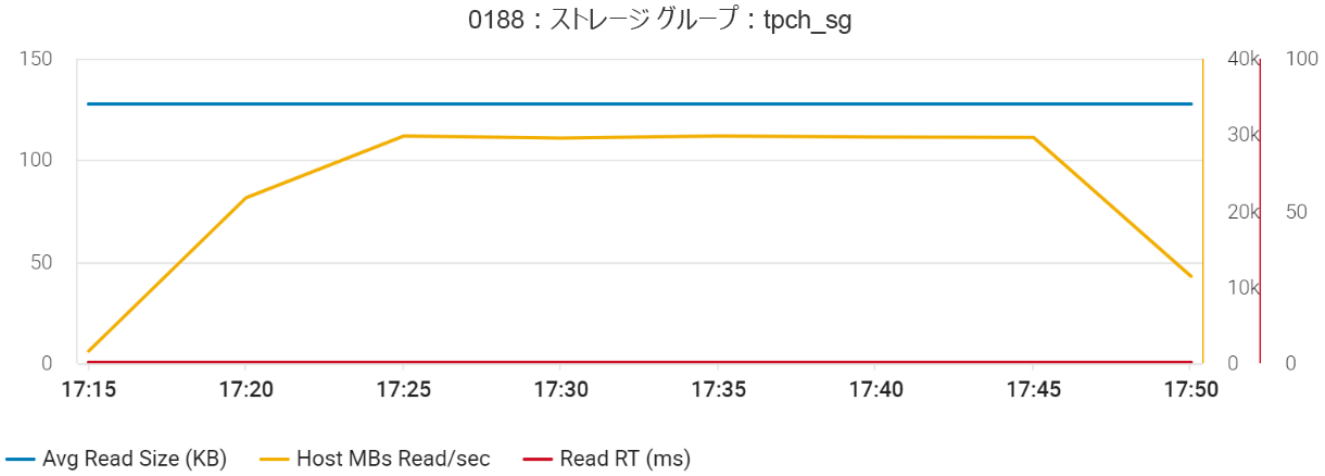

図 18.　テスト ケース 2Unisphere メトリック：Oracle データ ファイルのパフォーマンス

要約すると、テスト ケース 2 では、パーティション プルーニングのメリットがクエリにもたらされます。その結果、クエリによって要求されるデータの量が減少し、PowerMax キャッシュがクエリのパフォーマンスを向上させる機会が得られます。

圧縮パフォーマンス テスト

このセクションでは、ストレージ グループの圧縮の有無にかかわらず、Oracle ワークロードの優れたパフォーマンスを維持する PowerMax の機能を示します。

「PowerMax の圧縮と重複排除」で説明したように、Adaptive Compression Engine（ACE）では、圧縮するようマークされたストレージ グループに最もアクティブなデータ エクステントが属している場合でも、圧縮をすぐに実行しません。ACE では、割り当てられたストレージ容量の最もアクティブな 20%を展開した状態で維持します（一方でストレージ スペースは許容されます）。通常、最も頻繁にアクセスされるのは、最新のデータです。時間の経過とともに、新しいデータが書き込まれて頻繁にアクセスされるようになります。以前は「ビジー」とみなされたものが、あまりアクティブではなくなり、自動的に圧縮されます。

この方法は、データベースの実際のアクセス パターンに適用されるものですが、ベンチマーク ツールではこれを無視し、データベース全体でランダムに実行する傾向があります。SLOB の「ホットスポット」機能を使用すると、各ユーザー テーブルの一部にさらに頻繁にアクセスするようにして、実際の環境における動作をシミュレートすることができます。

PowerMax の圧縮テストを可能な限り現実に即したものにするために、SLOB をセミランダム データと併せてロードし、3.0:1 の圧縮率にしました。5 GB のバッファ キャッシュと SLOB のホットスポットを使用しました。この構成によるワークロードは、ストレージの読み取り I/O が 80%で、キャッシュの読み取りヒット率が 60%になりました。そのため、ストレージに送信された I/O 要求の 80%が読み取りとなり、OLTP タイプのワークロードを作成しただけでなく、圧縮される可能性があったデータに多くの要求がありました。40%の読み取りミスというのは、すべての読み取りのうち、少なくとも 40%のデータが PowerMax キャッシュに見つからず、（圧縮または展開されている）フラッシュ メディアからデータを取得する必要があることになります。

SLOB ワークロードの実行には、クラスター内の Dell サーバー2 台を使用しました。

次の表には、圧縮を無効にした場合の Oracle AWR から取得したテスト結果を示しています。「上位の時間計測イベント」では、AWR のデータ ファイル読み取りレイテンシーとして 0.28 ミリ秒が報告されました（`db file sequential read` 指標）。システム統計（グローバル）では、データ ファイルの合計 IOPS は 253,477 回（読み取り 184,270 回 + 書き込み 69,207 回）となっています。

Top Timed Events

	Wait		Event		Wait Time			Summary Avg Wait Time				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg Wait	%DB time	Avg	Min	Max	Std Dev	Cnt
*	User I/O	db file sequential read	330,518,461	0.00	93,995.64	284.39us	91.33	284.39us	283.83us	284.96us	800.63ns	2
*		DB CPU			17,645.57		17.14					2
*	System I/O	log file parallel write	2,879,516	0.00	1,244.34	432.14us	1.21	432.14us	431.63us	432.65us	724.31ns	2
*	System I/O	db file parallel write	2,794,976	0.00	148.83	53.25us	0.14	53.25us	53.00us	53.50us	352.55ns	2
*	Other	RMA: IPC0 completion sync	3,737	0.00	72.60	19.43ms	0.07	19.43ms	19.42ms	19.44ms	10.75us	2
*	User I/O	read by other session	117,935	0.00	36.27	307.54us	0.04	307.55us	306.00us	309.09us	2.18us	2
*	Other	LGWR any worker group	184,003	0.00	35.81	194.63us	0.03	194.65us	194.02us	195.28us	895.50ns	2
*	Other	LGWR worker group ordering	57,615	0.00	9.99	173.36us	0.01	173.36us	172.88us	173.85us	686.49ns	2
*	Application	enq: TX - row lock contention	709	0.00	4.99	7.04ms	0.00	7.04ms	6.85ms	7.22ms	264.03us	2
*	System I/O	control file sequential read	20,670	0.00	3.99	192.83us	0.00	192.83us	192.40us	193.26us	606.92ns	2

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
...							
physical read IO requests	331,383,685	184,240.69	111.74	92,120.34	1,491.23	91,065.88	93,174.80
physical read bytes	2,714,833,543,168	1,509,376,668.05	915,426.45	754,688,334.03	12,204,039.86	746,058,774.68	763,317,893.37
physical read total IO requests	331,437,622	184,270.68	111.76	92,135.34	1,491.02	91,081.03	93,189.65
physical read total bytes	2,748,847,708,672	1,528,287,657.14	926,895.84	764,143,828.57	12,090,945.49	755,594,239.03	772,693,418.11
physical read total multi block requests	32,882	18.28	0.01	9.14	0.18	9.02	9.27
physical reads	331,400,582	184,250.08	111.75	92,125.04	1,489.75	91,071.63	93,178.45
physical reads cache	331,400,580	184,250.08	111.75	92,125.04	1,489.75	91,071.63	93,178.45
physical reads cache prefetch	16,914	9.40	0.01	4.70	1.48	3.65	5.75
physical write IO requests	118,930,926	66,122.49	40.10	33,061.25	329.50	32,828.26	33,294.24
physical write bytes	994,186,862,592	552,741,974.33	335,234.16	276,370,987.16	3,105,245.41	274,175,247.08	278,566,727.25
physical write total IO requests	124,479,110	69,207.14	41.97	34,603.57	353.28	34,353.76	34,853.38
physical write total bytes	1,056,540,968,960	587,409,231.75	356,259.61	293,704,615.87	3,370,199.62	291,321,524.87	296,087,706.88

図 19.　ストレージ グループの圧縮が無効となっている AWR 統計表

次の図は、圧縮が有効な場合のテスト結果（Oracle AWR から取得）を示しています。「上位の時間計測イベント」では、AWR のデータ ファイル読み取りレイテンシーとして 0.31 ミリ秒が報告されました（`db file sequential read` 指標）。システム統計（グローバル）では、データ ファイルの合計 IOPS は 250,743 回（読み取り 181,296 回 + 書き込み 69,447 回）となっています。

PowerMax ストレージ システムの圧縮を有効にしている場合と無効にしている場合の 2 つの AWR レポートでは、Oracle データ ファイルの合計 IOPS に約 1%の違いがあり、データ ファイルの読み取りレスポンス タイムには、0.03 ミリ秒の違いがあります。このような違いをユーザーは認識しません。これは、ハイ パフォーマンスを維持しつつ、データ削減をサポートする PowerMax アーキテクチャの強みを示しています。

Top Timed Events

	Wait		Event		Wait Time			Summary Avg Wait Time				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg Wait	%DB time	Avg	Min	Max	Std Dev	Cnt
*	User I/O	db file sequential read	325,196,788	0.00	100,901.50	310.28us	92.11	310.30us	309.42us	311.18us	1.25us	2
		DB CPU			17,321.71		15.81					2
	System I/O	log file parallel write	2,591,175	0.00	1,128.66	435.58us	1.03	435.58us	434.94us	436.23us	911.07ns	2
	System I/O	db file parallel write	2,212,040	0.00	93.52	42.28us	0.09	42.75us	38.09us	47.42us	6.60us	2
	Other	RMA: IPC0 completion sync	3,756	0.00	72.92	19.41ms	0.07	19.41ms	19.40ms	19.42ms	14.87us	2
	User I/O	read by other session	123,383	0.00	40.96	331.99us	0.04	332.00us	331.70us	332.31us	431.06ns	2
	Other	LGWR any worker group	127,307	0.00	25.44	199.82us	0.02	199.83us	199.79us	199.86us	44.01ns	2
	Other	LGWR worker group ordering	46,589	0.00	8.23	176.59us	0.01	176.58us	175.98us	177.17us	844.21ns	2
	Application	enq: TX - row lock contention	761	0.00	7.23	9.50ms	0.01	9.56ms	7.75ms	11.36ms	2.55ms	2
	System I/O	control file sequential read	20,764	0.00	4.08	196.43us	0.00	196.44us	193.49us	199.39us	4.17us	2

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
...							
physical read IO requests	325,976,273	181,268.97	122.52	90,634.49	3,182.52	88,384.10	92,884.87
physical read bytes	2,670,499,569,664	1,485,012,115.26	1,003,728.73	742,506,057.63	26,058,874.29	724,079,650.91	760,932,464.35
physical read total IO requests	326,026,009	181,296.63	122.54	90,648.32	3,182.49	88,397.95	92,898.68
physical read total bytes	2,700,003,365,888	1,501,418,594.50	1,014,817.96	750,709,297.25	26,050,684.48	732,288,681.60	769,129,912.90
physical read total multi block requests	28,377	15.78	0.01	7.89	0.07	7.84	7.94
physical reads	325,988,719	181,275.89	122.53	90,637.95	3,181.01	88,388.63	92,887.26
physical reads cache	325,988,718	181,275.89	122.53	90,637.95	3,181.02	88,388.63	92,887.26
physical reads cache prefetch	12,488	6.94	0.00	3.47	1.49	2.42	4.52
physical write IO requests	119,899,846	66,673.94	45.07	33,336.97	302.29	33,123.22	33,550.72
physical write bytes	1,001,077,669,888	556,679,553.42	376,263.09	278,339,776.71	2,852,370.10	276,322,846.47	280,356,706.95
physical write total IO requests	124,887,655	69,447.56	46.94	34,723.78	319.18	34,498.09	34,949.47
physical write total bytes	1,057,279,619,584	587,932,349.54	397,387.04	293,966,174.77	3,056,270.11	291,805,065.45	296,127,284.09

図 20.　ストレージ グループの圧縮が有効となっている上位の時間計測イベント

PowerMax による作業時のデータ削減

次の例では、Oracle データベースにおける、PowerMax の圧縮と重複排除に関する使用とメリットを説明します。最初の例では、暗号化されていない Oracle データベースの圧縮および重複排除のメリットについて説明します。2 番目の例では、データベースが包括的に暗号化された場合の変化について説明します。暗号化により、データが全面的にランダムに表示され、圧縮のメリットが妨げられます。

どちらの例でも、データベースまたはストレージによって報告される Oracle データ ファイルの消費容量のみを報告しました。REDO ログの容量は、比較的小さいため、グラフでは報告されません。

この例は、データがセミランダムになるように変更された SLOB データベースを基にしています。この例が示すように、PowerMax ストレージ システムよって圧縮されたセミランダム データベースの圧縮率は 3.1:1 となっており、これは Oracle データベースに対して予想されるおよその圧縮率です。

暗号化されていない Oracle データベースの圧縮および重複排除

この例では、Oracle データベース（青色の左側のバー）と PowerMax ストレージ システム（緑色の右側のバー）の両方から報告された Oracle データ ファイルの消費容量を確認します。Oracle データベースが作成されると、データ ファイル容量は約 1.35 TB（Oracle データベースによって報告）になりました。`data_sg` ストレージ グループ（SG）での圧縮が有効になっていたため、実際に消費されたストレージはわずか 450 GB（PowerMax によって報告）でした。つまりデータ削減率（DRR）は 3.1:1

です。Oracle データベースでは、すべてのデータ ブロックに（そのコンテンツを問わず）固有のヘッダーがあるため、単一データベース内での重複排除のメリットがありません。

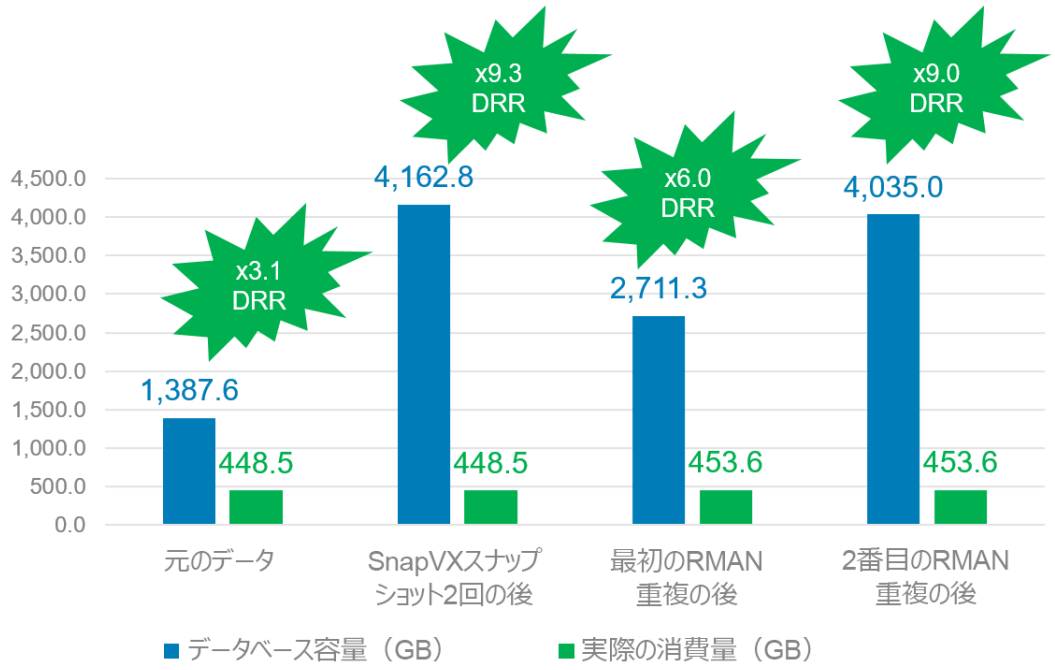

図 21.　例 1：PowerMax による Oracle データベースの圧縮と重複排除

次に、SnapVX スナップショットを 2 個作成し、それらを別のサーバーにリンク（提示）しました。スナップショットの作成とリンクにかかった時間はわずか数秒でした。その結果、元のデータベースのコピーが 3 個存在することになり、これらは約 4 TB（3 x 1.35 TB）の大きさでした。ストレージの検査時に容量は追加されておらず、DRR は 9.3:1 となりました。これは、データが変更された場合にのみ、PowerMax スナップショットによってストレージ容量が消費されるためです。その後、スナップショットを削除しました。

次に、RMAN の `DUPLICATE` コマンドを使用してデータベースのコピーを作成しました。RMAN によって、ターゲット データベース サーバーおよび ASM ディスク グループにソース データベースのバイナリー コピーが作成されました。RMAN では、ソース データベースのフル コピー作成にネットワークを使用したため、このプロセスには数時間かかりました。RMAN によるデータベース レベルでのデータベース クローン作成操作が完了すると、ソース データベースとクローン作成されたデータベースの合計容量が 2.64 TB になりました。ただし、ソースおよびターゲットのストレージ グループに関連づけられたストレージ容量は、わずか 450 GB で、DRR は 6.0:1 でした。

この結果の理由は、ASM アロケーション ユニット（AU）が Oracle 12.2 では 4 MB であり、以前のリリースでは 1 MB であるためです。128 KB の重複排除粒度を備えた PowerMax ストレージ システムにより、クローン作成されたデータベース エクステントがソースと同一であると識別され、全面的に重複排除されました。

最後に、RMAN の `DUPLICATE` コマンドを使用して 2 個目のデータベース コピーを作成しました。これで、ソース データベースと 2 つのコピーがあり、データベース レベルで合計 4 TB の容量になりました。ここでも、PowerMax ストレージ システムによって、データは全面的に重複排除され、3 個のデータベースに関連づけられたストレージ容量が 450 GB のまま維持され、DRR は 9:1 となりました。

暗号化された Oracle データベースの圧縮および重複排除

この例では、すべての Oracle データ ファイルの消費容量は、ストレージ システムから報告されています。一番左のバーは、データ ファイルのストレージ消費容量が約 1.35 TB であることを示しています。この例では、PowerMax ストレージ グループの圧縮は最初は有効になっていませんでした。

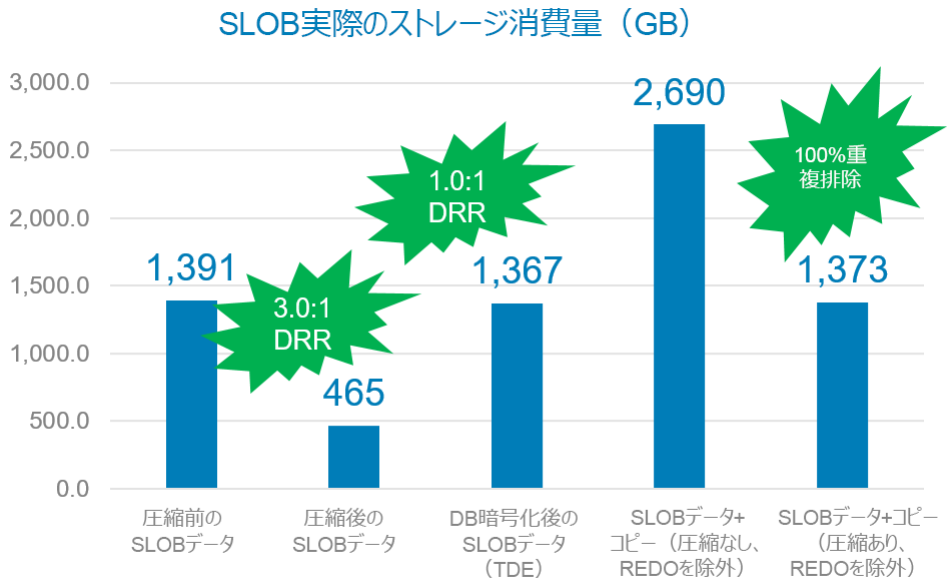

図 22.　例 2：PowerMax による暗号化された Oracle データベースの圧縮および重複排除

PowerMax の圧縮を `data_sg` に対して有効にし、バックグラウンド圧縮の完了を待ちました。プロセスの終了時、`data_sg` による消費量はわずか 465 GB となり、換算した DRR は 3.0:1 となっています。この結果は、圧縮がすでに有効になっている SG でデータベースを作成した場合の前述の例とそれほど違いがありません。

次に、すべてのテーブルスペースを暗号化するために、Oracle Transparent Database Encryption（TDE）を使用しました。DBA では特定のテーブル列またはいくつかのテーブルスペースのみを暗号化するよう選択できますが、私たちはデータベース全体を暗号化することの影響を確認したいと考えました。その結果、`data_sg` のストレージ消費量は、元のサイズである 1.35 TB まで増加しています。データベースの暗号化によってストレージ圧縮のメリットが相殺されていることがはっきりと分かります。

容量が追加されていないことをすでに確認しているため、ストレージ スナップショットは作成しませんでした。RMAN の `DUPLICATE` コマンドを使用してデータベースのクローンを作成しました。最初は、ターゲットのストレージ グループに対する圧縮が有効ではありませんでした。その結果、RMAN による操作完了後、ストレージの合計消費量が倍増しました。

最後に、ターゲットのストレージ グループで圧縮を有効にしました。PowerMax の重複排除により、100%の重複排除というメリットが再び実現し、ソースとターゲットの両方のストレージ グループを合わせたストレージ消費量が 1.35 TB に戻りました。

まとめ

PowerMax の圧縮が、Oracle データベースに対して非常に効率的であることが分かります。SLOB のセミランダム データベースでは、圧縮によって約 3:1 のデータ削減を達成しました。

SnapVX を使用してデータベースのコピーを作成する（推奨方法）と、操作が数秒で完了するうえ、容量の効率性を最大化するというメリットが実現します。

DBA が RMAN の DUPLICATE コマンドを使用してデータベースのクローンを作成する場合、データベース全体がネットワーク経由でコピーされるため、操作に時間がかかります。ただし、ASM AU の粒度が 1 MB または 4 MB であるため、PowerMax ストレージ システムでは、データを全面的に重複排除することができます。データがソース データベースに対する同一のバイナリー コピーであるためです。

CLI コマンドを使用したデータ削減管理

Unisphere を使用して新しいストレージ グループを作成する場合、PowerMax 圧縮はデフォルトで有効になっています。SG 作成ウィザードの圧縮チェックボックスをオフにすると、PowerMax 圧縮を無効にできます。Unisphere には、圧縮されたストレージ グループの圧縮率、圧縮されていないストレージ グループの潜在的な圧縮率などを示すビューとメトリックも含まれます。次のセクションでは、Solutions Enabler コマンド ライン インターフェイス（CLI）を使用して、これらの操作を実行する方法やデータ削減に関連した情報を表示する方法について説明します。

圧縮を有効にするには、ストレージ グループ（例にある data_sg）を PowerMax ストレージ アレイのストレージ リソース プール（SRP）に関連づける必要があります。圧縮を有効にして SRP を関連づけるには、次のコマンドを入力します。

```
# symsg -sg data_sg set -srp SRP_1 -compression
```

同様に、圧縮が有効になっているストレージ グループで圧縮を無効にするには、次のコマンドを入力します。

```
# symsg -sg data_sg set -srp SRP_1 -nocompression
```

ストレージ グループの圧縮率を表示するには、次のコマンドを入力します。

```
# symcfg list -tdev -sg data_sg -gb [-detail]
```

メモ：-detail オプションには、各圧縮プールのデータ アロケーションが含まれており、ここで排他的なアロケーションを確認できます。データが重複排除されていると、排他的なアロケーションは消費されません。

圧縮が無効となっている SG を含めたストレージ グループの推定圧縮率を表示するには、次のコマンドを入力します。

```
# symcfg list -sg_compression -by_compressibility -all
```

システム全体の効率性を表示するには、次のコマンドを入力します。

```
# symcfg list -efficiency -detail
```

PowerMax の Adaptive Compression Engine と重複排除の詳細については、『Data Reduction with Dell EMC PowerMax』を参照してください。

PowerMax のサービス レベル

サービス レベルの概要

PowerMax ストレージ システムなど、大容量でパワフルな NVMe フラッシュ ストレージでは多くの場合、多数のデータベースとアプリケーションが 1 個のストレージ システムに統合されています。PowerMax ストレージ システムは、サービス レベル（SL）を使用して、SL に従ってストレージ グループ（SG）の I/O レイテンシーを管理することにより、アプリケーションのパフォーマンス目標と優先順位を決定します。

デフォルトでは、PowerMax ストレージ システムによって、新しい SG に最適化された SL が割り当てられます。この SL は、システムによって提供可能な最良のパフォーマンスを取得しますが、最適化された SL によっても設定されている他のすべての SG と同じ優先度となります。このケースでは、ある SG（補助アプリケーションなど）からの急激な高負荷が、別の SG（重要でミッションクリティカルなアプリケーションなど）のパフォーマンスに影響を与える可能性があります。これは、システムの優先度とパフォーマンスの目標をすべて共有しているためです。個別の SL を使用することで、このような状況を回避できます。

SL のユース ケースには、「ノイジー ネイバー」のパフォーマンスを「ケージング」すること、テスト/開発システムよりも本番環境システムのパフォーマンスを優先させること、およびサービス プロバイダーや組織による「チャージバック」（クライアントがサービス レベルに対して支払いを行うシステム）を使用するうえでのニーズを満たすことが含まれます。

サービス レベルの仕組み

サービス レベルと目標レスポンス タイム

次の表は、サービス レベルの優先度とそれに関連するパフォーマンス目標を示しています。SL を使用すると、重要なシステムには Diamond、Platinum、または Gold などの優先度の高い SL を割り当て、Silver、Bronze、または Optimized などの優先度の低い SL を持つアプリケーションよりも優先度の高いパフォーマンス目標にすることができます。

また、一部の SL はレスポンス タイム制限が低くなっており、SG に割り当てることで、読み取りと書き込みのレイテンシーがその制限を下回らないようになります。SCM ドライブを使用すると、Diamond SL と Platinum SL の目標レスポンス タイムが短くなることにも注意してください。

表 4.　サービス レベルの優先度と制限

サービス レベル	SCM を使用しない目標レスポンス タイム（ミリ秒）	SCM を使用する目標レスポンス タイム（ミリ秒）	最短のレスポンス タイム（ミリ秒）
Diamond	0.6	0.4	該当なし
Platinum	0.8	0.6	該当なし
Gold	1.0	該当なし	該当なし
Silver	3.6	該当なし	約 3.6
Bronze	7.2	該当なし	約 7.2
最適化	該当なし	該当なし	該当なし

他の SL とは異なり、Optimized には特定のパフォーマンス目標がありません。Optimized ではない SL を持つ SG において、パフォーマンス目標の維持が困難な場合は、独自の目標を維持しようとする Optimized の SL を使用して SG にレイテンシーを付加することができます。

同様に、Optimized ではない SL を持つ SG において、パフォーマンス目標の維持が困難な場合は、優先度の低い SL が設定されている SG に対してレイテンシーを付加することができます。たとえば、Diamond の SG は Platinum の SG に影響を与え、Platinum の SG は Gold SG に影響を与えることができます。

サービス レベルと SCM ドライブ

PowerMaxOS Q3 2019 リリースでは、既存の NAND SSD フラッシュ ドライブに加えて、ストレージ クラス メモリー（SCM）の PowerMax サポートが導入されました。さまざまなドライブ テクノロジーのサポートにより、自動データ配置（ADP）と呼ばれる階層型ストレージが導入されています。ADP の機能は、サービス レベル（SL）の一部として含まれており、機械学習を使用して適切なドライブ タイプにデータを配置します。SL によって管理されるレスポンス タイムを改善するために、ADP は最もアクティブなデータをより高速なドライブ テクノロジーに配置します。

ADP の移動は、データの昇格またはデータの降格のいずれかとして発生します。昇格は SCM ドライブへのデータの移動であり、降格は SCM ドライブからのデータの移動です。PowerMaxOS は、SL の優先度とアクティビティ メトリックを使用して、昇格と降格を決定します。基本的な戦略については、「表 5」で説明されています。

表 5.　SCM は、サービス レベルに基づいて昇格および降格戦略を推進します

サービス レベル	優先度	詳細
プロモーション		
Diamond	最上位の昇格優先度	最適な使用率が保たれている間、PowerMax OS は Diamond SL を使用してすべてのデータを SCM ドライブに配置しようとします
Platinum、Gold、Optimized	すべてのデータが同じ優先度	
Silver、Bronze	昇格から除外	
降格		
Silver、Bronze	最上位の降格優先度	
Platinum、Gold、Optimized	同等の降格優先度	降格が行われるのは、優先度の高いデータまたはアクティビティの多い同じ優先度のデータ用に使用可能なスペースを SCM に作成する必要がある場合です。
Diamond		アクティビティの多い Diamond SL データが他にある場合、データは降格されます

PowerMax サービス レベルの詳細については、『Dell EMC PowerMax：PowerMaxOS のサービス レベルに関するテクニカル ホワイト ペーパー』を参照してください。

単一のデータベース ワークロードを持つサービス レベルの例

次の図は、Oracle データベースの単一ワークロードにおけるサービス レベルの一般的な影響を示しています。このテストでは、単一の OLTP ワークロードが中断または変更されることなく実行されました。data_sg SL のみが 30 分ごとに変更されました。

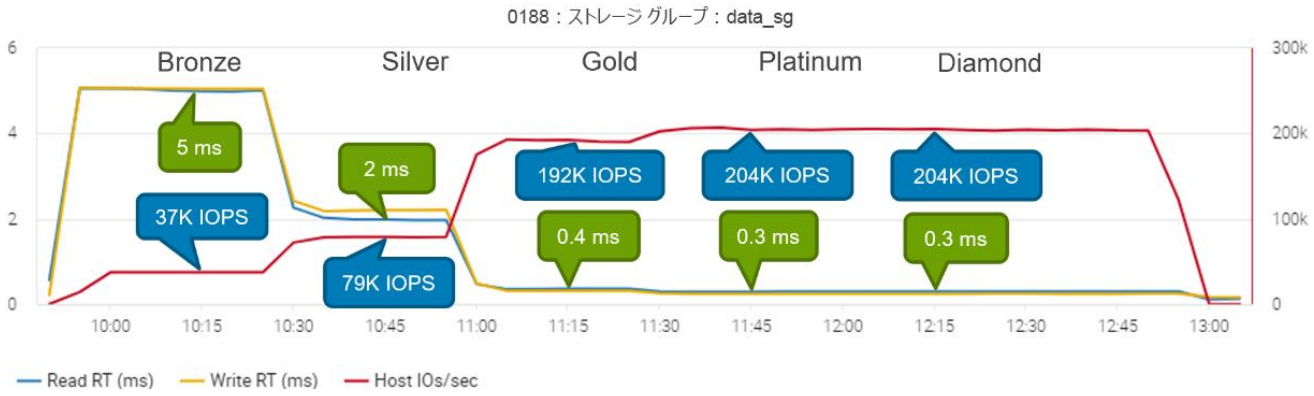

図 23.　単一の Oracle ワークロードにおけるサービス レベルの変更

Bronze SL では、平均 5 ミリ秒のレイテンシーと 37,000 IOPS のパフォーマンス レベルが適用されています。SL が Silver に変更されると、レイテンシーは 2 ミリ秒まで下がり、IOPS が 79,000 に増加しました。Gold の SL では、レイテンシーが 0.4 ミリ秒に短縮され、IOPS が 192,000 に増加しました。Platinum SL と Diamond SL は、どちらも 0.3 ミリ秒のレイテンシーと 204,000 IOPS で実行されたため、大きな違いはありませんでした。

SL が変更された場合は、その変更が PowerMaxOS ソフトウェア レイヤーで行われるため、すぐに有効になります。また、SL のレイテンシーは読み取りと書き込みの両方の I/O 応答時間に影響します。

2 つのデータベース ワークロードを持つサービス レベルの例

次の図は、2 個の Oracle データベースにおけるサービス レベルの影響を示しています。このテストでは、2 個の OLTP ワークロードが中断または変更されることなく実行されました。図の左側にあるように、上側の黄色のラインに示されるワークロードの SL は、Diamond の SL（重要でミッション クリティカルなアプリケーションのシミュレーション）に設定されています。下側の青色のラインに示されるその他のワークロードの SL は、Bronze の SL で開始され、Diamond の SL に到達するまで 30 分ごとに増加されました。

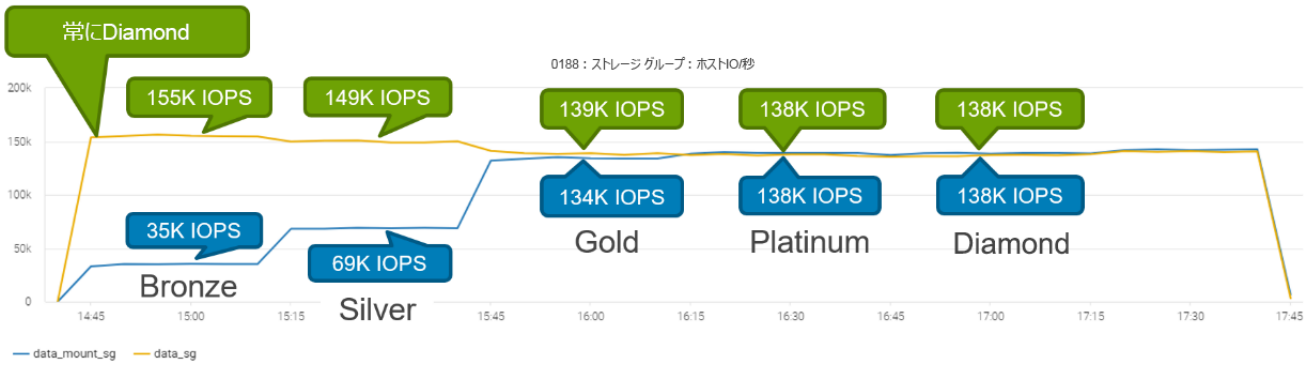

図 24.　2 個の Oracle ワークロードにおけるサービス レベルの変更

2 個のアプリケーションが同じ SL とシステム リソースを共有するまでの間、「ケージングされている」アプリケーションの SL が向上するとともに、Diamond の SL を使用したアプリケーションのリソースを徐々に消費していったことが分かります。この結果は、優先度の低い SL を優先度の低いアプリケーションに設定する価値を示しています。

PowerMax と Oracle データベースのベスト プラクティス

ストレージに関する考慮事項

ストレージ接続

一般的な SAN 構成では、HBA ポート（イニシエーター）とストレージのフロントエンド ポート（ターゲット）がスイッチに接続されています。スイッチ ソフトウェアでゾーンを作成し、イニシエーターとターゲットをペアリングします。各ペアリングで、サーバーとストレージ間に I/O が通過できる物理パスが作成されます。SAN スイッチを使用してサーバーとストレージの接続を構成する場合は、次のベスト プラクティスが適用されます。

- 冗長性と高可用性を実現するには、少なくとも 2 台のスイッチを使用して、障害やメンテナンスで 1 台が使用できなくなっても、サーバーがストレージへのアクセスを失わないようにします。
- （最初に 1 台のストレージ ダイレクターにすべてのポートを割り当ててから次に移動するのではなく）ストレージ エンジン、ダイレクター、I/O モジュールおよびポートに接続を分散させることで、最高のパフォーマンスと可用性を実現できます。
- サーバー イニシエーターをストレージ ポート ターゲットに接続するときは、スイッチを交差させないでください。つまり、スイッチ間リンク（ISL）は共有リソースであり、接続が制限され、使用率を予測できないことが多いため避けてください。
- サーバーからストレージへの接続を考慮する場合は、クラスター環境であっても、1 つのノードでデータのロードや RMAN バックアップを実行する場合があることに注意してください接続を適切に計画します。

  以下のポイントは、デバイスおよびサーバーからストレージへの接続ごとのパス数のガイドラインとなります。

  - 各 PowerMax ブリックには、フロントエンド I/O モジュールを保持する 2 つのダイレクターがあり、各 FC または FC-NVMe フロントエンド I/O モジュールには 4 つのポートがあります。1 つのブリック PowerMax 8000 は、ブリックごとに最大 24 個のフロントエンド ポートをサポートします（ダイレクターごとに 3 x I/O モジュール）。一方、2～8 個のブリックを備えた PowerMax 8000、および 1～2 個のブリックを備えた PowerMax 2000 は、それぞれ最大 32 個のフロントエンド ポート（ダイレクターごとに 4 個の I/O モジュール）をサポートします。PowerMax ダイレクターのレイアウトと I/O モジュールの詳細については、『Dell EMC PowerMax ファミリーの概要』ホワイトペーパーの 30～31 ページにある図 16、17、および 18 を参照してください。
  - ほとんどの OLTP ワークロード（一部のデータベース レポートおよびバッチと混合）では、4 つまたは 8 つのフロントエンド ポート（デバイスごとに 4 つまたは 8 つのパス）が非常に優れたスループット（IOPS）と中程度の帯域幅（GB/秒）を実現します。図 25 は、単一のブリック PowerMax 8000 へのデバイスごとに 8 つのパスがある接続の例を示しています。

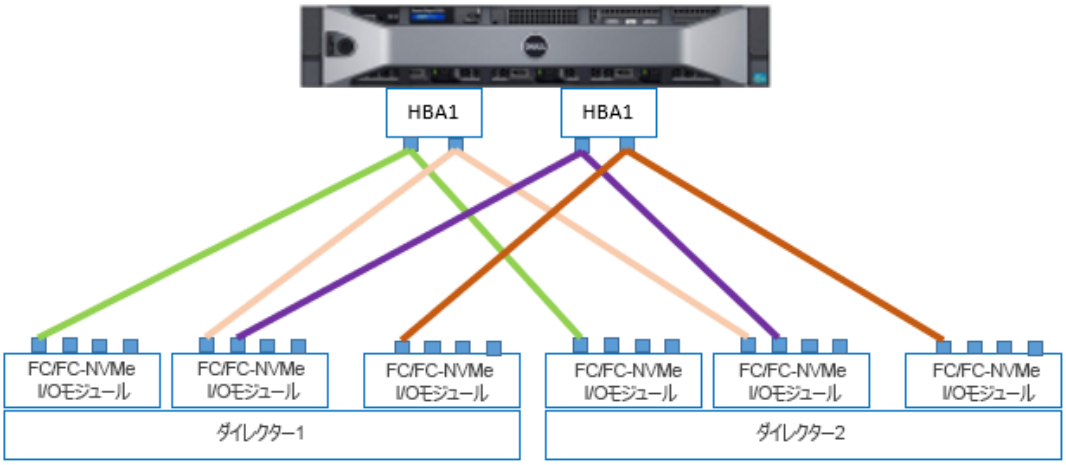

図 25.　デバイスごとに 8 つのパスがある接続の例（4 つのイニシエーター x 2 つのフロントエンド ポート）

- ハイパフォーマンスの OLTP データベース（レポートおよびバッチと混合）の場合、16 個のフロントエンド ポート（デバイスあたり 16 個のパス）によってスループットを最大限に高めることができます。図 26 は、単一のブリック PowerMax 8000 へのデバイスごとに 16 個のパスがある接続の例を示しています。

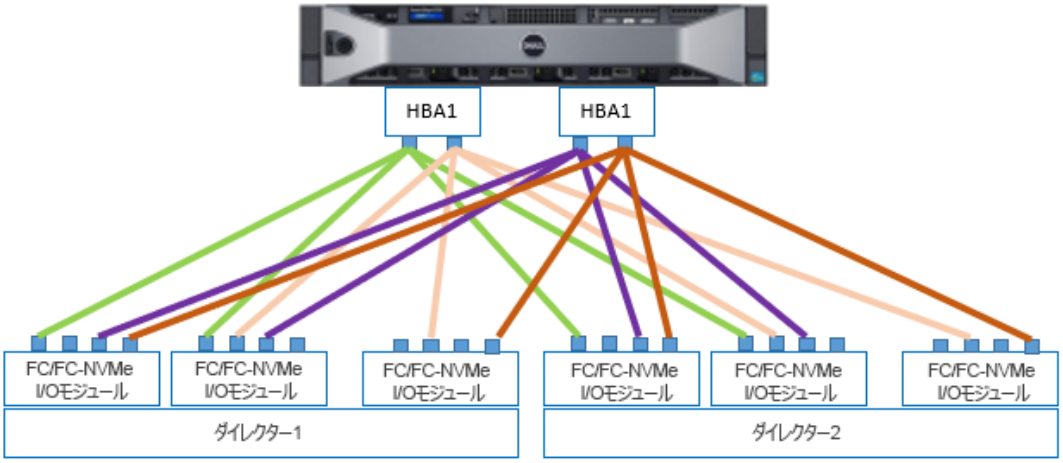

図 26.　デバイスごとに 16 つのパスがある接続の例（4 つのイニシエーター x 4 つのフロントエンド ポート）

- クエリ データセットが大きくて PowerMax キャッシュに収まらない、DSS に重点を置いたほとんどのワークロード（データ ウェアハウスと分析アプリケーション）の場合、16 個のフロントエンド ポートによって帯域幅（GB/秒）を最大化できます。データセットが PowerMax キャッシュ（100%読み取りヒット）に収まる場合は、追加のフロントエンド ポートによって帯域幅を拡大できます。
  - 100%読み取りヒットのハイパフォーマンス DSS テストでは、16 ポートで約 23 GB/秒を達成し、24 ポートで約 30 GB/秒を達成しました。パーティション プルーニング（約 30%の読み取りヒット）を行わない場合、データベース帯域幅は約 12 GB/秒にしか達しません。このことは、ほとんどのハイパフォーマンス環境では 16 ポートで十分であることを示しています。

FC または FC-NVMe プロトコルの選択とコア割り当て

FC プロトコルを使用する場合は、PowerMax 32Gb フロントエンド モジュールが構成され、FA（FC フロントエンド アダプター）として参照されます。FC-NVMe プロトコルを使用する場合、PowerMax 32Gb フロントエンド モジュールは同じハードウェアを使用しますが、FN（FC-NVMe フロントエンド アダプター）として構成および参照されます。

PowerMax 組み込み管理ではストレージへのアクセスが必要なため、システムがサーバー接続専用に FN を使用している場合でも、最初のエンジン（ブリック）のダイレクターごとに少なくとも 1 つのポートを FA として構成する必要があります。ポートが構成されると、そのダイレクターの CPU コアがポートに割り当てられます。

その結果、FC サーバー接続専用に構成されたシングルエンジン システムと FC-NVMe サーバー接続専用に構成されたシステムの間でストレージ CPU コアの割り当てを比較すると、他のすべての条件が同じであれば、FC-NVMe ポートをサポートするコアはわずかに少なくなります。

ほとんどの場合、この違いは重要ではなく、レイテンシーの改善や I/O アクセスの最適化などの FC-NVMe の利点を損なうものではありません。また、影響を受けるのは最初のエンジンだけであるため、構成されているエンジン（ブリック）が多いほど、違いは少なくなります。ただし、システムからの最大 IOPS が必要な場合（受け入れテストやベンチマークなど）は、FC-NVMe の代わりに FC を使用することでメリットを得られる可能性があります。

マスキング ビュー

PowerMax は、マスキング ビューを使用して、サーバーに表示されるデバイスを決定します。マスキング ビューには、ストレージ グループ（SG）、ポート グループ（PG）、イニシエーター グループ（IG）が含まれますマスキング ビューを作成すると、SG 内のデバイスは、IG 内のサーバー イニシエーターによって認識され、PG のポートによってストレージにアクセスできます。

マスキング ビューのコンポーネントのいずれかに変更が加えられると、マスキング ビューが自動的に更新されます。たとえば、SG にデバイスを追加すると、新しいデバイスがマスキング ビューのイニシエーターとポートを通じてサーバーで自動的に認識されるようになります。

ストレージ・グループ

ストレージ グループ（SG）は、一緒に管理されるデバイスのグループです。さらに、SG には他の SG を含めることができます。その場合、最上位の SG は親 SG と呼ばれ、下位レベルの SG は子 SG と呼ばれます。親/子 SG 構成では、デバイスは、子 SG のいずれかを直接使用するか、親 SG を使用して管理されるため、操作はすべての子 SG に影響します。たとえば、マスキング ビューには親 SG を使用し、データベースのバックアップ/リカバリー スナップショットおよびより詳細なパフォーマンス監視には子 SG を使用します。

- 細分性の高いパフォーマンス監視やデータベースのバックアップ/リカバリーに対応したスナップショットを必要としないデータベースの場合は、マスキング専用の 1 つの SG にすべてのデータベース デバイスを追加すれば十分です。
- ミッションクリティカルな Oracle データベースの場合は、次のデータベース コンポーネントをさまざまな ASM ディスク グループおよび一致する SG に分離することをお勧めします。
  - **data_sg**：データ ファイル、コントロール ファイル、UNDO テーブルスペース、システム テーブルスペースなどのデータベース データに使用されます。ログからデータを分離することで（data_sg と redo_sg を分離する）、ストレージ レプリケーションをデータベースのバックアップ/リカバリー操作やより細分性の高いパフォーマンス監視に使用できます。

- **redo_sg**：データベースの REDO ログに使用されます。
- **fra_sg**：データベースのアーカイブ ログおよびフラッシュバック ログ（使用されている場合）に使用されます。フラッシュバック ログは、アーカイブ ログよりもはるかに大きな容量を消費する可能性があることに注意してください。また、アーカイブ ログとは異なり、ストレージ レプリケーションで保護されている場合、フラッシュバック ログはデータ ファイルと整合性を維持する必要があります。このような理由から、アーカイブ ログとフラッシュバック ログは、異なる ASM ディスク グループおよび SG に分離することを検討してください。
- **grid_sg**：Grid Infrastructure（GI）に使用されます。GI は Oracle ASM または RAC（クラスター）を使用する場合に必要なコンポーネントです。シングル インスタンス（非クラスター化）導入環境でも、データベース データが GI 管理コンポーネントと混在しないように、この ASM ディスク グループと SG を作成することをお勧めします。

メモ：データベースのバックアップ/リカバリーに有効な高速のストレージ レプリカを活用できる ASM ディスク グループおよび一致する SG の詳細については、『Oracle database backup, recovery, and replications best practices with VMAX All Flash storage』を参照してください。

イニシエーター グループ

イニシエーター グループ（IG）には、ストレージ デバイスがマップされるサーバー イニシエーターの（HBA ポート）WWN（World Wide Name）のグループが含まれます。さらに、IG には他の IG を含めることができます。その場合、最上位の IG は親 IG と呼ばれ、下位レベルの IG は子 IG と呼ばれます。

データベースがクラスター化されている場合は、親/子 IG の導入が便利です。各子 IG には 1 台のサーバーのイニシエーターが含まれ、親 IG でそれらすべてを統合します。マスキング ビューが作成されると、親 IG が使用されます。クラスター ノードがクラスターに追加されたりクラスターから削除されても、マスキング ビューは変更されません。追加または削除されているノードと一致する子 IG を追加または削除することで親 IG のみが更新されます。

ポート グループ

ポート グループ（PG）には、ターゲット（ストレージ フロントエンド ポート）のグループが含まれています。マスキング ビューに配置されている場合、これらは、SG 内のデバイスにアクセスするために使用されるストレージ ポートになります。

物理接続は SAN ゾーン セットによって決まるため、管理をシンプルにするために、データベースで使用するすべてのストレージ ポートを PG に含めることをお勧めします。PG ポートと IG イニシエーター間の固有のパス関係は、ゾーン セットによって決まります。

マスキング ビュー

ミッションクリティカルでない環境の場合、1 つの SG 内にすべてのデバイスを含むデータベース全体のシンプルなマスキング ビューを作成し、単一のマスキング ビューを使用すれば十分です。

次のガイドラインは、データとログの SG を分離して、ストレージ スナップショットと細分性の高いパフォーマンス監視を使用したバックアップ/リカバリーを可能にする、ハイ パフォーマンスのミッションクリティカルなデータベースに適用されます。

この場合、data_sg と redo_sg は親 dataredo_sg SG SG の下に結合され、FRA は独自の SG に含まれます。次の表は、データベース用の 2 つのマスキング ビューと、クラスターまたは Grid Infrastructure 用に 1 つのマスキング ビューがあることを示しています。

表 6.　マスキング ビュー設計の例

マスキング ビュー	ストレージ グループ	子 SG	イニシエーター グループ	子 IG	ポート グループ
App1_DataRedo	App1_DataRedo	App1_Data、App1_Redo	App1_servers	Server1、Server2, …	PG1
App1_FRA	App1_FRA	（なし）	（同上）	（同上）	（同上）
グリッド	グリッド	（なし）	（同上）	（同上）	（同上）

データベースがクラスター化されている場合、IG はクラスターノードを含む親 IG になります。データベースがクラスター化されていない場合、IG には単一のサーバー イニシエーター（子 IG なし）を含めることができます。同様に、同様に、データベースがクラスター化されている場合は、「グリッド」ASM ディスク グループ デバイスを独自の SG およびマスキング ビューに含めることができます。データベースがクラスター化されていない場合、グリッド マスキング ビューはオプションです。

この設計にはいくつかの利点があります。

- データベース全体（App1_DataRedo）、またはデータ（App1_Data）と REDO ログ（App1_Redo）それぞれに対して、パフォーマンスを監視することができます。
- ストレージ コンシステントなスナップショットがリスタート ソリューションの一部として作成されている場合は、親 SG App1_DataRedo が使用されます。スナップショットがリカバリー ソリューションの一部として作成されている場合、リカバリー中に App1_Data のみの SG は、本番 REDO ログ（App1_Redo）を最新のデータベース トランザクションで上書きせずにリストアできます。
- SRDF を使用してデータベースをレプリケートする場合は、親 App1_DataRedo SG を使用して、ストレージ整合性のあるデータベース全体のレプリケーションを設定します。

次の例は、デバイスの作成からビューのマスキングに至るまでの、コマンドライン インターフェイス（CLI）の実行を示しています。マスキング ビューは、Unisphere でウィザードを使用して作成できます。CLI は、このようなコマンドがスクリプト化されて保存されている場合にのみ推奨されます。

コマンドライン インターフェイス（CLI）を使用したマスキング ビューの作成の例

- デバイスの作成：

```
set -x
export SYMCLI_SID=<SID>           # Storage ID
export SYMCLI_NOPROMPT=1

# Create ASM Disk Groups devices
symdev create -v -tdev -cap 40 -captype gb -N 3          # +GRID
symdev create -v -tdev -cap 200 -captype gb -N 16        # +DATA
symdev create -v -tdev -cap 50 -captype gb -N 8          # +REDO
symdev create -v -tdev -cap 150 -captype gb -N 4         # +FRA
```

メモ：FC-NVMe 接続用の新しいデバイスを作成するときは、上記の構文に「`-mobility`」フラグを追加してください。既存のデバイスを FC から FC-NVMe プロトコルに変更する場合は、次のような構文を使用して、デバイスのタイプを mobility に変更します。

```
symdev -devs <START:END> set -device_id mobility
```

- ストレージ グループの作成（デバイス ID は、上記のデバイス作成手順からの出力に基づいています）：

```
# SGs
symsg create grid_sg        # グリッド インフラストラクチャ用のスタンドアロン SG
symsg create fra_sg         # アーカイブ ログ用のスタンドアロン SG
symsg create data_sg        # データおよびコントロール ファイル デバイスの子 SG
symsg create redo_sg        # REDO ログ デバイスの子 SG
symsg create dataredo_sg    # データベース（データ+REDO）デバイスの親 SG

# 各 SG に適切なデバイスを追加
symsg -sg grid_sg addall -devs 12E:130                # デバイス ID を変更
symsg -sg data_sg addall -devs 131:133,13C:148        # 必要に応じて
symsg -sg redo_sg addall -devs 149:150
symsg -sg fra_sg addall -devs 151:154

# Add the child SGs to the parent
symsg -sg dataredo_sg add sg data_sg,redo_sg
```

- ポート グループの作成：

```
# PG
symaccess -type port -name 188_pg create       # 188 はストレージ SID
symaccess -type port -name 188_pg add -dirport 1D:4,1D:5,1D:6,1D:7
symaccess -type port -name 188_pg add -dirport 2D:4,2D:5,2D:6,2D:7
symaccess -type port -name 188_pg add -dirport 1D:8,1D:9,1D:10,1D:11
symaccess -type port -name 188_pg add -dirport 2D:8,2D:9,2D:10,2D:11
```

- イニシエーター グループの作成（4 つのサーバーを使用した 4 ノード RAC の例：dsib0144、dsib0146、dsib0057、dsib0058）。

メモ：デバイスの WWN は、Unisphere を使用するか、各データベース サーバーでコマンド `cat /sys/class/fc_host/host*/port_name` を実行することで見つけることができます。

```
# IG
symaccess -type initiator -name dsib0144_ig create
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa910b2
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa910b3
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa90f86
symaccess -type initiator -name dsib0144_ig add -wwn 10000090faa90f87

symaccess -type initiator -name dsib0146_ig create
symaccess -type initiator -name dsib0146_ig add -wwn 10000090faa910aa
```

```
symaccess -type initiator -name dsib0146_ig add -wwn 10000090faa910ab
symaccess -type initiator -name dsib0146_ig add -wwn 10000090faa910ae
symaccess -type initiator -name dsib0146_ig add -wwn 10000090faa910af

symaccess -type initiator -name dsib0057_ig create
symaccess -type initiator -name dsib0057_ig add -wwn 10000090fa8ec6e8
symaccess -type initiator -name dsib0057_ig add -wwn 10000090fa8ec6e9
symaccess -type initiator -name dsib0057_ig add -wwn 10000090fa8ec8ac
symaccess -type initiator -name dsib0057_ig add -wwn 10000090fa8ec8ad

symaccess -type initiator -name dsib0058_ig create
symaccess -type initiator -name dsib0058_ig add -wwn 10000090fa8ec6ec
symaccess -type initiator -name dsib0058_ig add -wwn 10000090fa8ec6ed
symaccess -type initiator -name dsib0058_ig add -wwn 10000090fa8ec720
symaccess -type initiator -name dsib0058_ig add -wwn 10000090fa8ec721

symaccess -type initiator -name db_ig create          # RAC の親 IG
symaccess -type initiator -name db_ig add -ig dsib0144_ig
symaccess -type initiator -name db_ig add -ig dsib0146_ig
symaccess -type initiator -name db_ig add -ig dsib0057_ig
symaccess -type initiator -name db_ig add -ig dsib0058_ig
```

- マスキング ビューの作成：

```
# MV
symaccess create view -name dataredo_mv -pg 188_pg -ig db_ig -sg dataredo_sg
symaccess create view -name fra_mv -pg 188_pg -ig db_ig -sg fra_sg
symaccess create view -name grid_mv -pg 188_pg -ig db_ig -sg grid_sg
```

デバイスの数とサイズ

PowerMax は、シン デバイスのみを使用します。つまり、アプリケーションがデバイスに書き込みを行うときにのみストレージ容量が消費されます。このアプローチでは、ストレージは実際に必要な場合にのみ使用されるため、フラッシュ容量を節約できます。

PowerMax デバイスのサイズは、数メガバイトから数テラバイトまで設定できます。そのため、容量が非常に大きいデバイスをわずかな数だけ作成したくなるかもしれませんが、次のことを考慮してください。

- Oracle ASM を使用する場合、ASM ディスク グループのデバイス（メンバー）は同様の容量であることが必要です。デバイスの最初のサイズが大きい場合は、ASM ディスク グループへの各容量の増分も大きくする必要があります。
- Oracle ASM のベスト プラクティスは、一度に 1 つずつデバイスを追加するのではなく、複数のデバイスをまとめて追加して、ASM ディスク グループの容量を増やすことです。この方法を使用すると ASM エクステントがリバランシング中に分離してホットスポットが回避されます。元のデバイスと同じサイズの複数のデバイスを ASM ディスク グループに同時に追加できるように、デバイス サイズの増分単位を設定します。
- 複数のデバイスを使用するもう 1 つのメリットは、サーバーがデバイス パスごとに I/O キューを作成し、複数の I/O キューが I/O の並列処理を強化して、キューイングの問題を防ぐことです。また、

データ転送が複数のデバイス間で発生する場合に、ローカルまたはリモート レプリケーション実行時のストレージ システムの並列処理が強化されるというメリットもあります。

- Oracle リリース 12.1 まで、ASM デバイス サイズは 2 TB に制限されていました。Oracle リリース 12.2 以降では、ASM ははるかに大きなデバイス サイズをサポートします。

すべてのデータベースに適したサイズが 1 つあるわけではありませんが、デバイスのサイズと数については、次のことをお勧めします。

- ハイパフォーマンス データベースには、多くの場合、8～16 個のデータ デバイスと 4～8 個の REDO ログ デバイスで十分です。12.1 以前の Oracle リリースの場合、これによって最大 32 TB（16 データ デバイス x 2 TB = 32 TB）が可能になります。32 TB を超えるデータベースでは、データベースの容量要件を満たすためにより多くのデバイスが必要になる場合があります。Oracle リリース 12.2 以降を使用している場合は、2 TB の制限に縛られなくなりました。
- SRDF/Sync、SRDF/Metro、PowerProtect（SnapVX を使用）などのストレージ レプリケーションを導入する場合は、レプリケーションの同時実行性を高めるために、上記の数値を 2 倍にすることをお勧めします。つまり、16～32 個のデータ デバイスと 8～16 個の REDO ログ デバイスを使用してください。こうすることによって、より高いレプリケーション帯域幅を実現できます。

メモ：すべての PowerMax デバイスはシンであり、書き込みが行われる場合を除きストレージ容量は消費されませんが、デバイスのサイズが大きいほど、消費されるメタデータ量が多くなります（これは使用可能なストレージ キャッシュに影響します）。したがって、予想される容量のニーズに基づいて、デバイスのサイズを適度に設定する必要があります。

サーバの検討事項

パーティションのアライメント

Linux では必須ではありませんが、Oracle では ASM デバイスごとに 1 つのパーティションを作成することを推奨しています。デフォルトで、Oracle Linux（OL）および Red Hat Enterprise Linux（RHEL）リリース 7 以降では、デフォルトのオフセットが 1 MB であるパーティションが作成されます。ただし、OL または RHEL の以前のリリースでは、63 ブロックのパーティション オフセット（63 x 512 バイト = 31.5KB）がデフォルトに設定されていました。

PowerMax は 128 KB のトラック サイズを使用するため、0 オフセット（パーティションが作成されない場合）または 1 MB のオフセットが完全に対応します。

デフォルトで 31.5 KB のオフセットになっている旧リリースの Linux で ASM デバイス用のパーティションを作成する場合は、パーティションのオフセットを 1 MB に調整することを強くお勧めします。

パーティションの作成と調整には、`parted` コマンドまたは `fdisk` コマンドを使用します。 多くの場合、parted の方が使いやすく、スクリプト作成も簡単です。

次の手順は、`fdisk` コマンドの使用方法を示しています。

1. デバイス上に 1 つのプライマリー パーティションを作成します。
2. fdisk **エキスパート** モードに切り替えるには、**x** を使用します。
3. パーティション**開始**オフセットを変更するには、**b** を使用します。
4. 1 MB のオフセットの場合は 2,048 を入力します（2,048 x 512 バイト ブロック）。
5. オプションで、パーティション テーブル レイアウトを**出力**するには、**p** を使用します。
6. パーティション テーブルを**作成**するには、**w** を使用します。

次の例は、PowerPath デバイスで Linux `parted` コマンドを使用する方法を示しています（たとえば、デバイス マッパー用に同様のスクリプトを作成できます）。

```
for i in {a..h}; do
  parted -s /dev/emcpower$i mklabel msdos
  parted -s /dev/emcpower$i mkpart primary 2048s 100%
  chown oracle.oracle /dev/emcpower$i1
done
fdisk -lu # サーバー デバイスとそのパーティション オフセットを一覧表示
```

RACを使用する場合、他のノードは新しいパーティションを認識しません。他のすべてのノードでパーティション テーブルを再起動または読み取り/書き込みすると、この状態が解決されます。例：

```
for i in {a..h}; do
  fdisk /dev/emcpower$i << EOF
w
EOF
  chown oracle.oracle /dev/emcpower$i1
done
fdisk -lu # サーバー デバイスとそのパーティション オフセットを一覧表示
```

マルチパス ソフトウェア

マルチパス ソフトウェアは、以下の理由から、データベースの導入にとって重要です。

- **ロード バランシングとパフォーマンス**—ストレージ デバイスごとに数千 IOPS を処理する必要が生じる場合があります。オールフラッシュ ストレージを使用すると、使用するデバイスの数は少なくなりますが、容量が大きくなる傾向があり、その結果、デバイスあたりの IOPS 要求が増加します。マルチパス ソフトウェアを使用すると、複数の HBA ポート（イニシエーター）とストレージ ポート（ターゲット）を介してデバイスへの読み取りおよび書き込み I/O を処理できます。これにより、ポート全体に負荷を分散して、単一パスまたはポート I/O 制限を回避することができます（「ロード バランシング」）。
- **パス フェールオーバー**—マルチパス ソフトウェアは、アプリケーションが使用する疑似デバイス名またはエイリアスを作成します。一方、疑似デバイスへの I/O は、イニシエーターとターゲットの間で使用可能なすべての異なるパスによって処理されます。パスが動作を停止すると、マルチパス ソフトウェアは自動的に I/O を残りのパスに送ります。パスが復帰すると、ソフトウェアはそのパスへの I/O のサービスを自動的に再開します。その間、アプリケーションは、アクティブなパスに関係なく、疑似デバイスを引き続き使用します。
- **マルチパス疑似デバイスは Oracle ASM と連携**—Oracle ASM は、各ストレージ デバイスの単一の表示を必要とし、そのデバイスへの複数のパスが見つかった場合にデバイスを ASM 候補としてリストしません。マルチパス疑似名（またはユーザーフレンドリーなエイリアス）を使用することにより、ASM は単一の表示を認識するようになり、マルチパス ソフトウェアはその疑似デバイスのさまざまなパスに I/O を分散させます。

FC-NVMe マルチパス オプション

FC-NVMe は比較的新しいため、Linux OS リリースとマルチパス ソフトウェアに関する考慮事項がいくつかあります。一部の旧 Linux リリースでは、FC-NVMe 対応であることが主張されていますが、このセクションでは、FC-NVMe が最適な選択肢であるとは限らない理由について説明します。

FC-NVMe のハイパフォーマンスと低レイテンシーを実現するには、マルチパス ソフトウェアがプロトコルで行われた変更と最適化に対処する必要があります。Dell EMC PowerPath 7 は、FC-NVMe マルチパス サポートを提供する最初の PowerPath リリースです。PowerPath 7 は、パス フェールオーバー、ロード バランシング、ハイパフォーマンスなど、FC-NVMe のマルチパス機能を完備しています。

FC-NVMe 用の Linux ネイティブ マルチパス ソフトウェアを検討する場合は、次の 2 つのオプションがあります。

- Linux デバイス マッパー（DM）
- 新しい FC-NVMe マルチパス

DM は NVMe および FC-NVMe デバイスで適切に動作し、旧 Linux リリースにも対応していますが、その目的のために最適化されておらず、Dell が実施したどのテストでもパフォーマンスが低下しました。その結果、FC-NVMe に DM を使用することはお勧めしません。

したがって、ネイティブ マルチパスの場合は、新しい FC-NVMe マルチパス ソフトウェアを使用してください。このとき、以下の点に留意してください。

- FC-NVMe ネイティブ マルチパスをサポートする初期の Linux リリースでは、ロード バランシングなしでパス フェールオーバーのみが有効になりました。パス I/O ポリシーは「numa」として設定され、アクティブなパスは 1 つしか許可されず、その他のパスはすべて、フェールオーバーの準備ができたスタンバイ状態でした。この構成はハイパフォーマンスを促進しないため、お勧めしません。
- その後の Linux リリースでは、FC-NVMe ネイティブ マルチパスに「ラウンドロビン」パス I/O ポリシーが追加されました。このポリシーは、ロード バランシングとフェールオーバーの両方を可能にし、ハイパフォーマンスをサポートします。

重要： 上記の理由により、「ラウンド ロビン」パス I/O ポリシーを実装する PowerPath（FC-NVMe サポート付き）または Linux ネイティブ FC-NVMe マルチパス（DM と混同しないでください）のいずれかを使用することをお勧めします。

このホワイトペーパーの執筆時点で利用可能な最小マルチパス オプションと OS リリースは、次の表にまとめられています。

表 7. FC-NVMe Linux OS リリースおよびマルチパス オプション

オペレーティング システム	PowerPath	「ラウンド ロビン」I/O ポリシーを備えたネイティブ MP
SuSE Enterprise Linux	使用可能[4]	使用可能[5]

[4] Linux 7.0 および SLES 15 用の PowerPath の最小要件。Dell のラボ テストでは、カーネル 4.12.14-25.25 の最小要件が示されました。

[5] SuSE サポート アップデート（リンク）に基づいて、NVMeoF マルチパス「ラウンド ロビン」ポリシーが SLES12-SP4 および SLES15-SP4 で導入されました。Dell のラボ テストでは、カーネル 4.12.14-150 の最小要件が示されました。

オペレーティング システム	PowerPath	「ラウンド ロビン」I/O ポリシーを備えたネイティブ MP
Red Hat	使用可能[6]	使用可能[7]
Oracle® Linux	使用可能[8]	使用可能[9]

メモ： Dell EMC がサポートする構成の詳細については、Dell EMC eLab Navigator ノート『Dell EMC PowerMaxOS 5978.444.444 & 5978.479.479 – 32G FC-NVMe Support Matrix』を参照してください。

FC マルチパス オプション

FC プロトコルを使用する Linux 上の Oracle では、3 つのマルチパス オプションが利用可能です。

- Dell EMC PowerPath ソフトウェア（ベア メタル、つまり PowerPath/VE for VMware の場合）
- Linux デバイス マッパー（DM）ネイティブ マルチパス ソフトウェア
- VMware ネイティブ マルチパス

メモ：iSCSI プロトコルも PowerMax 上の Oracle データベースに使用するための有効なオプションですが、このホワイトペーパーでは取り上げていません。

3 つのオプションはすべて、FC で長年利用可能であり、パス フェールオーバー、ロード バランシング、ハイパフォーマンスを実現します。

デバイスの権限

サーバーが再起動すると、すべてのデバイスは root ユーザー権限をデフォルトで受信します。ただし、Oracle ASM デバイス（「ディスク」）には、Oracle ユーザー権限が必要です。Oracle ASM デバイスの権限設定はブート シーケンスの一部（グリッド インフラストラクチャと ASM の起動時）であることが必要です。

Oracle ASMlib では、デバイスの権限が自動的に設定されます。ASMlib を使用しない場合、udev ルールを使用すると、ブート シーケンス中にデバイスの権限を最も簡単に設定できます。

udev ルールは、`/etc/udev/rules.d/`ディレクトリー内のテキスト ファイルに追加されます。このルールは、このディレクトリー内のファイル名の前のインデックス番号に基づく順序で適用されます。

ルールを含むテキスト ファイルのコンテンツでは、Oracle デバイスを正しく識別する必要があります。たとえば、すべてのパーティション付きデバイスに Oracle 権限が含まれる可能性がある場合は、それらすべての

[6] Linux 7.1 および RHEL 8.0 用の PowerPath の最小要件。

[7] RHEL 8.1 の最小要件。

[8] Linux 7.1 および Oracle Linux 7.7/UEK5u2 用の PowerPath の最小要件。

[9] Oracle Linux 7.7/UEK5u2 の最小要件。Dell のラボ テストでは、カーネル 4.14.35-1902.5.2 の最小要件が示されました。

デバイスに適用される一般的なルールを設定します。すべてのデバイスが同じアクセス権を使用できるわけではない場合は、デバイスを個別に指定する必要があります。ほとんどの場合、UUID/WWN に基づいて指定する必要があります。

Oracle ユーザー権限を受け取るためにデバイスが個別に識別される場合は、udev ルールを使用して、簡単に識別できるように各デバイスのユーザーフレンドリーなエイリアスを作成することもできます。Linux デバイス マッパーを使用する場合は、`/etc/multipath.conf` ファイルを使用してデバイス エイリアスを作成することもできます。

デバイスを識別してアクセス権を設定する方法とオプションは多数あり、次の例はそれらを完全に網羅したリストではありません。

Linux 上の Oracle ASM では、パーティションをデバイスごとに 1 つずつ作成する必要はありませんが、Oracle はそれを推奨しており、パーティション 1 を持つすべてのデバイスに Oracle ユーザー権限を割り当てる簡単な方法を提供しています。

PowerPath の例（FC と FC-NVMe の両方）

PowerPath を使用する場合、PowerPath デバイスは再起動後も永続的であり、変更されないため、ユーザーフレンドリーなエイリアスなしで/dev/emcpower 表記を使用することがよくあります。そのため、次の例に示すように、udev ルールはすべての疑似デバイスのパーティション 1 に共通にすることができます。

```
# vi /etc/udev/rules.d/85-oracle.rules
ACTION=="add|change", KERNEL=="emcpower*1", OWNER:="oracle",
GROUP:="oracle", MODE="0660"
```

FC または FC-NVMe のどちらを使用しているかにかかわらず、PowerPath 疑似デバイスの命名規則（ひいては、デバイスのアクセス権）は同じです。

Linux デバイス マッパーの例（FC のみ）

デバイス マッパーを使用する場合、疑似デバイス名またはエイリアスを設定する方法は複数あります。たとえば、エイリアスの設定は`/etc/multipath.conf` ファイルで行うことも、udev ルール ファイルで直接行うこともできます。エイリアスの設定方法に基づいて、udev ルールを適用してデバイスのアクセス権を設定できます。

次の例では、ユーザーフレンドリーなエイリアスが`/etc/multipath.conf` ファイルに設定されていることを前提としています（手順については、「Linux デバイス マッパーの例」を参照してください）。この場合、udev ルールは、パーティション 1 のすべてのエイリアスに共通にすることができます（たとえば、「ora_prod_data1p1」や「ora_prod_redo3p1」などのデバイス エイリアスが作成された場合）。

```
# vi /etc/udev/rules.d/12-dm-permissions.rules
ENV{DM_NAME}=="ora_prod_*p1", OWNER:="oracle", GROUP:="oracle",
MODE:="660"
```

Linux ネイティブ FC-NVMe マルチパスの例

ネイティブ マルチパスで FC-NVMe を使用すると、パーティション 1 のデバイスは、/dev/nvmeXXp1 などのマルチパス疑似名で表示されます。エイリアスを必要としない場合は、次の例に示すように、一般的な udev ルールを適用できます。

```
# vi /etc/udev/rules.d/12-dm-permissions.rules
KERNEL=="nvme*p1", ENV{DEVTYPE}=="partition", OWNER:="oracle",
GROUP:="oracle", MODE:="660"
```

VMware の例（FC のみ）

VMware マルチパスはデバイスを/dev/sd ブロック デバイスとして VM に提示し（マルチパスは ESXi レベルで動作しています）、デバイスが追加または削除されるとブロック デバイスの割り当てが変更される場合があります。そのため、各デバイスの UUID に基づいてデバイスのアクセス権を個別に指定できます（手順については、「VMware ネイティブ マルチパスの例」を参照してください）。

RAC ノード間で一貫したデバイス名

Oracle RAC を使用する場合、同じストレージ デバイスがクラスタ ノード全体で共有されます。ASM は、ASM ディスク グループ内のデバイスに独自のラベルを付けます。したがって、ASM では、RAC ノード間でサーバー デバイス名を一致させる必要がありません。しかし、この操作によってユーザーのストレージ管理操作が容易になることがよくあります。このセクションでは、サーバーのデバイス名を照合する方法について説明します。

PowerPath の例

クラスター ノード間で PowerPath 仮想デバイス名を一致させるには、次の手順を実行します。

1. 1 台目のサーバー（クラスター ノード）で `emcpadm export_mappings` コマンドを使用して、PowerPath 構成で XML ファイルを作成します。

   ```
   # emcpadm export_mappings -f /tmp/emcp_mappings.xml
   ```

2. ファイルを他のノードにコピーします。
3. 他のノードで、マッピングをインポートします。

   ```
   # emcpadm import_mappings -f /tmp/emcp_mappings.xml
   ```

メモ：PowerPath データベースは`/etc/emcp_devicesDB.idx` ファイルと `/etc/emcp_devicesDB.dat` ファイルに保持されます。これらのファイルをいずれかのサーバーから別のサーバーにコピーし、その後再起動することができます。emcpadm export/import の方法を使用して、サーバー間で PowerPath デバイス名を一致させることをお勧めします。この場合のファイルのコピーは、他のサーバーの既存の PowerPath マッピングを上書きするショートカットです。

Linux デバイス マッパーの例

デバイス マッパー（DM）は、UUID を使用して、RAC ノード間で永続的にデバイスを識別できます。UUID を使用するだけで十分ですが、さらに便利なエイリアス（たとえば、`/dev/mapper/ora_data1`、`/dev/mapper/ora_data2` など）を使用することもできます。

次の例は、`/etc/multipath.conf` DM 構成ファイルをエイリアスを使用して設定する方法を示しています。デバイスの UUID を検索するには、`scsi_id -g /dev/sdXX` Linux コマンドを使用します。コマンドがまだインストールされていない場合は、`sg3_utils` Linux パッケージをインストールして追加してください。

```
# /usr/lib/udev/scsi_id -g /dev/sdb
360000970000198700067533030314633
```

```
# vi /etc/multipath.conf
...
multipaths {
        multipath {
                wwid                    360000970000198700067533030314633
                alias                   ora_data1
        }
        multipath {
                wwid                    360000970000198700067533030314634
                alias                   ora_data2
        }
...
}
```

RAC ノード間でエイリアスを一致させるには、`/etc/multipath.conf` 構成ファイルを他のノードにコピーして、それらのノードを再起動します。再起動を避けたい場合は、次の手順に従ってください。

1. 1 台のサーバーですべてのマルチパス デバイスを設定し、他のサーバーでマルチパスを停止します。

```
# service multipathd stop
# service multipath -F
```

2. 1 台目のサーバーから他のすべてのサーバーにマルチパス構成ファイルをコピーします。ユーザーフレンドリーな名前の整合性を確保する必要がある場合は、`/etc/multipath/bindings` ファイルを 1 台目のサーバーから他のすべてのサーバーにコピーします。エイリアスの整合性を確保する必要がある場合（エイリアスは `multipath.conf` で設定されます）、`/etc/multipath.conf` ファイルを 1 台目のサーバーから他のすべてのサーバーにコピーします。
3. 他のサーバーでマルチパスを再起動します。

```
# service multipathd start
```

VMware ネイティブ マルチパスの例

VMware 仮想マシン（VM）で Oracle を実行している場合、PowerPath/VE または VMware ネイティブ マルチパスは有効ですが、VM デバイスは/dev/sdXX として表示されます。Udev ルールを使用して、RAC ノード間で一致するデバイス エイリアスを作成し、Oracle ユーザー権限をデバイスに割り当てることができます。

VM 上のデバイス UUID を識別できるようにするには、VM の `disk.EnableUUID` を有効にします。次の手順に従います。

1. vSphere で、VM の電源をオフにします。
2. VM を右クリックして［**設定の編集**］を選択します。
3. ［**VM オプション**］タブを選択します。
4. ［**詳細設定**］オプションを展開し、［**構成の編集**］をクリックします。
5. 次の図に示すように、［**構成パラメーターの追加**］を選択して、パラメーター disk.EnableUUID.を追加します。パラメーターを TRUE に設定します。

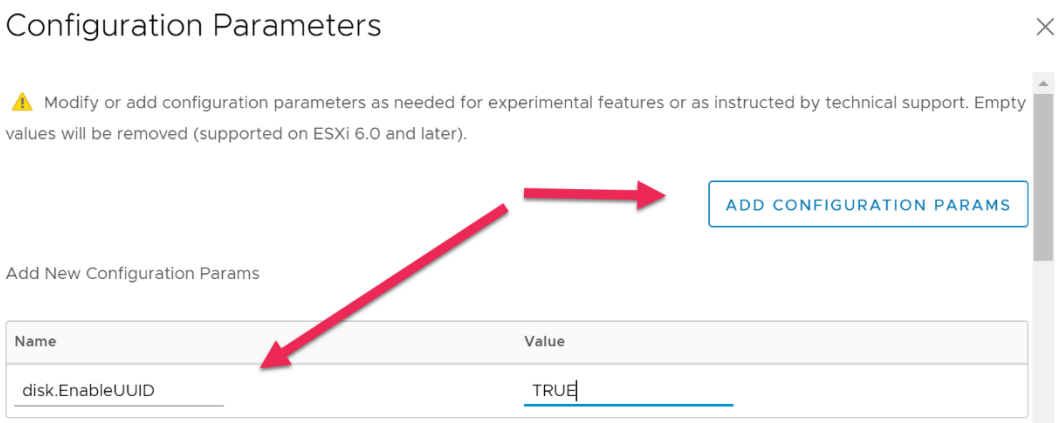

図 27.　UUID デバイス識別を有効にする

6. VM を再起動します
7. 次に示すように、`scsi_id` コマンドを使用して、デバイスの UUID を識別します。

メモ： RHEL 7 以降では、`scsi_id` コマンドは`/lib/udev/scsi_id` にあります。前のリリースでは、`/sbin/scsi_id` にありました。

```
# /lib/udev/scsi_id -g -u -d /dev/sdb
36000c29127c3ae0670242b058e863393
```

8. UUID に基づいて、パーティション 1 を持つすべてのデバイスに Oracle ユーザー権限を設定する udev ルールを作成します。次の例では、ASM ディスク文字列を使用して設定できるように、デバイスにエイリアスが指定されています。

```
# /lib/udev/scsi_id -g -u -d /dev/sdb
36000c29127c3ae0670242b058e863393
```

```
# cd /etc/udev/rules.d/
# vi 99-oracle-asmdevices.rules
KERNEL=="sd*1", SUBSYSTEM=="block", PROGRAM=="/usr/lib/udev/scsi_id -g -u
-d /dev/$parent", RESULT=="36000c29127c3ae0670242b058e863393",
SYMLINK+="ora-data1", OWNER="oracle", GROUP="oracle", MODE="0660"
...
```

ブロック マルチキュー（MQ）

最近の Linux カーネルでは、NVMe デバイスと超低レイテンシーおよび高 IOPS の必要性に対応するように、サーバー オペレーティング システムのブロック I/O デバイス ドライバーが書き直されました。新しい設計は、ブロック マルチキューまたは略して MQ と呼ばれます。SCSI ブロック デバイスがサーバー（/dev/sd）に提示されると、MQ はデフォルトで有効になりません。これは、SCSI ブロック デバイスが従来の回転式ハード ディスク ドライブ（HDD）でサポートされる可能性があるためです。ストレージがオールフラッシュである場合（PowerMax など）は、「付録 Iblk-mq と scsi-mq」で説明されているように、SCSI デバイスに対して MQ を手動で有効にすることができます。

PowerMax FC-NVMe を使用している場合など、NVMe デバイス（/dev/nvme）としてサーバーに提示されるデバイスは、自動的に MQ に対して有効になります。この機能を有効にするために追加の変更は必要ありません。

おそらくフラッシュ メディアの採用が進んでいることが理由で、RHEL 8.0 では、SCSI（/dev/sd）デバイスでも MQ がデフォルトになっています。

以前のテストでは、MQ を使用するとパフォーマンスが大幅に向上することがわかりました。一方、最近のテストでは、新しい CPU、32Gb SAN、4 ノードの Oracle 19c RAC を使用した場合に、サーバーは「ボトルネック」ではなくなり、MQ による目立ったメリットは確認されませんでした。

Dell EMC は、サーバーが頻繁に使用される FC 環境で MQ を有効にしてテストし、Oracle FC の導入に大きなメリットがあるかどうかを確認することをお勧めしています。FC-NVMe の導入環境では、MQ が自動的に有効になることに注意してください。

Oracle ASM のベスト プラクティス

ASM ディスク グループ

ミッションクリティカルな Oracle データベースの場合は、次のデータベース コンポーネントをさまざまな ASM ディスク グループおよび一致する SG に分離することをお勧めします。

- **+DATA—**データ ファイル、コントロール ファイル、UNDO テーブルスペース、システム テーブルスペースなどのデータベース データに使用される ASM ディスク グループ。ログからデータを分離することで（+DATA と+REDO ASM ディスク グループを分離する）、ストレージ レプリケーションをデータベースのバックアップ/リカバリー操作やより細分性の高いパフォーマンス監視に使用できます。
- **+REDO—**データベースの REDO ログに使用される ASM ディスク グループ。
- **+FRA—**データベースのアーカイブ ログおよびフラッシュバック ログ（使用されている場合）に使用される ASM ディスク グループ。フラッシュバック ログは、アーカイブ ログよりもはるかに大きな容量を消費する可能性があることに注意してください。また、アーカイブ ログとは異なり、ストレージ レプリケーションで保護されている場合、フラッシュバック ログはデータ ファイルと整合性を維持する必要があります。このような理由から、アーカイブ ログとフラッシュバック ログは、異なる ASM ディスク グループおよび SG に分離することを検討してください。
- **+GRID—**Oracle ASM または RAC（クラスター）を使用する場合に必要なコンポーネントである Grid Infrastructure（GI）に使用される ASM ディスク グループ。シングル インスタンス（非クラスター化）導入環境でも、データベース データが GI 管理コンポーネントと混在しないように、この ASM ディスク グループと SG を作成することをお勧めします。

  すべての ASM ディスク グループは、標準冗長性（2 つのミラー）を使用する可能性のある +GRID を除き、外部の冗長性（ASM ミラーリングなし）を使用する必要があります。+GRID にはユーザー データが含まれていないため、小規模のままになります。標準冗長性に設定すると、Oracle は 1 つではなく 3 つのクォーラム ファイルを作成します（外部の冗長性が使用される場合など）。3 つのクォーラム ファイルがあると、データベース アクティビティが集中しているときにノードがクォーラムへの登録を試みている間の遅延を回避するのに役立ちます。

メモ：データベースのバックアップ/リカバリーに有効な高速のストレージ レプリカを活用できる ASM ディスク グループおよび一致する SG の考慮事項について詳しくは、『Oracle database backup, recovery, and replications best practices with VMAX All Flash storage』を参照してください。

ASM ストライピング

デフォルトでは、ASM は 1 MB のアロケーション ユニット（AU）サイズ（リリース 12.2 では 4 MB）を使用し、ストライプ深度として AU を使用してディスク グループ全体にデータをストライピングします。このデ

フォルトの ASM ストライピング方式は「粗いストライピング」と呼ばれ、OLTP タイプのアプリケーションに最適です。DBA は、AU のサイズをデフォルトから増やすことを決定する場合がありますが、そうすることによる明確なメリットはありません。

AS には、「きめ細かなストライピング」と呼ばれる代替ストライピング方式があります。きめ細かなストライピングを使用すると、ASM はディスク グループ内で 8 台のデバイス（使用可能な場合）を選択し、それぞれに AU を割り当てて、各 AU を 128 KB のチャンクにさらにストライピング（分割）します。次に 128 KB のチャンクをいっぱいにすることで、8 台のデバイス全体にラウンドロビン方式でデータを割り当てます。8 つの AU がすべていっぱいになると、別の 8 台のデバイスを選択してプロセスを繰り返します。

きめ細かなストライピングは、主にシーケンシャル書き込みを行う Oracle オブジェクトに対して PowerMax が推奨するストライピング方式です。シーケンシャル書き込みでは I/O が大きくなる傾向にあるため、これらを 128 KB のストライプに分割することで、レイテンシーが改善され、PowerMax は（そのトラック サイズも 128 KB であるため）より効率的に書き込みを処理できるようになります。

このため、REDO ログには ASM のきめ細かなストライピングを使用することをお勧めします。この方法は、インメモリー データベース（トランザクションが高速で REDO 書き込みの負荷が大きくなる可能性がある）や、バッチ データのロード（ここでも REDO ログの書き込み負荷は重い）で特に役立ちます。

Oracle の Temp ファイルが I/O 集約型になる可能性のあるデータ ウェアハウスでは、Temp ファイルもきめ細かなストライピングのメリットを得ることができます。

各 Oracle ASM ファイル タイプのストライピングのタイプは、ASM テンプレートで保持されます。各 ASM ディスク グループには、独自のテンプレート セットがあります。テンプレートの変更（REDO ログ テンプレートをきめ細かに変更するなど）は、テンプレートが変更された ASM ディスク グループにのみ適用されます。

さらに、既存の ASM アロケーションは、テンプレートの変更の影響を受けず、新しいエクステントのみに影響を受けます。そのため、+REDO ASM ディスク グループの REDO ログ テンプレートを変更する場合は、後で新しい REDO ログ ファイルを作成し、データベースがそれらを使用していることを確認してから、古いものを削除する必要があります。REDO ログの作成と削除には時間がかからず、データベースの実行中に行えます。

ASM テンプレートを調査するには、ASM インスタンスから次のクエリーを実行します。

```
SQL> select DG.name Disk_Group, TMP.name Template, TMP.stripe from
v$asm_diskgroup DG, v$asm_template TMP where
DG.group_number=TMP.group_number order by DG.name;
```

+REDO ASM ディスク グループ内のデータベース REDO ログ テンプレートを変更するには、次のコマンドを実行します。

```
SQL> ALTER DISKGROUP REDO ALTER TEMPLATE onlinelog ATTRIBUTES
(FINE);
```

+TEMP ASM ディスク グループの一時ファイル テンプレートを変更するには、次のコマンドを実行します。

```
SQL> ALTER DISKGROUP TEMP ALTER TEMPLATE tempfile ATTRIBUTES
(FINE);
```

Oracle シーケンシャル読み取りの I/O サイズ

通常、OLTP ワークロードは、一度に 1 つずつレコードを読み取ったり更新したりします。そのため、Oracle での読み取りや変更は、シングル データベース ブロック（通常は 8 KB のサイズ）で行われます。これは、OLTP ワークロードを構成するデータ ファイルへの読み取りおよび書き込み操作の I/O サイズでもあります。ただし、レポート、リスト、複数のソースからのデータのマージ、述語の評価など、クエリでデータのセットをフェッチする必要がある場合、Oracle は 1 回の操作でマルチブロック読み取りを実行します。

Oracle がマルチブロック読み取りを実行すると、最大 1 MB のサイズの大きな I/O が発行されます。大規模読み取りのサイズは、データベース パラメーター`db_file_multiblock_read_count`（MBRC）によって制御できます。このデータベース パラメーターは、マルチブロック読み取り I/O 操作の Oracle 最大サイズを決定します。最大 I/O サイズは、MBRC と Oracle ブロック長の乗算として計算されます。8 KB のデータベース ブロック長で MBRC を 16 に設定した場合の結果は、最大 128 KB のデータベース読み取り I/O サイズ（16 x 8KB = 128KB）になります。MBRC が 128 に設定されている場合、結果は最大 1 MB のデータベース読み取り I/O サイズ（128 x 8 KB = 1,024 KB）になります。

Dell のテストでは、2 つのオプション間の帯域幅にごくわずかな違いがあることがわかりました。ただし、I/O サイズは IOPS とレイテンシーに大きな影響を及ぼします。128KB の I/O サイズは 1MB の I/O サイズの 8 分の 1 であるため、同じ帯域幅を実現するためにより多くの IOPS が生成されます。ただし、各 I/O が小さいため、レスポンス タイムは 1MB I/O よりもはるかに短くなります。

ほとんどの環境で OLTP ワークロードと DSS ワークロード（シングルブロック操作とマルチブロック操作）が混在して実行されるため、128KB の I/O サイズを使用することをお勧めします。混合環境では、マルチブロック操作のレイテンシーを短縮できるため、シングルブロック I/O の処理の待ち時間が長くなりません。システムが専用のデータ ウェアハウス（主にマルチブロック読み取り）である場合でも、帯域幅は非常に似ているため、システム全体の使用率に合計 IOPS が問題にならない限り、低レイテンシーがメリットであることに変わりありません。

4 KB の REDO ログ セクター サイズ

Oracle 11gR2 には、REDO ログのブロック長をデフォルトの 512 バイトから 4 KB に変更する機能が導入されました。導入理由の 1 つは、一部のドライブがネイティブのブロック長に 4 KB を使用しているためです（SSD ドライブなど）。もう 1 つの理由は、従来の 512 バイトから 4 KB にブロック長を増やすことで、高密度 HDD に関連づけられたメタデータ オーバーヘッドを削減するためです。

PowerMax ストレージを使用する場合、REDO ログのブロック長をデフォルトのセクターあたり 512 バイトから変更すべきでない大きな理由として次の 2 つがあります。

- データベースは、フラッシュ ドライブに直接書き込みを行いません。代わりに、PowerMax ストレージ システムへのすべての書き込みは、PowerMax キャッシュに保存されます。キャッシュで書き込みを統合し、後で最適化された書き込みをフラッシュ メディアに提供できます。したがって、このような変更でドライブに直接メリットがもたらされることはありません。
- 大きくなることの多い redo wastage が増えます。Oracle データベースが頻繁にコミットされる場合、ログ バッファの即時書き込みが必要です。4 KB のブロックと頻繁なコミットによって、REDO ログ バッファーがほとんど空になり、不要な書き込みオーバーヘッドと REDO の浪費が生じることがあります。

Linux カーネル I/O スケジューラの選択

Linux ブロック I/O ドライバーの一部は、I/O スケジューラ（I/O エレベータとも呼ばれる）です。I/O スケジューラは、送信キュー内の I/O を取得し、その順序を変更して、たとえば、書き込みよりも読み取りを優先したり、小さい I/O を大きい I/O に統合することができます。

Red Hat によると、最適なデータベース レイテンシーを得るために選択される I/O スケジューラは Deadline です。RHEL 7 以降、Deadline がデフォルトの I/O スケジューラになりました。ただし、以前のリリースでは CFQ でした。

VMAX All Flash 以降の PowerMax ストレージ システムでの Dell のテストによると、Deadline は優れたパフォーマンスを実現しました。CFQ のパフォーマンスは良好ではありませんでした。したがって、PowerMax ストレージ システムで Oracle データベースに Linux I/O スケジューラを選択する場合は、Deadline を使用することをお勧めします。

Linux I/O スケジューラを使用する代わりに、blk-mq（MQ）を使用した Linux ブロック I/O ドライバーの新しい機能拡張を使用できます。詳細については、付録 Iblk-mq と scsi-mq を参照してください。FC-NVMe を使用する場合、MQ はデフォルトで有効になります。

付録

付録 Iblk-mq と scsi-mq

blk-mq とは

Linux ブロック デバイスの I/O レイヤーは、HHD のパフォーマンスを最適化するよう設計されています。これは、ブロック デバイスあたりの I/O 送信の単一キュー（SQ）、I/O が送信キューで送信、削除、順序変更されるたびにすべての CPU コアで共有される単一のロック機構、および非効率なハードウェア割り込み処理に基づいていました。詳細については、『Linux Block IO: Introducing Multi-queue SSD Access on Multi-core Systems』と『Improving Block-level Efficiency with scsi-mq』を参照してください。

プライマリー ストレージ（フラッシュ ストレージ デバイス）としての Non-Volatile Memory（NVM）の使用が増えるにつれ、I/O ボトルネックはストレージ メディアからサーバーI/O レイヤーにシフトしました。この移行により、新たな設計への扉が開きました。それはブロック マルチキュー（MQ）で、blk-mq とも呼ばれます。

メモ： SCSI タイプのブロック デバイス（`/dev/sd`）を使用した blk-mq の実装は、scsi-mq と呼ばれます。

---

blk-mq には 2 レイヤー設計が導入されています。この設計では、各ブロック デバイスが複数のソフトウェア I/O 送信キュー（CPU コアあたり 1 つ）を備え、最終的にデバイス ドライバーの 1 つのキューに展開されます。キューの処理は、I/O を送信するコアが処理する FIFO の順序に基づいています。割り込みや共有ロック メカニズムは不要になりました。

現在の設計では、I/O パターンがランダムまたはシーケンシャルどちらでも NVM メディアのパフォーマンスは影響を受けないため、I/O の順序変更（スケジューラ）は省略されます。ただし、I/O スケジューリングは、「`mq-deadline`」（RHEL 8）などのカーネル I/O スケジューラを使用して導入できます。

VMAX All Flash と PowerMax ストレージ システムで blk-mq をテストしています。サーバーがボトルネック（古い CPU、高いサーバー使用率）であった場合、MQ は優れたパフォーマンスとサーバー効率のメリットをもたらしました。新しいサーバー、CPU、SAN（32Gb）では、このようなメリットはありませんでした。

また、FC-NVMe プロトコルを使用する場合、MQ は`/dev/nvme` デバイスに対してデフォルトですでに有効になっています。フラッシュ ストレージの採用が進んでいることを背景に、RHEL 8 以降では、FC プロトコル（`/dev/sd` デバイス用）を使用している場合でも、MQ がデフォルトで有効になっています。サーバーがボトルネックになる可能性がある FC 環境で、デフォルトでまだ有効になっていない場合は、blk-mq を試すことをお勧めします。

blk-mq の使用について、次の点に注意してください。

- blk-mq/scsi-mq の PowerPath サポートについては、Linux 向け PowerPath ファミリーのリリース ノートを参照してください。

- 当社のテストでは、Linux のネイティブ マルチパスは blk-mq では正常に動作していますが、Linux カーネルのバージョンが 4.x である場合に限られます。たとえば、OL/UEK7.4 以降または RHEL8.0 以降の場合です。
- OL8.x および RHEL8.x では、blk-mq/scsi-mq はすでにデフォルトで有効になっています。
- blk-mq は簡単に有効化/無効化できます。したがって、Linux でカーネル 4.x と FC プロトコルを使用している VMAX All Flash および PowerMax のお客様は、blk-mq がハイパフォーマンス データベースにパフォーマンスとサーバー効率のメリットを提供するかどうかを確認してください。

blk-mq の有効化または無効化

Linux カーネルは、NVMe プロトコルで提示されるデバイス（`/dev/nvme` として表示されるデバイス）に対してデフォルトで blk-mq を有効にします。FC プロトコルを介して提示されるデバイス（`/dev/sd` として表示されるデバイス）の場合、blk-mq が有効になっていない可能性があります。

FC デバイスで MQ が有効になっているかどうかを確認するには、次のコマンドを実行します。最初のコマンドは MQ が一般的に有効になっているかどうかを判断し、2 番目のコマンドはデバイス マッパー マルチパスを使用している場合にのみ有効です。

```
# cat /sys/module/scsi_mod/parameters/use_blk_mq
N
# cat /sys/module/dm_mod/parameters/use_blk_mq
N
```

FC デバイスで blk-mq を有効にするには、`/boot/grub2/grub.cfg` を更新します。これを行うには、以下に示すように GRUB_CMDLINE_LINUX パラメーターを編集します。前述したとおり、dm_mod.use_blk_mq パラメーターは、デバイス マッパーを使用している場合にのみ有効です。

```
# vi /etc/default/grub
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="crashkernel=auto resume=UUID=7b5d3708-53b9-482e-80f6-
01d4086f30b2 rhgb quiet scsi_mod.use_blk_mq=1 dm_mod.use_blk_mq=y"
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true
```

`scsi_mod.use_blk_mq=1` パラメーターを指定すると、カーネル レベルで SCSI タイプのブロック デバイスに対して blk-mq が有効になります（ここで `0` を指定すると無効になります）。

`dm_mod.use_blk_mq=y` パラメーターを指定すると、デバイス マッパー（DM）の Linux ネイティブ マルチパスに対して blk-mq が有効になります（ここで `n` を指定すると無効になります）。

grub.cfg ファイルを再作成し、変更を有効にするためにサーバーを再起動してください。

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

付録 II.Oracle ASM オンライン ストレージの再利用

削除されたデータとストレージの再利用

データベース クライアントがデータを削除すると、ASM は追加の空き容量を認識しますが、ストレージ システムは認識しません。削除されたデータに基づくストレージ エクステントは、割り当てられたままになります。ストレージ エクステントは次の方法で再利用できます。

- ストレージ デバイスを除去して削除します。
- `symsg free` または `symdev free` コマンドを（デバイスがサーバーに対して not-ready になった後に）実行します。これにより、実際のデバイスではなくデバイス全体の内容が消去されます。
- デバイス全体またはパーティションで Linux `blkdiscard` コマンドを使用し、SCSI unmap コマンドを使用して容量を効率的に解放します。この方法でも、デバイスまたはパーティションのコンテンツが再び消去されますが、最初にデバイスをサーバーから削除する必要はありません。
- `symsg reclaim` または `symdev reclaim` コマンドを実行します。これらのコマンドで、ゼロ化されたすべてのストレージ エクステント（128 KB）が解放（再利用）されます。ただし、削除された ASM エクステントにゼロは含まれません。

Oracle ASM フィルター ドライバーのオンライン ストレージの再利用では、ASM ディスク グループがオンラインおよびアクティブな状態なまま、削除されたデータのストレージ スペースを再利用できます。

ASM フィルター ドライバー

ASM フィルター ドライバー（AFD）は、Oracle ASM ディスクの I/O パスに存在するカーネル モジュールです。このドライバーは、Grid Infrastructure リリース 12 から利用できます。

AFD には、ASM より多くのメリットがあります。次のようなメリットがあります。

- **Oracle プロセスで発生していない書き込みから ASM デバイスを保護**：たとえば、次のコマンドのいずれかを実行しても ASM ディスクに問題は生じません。
  - `dd if=/dev/zero of=<my_ASM_device>`
  - `blkdiscard <my_ASM_device>`

  保護は、AFD ラベルがデバイスから削除された場合にのみ無効になります。
- **ASM デバイスのラベル付けによって管理が容易に**：ASM ディスク グループを作成するときに、AFD はディスク グループ名とディスク番号に基づく自動ラベルを付けることができます。ユーザーが自分でラベルを付けることもできます。
- **ASM ディスク グループのストレージ オンラインの再利用：**ASM ディスク グループをオンライン状態にしたままで、ASM ディスク グループとストレージ システム内で削除されたスペースが再利用されます。

AFD オンライン ストレージの再利用を使用する場合

ASM では、削除されたスペースを効率的に再利用できます。たとえば、データ ファイルが削除され、同じ容量の新しいファイルが作成された場合は、削除されたストレージ スペースを再利用する必要はありません。ASM はそのスペースを再利用するだけです。

従来のデータベースや多くの容量を消費するデータベースのコピーなど大規模なデータ セットが ASM で削除された場合は、アレイ内のこのストレージを再利用して、他のアプリケーションで利用することができます。

ASM ストレージの再利用中、ASM はまず手動でリバランシングを実行します。これによって削除されたデータで作成されたギャップに ASM エクステントが移動し、ASM ディスク グループが最適化（圧縮）さ

れます。ASM ディスク グループが圧縮されると、High Watermark（HWM）はその新しい割り当て容量に基づいて更新されます。次に、ASM は SCSI unmap コマンドをストレージ システムに送信して、新しい HWM より上のスペースを再利用します。再利用は効率的かつ高速に行われます。

AFD オンライン ストレージの再利用を使用する方法

AFD ストレージの再利用を使用するには、ASM ディスク グループに 12.1（またはそれ以降）の互換性設定が必要です。

AFD ストレージの再利用を有効にするには、次のコマンドを使用して ASM ディスク グループに THIN 属性を与えます。

```
ALTER DISKGROUP <NAME> SET ATTRIBUTE 'THIN_PROVISIONED'='TRUE';
```

この後、次のコマンドを使用して、ASM ディスク グループのストレージを何度でも再利用できます。

```
ALTER DISKGROUP <NAME> REBALANCE WAIT;
```

WAIT オプションを使用すると、操作が完了したときにのみプロンプトが表示されます。

AFD オンライン ストレージの再利用の例

次の例は、AFD オンライン ストレージの再利用の値を示しています。

まず、容量の削減を示すために空の ASM ディスク グループを使用します（実環境では、ASM ディスク グループは空になりません）。

1. ASM ディスク グループに THIN 属性を指定します。

```
ALTER DISKGROUP TEST SET ATTRIBUTE
'THIN_PROVISIONED'='TRUE';
```

2. 300 GB のテーブルスペースを追加します。

```
CREATE BIGFILE TABLESPACE TP1 DATAFILE '+TEST' size 300G
ONLINE;
```

新しいデータ ファイルが作成されると、Oracle はスペースを初期化して、そのスペースはストレージ システムに割り当てられます。

3. テーブルスペースを削除します。

```
DROP TABLESPACE TP1 INCLUDING CONTENTS AND DATAFILES;
```

ストレージ レベルでは、次の図に示すように、スペースはまだ解放されていません。

```
      S Y M M E T R I X   T H I N   D E V I C E S
-------------------------------------------------------
                                          Total      Comp
        Bound       Flgs       Total     Allocated   Ratio
Sym   Pool Name     EMPT         GBs        GBs (%)
----- ------------ ---- ---------- ---------- --- ------
00151 -            F..B      150.0       75.0  50  1.0:1
00152 -            F..B      150.0       75.1  50  1.0:1
00153 -            F..B      150.0       75.1  50  1.0:1
00154 -            F..B      150.0       75.0  50  1.0:1

Total                     ---------- ---------- --- ------
  GBs                         600.0      300.2  50
```

図 28.　AFD を再利用する前の PowerMax ストレージ グループ

4. ASM ディスク グループをリバランシングします。

```
ALTER DISKGROUP TEST REBALANCE WAIT;
```

次の図は、小さい ASM メタデータ（空の ASM ディスク グループで開始したため）以外、スペースは消費されていないことを示しています。

```
      S Y M M E T R I X   T H I N   D E V I C E S
-------------------------------------------------------
                                          Total      Comp
        Bound       Flgs       Total     Allocated   Ratio
Sym   Pool Name     EMPT         GBs        GBs (%)
----- ------------ ---- ---------- ---------- --- ------
00151 -            F..B      150.0        0.0   0  1.0:1
00152 -            F..B      150.0        0.1   0  1.0:1
00153 -            F..B      150.0        0.1   0  1.0:1
00154 -            F..B      150.0        0.1   0  1.0:1

Total                     ---------- ---------- --- ------
  GBs                         600.0        0.2   0
```

図 29.　AFD 再利用後の PowerMax ストレージ グループ

付録 III.PowerMax の圧縮と重複排除の概要 http://kernel.dk/systor13-final18.pdf

次のセクションでは、PowerMax でのサーバーI/O の圧縮と重複排除について詳しく説明します。

データベース サーバーから新しい書き込みが届くと、次の図に示すように、それらの書き込みは PowerMax キャッシュに登録され、サーバーに対してすぐに確認されて、書き込みレイテンシーが低くなります。

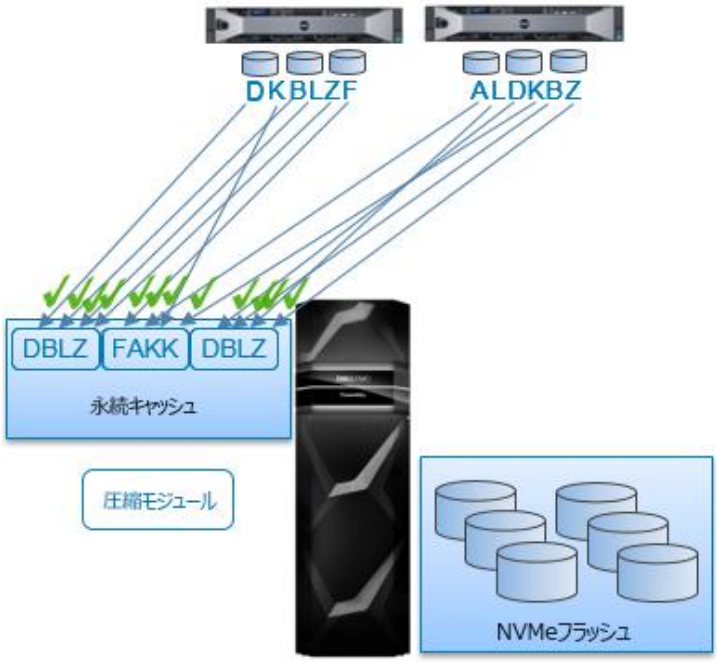

図 30.　重複排除ステップ 1：サーバーがレジスタを PowerMax キャッシュに書き込む

PowerMax キャッシュは永続的であるため、データを NVMe フラッシュ メディアにすぐに書き込む必要はありません。Oracle は、同じまたは隣接するデータベース ブロックへの書き込みを複数回継続できます。

PowerMax が NVMe フラッシュ ストレージにデータを書き込むときに、そのストレージ グループで圧縮が有効な場合、新しいデータを含む 128 KB のキャッシュ スロットがハードウェア圧縮モジュールに送信され、データはそこで圧縮されて、ハッシュ ID が生成されます。

次の図に示すように、キャッシュ スロットの一意性がテストされて、実際に一意である場合は、データの圧縮バージョンが適切な圧縮プールに格納され、シン デバイス ポインタが更新されてデータの新しい場所を指し示します。

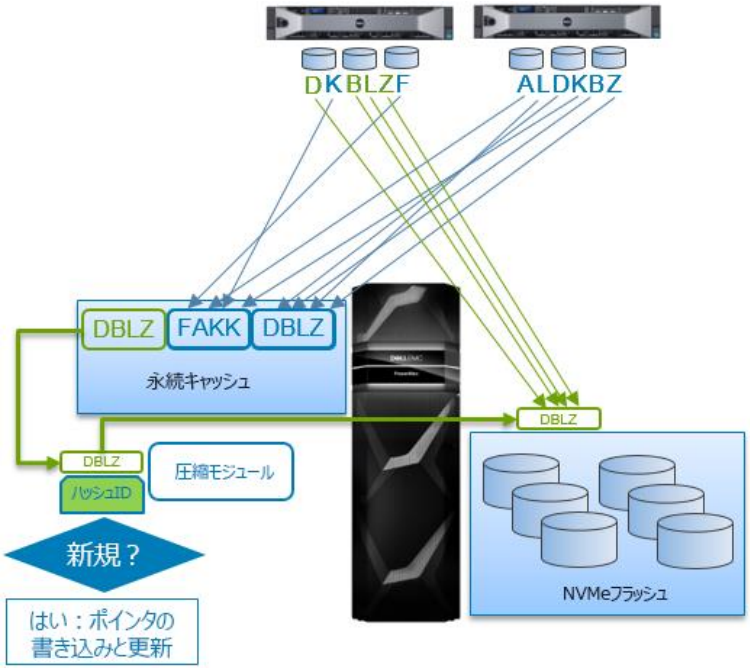

図 31.　重複排除ステップ 2：キャッシュ スロットが圧縮され、一意性がチェックされる

圧縮されたデータが一意でない場合、つまりその同じデータの以前の同一コピーがすでに圧縮された状態で PowerMax に保存されている場合、データが再び保存されることはありません。代わりに、次の図に示されているように、シン デバイスのポインタが更新され、データの既存の圧縮バージョンを指し示すだけです。

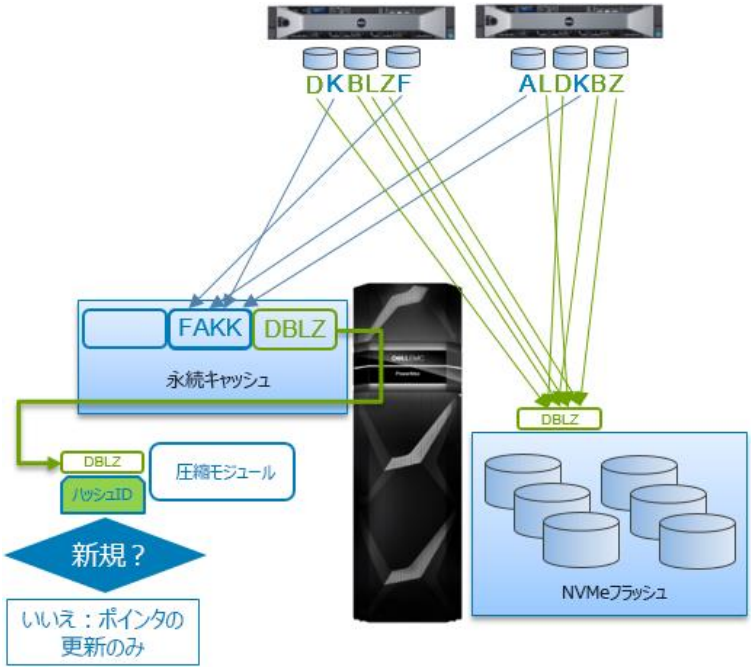

図 32.　重複排除ステップ 3：一意でないデータの重複排除

この例では重複排除（同じデータの複数のコピーを PowerMax ストレージ システムに 1 回だけ保存する）の機能を示します。

付録 IV.Linux iostat コマンド

読み取り I/O レイテンシーが予想よりも高い場合は、Linux `iostat` コマンドを使用して調査することをお勧めします。次のコマンドを実行します。

`iostat -xtzm <インターバル> <イテレーション> [optional: <スペースで区切られた特定のデバイスのリスト>]`

出力をファイルにキャプチャします。何度か繰り返した後、コマンドを停止してファイルを検査します。最初のインターバルを無視し、インターバル 2 以上に焦点を当てます。

ファイルのサイズが大きいため、モニターする特定のデバイスを含めるか、1 つの Oracle データ ファイルの仮想名を見つけて、次の図のように 1 つのデバイス[10]に焦点を当てます。

```
Device:         rrqm/s   wrqm/s     r/s     w/s   rsec/s   wsec/s avgrq-sz avgqu-sz   await  svctm  %util
dm-395            0.00     0.00 2145.33  937.33    16.76     7.51    16.13     2.99    0.97   0.28  87.77
```

デバイスにキューイングされる I/O の数（`avgqu-sz`）、キュー タイムを含む I/O の処理にかかる時間（`await`）、キューを離れるときの I/O の処理にかかる時間（`svctm`）を書き留めます。`await` 時間が長く、`svctm` 時間が短い場合は、サーバーがキューイングの問題を抱えている可能性があり、さらに多くの LUN（追加の I/O キュー）とおそらくストレージへのパスも必要です。`svctm` 時間が長い場合は、SAN またはストレージ性能のボトルネックを示している可能性があります。

次の表に、一部の `iostat` メトリックの概要とその使用方法に関するアドバイスを示します。

表 8. Linux `iostat` と flags xtz - メトリックの概要

メトリック	説明	コメント
デバイス	`/dev` ディレクトリーにリストされているデバイス名	マルチパスを使用する場合、各デバイスには仮想名（`dm-xxx` または `emcpowerxxx` など）があり、各パスにはデバイス名（`/dev/sdxxx` など）があります。仮想名を使用して、すべてのパスにわたって集計されたメトリックを調べます。
r/s、w/s	デバイスに対して発行された 1 秒あたりの読み取り/書き込み要求の数。	r/s と w/s は、デバイスのサーバーIOPS 要求を提供します。これらのメトリック間の比率は、読み取り/書き込み比を示します。
rMB/s、wMB/s	デバイスに対する 1 秒あたりの読み取り/書き込みの MB 単位の量（セクターあたり 512 バイト）	デバイスの帯域幅性能を確認します。 rMB/s を r/s で割ることによって、平均読み取り I/O サイズを確認できます。 同様に、読み取られた rMB/s を w/s で割ることによって、平均書き込みサイズを判断できます。
avgrq-sz	デバイスに発行された要求の平均サイズ（セクター単位）	ほとんどのパフォーマンスの問題では、キューのサイズはさほど重要となりません。次のパラメーター「avgqu-sz」に焦点を当てる必要があります。

[10] Oracle ASM ストライピングでは、I/O をすべてのデータ デバイスに均等に分散する必要があります。

メトリック	説明	コメント
avgqu-sz	デバイスに発行された要求の平均キュー長	デバイスにキューイングされたリクエストの数。キューが大きい場合、レイテンシーが増加します。デバイスがサービス レベルの低い SG にある場合は、サービス レベルの向上を検討してください。それ以外の場合、ストレージ システムが過剰に使用されていない場合は、サーバーレベルでの I/O 分散を向上させるためにデバイスまたはパスの追加を検討してください。
await	デバイスに対して発行された I/O 要求の平均時間（ミリ秒）。これには、キュー内の要求に費やされた時間と、それらの処理に費やした時間が含まれます。	キューイング時間を含む `await` 時間が `svctm` 時間よりはるかに長い場合は、サーバーのキューイングの問題を示している可能性があります。前述の avgqu-sz メトリックを参照してください。
svctm	デバイスに発行された I/O 要求の平均サービス時間（ミリ秒単位）	アクティブ デバイスの場合、`await` 時間は、予想されるサービス レベルの時間内である必要があります（たとえば、フラッシュ ストレージの場合は 1 ミリ秒以下、15k rpm ドライブの場合は最大 6 ミリ秒など）。

付録 VOracle AWR I/O 関連情報

ピーク ワークロード期間の AWR レポートを収集して、潜在的なボトルネックを特定します。AWR は、レポート期間中にすべてのメトリックを平均するため、24 時間のレポートは一般には役に立たちません。有効な AWR レポートは、ワークロードが安定し、高い状態の短時間（15 分、30 分、1 時間など）について生成されます。

Oracle RAC を使用している場合は、各インスタンスに対して個別に、またはクラスター全体に対して AWR レポートを作成できます。インスタンス AWR のメトリックは、その特定のデータベース サーバーのワークロードのみを表します。RAC AWR のメトリックは、クラスタ全体のワークロードを表します。次の例に、両方のタイプを示します。Oracle のリリースごとに、レポートに変更が加えられています。

AWR ロードプロファイル

インスタンス AWR レポートの［Load Profile］領域には、次の図に示すように、「Physical reads (blocks)」、「Physical writes (blocks)」、「Logical reads」メトリックが含まれます。これらのメトリックの単位はデータベース ブロックです。ブロック ユニットを I/O メトリックに直接変換することはできません。ただし、これらの数値により、読み取り/書き込み比や、バッファ キャッシュ（論理読み取り）と実際の読み取り I/O（物理読み取り）で満たされる読み取り回数の比較など、データベース I/O プロファイルを把握できます。

通常は、高い割合の OLTP 読み取りワークロードがデータベース キャッシュ（論理読み取り）によって満たされることが予測されます。ベンチマークでは、より多くの I/O を生成するデータベース キャッシュのサイズが制限され、数値は互いに近くなります。

Oracle 12c データベースの AWR レポートでは、ロード プロファイルに実際の I/O メトリックも示されます（ハイライト表示されたメトリックの 2 番目のグループで示されています）。

	Per Second	Per Transaction	Per Exec	Per Call
DB Time(s):	80.9	0.2	0.04	87.89
DB CPU(s):	6.7	0.0	0.00	7.24
Background CPU(s):	1.3	0.0	0.00	0.00
Redo size (bytes):	10,801,337.4	20,760.1		
Logical read (blocks):	145,908.9	280.4		
Block changes:	68,121.0	130.9		
Physical read (blocks):	132,661.0	255.0		
Physical write (blocks):	36,090.4	69.4		
Read IO requests:	132,660.8	255.0		
Write IO requests:	35,774.8	68.8		
Read IO (MB):	1,036.4	2.0		
Write IO (MB):	282.0	0.5		
IM scan rows:	0.0	0.0		
Session Logical Read IM:	0.0	0.0		
Global Cache blocks received:	6.7	0.0		
Global Cache blocks served:	5.5	0.0		
User calls:	0.9	0.0		
Parses (SQL):	1.2	0.0		
Hard parses (SQL):	0.0	0.0		
SQL Work Area (MB):	0.1	0.0		
Logons:	0.2	0.0		
Executes (SQL):	2,081.3	4.0		
Rollbacks:	0.0	0.0		
Transactions:	520.3			

図 33.　単一のインスタンス AWR レポートの［Load Profile］セクション

クラスターAWR レポートにも、次の図に示すように同様の情報が表示されます。

System Statistics - Per Second

I#	Logical Reads/s	Physical Reads/s	Physical Writes/s	Redo Size (k)/s	Block Changes/s	User Calls/s	Execs/s	Parses/s	Logons/s	Txns/s
1	145,908.93	132,661.01	36,090.44	10,548.18	68,121.04	0.92	2,081.30	1.22	0.15	520.29
2	143,178.74	131,490.11	36,014.32	10,523.52	68,144.06	1.10	2,073.29	1.32	0.15	518.29
3	144,388.30	121,627.96	33,794.64	9,732.37	63,366.03	0.88	1,924.26	1.22	0.15	481.03
4	43.82	1.16	0.49	1.77	5.35	0.49	0.85	0.75	0.14	0.01
Sum	433,519.79	385,780.24	105,899.88	30,805.83	199,636.47	3.40	6,079.70	4.51	0.58	1,519.61
Avg	108,379.95	96,445.06	26,474.97	7,701.46	49,909.12	0.85	1,519.92	1.13	0.15	379.90
Std	72,232.72	64,486.05	17,681.74	5,147.09	33,344.97	0.26	1,015.29	0.25	0.00	253.91

図 34.　クラスターAWR レポート：ロード プロファイル

AWR の上位のフォアグラウンド イベント

データベースは CPU と I/O に対してほとんどの時間待機するのが理想的です。この状態はシステムが物理的な制限に達して稼働していることを示します。AWR レポートの `db file sequential read` フィールド（実際にはランダム読み取りを意味する）に、ストレージ タイプとアプリケーションのニーズに適した平均待機時間が示されているか確認します。たとえば、次の図では 596 マイクロ秒（0.6 ミリ秒）の I/O レイテンシーになっています。

Top Timed Events

	Wait		Event		Wait Time			Summary Avg Wait Time				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg Wait	%DB time	Avg	Min	Max	Std Dev	Cnt
*	User I/O	db file sequential read	693,729,747	0.00	413,493.42	596.04us	95.71	562.43us	460.79us	605.79us	68.10us	4
*		DB CPU			38,904.63		9.01					4
*	System I/O	log file parallel write	3,016,244	0.00	3,746.84	1.24ms	0.87	1.14ms	864.59us	1.35ms	205.44us	4
*	Configuration	free buffer waits	194,424	0.00	1,212.68	6.24ms	0.28	6.06ms	5.43ms	6.49ms	558.35us	3
*	System I/O	db file parallel write	9,081,166	0.00	971.55	106.99us	0.22	182.77us	93.48us	415.78us	155.58us	4
*	User I/O	read by other session	1,135,914	0.00	670.04	589.87us	0.16	590.16us	577.44us	597.56us	11.06us	4
*	Other	LGWR any worker group	436,390	0.00	366.63	840.14us	0.08	838.70us	820.94us	856.46us	25.12us	4
*	Cluster	gc cr grant 2-way	1,360,847	0.00	191.31	140.58us	0.04	144.66us	136.86us	158.86us	10.03us	4
*	Other	RMA: IPC0 completion sync	7,542	0.00	146.36	19.41ms	0.03	19.41ms	19.39ms	19.42ms	14.07us	4
*	Other	LGWR worker group ordering	98,744	0.00	96.95	.98ms	0.02	.98ms	759.74us	1.19ms	305.37us	4
1	User I/O	db file sequential read	238,457,150	0.00	140,648.51	589.83us	96.46					
1		DB CPU			12,009.95		8.24					
1	System I/O	log file parallel write	1,049,286	0.00	1,417.43	1.35ms	0.97					
1	System I/O	db file parallel write	3,536,292	0.00	402.01	113.68us	0.28					
1	User I/O	read by other session	372,299	0.00	221.69	595.46us	0.15					
1	Other	LGWR any worker group	235,860	0.00	202.00	856.46us	0.14					
1	Configuration	free buffer waits	20,671	0.00	112.19	5.43ms	0.08					
1	Other	RMA: IPC0 completion sync	1,885	0.00	36.55	19.39ms	0.03					
1	Other	LGWR worker group ordering	47,967	0.00	36.44	759.74us	0.02					
1	Application	enq: TX - row lock contention	1,685	0.00	30.62	18.17ms	0.02					
2	User I/O	db file sequential read	236,444,916	0.00	140,281.50	593.29us	96.21					
2		DB CPU			11,943.30		8.19					
2	System I/O	log file parallel write	930,363	0.00	1,064.95	1.14ms	0.73					
2	System I/O	db file parallel write	3,496,389	0.00	378.03	108.12us	0.26					
2	[illegible]	free buf[illegible]aits	54,05[illegible]	[illegible]	[illegible]	[illegible]	[illegible]					

図 35. クラスターAWR レポート：上位の時間計測イベント

`log file parallel write` メトリックは、Oracle Log Writer が REDO ログをストレージに書き込む速度を示します。Oracle は、負荷（最大 1 MB のサイズ）に基づいて異なる I/O サイズでの REDO ログの書き込みを生成できます。I/O が大きいほど、完了までに長い時間がかかります。ただし、この例では、`log file parallel write` メトリックは 1.24 ミリ秒の書き込みレイテンシーを示しています。これは、大規模な I/O においては素晴らしい数字です。

AWR データ ファイルの読み取り/書き込み I/O メトリック

AWR レポートで IOPS と MB/秒の I/O 関連メトリックを見つけるには、`physical read total IO requests`、`physical write total IO requests`、`physical read total bytes`、`physical write total bytes` のメトリックを確認します。これらのメトリックは、読み取り IOPS、書き込み IOPS、読み取り帯域幅、書き込み帯域幅を示します。

次の図は、クラスターが 1 秒あたり 385,808 の読み取り I/O、1 秒あたり 108,197 の書き込み I/O、2.96 GB/秒の読み取り帯域幅（3,179,107,539 / 1024 / 1024 / 1024 でバイト/秒から GB/秒に変更）、0.84 GB/秒の書き込み帯域幅（900,994,499 / 1024 / 1024 / 1024）を実行したことを示しています。当然ですが、DSS ワークロード中は帯域幅の方が関心が高くなります。

System Statistics (Global)

				per Second			
Statistic	Total	per Second	per Trans	Average	Std Dev	Min	Max
...							
physical read IO requests	695,479,821	385,769.18	253.86	96,442.30	64,484.59	1.16	132,660.80
physical read bytes	5,697,533,935,616	3,160,311,685.17	2,079,679.61	790,077,921.29	528,269,759.32	9,529.13	1,086,758,960.36
physical read total IO requests	695,550,651	385,808.47	253.89	96,452.12	64,483.72	12.72	132,673.20
physical read total bytes	5,731,419,739,648	3,179,107,539.01	2,092,048.40	794,776,884.75	527,375,543.67	6,402,809.29	1,093,548,670.12
physical read total multi block requests	32,948	18.28	0.01	4.57	2.64	0.68	6.39
physical reads	695,499,748	385,780.24	253.87	96,445.06	64,486.05	1.16	132,661.01
physical reads cache	695,480,034	385,769.30	253.86	96,442.32	64,484.63	1.16	132,661.01
physical reads cache prefetch	1,632	0.91	0.00	0.23	0.17	0.00	0.41
physical reads direct	19,713	10.93	0.01	10.93		10.93	10.93
physical reads direct (lob)	17	0.01	0.00	0.01		0.01	0.01
physical reads direct temporary tablespace	19,696	10.93	0.01	10.93		10.93	10.93
physical write IO requests	189,229,458	104,961.94	69.07	26,240.49	17,525.92	0.36	35,774.79
physical write bytes	1,564,020,015,104	867,531,829.36	570,889.19	216,882,957.34	144,848,823.42	4,021.59	295,652,855.92
physical write total IO requests	195,062,463	108,197.40	71.20	27,049.35	18,062.80	0.87	36,894.04
physical write total bytes	1,624,347,823,104	900,994,499.05	592,909.68	225,248,624.76	150,433,900.75	12,754.92	307,110,805.44
physical write total multi block requests	1,102	0.61	0.00	0.15	0.18	0.01	0.41
...							

図 36.　クラスターAWR レポート：システム統計

AWR と REDO ログ スイッチ

REDO ログは、Oracle データベースの復元性とパフォーマンスの鍵となります。ログへの Oracle 書き込みサイズは、512 バイトから最大 1 MB までです。Oracle は、ログ バッファとログ ファイルの空き容量や時刻など、複数の条件に基づいて、次のログ ファイルに切り替えます。

Oracle が 1 時間あたりわずかな回数しかログ ファイルを切り替えないように REDO ログのサイズを構成します。ログ ファイルの数が十分にあることを確認して、アーカイブ プロセスがログの切り替え時に完了するのを待たないようにします。

ログの切り替え回数は、次の図に示すようにインスタンス AWR レポートに表示されます。`Total` の数値は AWR レポート期間に発生した切り替え回数を示しています。`per Hour` 値は、AWR レポート中のアクティビティに基づいて導かれた見積り数です。

Instance Activity Stats - Thread Activity

Statistic	Total	per Hour
log switches (derived)	2	3.99

図 37.　インスタンス AWR レポート：log_switch