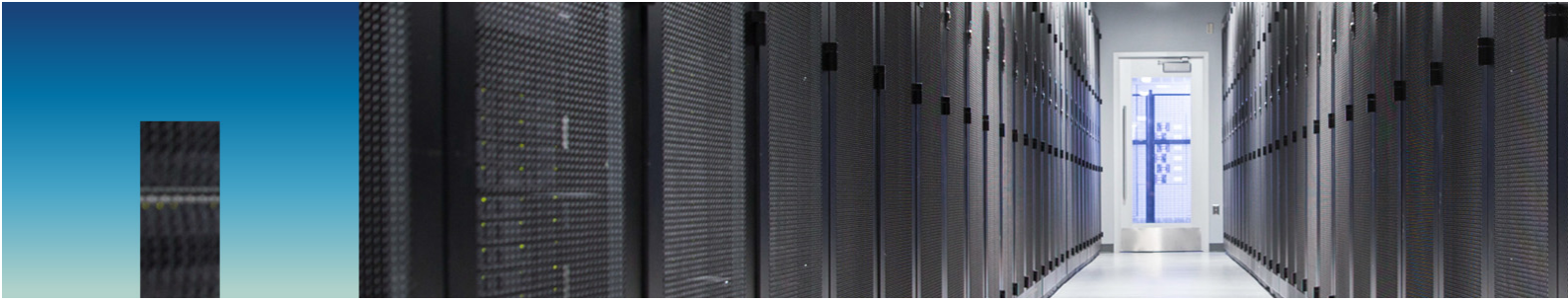


テクニカル レポート

Data ONTAP を基盤にした Oracle データベース

ネットアップ、Jeffrey Steiner
2016 年 11 月 | TR-3633

重要

本レポートに指定された環境、構成、バージョンがお客様の環境に対応しているかどうかは、Interoperability Matrix Tool（IMT）を参照してください。

NetApp®

目次

表一覧

図一覧

1 はじめに

NetApp® clustered Data ONTAP®は、インライン圧縮、ハードウェアの無停止アップグレード、他社製ストレージ アレイからの LUN インポートなど、様々な機能を標準搭載した強力なデータ管理プラットフォームです。クラスタは最大で 24 ノード構成が可能なうえ、データ サービスに、Network File System（NFS）、Common Internet File System（CIFS）、iSCSI、Fibre Channel（FC）、Fibre Channel over Ethernet（FCoE）のプロトコルを同時に使用できます。また、NetApp Snapshot®テクノロジをベースに、何万ものオンライン バックアップや完全に動作可能なデータベース クローンを作成することもできます。

これら Data ONTAP の豊富な機能セットに加えて、ユーザにはデータベースのサイズ、パフォーマンス要件、データ保護のニーズなど、様々な要件が存在します。ネットアップ ストレージは、VMware ESX の仮想環境で稼働する約 6,000 のデータベースから、996TB のシングル インスタンス データ ウェアハウス（規模は拡大中）まで、あらゆる環境に導入されています。そのため、ネットアップ ストレージを基盤にして Oracle データベースを構築するにあたって、明確なベストプラクティスというものはほとんどありません。

本ドキュメントでは、ネットアップ ストレージ環境で Oracle データベースを運用するにあたっての要件を、2 つの方法で解説します。1 つは、明確なベストプラクティスがある場合、それを具体的に紹介する方法。もう 1 つは、設計の際に考慮すべき多数の事柄を確認していく方法です。Oracle 向けストレージ ソリューションの設計者は、それぞれのビジネス要件を基に、この考慮事項に対処しなければなりません。

本ドキュメントではまず、すべての環境に共通する一般的な考慮事項を説明し、続いて、使用する仮想化ソリューションや OS ごとに固有の推奨事項を解説します。ファイルシステムのレイアウトの選択や NFS ロックの解除など、特殊なトピックについては付録で取り上げます。

解説は主に、clustered Data ONTAP 環境が前提になっていますが、7-Mode システムに当てはまる部分も多数あります。

2 NetApp ONTAP の導入オプション

NetApp ONTAP ソフトウェアはネットアップ データ ファブリックの基盤です。データベースの運用という観点から考えれば、ONTAP によって、必要なときにどこからでもデータにアクセスできるようになるということです。例えば、オンプレミスの All Flash FAS（AFF）システムで稼働しているミッションクリティカルなデータベースを、ハイパースケール クラウド プロバイダの ONTAP Cloud 環境にレプリケートし、そこからクローンを何十個も作成して開発に利用するといったことが可能になります。規模の小さいオフィスの場合は、既存のハードウェアに ONTAP Select を導入することで、データセンターにあるメインのシステムと同じようにデータを管理できます。

Oracle データベースを含め、どのワークロードを実行する場合でも、実際のベストプラクティスは ONTAP の導入方法によって大きく異なります。ONTAP はどこで稼働しようと ONTAP ですが、ONTAP の導入オプションの選択には、ビジネス要件、クラウド戦略、レプリケーション要件、SLA、サイト間で使用できる帯域幅が大きく関わってきます。

以降のセクションでは、様々なオプションの核となる条件を解説します。本書にない詳細については、公式の製品ドキュメントをご覧になるか、ネットアップの担当者にお問い合わせください。

2.1 ONTAP 搭載の All Flash FAS と FAS コントローラ

ONTAP 搭載の AFF や FAS コントローラは、パフォーマンスとデータの制御性で常に業界をリードするソリューションです。このソリューションは、標準的なオプションとして 20 年以上にわたり、数千のお客様に利用されています。ONTAP が提供するソリューションはあらゆる環境に対応可能で、その種類は、ミッションクリティカルな 3 つのデータベースが稼働する環境から、6 万のデータベースが稼働するサービス プロバイダ環境、ペタバイト規模のデータベースの瞬時のリストア、1 つのデータベースから作成された数百個のクローンをサポートするデータベース サービスと多岐にわたります。

2.2 クラウド向け NetApp Private Storage

NetApp Private Storage（NPS）は、大量のデータが発生するワークロードをパブリック クラウドで処理したいというニーズに応えるため、ネットアップが投入したオプションです。パブリック クラウド ストレージ オプションは多数ありますが、パフォーマンス、制御性、拡張性に限りがあるものがほとんどで、データベース ワークロードに関して言えば、次の点が大きな足かせになります。

- パブリック クラウド ストレージ オプションでは、最新のデータベース ワークロードに必要な IOPS レベルまで拡張したくても、コストや効率性、管理性のために拡張できない。
- パブリック クラウド プロバイダの IOPS 機能が物理的に要件を満たしていても、大抵の場合、I/O レイテンシがデータベース ワークロードの要件に合わない。データベースをオールフラッシュ ストレージ アレイに移行したり、レイテンシをミリ秒単位ではなくマイクロ秒単位で測定すると、この状況がますます当てはまる。
- パブリック クラウド ストレージの可用性は概ね優れているが、ミッションクリティカルな環境の厳しい要件を満たせるほどではない。
- パブリック クラウド ストレージ サービスにもバックアップとリカバリ機能があるが、大半のデータベースに求められるゼロの RPO やほぼゼロの RTO を達成できることはほとんどない。データベースのデータ保護には、Snapshot をベースにした文字通り瞬時のバックアップとリカバリが必要。クラウド内のどこかにデータを転送するバックアップとリカバリでは不十分。
- ハイブリッド クラウド環境では、オンプレミスとクラウド ストレージ システムの間でデータを移動できることが必須条件。ストレージ管理のための共通の基盤も必要。
- 多くの政府機関はデータ主権を法律で厳格に定めており、国外にデータを持ち出すことを禁じている。

NetApp Private Storage システムは、パブリック クラウド プロバイダ（Amazon AWS、Microsoft Azure、IBM SoftLayer）に最大限のストレージ パフォーマンス、制御性、柔軟性を提供します。これは、データセンターに配置した AFF や FAS システムをパブリック クラウドに直接接続することで実現します。そのため、ハイパースケーラ ストレージの制限を一切受けることなく、ハイパースケーラ コンピューティング レイヤの能力をフルに活用できます。さらに、アプリケーションのバイナリ、データベース、データベースのバックアップ、アーカイブなど、すべてのデータが常にシステムに格納されているので、クラウドに依存しないマルチクラウド アーキテクチャが実現します。時間や帯域幅、費用をかけて、異なるクラウド プロバイダ間でデータを移動する必要はありません。

お客様の中には、NPS モデルを使用して独自の取り組みを進めている企業もいくつかあるほどです。例えばよく見るのが、自社のデータセンター施設から、ハイパースケール クラウド プロバイダの 1 つにいつでも高速アクセスできるようにする使い方です。別の例では、ハイパースケール クラウド プロバイダへの高速アクセスに、その機能を備えたコロケーション施設を使用しているお客様もいらっしゃいます。この場合、基本的に従量課金制で利用できるオンデマンドの Amazon AWS、Azure、SoftLayer を仮想サーバのソースとして使用します。場合によっては、普段の運用は何も変わらないお客様もいらっしゃいます。単純に、従来の仮想インフラに代わる、強力かつ柔軟でコスト効率に優れた方法としてハイパースケーラ サービスを利用する場合です。

NPS はサービスとしても利用できます（NPSaaS）。データベース環境は要件がきわめて厳しいので、コロケーション施設に NPS システムを導入する例がどうしても多くなるのですが、中には、クラウド サーバとクラウド ストレージの両方を投資支出ではなく運用コストとして活用したいというお客様もいらっしゃいます。この場合、ストレージ リソースを、純粋なオンデマンド サービスとして必要に応じて提供できなければなりませんが、こうしたお客様のために、現在数社のプロバイダが NPS をサービスとして提供しています。

2.3 ONTAP Select

お客様所有の仮想インフラに ONTAP Select を導入すると、コモディティ ハードウェアに内蔵されたドライブに ONTAP のインテリジェントな機能とデータ ファブリックへの接続が提供され、ONTAP とゲスト オペレーティング システムで同じハードウェアを共有する高度な統合インフラが実現します。ONTAP を基盤にした Oracle のベストプラクティスには何も影響しません。一番の懸念はパフォーマンスですが、ONTAP Select はこの点でも十分な機能を提供します。ハイエンドの AFF システムの最大パフォーマンスには及びませんが、データベースに 30 万 IOPS が求められることはまずありません。一般的なデータベースなら、ONTAP Select で達成できる約 5,000～10,000IOPS で十分です。またデータベースは、ストレージの IOPS よりもレイテンシから大きな影響を受けることがほとんどなので、ONTAP Select を SSD に導入して解決するとよいでしょう。

2.4 ONTAP Cloud

ONTAP Cloud は ONTAP Select とほぼ同じ製品です。ただしこちらは、お客様の仮想インフラの代わりにハイパースケーラ クラウド環境に導入することで、ハイパースケーラのストレージ ボリュームにインテリジェントな機能とデータ ファブリックへの接続を提供します。ONTAP を基盤にした Oracle のベストプラクティスには何も影響しません。ONTAP Cloud では主にパフォーマンスと、多少ですがコストを考慮する必要があります。ONTAP Cloud のパフォーマンスは、クラウド プロバイダが管理する基盤のボリュームのパフォーマンスから部分的な制約を受けますが、その分、ストレージの管理性が向上します。

ONTAP Cloud のキャッシュ機能によってパフォーマンスが向上する場合もあります。ただし IOPS とレイテンシに関してはパブリック クラウド プロバイダ次第なので、多少の制約が常に発生します。これは、データベースに十分なパフォーマンスが得られないということではなく、単純に、最大パフォーマンスが物理的な AFF システムなどを導入した場合よりも低くなるということです。また、ONTAP Cloud が対応しているクラウド プロバイダ各社では、提供するストレージ ボリュームのパフォーマンス向上が継続的に図られています。

現在、ONTAP Cloud のユースケースは開発とテストが主ですが、本番用システムに ONTAP Cloud を使用しているお客様もいらっしゃいます。特に注目すべきは、ストレージのパフォーマンスに関する制約を、Oracle インメモリ データベースを使用することで解決している例です。確かにこの方法なら、データベース サーバをホストしている仮想マシンの RAM に、より多くのデータが格納されるので、ストレージのパフォーマンス要件を軽減できます。

3 Data ONTAP の設定

Data ONTAP OS の設定を徹底的に解説することは、本ドキュメントの趣旨ではありません。また、大規模なエンタープライズ リソース プランニング データベースを 3 つ構築するのに、2,000 の仮想データベースで構築された環境向けのベストプラクティスは適さないでしょう。データ保護の要件がわずかに違うだけで、ストレージの設計に大きな影響を及ぼしかねません。本セクションでは、基本的な事項をいくつか確認していきます。より包括的な設定例は「付録 1：ファイルシステムのレイアウト」をご覧ください。設計に関して総合的な支援が必要な場合は、ネットアップまたはネットアップのパートナーにお問い合わせください。

3.1 RAID レベル

ネットアップ ストレージを構成しようとすると、RAID レベルに関して不明点が生じることがあります。Oracle の古いガイドや Oracle の設定方法に関する書籍には、RAID ミラーリングの使用には注意が必要なことや、一部の RAID タイプの使用を避けるよう書かれているものが多く見られます。見解は確かな根拠に基づいていますが、これらの資料の内容は、RAID 4 や、ONTAP に使用されている NetApp RAID-DP®、RAID-TEC™のテクノロジには当てはまりません。

RAID 4、RAID 5、RAID 6、RAID-DP、RAID-TEC はいずれもパリティを活用して、ドライブ障害によるデータ損失を防ぎます。これらの RAID オプションはミラーリングよりもはるかに優れたストレージ効率を発揮するのですが、大半の RAID 実装では書き込み処理に影響するデメリットがあることも事実です。他の RAID 実装の場合、書き込み処理を終わらせるには、ディスクのデータを何度も読み取ってパリティ データを生成しなければならないことがその原因で、このプロセスは一般に「RAID ペナルティ」と呼ばれています。

しかし ONTAP を活用すれば、RAID ペナルティは発生しません。NetApp WAFL®（Write Anywhere File Layout）が RAID レイヤに組み込まれているためで、書き込み処理が RAM で 1 つにまとめられ、パリティの生成も含めた完全な RAID ストライプとして用意されます。WAFL 搭載の Data ONTAP なら、書き込みを終わらせるために読み取りを実行する必要がないので、RAID のペナルティとは無縁です。レイテンシが重要な処理（Redo ログなど）でパフォーマンスが妨げられたり、データファイルのランダムな書き込みで、パリティ生成による RAID のペナルティが発生することがありません。

信頼性に関しては、統計上、RAID-DP ですら RAID ミラーリングよりはるかに保護性に優れています。主に問題になるのは、RAID のリビルド時にディスクに大きな負荷がかかることですが、RAID セットのミラーリングでは、RAID セットのパートナーにリビルドする際、ディスク障害によってデータ損失が発生するリスクがあり、その確率は、RAID-DP セットで三重ディスク障害が発生するリスクよりもはるかに高率です。

3.2 容量制限

予測性可能な高パフォーマンスをストレージ アレイに提供するには、メタデータやデータ構成のための空きスペースがいくらか必要です。空きスペースとは実際のデータに使用されていないスペースのことで、アグリゲートに割り当てられていないスペースや、コンスティチュエント ボリューム内の未使用のスペースを含みます。シンプロビジョニングも考慮する必要があります。例えば、あるボリュームに含まれている 1TB の LUN の容量のうち、実際のデータに使用されているのは 50%だけだとします。これはシンプロビジョニング環境では、500GB のスペースが消費されていると正しく表示されますが、フルプロビジョニング環境では、1TB の容量がすべて使用中と表示され、使用されていない 500GB のスペースが隠れてしまいます。このスペースは実際のデータに使用されているわけではないので、本来なら、空きスペースの合計の計算に含めるべきです。

以降のセクションでは、データベースに使用するストレージ システムについてのネットアップの推奨事項を説明します。

SSD アグリゲート（AFF システムを含む）

ネットアップは、最低 10%の空きスペースを確保することを推奨しています。これには、アグリゲートやボリュームの空きスペース、フルプロビジョニングのために割り当てられているものの実際のデータには使用されていない空きスペースなど、未使用のスペースがすべて含まれます。

推奨する空きスペース 10%は控えめな数字です。SSD アグリゲートの場合、90%以上の利用率でもパフォーマンスに影響することなくデータベース ワークロードをサポートできますが、それではやがて、アグリゲートのスペースを使い果たしてしまう恐れがあります。

HDD アグリゲート（Flash Pool アグリゲートを含む）

ネットアップは、最低 15%の空きスペースを確保することを推奨しています。これには、アグリゲートやボリュームの空きスペース、フルプロビジョニングのために割り当てられているものの実際のデータには使用されていない空きスペースなど、未使用のスペースがすべて含まれます。

利用率が 85%未満なら、無視できないほどの影響がパフォーマンスに及ぶことはないはずです。90%に近づくと、一部のワークロードで多少のパフォーマンス低下が目立つようになるかもしれません。95%に達すると、ほとんどのデータベース ワークロードでパフォーマンスが低下します。

3.3 Snapshot ベースのバックアップ

ファイルシステムのレイアウトで検討すべき最も重要なことは、NetApp Snapshot テクノロジの活用を計画することです。主に 2 つの方法があります。

- クラッシュ整合性のあるバックアップ
- Snapshot で保護されたホット バックアップ

データベースのクラッシュ整合性のあるバックアップを作成するには、データベースの構造全体（データファイル、Redo ログ、制御ファイルなど）をある特定の時点でキャプチャすることが必要です。データベースが 1 つの NetApp FlexVol®フレキシブル ボリュームに格納されている場合は、任意の時点で Snapshot を作成すればよいので、このプロセスは簡単です。データベースが複数のボリュームにわたって格納されている場合は、整合グループ（CG）Snapshot を作成する必要があります。CG Snapshot コピーの作成方法は何通りかあり、NetApp Snap Creator®フレームワークのほかに、NetApp SnapManager® for Oracle（SMO）、NetApp SnapDrive® for UNIX、ユーザが保持しているスクリプトを使用できます。

クラッシュ整合性のある Snapshot バックアップは主に、ポイントインバックアップ リカバリで十分な場合に使用します。一部の状況下ではアーカイブ ログで対応できますが、よりきめ細かなポイントインタイム リカバリが必要な場合は、ホット バックアップの使用を推奨します。

Snapshot ベースのホット バックアップの基本的な作成手順は次のとおりです。

1. データベースを `backup` モードにします。
2. データファイルをホストしているすべてのボリュームの Snapshot コピーを作成します。
3. `backup` モードを終了します。
4. `alter system archive log current` コマンドを実行し、ログをアーカイブします。
5. アーカイブ ログをホストしているすべてのボリュームの Snapshot コピーを作成します。

この手順により、バックアップ モードのデータファイルと、バックアップ モード時に生成された重要なアーカイブ ログを含む Snapshot コピーが 1 組作成されます。データベースのリカバリには、この 2 つが必要です。制御ファイルなどのファイル類も保護すると便利ですが、保護が必須となるのはデータファイルとアーカイブログだけです。

戦略はお客様によって様々に異なるかもしれませんが、最終的にはこのセクションで概要を述べた原則が、ほぼすべての戦略の大本になります。

3.4 Snapshot ベースのリカバリ

Oracle データベースのボリューム レイアウトを設計する際には、ボリュームベース NetApp SnapRestore®（VBSR）テクノロジを使用するかどうかを最初に決定する必要があります。

ボリュームベース SnapRestore を使用すると、ボリュームを、以前のある時点の状態にほぼ瞬時にリバートできます。ただし、VBSR ではボリュームのデータがすべてリバートされるので、場合によっては適切でないユースケースがあるかもしれません。例えばデータベース全体が、データファイル、Redo ログ、アーカイブ ログも含めて 1 つのボリュームに格納されている場合、このボリュームを VBSR でリストアすると、最新のアーカイブログと Redo ログが破棄されてデータを失う結果になってしまいます。

通常のリストアに VBSR は必要ありません。データベースの多くは、ファイルベースの Single-File SnapRestore（SFSR）を使用するか、Snapshot コピーで複製したファイルをアクティブなファイルシステムに戻すだけでリストアできます。

VBSR は、データベースが巨大な場合やできるだけ迅速なリカバリが必要な場合に推奨される方法で、使用にあたってはデータファイルを分離する必要があります。NFS 環境では、リカバリするデータベースのデータファイルを専用のボリュームに格納して、他のファイル タイプの影響を受けないようにしてください。SAN 環境の場合は、専用の FlexVol に配置された専用の LUN に格納してください。ボリューム マネージャを使用する場合は（Oracle Automatic Storage Management[ASM]を含む）、ディスクグループもデータファイル専用にしてください。

データファイルをこのように分離すれば、他のファイルシステムに悪影響を与えることなく、データファイルを以前の状態にリバートできます。

clustered Data ONTAP 8.2 の強化機能

clustered Data ONTAP 8.2 は、リストア機能が大幅に強化されています。以前のバージョンの Data ONTAP でファイルレベルのクローンを作成しようとすると、アクティブなファイルシステムを使用するしかなかったのですが、8.2 では、Snapshot コピーからファイルレベルのクローンを作成できるようになりました。その結果、ファイルシステム レイアウトが 1 つのボリュームに複数の種類のデータベース ファイルを含んでいる場合はもとより、1 つのボリュームに複数のデータベースを含んでいる場合でも、一段と簡単に使用できるようになりました。

8.2 より前のバージョンでは、10TB のデータベースのリストアを十分な速さで処理するには、ほとんどの場合、データファイルを専用ボリュームに分離する必要がありました。関係のないファイルがボリュームに格納されていると、リストア プロセスでそれが削除されるため、VBSR を使用できず、代わりにデータを複製することでリカバリを実行していました。このプロセスは、SnapManager for Oracle（SMO）のようにアレイ内で内部複製処理を実行できる製品で処理すれば、今なおきわめて高速ですが、VBSR ほどではありません。

clustered Data ONTAP 8.2 では、ファイルや LUN のクローンを Snapshot コピーから直接作成できます。この処理はほぼ瞬時に完了し、スペース効率にも優れているため、大規模データベースの高速リカバリに VBSR を使用する必要がなくなります。しかも、複数のデータベースで同じボリュームを共有できます。

LUN ベースの環境の場合は、データファイルを専用のディスクグループと LUN に格納すると、ひとまとまりでリストアできます。データファイルのディスクグループに他のファイルが格納されていると、Snapshot コピーからクローンを作成する際にこのファイルが削除されるため、Snapshot コピーを高速リカバリに使用できません。

注： Snapshot コピー ベースのクローニングでファイルをリストアすると、バックグラウンド処理により、メタデータがすべて更新されます。パフォーマンスへの影響はありませんが、このバックグラウンド処理が完了するまで Snapshot コピーは作成できません。処理速度は約 5GBps（18TB/時）です。これは、リストアするファイルの合計サイズに基づきます。

3.5 Snapshot リザーブ

SAN 環境では、Oracle データを格納した各ボリュームで `percent-snapshot-space` をゼロに設定します。LUN 環境では、Snapshot コピー用にスペースをリザーブしてもメリットはありません。フラクショナル リザーブを 100 に設定すると、LUN を格納したボリュームの Snapshot コピー用に、ボリューム内に十分な空きスペースを確保し、すべてのデータの書き替えを 100%吸収する必要が生じます。ただし、この空きスペースに Snapshot リザーブは含まれません。フラクショナル リザーブの値を 100 未満に設定すると、その値に応じた空きスペースが必要になりますが、この場合も、Snapshot コピーのリザーブは含まれません。つまり LUN 環境の場合、Snapshot コピー用にスペースをリザーブしても無駄ということです。

一方 NFS 環境には、2 通りのオプションがあります。

- Snapshot コピーによって消費が予想されるスペースを基に、`percent-snapshot-space` を設定する。
- `percent-snapshot-space` をゼロに設定し、Snapshot コピーによって消費されるアクティブなスペースを一括で管理する。

1 つ目のオプションを用いる場合は、`percent-snapshot-space` をゼロ以外の値（通常は 20%前後）に設定します。このスペースはユーザには表示されませんが、この値を設定することでスペースの利用が制限されるわけではありません。リザーブが 20%のデータベースで 30%の書き替えが発生した場合は、リザーブされている 20%だけでなく、リザーブされていないスペースも Snapshot コピーに使用することができます。

リザーブの値を 20%などに設定することには、Snapshot コピーにいつでも使用可能なスペースを確保できるという大きな利点があります。例えば、1TB のボリュームに 20%のリザーブを設定すれば、データベース管理者（DBA）が格納できるデータは 800GB に制限され、少なくとも 200GB のスペースが Snapshot コピー用に保証されます。

percent-snapshot-space をゼロに設定すると、ボリューム内のスペースがすべてエンドユーザに表示され可視性が向上します。仮に Snapshot コピーを使用する 1TB のボリュームが表示された場合、このスペースはアクティブなデータと Snapshot の書き替えによって共有されることを DBA の方は理解しておいてください。

エンドユーザの場合、オプション 1 と 2 のどちらを選んでも明確な違いはありません。

3.6 read_realloc

Oracle データファイルの場合、書き込みアクティビティのほとんどはランダムな上書き処理です。上書きが発生すると、変更のあったデータがストレージ システムの新しい物理的な場所に配置されます。この操作が、一般にパフォーマンスが最も重視される I/O タイプであるランダム I/O に影響することはありません。ただし、シーケンシャル I/O のスループットには影響します。というのは、マルチブロック読み取り要求への応答を集約し、先読みを実行するには、ストレージ システムで実行される物理ディスク I/O が増加するためです。

All Flash FAS（AFF）システムの場合、I/O の増加は大したことではありませんが、回転式メディアのアレイの場合は（Flash Pool アグリゲートも含む）ドライブ ヘッドの回転が増えるので、結果としてレイテンシが上昇し、スループットが低下します。ボリュームで read_realloc を有効にすると、ファイルシステムのレイアウトをリアルタイムで最適化できます。WAFL ボリュームのデータの配置が不適切な場合、対処が必要となる問題は、そのほとんどが読み取りアクティビティに起因します。ブロックの読み取りが完了したあと、そのデータを 1 つの連続する RAID ストライプとしてドライブに書き戻す処理に大きな負荷はかかりません。read_realloc オプションを使用すると、全体的なパフォーマンスに影響することなく、この処理を実行できます。

例えば、テーブルのフル スキャンを実行すると、データファイルのシーケンシャル リードが発生します。この時 read_realloc が有効になっていると、ディスク上での配置が最適ではないブロックが検出されます。これで問題は 90%解決したも同然です。問題のブロックはその時点でストレージ システムの RAM にあるので、データベース サーバからの読み取り要求が処理されたあと、次のステップとして、read_realloc によってブロックが最適な配置でディスクに書き戻されます。次回テーブルのフルスキャンを実行したとき、データは最適な状態になっています。長期的に見ても、read_realloc を使用することによってデータが定期的にクリーンアップされ、ディスク上のデータファイル レイアウトが最適化されます。

read_realloc には、一般的な on と space_optimized の 2 つのオプションがあります。一般的な設定では、アクティブなファイルシステムと Snapshot コピーに含まれるブロックの両方に関して、ブロック レイアウトが最適化されます。その結果、Snapshot コピーがある場合には消費されるスペースが増えますが、一方で、アクティブなファイルシステムや Snapshot コピー、クローンでシーケンシャル読み取りを実行する際のパフォーマンスが向上するというメリットがあります。space_optimized を使用した場合、Snapshot コピーに含まれているブロックは再配置されません。

この 2 つのパラメータはいつでも変更可能ですが、環境全体で一度に read_realloc を有効にすることは避けてください。必要な処理が増えてパフォーマンスに影響する恐れがあります。1 日につき、データファイルを格納したボリューム 1～2 個で有効にするのが安全です。

以下にネットアップの推奨事項を記載します。

- データファイルを格納しているボリュームに read_realloc を設定し、スペースの消費状態を監視します。

  ボリュームにアーカイブ ログや制御ファイル、他の Oracle ファイル データが含まれているボリュームの場合、このオプションを有効にする必要はありませんが、有効にしても問題にはなりません。
- Snapshot コピーがスペースを過剰に消費している状況が観察される場合は、設定を space_optimized に変更します。
- 前述のように、read_realloc は AFF システムには適用されません。

3.7 Data ONTAP とサードパーティのスナップショット

Oracle Doc ID 604683.1 には、サードパーティのスナップショットのサポート要件と、バックアップおよびリストア処理に使用可能な複数のオプションが説明されています。

サードパーティ ベンダーは、自社のスナップショットが以下の要件に沿っていることを保証しなければなりません。

- スナップショットが、Oracle が推奨するリストアおよびリカバリ処理に統合可能である。
- スナップショットが、作成時点でデータベースとのクラッシュ整合性がある。
- スナップショット内の各ファイルについて書き込み順序が保持されている。

Data ONTAP とネットアップの Oracle 向け管理製品は、以上の要件を満たしています。

3.8 クラスタの運用 ― テイクオーバーとスイッチオーバー

ストレージのテイクオーバーとスイッチオーバーを適切に使用するには、これらのテクノロジがどういった機能なのかを理解しなければなりません。

- 通常の状態では、あるコントローラへの書き込みは、パートナーに同期ミラーリングされます。NetApp MetroCluster™環境の場合、書き込みはリモートのコントローラにもミラーリングされます。書き込みがすべての場所の不揮発性メディアに格納されるまで、ホスト アプリケーションに確認応答は返されません。
- 書き込みデータを格納するメディアは不揮発性メモリ（NVMEM）と呼ばれます。不揮発性ランダム アクセス メモリ（NVRAM）と呼ばれる場合もあります。機能はジャーナルですが、書き込みキャッシュと捉えることができます。通常の処理で NVMEM のデータが読み取られることはなく、ソフトウェアやハードウェアに障害が発生した際のデータ保護にのみ使用されます。ディスクに書き込むデータは NVMEM ではなく、システムの RAM から伝送されます。
- テイクオーバー処理では、高可用性（HA）ペアを構成する 1 つのノードがパートナーの処理を引き継ぎます。スイッチオーバーも基本的に同じですが、こちらは MetroCluster 構成が対象で、リモート ノードがローカル ノードの機能を引き継ぎます。

定期的なメンテナンス作業では、ネットワーク パスの変更によって生じるごく一時的なデータベースの運用停止を除いて、ストレージのテイクオーバーやスイッチオーバーは透過的に実行されなければなりません。ネットワークの設定は複雑なので、どうしてもエラーが起こりがちです。そのためネットアップでは、ストレージ システムを本稼働させる前に、データベースを使用してテイクオーバーとスイッチオーバーの処理を徹底的にテストすることを強く推奨しています。これ以外に、ネットワーク パスがすべて正しく設定されていることを確認する方法はありません。SAN 環境では、`sanlun lun show -p` コマンドの出力を注意深くチェックし、必要なプライマリ パスとセカンダリ パスがどちらも使用可能になっていることを確認します。

テイクオーバーやスイッチオーバーを強制的に実行するときは注意が必要です。テイクオーバーやスイッチオーバーでストレージ設定を強制的に変更するということは、ディスクを所有しているコントローラの状態を無視して、別のノードに無理やりディスクを制御させるということを意味します。テイクオーバーの不適切な強制は、データの損失や破損につながりかねません。強制的なテイクオーバーやスイッチオーバーは、NVMEM のコンテンツの削除を招く恐れがあるからです。テイクオーバーやスイッチオーバーの完了後に NVMEM のデータが失われていた場合、データベース側から見ると、ディスクに格納されていたデータが少し前の状態にリバートされるかもしれないということです。

一般的な HA ペアの場合、強制テイクオーバーが必要になることはまずありません。ほぼすべての障害シナリオで、ノードがシャットダウンされると、パートナー ノードにそれが通知されてフェイルオーバーが自動的に実行されます。ただし、ノード間のインターコネクトで障害が発生し、その後一方のコントローラが失われるローリング エラーなど、一部の例外的なケースでは強制テイクオーバーが必要です。このような状況では、コントローラ障害の前にノード間のミラーリングが失われるため、障害が発生していないコントローラに処理中の書き込みを複製することができません。そこで強制的なテイクオーバーが必要になりますが、その場合データが失われる可能性があります。

MetroCluster のスイッチオーバーにも同じ論理が当てはまります。通常の場合、スイッチオーバーはほぼ透過的です。ところが災害時には、セカンダリ サイトと災害発生サイトの間の接続が失われることがあります。セカンダリ サイト側から見れば、この問題は、サイト間の接続が中断されただけのことで、元のサイトでは今もデータの処理が続いている可能性があります。ノードがプライマリ コントローラの状態を確認できなければ、強制スイッチオーバーを実行するしかありません。

ネットアップでは、以下の対策を施すよう推奨しています。

- テイクオーバーやスイッチオーバーを誤って強制的に実行しないよう、厳重に注意します。強制実行は普通は必要なく、強制的な変更はデータ損失を招く恐れがあります。
- テイクオーバーやスイッチオーバーの強制実行が必要な場合は、データベースがシャットダウンされていること、ファイルシステムがすべてディスマウントされていること、ASM インスタンスがすべてシャットダウンされていること、論理ボリューム マネージャ（LVM）ボリューム グループがすべて活動停止になっていることを確認してください。
- MetroCluster の強制スイッチオーバー イベントでは、障害が発生したノードを、障害が発生していないすべてのストレージ リソースからフェンシングします。詳細については、該当する Data ONTAP バージョンの『MetroCluster 管理およびディザスタ リカバリ ガイド』を参照してください。

MetroCluster と複数のアグリゲート

MetroCluster は同期レプリケーション テクノロジですが、接続が中断すると非同期モードに切り替わります。これはお客様から寄せられる最も多く寄せられるリクエストで、同期レプリケーションを保証したテクノロジでは、サイト間の接続が中断すると、データベースの I/O が完全に停止してサービスを提供できなくなるからです。

MetroCluster の場合、接続が再開するとアグリゲートの再同期がすぐに始まります。他のストレージ テクノロジと異なり、すべてのデータの完全なミラーリングを再度実行する必要がなく、変更による差分のみが転送されます。

複数のアグリゲートにまたがって格納されているデータベースでは、災害が連続して発生した場合にデータ リカバリに追加の手順が必要になるという、ちょっとしたリスクがあります。特に、（a）サイト間の接続が中断、（b）接続が再開、（c）アグリゲートの一部のみが同期された状態になり、その後（d）プライマリ サイトが失われる、といった場合、セカンダリ サイトでは、アグリゲート同士が互いに同期していない状態になります。この場合、データベースの中に互いに同期していない部分があるので、データベースを起動するにはリカバリが必要です。データベースが複数のアグリゲートにまたがって格納されている場合は、Snapshot ベースのバックアップと、数多くあるツールのいずれかを活用して、この異常な事態からすばやくリカバリすることは可能かどうかを検証することを強く推奨します。

4 Storage Virtual Machine と論理インターフェイス

このセクションでは、管理に関する重要な原則をおおまかに説明します。より包括的な説明は、ご使用の Data ONTAP バージョンに対応する『clustered Data ONTAP ネットワーク管理ガイド』を参照してください。データベース アーキテクチャの他の要素同様に、Storage Virtual Machine（SVM、旧称 Vserver）と論理インターフェイス（LIF）の設計については、拡張性の要件とビジネス ニーズによって最適なオプションが大きく変わってきます。

LIF の戦略策定にあたっては、主に以下の事項を考慮してください。

- **パフォーマンス：**ネットワーク帯域幅が十分かどうか。
- **耐障害性：**設計に単一点障害（Single Point of Failure）があるかどうか。
- **管理性：**ネットワークを無停止で拡張可能かどうか。

上記の考慮事項は、ホストからスイッチ、ストレージ システムに至る、エンドツーエンドのソリューションに該当します。

4.1 Storage Virtual Machine

SVM はストレージの基本の機能ユニットです。そこでわかりやすくするために、VMware ESX サーバ上のゲストとの比較で説明します。初めてインストールしたときの ESX には、ゲスト OS のホスト機能やエンドユーザのアプリケーションをサポートする機能など、設定済みの機能は何もありません。仮想マシン（VM）を定義するまでは空のコンテナです。clustered Data ONTAP もほぼ同じです。インストールしただけではこの OS にデータを処理する機能はなく、SVM を定義しなければなりません。SVM の特性がデータ サービスを定義します。

お客様の中には、プライマリ SVM を 1 つ運用して日常的な要件のほとんどに対処し、さらにいくつかの SVM によって次のような特殊なニーズに対応している企業があります。

- 専門チームが管理する業務上重要なデータベースを格納する SVM
- 開発グループ向けの SVM。他から独立した専用ストレージをグループで管理できるよう、管理者によって完全に制御
- 人事情報や財務レポートのデータなど、機密性の高いビジネス データを格納する SVM。管理するチームの限定が必要

マルチテナント環境では、各テナントのデータに専用 SVM を割り当てることができます。SVM の最大数はクラスタ ノードあたり 125 個前後が推奨されますが、通常はこの最大数に達する前に LIF が最大数に達します。またマルチテナント環境は、ネットワーク セグメントを基に分離した方が、複数の専用 SVM に分離するより適切です。

4.2 LIF のタイプ

LIF には複数のタイプがあります。Data ONTAP の公式製品ドキュメントには、このトピックに関して、より包括的な情報が記載されていますが、ここでは LIF を機能の観点から次のグループに分類します。

- **クラスタ管理およびノード管理 LIF :** ストレージ クラスタの管理に使用する LIF。
- **SVM 管理 LIF :** SVM へのアクセスを、Data ONTAP の API（NetApp Manageability SDK）を通じて許可するインターフェイス。Snapshot コピーの作成やボリュームのサイズ変更などの機能に対応。SMO などの製品では SVM 管理 LIF へのアクセスが必要です。
- **データ LIF :** FC、iSCSI、NFS、CIFS データを伝送するインターフェイス。

注 : NFS トラフィックの管理に使用するデータ LIF を有効にするには、ファイアウォール ポリシーを `data` から `mgmt` に変更するか、HTTP、HTTPS、SSH を許可する別のポリシーに変更します。この変更により、NFS データ LIF と、それとは別の管理 LIF の両方にアクセスするよう各ホストを設定する必要がなくなるので、ネットワーク設定が簡易化されます。iSCSI と管理トラフィックに関しては、確かにどちらも IP プロトコルを使用しますが、両者に対応するようにインターフェイスを設定することはできません。iSCSI 環境の場合は、独立した管理 LIF が必要です。

4.3 SAN LIF の設計

SAN 環境の場合、マルチパスを使用するため LIF の設計は比較的簡単です。最新のすべての SAN 実装では、クライアントが複数のネットワーク パス経由でデータにアクセスし、アクセスに最も適したパスを 1 つまたは複数選択することができます。この結果、SAN クライアントは、最適なすべてのパスにわたって I/O を自動で分散できるので、パフォーマンスに関しては LIF の設計は簡単です。

あるパスが使用不可能になると、クライアントによって別のパスが自動で選択されます。設計のしやすさは、一般に SAN LIF の管理性の向上にもつながっています。ただしこれは、SAN 環境の方が常に簡単に管理できるということではありません。SAN ストレージには、このほかにも NFS よりもはるかに複雑な要素が多くあります。ここで言いたいのは、SAN LIF は設計が容易だということだけです。

パフォーマンス

SAN 環境の LIF のパフォーマンスに関しては、帯域幅を考慮することが最も重要です。例えば、4 ノードの Data ONTAP クラスタの各ノードに 16Gb FC ポートを 2 つずつ構成すると、ノード 1 つにつき最大で 32Gb の帯域幅を提供できます。I/O はポート間で自動的に分散され、すべての I/O が最適なパスに転送されます。

耐障害性

SAN LIF はフェイルオーバーができません。SAN LIF に障害が発生すると、クライアントのマルチパス機能によってパスの損失が検出され、別の LIF に I/O がリダイレクトされます。

管理性

NFS 環境では、クラスタ内でのボリュームの再配置に LIF の移行が伴うことが多いため、この移行はきわめて一般的なタスクです。SAN 環境の場合は、ボリュームを再配置しても LIF を移行する必要はありません。ボリュームの移動が完了すると Data ONTAP がパスの変更を SAN に通知し、SAN クライアントが改めてパスを自動で最適化します。SAN 環境で LIF の移行が必要になるのは、主に、物理ハードウェアを大幅に変更したときです。例えば、コントローラの無停止アップグレードが必要な場合は、SAN LIF を新しいハードウェアに移行します。FC ポートの障害が検出された場合も、LIF を未使用のポートに移行します。

設計に関する推奨事項

ネットアップでは主に、次のことを推奨しています。

- パスは必要以上に作成しないでください。パスの数が多すぎると管理が全体的に複雑化し、一部のホストで、パスのフェイルオーバーによる問題が発生する恐れがあります。さらにホストによっては、SAN ブートなどの設定の際にパスの数が制限されるという予期せぬ事態に見舞われます。
- LUN に、ストレージへのパスが 5 つ以上必要になることはほとんどありません。LUN にパスをアドバタイズするノードを 3 つ以上に増やしても、得られる価値には限界があります。なぜなら、LUN を所有するノードと、そのノードの HA パートナーに障害が起きると、その LUN をホストしているアグリゲートにアクセスできなくなるからです。こうした状況では、プライマリ HA ペア以外のノードにパスを作成していても役に立ちません。
- 参照可能な LUN パスの数は FC ゾーンに含めるポートを選択することで管理できますが、一般には、ターゲットに設定可能なポイントすべてを FC ゾーンに含め、LUN の可視性を Data ONTAP レベルで制御する方が簡単です。
- clustered Data ONTAP 8.3 以降では、選択的な LUN マッピング（SLM）機能をデフォルトで使用できます。SLM を使用すると、新しい LUN のアドバタイズを、基盤にあるアグリゲートを所有しているノードと、そのノードの HA パートナーから自動で実行できます。この方法を用いれば、ポートのアクセス性を制限するためにポートセットを作成したりゾーニングを設定する必要がありません。必要最小限のノードで LUN をそれぞれ使用し、最適なパフォーマンスと耐障害性を実現できます。

  LUN を、所有者である HA ペア以外に移行する場合は、`lun mapping add-reporting-nodes` コマンドを使用して新しいノードを追加すると、追加された新しいノードで LUN がアドバタイズされます。これにより、LUN に新しい SAN パスが作成されて LUN の移行が完了します。ただし、ホストがこの新しいパスを使用するには、パスの検出処理が必要です。
- 間接トラフィックを過度に気にする必要はありません。大量の I/O が発生する環境ではレイテンシがマイクロ秒単位で重要になるので、間接トラフィックを避けることが肝要ですが、通常のワークロードに関して言えば、パフォーマンスに認められる影響はごくわずかです。
- ゾーニングでは、セクション13.1 に記載のルールに従ってください。

4.4 NFS LIF の設計

NFS は SAN プロトコルに比べて、複数のデータ パスを定義する能力に限りがあります。NFSv4.1 の拡張である Parallel NFS（pNFS）は、この制限に対処していますが、Oracle データベースは pNFS には未対応なので、本ドキュメントでは取り上げません。

パフォーマンスと耐障害性

SAN LIF のパフォーマンス測定は主に、すべてのプライマリ パスを合わせた総帯域幅を計算すれば済むことですが、NFS LIF のパフォーマンスを割り出すには、正確なネットワーク構成を詳しく確認しなければなりません。例えば、10Gb ポートを 2 つ構成する場合、物理ポートとして構成することもできれば、Link Aggregation Control Protocol（LACP）インターフェイス グループとして構成することもできます。インターフェイス グループとして構成されている場合は、複数のロード バランシング ポリシーを使用し、トラフィックを切り替えるかルーティングするかによって負荷が異なる方法で分散されるように設定できます。最後に、Direct NFS（DNFS）が提供するロード バランシング構成は、現時点ではどの OS の NFS クライアントにも見られません。

SAN プロトコルと異なり、NFS の場合はプロトコル レイヤで耐障害性を実現しなければなりません。例えば、LUN は設定で常にマルチパスが有効化されるので、ストレージ システムへの、FC プロトコルを使用する複数の冗長チャネルが提供されます。一方 NFS ファイルシステムは、1 つの TCP / IP チャネルが使用可能かどうかに依存し、このチャネルは物理レイヤでしか保護できません。ポートのフェイルオーバーや LACP ポート アグリゲーションなどのオプションがあるのは、こうした理由からです。

NFS 環境では、パフォーマンスと耐障害性がどちらもネットワーク プロトコル レイヤで提供されます。そのため、両者は互いに関連するトピックとして一緒に論じなければなりません。

ポート グループへの LIF のバインド

LIF をポート グループにバインドするには、LIF の IP アドレスを物理ポート グループに関連付けます。物理ポートを 1 つに集約するには、主に LACP を用います。LACP のフォールト トレランスは、LACP グループを構成するポートをそれぞれ監視し、障害が発生したポートをグループから取り除くという、実にシンプルな機能です。しかし、パフォーマンスに関する LACP の機能については多くの誤解が見られます。

- LACP は、エンドポイントに合わせるためにスイッチ上で設定する必要がありません。例えば、Data ONTAP を IP ベースのロード バランシングで設定し、スイッチに MAC ベースのロード バランシングを使用することができます。
- LACP 接続を使用するエンドポイントは、それぞれが別々にパケット転送ポイントを選択できますが、受信に使用するポートは選択できません。これは、Data ONTAP から特定のデスティネーションに送信されるトラフィックは特定のポートに結び付けられるが、リターン トラフィックは別のインターフェイスに届く可能性があることを意味します。ただし、これが問題になることはありません。
- LACP では、トラフィックが常に均等に分散されません。このため一般に、多数の NFS クライアントを持つ大規模環境では、LACP アグリゲーションのすべてのポートが均等に使用されます。しかし環境内の NFS は、ファイルシステム 1 つにつき 1 つのポートの帯域幅しか使用できず、アグリゲーション全体を使用することができません。
- Data ONTAP ではラウンドロビンベースの LACP ポリシーを使用できますが、スイッチからホストへの接続にこのポリシーを適用することはできません。例えば、ホスト側と Data ONTAP 側でそれぞれ 4 つのポートをまとめて LACP トランク グループを構成しても、ファイルシステムの読み取りには 1 つのポートしか使用できません。Data ONTAP 側では、データの伝送に 4 つのポートをすべて使えますが、現在のスイッチ テクノロジでは、スイッチからホストへのデータ送信に 4 つのポートをすべて使用することはできません。使用できるのは 1 つだけです。

多数のデータベース ホストで構成された大規模環境の場合、最も一般的に用いられるのは、IP ロード バランシングを使用して、適切な数の 10Gb インターフェイスで LACP アグリゲートを構築する方法です。この方法なら、クライアントの数が十分にあるかぎり、Data ONTAP 側ですべてのポートを均等に使用できます。LACP トランキングの場合、負荷を動的に再分散することができないため、構成に含まれるクライアントの数が減るとロード バランシングが機能しなくなります。

接続が確立すると、一方向のトラフィックは 1 つのポートでのみ処理されます。例えば、あるデータベースが NFS ファイルシステムに対してテーブルのフル スキャンを実行していて、接続に 4 ポートの LACP トランクを使用している場合、データの読み取りには 1 枚の NIC のみが使用されます。この環境にあるデータベース サーバが 3 台だけの場合、3 台すべてが同じポートからデータを読み取り、残りの 3 つのポートがアイドル状態という状況もありえます。

物理ポートへの LIF のバインド

物理ポートに LIF をバインドすると、ネットワーク構成をきめ細かく制御できるようになります。これは、Data ONTAP システム上のある IP アドレスは、一度に 1 つのネットワーク ポートにだけ関連付けられるからです。フェイルオーバー グループとフェイルオーバー ポリシーを設定すれば、耐障害性も実現できます。

フェイルオーバー ポリシーとフェイルオーバー グループ

ネットワーク停止時の LIF の動作を制御するのが、フェイルオーバー ポリシーとフェイルオーバー グループです。設定オプションは、Data ONTAP のバージョンが変わるごとに変更されています。具体的な詳細は、ご使用のバージョンの Data ONTAP に対応する『clustered Data ONTAP ネットワーク管理ガイド』を参照してください。

clustered Data ONTAP 8.2 以前に関しては、以下の一般的な推奨事項に従ってください。

1. ユーザが定義したフェイルオーバー グループを設定します。
2. フェイルオーバー グループに、ストレージ フェイルオーバー（SFO）パートナー コントローラのポートを含め、ストレージのフェイルオーバー時に LIF がアグリゲートに従って移動するようにします。このように設定すれば、間接トラフィックの生成を回避できます。
3. パフォーマンス特性が元の LIF と一致するフェイルオーバー ポートを使用します。例えば、10Gb の 1 つの物理ポート上の LIF には、10Gb ポート 1 つだけで構成されたフェイルオーバーグループを含め、4 ポートの LACP LIF は、別の 4 ポート LACP LIF にフェイルオーバーするようにします。
4. プライマリ コントローラにフェイルオーバー ポリシーを設定します。

clustered Data ONTAP 8.3 では、ブロードキャスト ドメインを基に LIF のフェイルオーバーを管理できます。この機能により、所定のサブネットにアクセスするポートをすべて定義したり、Data ONTAP によって適切なフェイルオーバーLIF を選択することが可能です。一部のお客様はこの方法を使用できますが、予測性がないため、高速データベース ストレージ ネットワーク環境では限界があります。例えば、ファイルシステムへのルーティン アクセス用の 1Gb ポートと、データファイル I/O 用の 10Gb ポートを使用する環境があるとします。この 2 つのタイプのポートが同じブロードキャスト ドメインにあると、LIF のフェイルオーバーによって、データファイル I/O が 10Gb ポートから 1Gb ポートに移ることがあります。

ネットアップでは、Data ONTAP 8.2 によって LIF のフェイルオーバーに使用するポートを定義する方法を推奨しています。以下の推奨事項を考慮してください。

5. ユーザが定義したフェイルオーバー グループを設定します。
6. フェイルオーバー グループに SFO パートナー コントローラのポートを含め、ストレージのフェイルオーバー時に LIF がアグリゲートに従うようにします。これにより、間接トラフィックの生成を回避します。
7. パフォーマンス特性が元の LIF と一致するフェイルオーバー ポートを使用します。例えば、10Gb の 1 つの物理ポート上の LIF には、10Gb ポート 1 つだけで構成されたフェイルオーバーグループを含め、4 ポートの LACP LIF は、別の 4 ポート LACP LIF にフェイルオーバーするようにします。これらのポートが、ブロードキャスト ドメインに定義されたポートのサブセットになります。
8. SFO パートナーのみにフェイルオーバー ポリシーを設定します。こうすることで、フェイルオーバー時に LIF がアグリゲートに従って移動します。

自動リバート

`auto-revert` パラメータは必要に応じた値に設定します。ほとんどの環境では、LIF がホーム ポートにリバートするよう true に設定することが好まれます。ただし場合によっては、予期せぬフェイルオーバーが発生した際、LIF がホーム ポートにリバートする前に調査できるよう、このパラメータを false に設定することもあります。

ボリュームに対する LIF の比率

よくある誤解の一つに、ボリュームと NFS LIF は 1:1 の関係にしなければならないという考えがあります。ボリュームをクラスタ内の任意の場所に移動する際に、新たなインターコネクト トラフィックの発生を抑えたいという場合はこの設定が必要ですが、すべての場合に必要なわけではありません。インタークラスタ トラフィックについては確かに考慮が必要ですが、少々の発生は問題になりません。clustered Data ONTAP 向けに作成され公表されているベンチ―マークの多くには、大量の間接 I/O が含まれています。

例えば、パフォーマンスが重要なデータベースの数があまり多くなく、合計で 40 個のボリュームしか必要としないデータベース プロジェクトの場合、ボリューム対 LIF が 1:1 の戦略、つまり必要な IP アドレスの数が 40 個の構成が妥当かもしれません。この構成なら、任意のボリュームを任意の場所に、関連付けられている LIF と一緒に移動できます。トラフィックは常に直接伝送され、レイテンシのすべての要因をマイクロ秒レベルに押さえることができます。

逆の例はホストされた大規模環境です。この場合は、顧客数対 LIF が 1:1 の関係でさらに容易に管理できます。長期的に見れば、ボリュームを別のノードに移す必要が生じて、間接トラフィックが多少発生するかもしれませんが、インターコネクト スイッチのネットワーク ポートが過負荷にならないかぎり、パフォーマンスへの影響が明らかになることはありません。パフォーマンスに懸念がある場合は、ノードを追加して新しい LIF を設定したり、次回のメンテナンス期間にホストを更新して、構成から間接トラフィックを取り除くことが可能です。

5 圧縮、コンパクション、重複排除

圧縮と重複排除の 2 つの Storage Efficiency オプションを使用すると、所定の量の物理ストレージに格納できる論理データの量を増やすことができます。圧縮とは、簡単に言うと、データのパターンを検出してスペースが低減するようにエンコードする数学的なプロセスです。一方重複排除は、データ内で繰り返されるブロックを検出し、余計なコピーを取り除きます。どちらも結果にはそれほど違いはありませんが、仕組みが大きく異なるため管理の仕方も異なります。

5.1 圧縮

データベースの圧縮方法はいくつかあります。最近まで、データベースは大量のスピンドルを使用しないと、パフォーマンスを十分に提供できない場合がほとんどだったため、圧縮による効果は限られていました。満足なパフォーマンスが得られるストレージ アレイを構築しようとすると、アレイが提供する容量が必要以上になってしまうことが一般的でしたが、この状況はソリッドステート ストレージの普及で変わりました。優れたパフォーマンスを得るために、ドライブを過度にオーバープロビジョニングする必要がなくなったのです。

圧縮を使用しなくても、データベースをソリッドステート ストレージ プラットフォームに移行すれば（HDD との混在構成かオール SSD かは問わない）、I/O のサポートのためだけにドライブを購入する必要がなくなり、コストを大幅に削減できます。例えば、最近の大規模データベース プロジェクトに使用されたストレージ構成をいくつか対象に、NetApp Flash Cache™または Flash Pool™のインテリジェント データ キャッシングを使用した場合と使用しなかった場合でコストを比較したところ、この 2 つのフラッシュ テクノロジによって、コストが約半分に削減されたことがわかりました。IOPS 密度が高いフラッシュ メディアを使用したことで、本来なら必要だった回転式ディスクとシェルフの数を大きく減らせたことがその理由です。

上述のように、ソリッド ステート ドライブ（SSD）では IOPS が向上するため、ほぼすべてのケースでコストを削減できますが、さらに圧縮を使用すれば、ソリッド ステート メディアの実効容量を増やして、削減効果をさらに高められます。圧縮はデータベース自体から実行することも可能ですが、Oracle 環境ではこの方法がとられることは滅多にありません。組み込みの圧縮オプションでは日々変化するデータに対応することができず、高度な圧縮オプションはライセンス コストが高くつくからです。加えて、Oracle データベース自体も安くないことを考えれば、実際のデータベース処理ではなく、データの圧縮と解凍を実行するために CPU に高いライセンス コストを払うことには意味がありません。それよりも、圧縮処理をストレージ システムにオフロードする方が賢明です。

圧縮の効果はオール SSD 環境に限定されず、Flash Pool アグリゲートなどのハイブリッド オプションにもメリットをもたらします。データが SSD レイヤで圧縮されると、SSD のストレージ容量が増えるという実際的な効果が得られます。

Data ONTAP 8.3.1

Data ONTAP 8.3.1 は、様々なサイズのブロックに作用するインライン圧縮手法である「アダプティブ圧縮」を採り入れています。パフォーマンスへの影響は最小限で、圧縮を有効にすることで全体的なパフォーマンスが向上する場合もあります。また、Data ONTAP 8.3 以前で提供されていた圧縮機能を、新たに「二次圧縮」という名前で使用できます。

圧縮の使い方に、これさえ守ればよいというベストプラクティスはありません。最適なオプションは業務の処理方法によって異なります。大半のデータベースはアダプティブ圧縮が有効なボリュームに配置でき、データを分離したり、ファイルを特別な方法で扱う必要がありません。アダプティブ圧縮は複数のデータ タイプと I/O パターンに対応します。

アダプティブ圧縮

アダプティブ圧縮は、Oracle ワークロードを使用した徹底的なテストを通じて、パフォーマンスへの影響がごくわずかであることが確認されています。レイテンシがマイクロ秒単位で測定されるオールフラッシュ環境でも同様です。最初のテストでは、一部のお客様から、圧縮によってパフォーマンスが向上したという報告もありました。これは、圧縮によって、データベースに使用できる Flash Pool の SSD の容量が増えたことによります。

Data ONTAP は物理ブロックを 4KB 単位で管理するので、8KB のブロックを使用する一般的な Oracle データベースの場合、最大で 2:1 の圧縮を実現できます。実際のユーザ データを使った初期のテストではこれに近い圧縮率が観察されましたが、実際の結果は、格納されているデータのタイプによって異なります。

二次圧縮

二次圧縮では、使用するブロックのサイズがアダプティブ圧縮よりも大きく、32KB に固定されます。このためデータを効率よく圧縮できますが、二次圧縮は主に、保存中のデータやシーケンシャルに書き込まれるデータなど、最大限の圧縮が必要なデータ向けに設計された機能です。

ネットアップでは、アーカイブ ログや Recovery Manager（RMAN）のバックアップなどに二次圧縮を使用することを推奨しています。こうしたタイプのファイルはシーケンシャルで書き込まれており、更新されることはありません。だからと言ってアダプティブ圧縮は勧められないということではありませんが、格納中のデータが大量にある場合は、アダプティブ圧縮よりも二次圧縮の方が削減効果が高まります。

データ量が膨大で、データファイルそのものが読み取り専用またはほとんど更新されない場合は、二次圧縮の使用を検討してください。32KB のブロック サイズを使用するデータファイルには、同じく 32KB のブロック サイズを使用する二次圧縮を使用した方が、高い圧縮効果が得られます。ただし、圧縮するボリュームに、ブロック サイズが 32KB 以外のデータが格納されていないことを注意して確認する必要があります。また、二次圧縮は、頻繁に更新されないデータのみに使用してください。

注意

RMAN バックアップには二次圧縮と重複排除を併用しないでください。理由は、バックアップ データへのわずかな変更も 32KB の圧縮領域に影響するからです。領域がずれた場合、圧縮されたデータはファイル全体にわたって変わってしまう結果となり、続けて重複排除を実行すると、重複排除エンジンは圧縮後のバックアップを別のものとみなします。RMAN バックアップの重複排除が必要な場合は二次圧縮を使用せず、アダプティブ圧縮を使用することを推奨します。アダプティブ圧縮の方が作用するブロック サイズが小さいので、重複排除による効率化の妨げになりません。同じ理由から、ホスト側で実行する圧縮も重複排除による効率化の妨げになります。

圧縮とシンプロビジョニング

圧縮はシンプロビジョニングの一形態です。例えば、100GB のボリュームを使用している 100GB の LUN を 50GB に圧縮するとします。LUN を圧縮してもボリュームは 100GB のままなので、実際の削減効果は目に見えません。節約したスペースをシステムのどこからでも使用できるようにするには、まずボリュームのサイズを削減することが必要です。100GB の LUN にあとから変更が加えられた結果、データの圧縮効果が下がった場合、やがて LUN のサイズが拡大してボリュームがいっぱいになる恐れがあります。シンプロビジョニングを使用すると、使用可能な容量を大きく増やし、コストも大幅に削減できます。ただしスペース利用率を常に監視して、突然の容量不足に陥らないようにすることが必要です。

アライメント

データベース環境でアダプティブ圧縮を実行するには、圧縮ブロックのアライメントに関して考慮が必要です。特定のブロックでランダム オーバーライトが発生するデータの場合、アライメントは唯一の考慮事項で、アプローチの方法としてはファイルシステム全体のアライメントによく似ています（「ゾーニング」のセクションを参照）。

例えば、Oracle のデータファイルに対する 8KB の書き込みは、ファイルシステムの 8KB の境界に合っている場合のみ圧縮されます。つまり書き込みが、ファイルの 1 つ目の 8KB に収まり、次に 2 つ目の 8KB、続いて 3 つ目の 8KB にと収まっていく必要があるということです。RMAN バックアップやアーカイブ ログなどは、複数のブロックにわたってデータが書き込まれるシーケンシャル書き込み処理で、すべてが圧縮されます。アライメントを考慮する必要はありません。注意が必要となるのは、データファイルのランダム オーバーライトのみです。

NFS

NFS では、データファイルがアライメントされます。データファイルの各ブロックがファイルの始点を基にアライメントされます。

SAN

SAN 環境では、圧縮を最適化するためにはデータを 8KB の境界に合わせてアライメントする必要があります。SAN の場合、LUN とファイルシステムの 2 つの面でアライメントが必要です。LUN は、ディスク デバイス全体として構築するか（パーティショニングなし）、8K の境界に合わせたパーティショニングで構築する必要があります。後述の OS 別セクションでは、圧縮とアライメントについて構成別に詳しく解説していますので、参照してください。

Data ONTAP 8.3.1 以降に関する推奨事項

Data ONTAP 8.3.1 以降については、以下のことを推奨します。

- 最も簡単な圧縮の活用方法は、すべてのデータベース ボリュームでアダプティブ圧縮を有効にすることです。前述のように、アダプティブ圧縮はどの I/O パターンにも適していますが、以下の例外にだけ注意してください。
  - ボリュームがシンプロビジョニングされていないことが確実な場合は、圧縮してもメリットはないので圧縮を有効にしないでください。
  - アーカイブ ログが大量に保持されている場合は、二次圧縮が適用されているボリュームにログを移動することで、大幅な効率化を達成できます。
  - データベースの中には、Redo ログが頻繁に作成されるものがあります。Redo ログは比較的小さいうえ、絶えず上書きされるので、圧縮によるスペース削減効果はごくわずかです。Redo ログのデータは、圧縮を適用しないボリュームに移動してください。
  - データファイルに圧縮不可能なデータが大量に含まれている場合（圧縮が常に有効になっている場合や暗号化が使用されている場合）は、圧縮を使用しないボリュームにデータを格納します。

- 上記の例外に過度に気を配る必要はありません。Data ONTAP では圧縮を有効にするタイミングを柔軟に選択できます。ここに挙げた例外は選択の幅を広げることがその目的です。ほとんどのテストにおいて検知されるほどの圧縮効果はないとしても、効果はゼロではありません。お客様によっては、プラットフォームのパフォーマンスをできるだけ高め、レイテンシをマイクロ秒単位で最小化することが望まれることもあり、その場合、圧縮は最適なオプションにならないかもしれません。

Data ONTAP 8.3 以前に関する推奨事項

Data ONTAP 8.3 以前で Data ONTAP の圧縮機能を使用する場合は注意が必要です。なぜなら 8.3 以前は、圧縮できるブロック サイズが 32KB なのに対し、一般的なデータベースは 8KB のブロックを使用するからです。そのため 1 つの 8KB ブロックを更新するのに、Data ONTAP では Oracle のブロックを 4 つ読み取ってから 8KB のユニットを 1 つ更新し、それをディスクに書き込まなくてはなりません。

一部のお客様は、アーカイブ ログなどのシーケンシャルで書き込まれたデータや、データファイル内のアーカイブ データのように更新頻度の低いデータに Data ONTAP 8.3 以前の圧縮機能をうまく活用しています。

- Oracle データを格納しているボリュームでは、圧縮を有効にしないでください。ただし、（a）変更率がきわめて低い場合や、（b）書き込みがシーケンシャルで、更新が前提条件ではないデータ（RMAN バックアップ ファイルやアーカイブ ログなど）は除きます。

Data ONTAP のバージョン固有の注意

Data ONTAP のバージョンごとに以下の注意事項があります。

- Data ONTAP 8.2 では、ボリュームで圧縮を使用すると、データが Flash Pool にキャッシュされなくなります。
- Data ONTAP 8.3 では、圧縮後のブロックが Flash Pool の読み取りキャッシュの対象になりますが、書き込みキャッシュの対象にはなりません。
- Data ONTAP 8.3.1 では、圧縮後のブロックが Flash Pool の読み取りキャッシュと書き込みキャッシュの対象になります。
- 圧縮後のボリュームに Flash Cache を使用することができますが、データは圧縮されていない状態でフラッシュ レイヤに格納されます。

注：圧縮とフラクショナル リザベーションの相互作用については、「フラクショナル リザベーション」セクションの説明を参照してください。

5.2 インライン データ コンパクション

インライン データ コンパクションは ONTAP 9 で登場した機能で、圧縮効率を向上させます。前述のように、アダプティブ圧縮では 4K の WAFL ブロックに 8K の IO が格納されるため、実現する削減率は最大でも 2:1 です。二次圧縮のような圧縮手法は、8K よりも大きいサイズのブロックを使用するので効率性に優れていますが、小さなブロックを上書きするデータには適しません。32KB のデータ ユニットを解凍して 8K 分を更新し、再度圧縮してからディスクに改めて書き込む処理では、オーバーヘッドが発生します。

インライン データ コンパクションでは、論理的な WAFL ブロックを物理的な WAFL ブロックに格納することが可能です。例えば、テキストや部分的にフルのブロックのように極限まで圧縮可能なデータを格納したデータベースは、8KB から 1KB に圧縮することができますが、コンパクションを使用しない場合、この 1KB のデータによって、4KB ブロック全体がその後も占有されます。インライン データ コンパクションを使用すれば、圧縮した 1KB のデータを、他の圧縮データと一緒に 1KB ちょうどの物理スペースに格納できます。これは圧縮テクノロジというより、単純に言えば、ディスク スペースをより効率よく割り当てる方法なため、検出できるほどの影響がパフォーマンスに及ぶことはありません。

得られる削減効果の程度は様々です。圧縮済みのデータや暗号化データは、通常それ以上は圧縮することができないため、コンパクションによるメリットはありません。初期化されたばかりの Oracle データファイルで、格納されているデータがせいぜいメタデータのブロックとゼロ ブロックの場合は、最大 80:1 まで圧縮できます。これによりもたらされる可能性ははかり知れません。どれくらいの削減効果を実現できるかを検証する一番の方法は、ネットアップのスペース削減試算ツール（SSET）を使用することです。このツールは NetApp Field Portal から、またはネットアップの担当者経由で入手可能です。

5.3 重複排除

重複排除を主な削減機能として Oracle データベースのファイルに使用することは避けてください。ほとんど効果はありません。Oracle のブロックにはデータベース全体で一意のヘッダーと、ほぼ一意のトレーラが含まれています。重複排除によってスペースは 1 パーセント削減できますが、深刻なオーバーヘッドが発生するという犠牲が伴います。

ブロック サイズが 16K と大きいデータベースの場合、最大で 15%のスペース削減効果が見られたケースがありますが、ごく数例にすぎません。各ブロックの最初の 4KB には、データベース全体で一意のヘッダーが含まれ、最後の 4KB ブロックにはほぼ一意のトレーラが含まれています。間のブロックが重複排除の対象候補ですが、実際には、初期化（ゼロ化）したデータを重複排除するのとほとんど変わりません。

多くの競合企業は、Oracle データベースが何回も複製されていると仮定すれば、重複排除が可能だと主張しています。この点では、ネットアップの重複排除も有効ですが、Data ONTAP には NetApp FlexClone®というより優れたオプションがあります。最終的な結果はどちらも同じで、基盤の物理ブロックのほとんどを共有する Oracle データベースのコピーが複数作成されます。FlexClone を使用すれば、時間をかけてデータファイルを複製し、その後重複を排除するよりも、はるかに効率的です。FlexClone では、重複を排除するのではなく、二重になる部分が最初から作成されません。

同じデータファイルから作成されたコピーが複数存在するまれなケースについては、重複排除を使用できます。

Oracle データを格納しているボリュームでは、重複排除を有効にしないことを推奨します。ただしそのボリュームに、同じデータのコピーが複数含まれていることがわかっている場合、例えば、同じボリュームに対して繰り返し作成される RMAN バックアップなどが含まれている場合は別です。

6 シンプロビジョニング

シンプロビジョニングとは、実際に使用できるスペース以上のスペースをストレージ システムに構成する手法です。この手法には様々な形があり、Data ONTAP が Oracle データベース環境に提供する多数の機能に欠かせないテクノロジです。

Snapshot を使用する場合、シンプロビジョニングが関わることがほとんどです。例えば、ネットアップ ストレージ環境で稼働する一般的な 10TB のデータベースには、30 日分の Snapshot コピーが含まれています。この場合、アクティブなファイルシステムに表示されるデータは約 10TB、Snapshot コピー専用のスペースが 300TB となります。合わせて 310TB のストレージが、通常は約 12～15TB のスペースに配置されています。アクティブなデータベースは 10TB を消費しますが、残りの 300TB のデータは、元のデータに加えられた変更のみが格納されるため 2～5TB のスペースしか必要としません。

クローニングもシンプロビジョニングの一例です。ネットアップの大手取引先の一つは、80TB のデータベースから 40 個のクローンを作成して開発に使用しています。もし、40 人の開発者がすべてのデータファイルのすべてのブロックを上書きしたら、3.2PB を超えるストレージが必要になりますが、実際には書き替え率が低く、変更されたブロックのみがディスクに格納されるので、必要なスペースはすべて合わせても 40TB 弱です。

6.1 スペース管理

Oracle 環境をシンプロビジョニングする場合は、データの変更率が予期せず上昇することがあるので注意が必要です。例えば、テーブルの再インデックス付けの際に Snapshot コピーによるスペース消費が急増したり、RMAN バックアップの配置場所が不適切な場合にきわめて短時間で大量のデータが書き込まれることがあります。また、データファイルの拡張時にファイルシステムの空きスペースが不足して、Oracle データベースのリカバリが困難になることもあります。

幸い、こうしたリスクは `volume-autogrow` ポリシーや `snapshot-autodelete` ポリシーを注意して設定することで対処可能です。名前からわかるように、この 2 つのオプションを使ってポリシーを作成すると、Snapshot コピーが消費するスペースを自動で削除したり、ボリュームを拡張して追加データに対応することができます。オプションは多数提供されているので、ニーズに合わせて選択してください。

これらの機能の包括的な説明は、『Data ONTAP 論理ストレージ管理ガイド』を参照してください。

6.2 LUN シンプロビジョニング

Oracle 環境では、データファイルの作成時にファイルが初期化されてフル サイズになるため、アクティブな LUN のシンプロビジョニングはあまり有効ではありません。ファイルシステム環境のアクティブな LUN をシンプロビジョニングして効率化しても、削除したデータや消去したデータによって、ファイルシステムのうち割り当てられていない空白スペースが消費されていき、次第にその効果が失われます。

唯一の例外は、論理ボリューム マネージャ（LVM）を使用する場合です。Veritas VxVM や Oracle ASM などの LVM を使用すると、基盤の LUN を複数のエクステントに分割して必要なときだけ使用できます。例えば、最初は 2TB のサイズだったデータベースがやがて 10TB に拡大した場合、このデータベースを、シンプロビジョニングされ、LVM ディスクグループにまとめられた、10TB の LUN に配置することができます。こうすれば、データベースの作成時に消費されるディスク スペースは 2TB だけで、データベースの拡大に対応するためにエクステントが割り当てられたときに初めて追加のスペースが必要になります。このプロセスなら、スペースを監視しているかぎり安全です。

6.3 フラクショナル リザベーション

フラクショナル リザーブは、ボリューム内での、スペース効率に関連した LUN の動作を決定します。`fractional-reserve` オプションを 100%に設定した場合、ボリューム内のあらゆるデータ パターンのすべてのデータを、ボリュームのスペースを使い切ることなく、100%書き換えることが可能です。

Snapshot を例に、1TB のボリュームに配置された 1 つの 250GB の LUN にデータベースが格納されているとしましょう。Snapshot コピーを作成すると、即座にボリュームのスペースから 250GB が新たにリザーブされ、これによって、どのような場合もボリュームがスペース不足にならないことが保証されます。データベース ボリュームのすべてのバイトの上書きが必要になることはほとんどありえないため、フラクショナル リザーブの使用は一般には無駄です。絶対に起こらないイベントのためにスペースをリザーブする理由はありません。ただし、ストレージ システムのスペース消費を監視できない状況で、スペース不足が起きないことを保証しなければならない場合は、Snapshot コピー用に 100%のフラクショナル リザーブが必要となることがあります。

6.4 圧縮と重複排除

圧縮と重複排除はどちらもシンプロビジョニングの一形態です。例えば、50TB のデータベース容量を 30TB に圧縮すれば、結果として 20TB が削減されます。圧縮によるメリットを活かすには、削減した 20TB の一部を他のデータに使用するか、あるいは 50TB 未満のストレージを購入する必要があります。つまり、ストレージ システムの正式な容量よりも多くのデータを格納できます。データベースから見れば、50TB のデータがあるのにディスクは 30TB しか使用されていないことになります。

データベースをどれだけ圧縮できるかは常に変わる可能性があるので、実際のスペースの消費が増える結果になることもあります。消費量が増えるということは、他の形態のシンプロビジョニングと同様に、監視機能および `volume-autogrow` や `snapshot-autodelete` によって圧縮を管理する必要があるということです。

圧縮と重複排除については、「圧縮」と「重複排除」のセクションでさらに詳しく取り上げます。

6.5 圧縮とフラクショナル リザベーション

圧縮はシンプロビジョニングの一形態です。フラクショナル リザベーションは圧縮の使用に影響します。スペースは Snapshot コピーの作成前にリザーブされる点に注意してください。通常フラクショナル リザーブが重要になるのは Snapshot コピーが存在する場合のみで、Snapshot コピーがなければフラクショナル リザーブは重要ではありません。ただし圧縮については例外です。圧縮が有効なボリュームで LUN を作成すると、Snapshot コピーに対応するためのスペースが確保されます。設定の際はこの動作に困惑するかもしれませんが、これは想定される動作です。

例えば 10GB のボリュームに、2.5GB に圧縮された 5GB の LUN が格納されているとします。Snapshot コピーはありません。以下の 2 つのシナリオを考えてください。

- フラクショナル リザーブ = 100（7.5GB を消費）
- フラクショナル リザーブ = 0（2.5GB を消費）

1 つ目のシナリオでは、現在のデータ用に 2.5GB のスペースと、圧縮前のデータが 100%書き替えられた場合に Snapshot コピーが使用する 5GB のスペースが確保されます。2 つ目のシナリオでは余分にリザーブされたスペースはありません。

少し複雑かもしれませんが、実際にこうした状況に遭遇することはまずありません。圧縮には当然シンプロビジョニングが伴い、そして LUN 環境でのシンプロビジョニングにはフラクショナル リザベーションが必要です。圧縮されたデータが圧縮できないデータで上書きされることはいつでも起こりえます。したがって、圧縮による削減効果を実現するためには、ボリュームをシンプロビジョニングする必要があります。

リザーブの設定に関する推奨事項は以下のとおりです。

- 基本的な容量監視プロセスが実施されていて、`volume-autogrow` と `snapshot-autodelete` を使用できる場合は `fractional-reserve` を 0 に設定します。
- 監視プロセスがない場合や、何らかの事情でスペースを使用できない場合は、`fractional-reserve` を 100 に設定します。

7 パフォーマンスの比較とベンチマーク

データベース ストレージのパフォーマンスを正確にテストすることは、きわめて複雑なプロセスです。IOPS とスループットの理解が必要なだけでなく、フォアグラウンドとバックグラウンドの I/O 処理の違い、データベースにレイテンシが及ぼす影響、同じくストレージ パフォーマンスに影響する OS とネットワークの様々な設定も理解しなければなりません。さらに、ストレージとは無関係のデータベース タスクを考慮する必要もあります。ストレージ パフォーマンスがパフォーマンスの制限要因でなくなった場合、ストレージ パフォーマンスを最適化してもある時点でそのメリットはなくなります。

現在データベース ユーザはほとんどがオールフラッシュ アレイを選択しており、これも新たな考慮事項を生み出しています。例えば、2 ノードの AFF8080 システムでパフォーマンスをテストする場合を考えてみましょう。

- 読み取り / 書き込みの比率が 75 対 25 の場合、2 台の AFF8080 ノードを使用して 30 万回以上のランダムデータベース IOPS を、レイテンシ 1 ミリ秒未満で達成できます。これは、ほとんどデータベースに現在求められるパフォーマンス要件を大きく超えているため、想定されるパフォーマンスの向上を予測することは困難です。ストレージがボトルネックになる可能性はほとんどなくなるでしょう。
- ネットワーク帯域幅が、パフォーマンスの制限要因としてますます一般化しつつあります。例えば、回転式ディスクを使用したソリューションは I/O レイテンシがきわめて高いことから、データベース パフォーマンスのボトルネックになることがよくあります。レイテンシによる制限がオールフラッシュ アレイによって取り除かれると、多くの場合ネットワークが新たな障壁となります。特に、実際のネットワーク接続を可視化することが難しい仮想環境とブレード システムでは顕著です。この場合、帯域幅の制限のためにストレージ システム自体をフルに活用できないと、パフォーマンスのテストが複雑になる可能性があります。

- オールフラッシュ アレイではレイテンシが大幅に向上するので、オールフラッシュ アレイと回転式ディスク混在のアレイのパフォーマンスを比較することは一般には不可能です。テストの結果は大抵の場合意味がありません。
- オールフラッシュ アレイでピーク時の IOPS パフォーマンスを比較することも、データベースがストレージ I/O の制限を受けないことを考えれば、ほとんどの場合意味がありません。例えば、あるアレイが 50 万 IOPS を維持できるのに対し、別のアレイが維持できるのが 30 万 IOPS だとします。データベースの稼働時間の 99%が CPU 処理に費やされているとすれば、この違いは実環境では重要ではありません。ワークロードがストレージ アレイの能力をフルに活用することはありません。対照的に統合プラットフォームでは、ストレージ アレイの能力を最大限活用することが期待されるため、ピーク時の IOPS の能力が重要になるかもしれません。
- いずれのストレージ テストでも、レイテンシと IOPS を必ず考慮してください。市場に出回っているストレージ アレイの多くは桁外れの IOPS を謳っていますが、このレベルの IOPS はわずかなレイテンシによって意味をなさなくなります。オールフラッシュ アレイの場合、一般的な目安は 1 ミリ秒です。テスト方法としては、IOPS の最大値を測定するよりも、平均レイテンシが 1 ミリ秒を上回らない範囲でストレージ アレイが維持できる IOPS を確認することをお勧めします。

7.1 calibrate_io

`calibrate_io` コマンドを、ストレージ システムのテストや比較、ベンチマークに使用することは避けてください。Oracle のドキュメントにあるように、このプロシージャはストレージの I/O 性能をキャリブレート（調整）します。

キャリブレーションはベンチマークと同じではありません。このコマンドの目的は、I/O を実行してデータベース処理をキャリブレートし、ホストに対して実行される I/O のレベルを最適化して効率性を高めることです。`calibrate_io` 処理で実行される I/O のタイプは、データベース ユーザの実際の I/O を表していないため、結果の予測がつかないばかりか、再現さえできないことがよくあります。

7.2 SLOB2

Silly Little Oracle Benchmark（SLOB2）は、データベースのパフォーマンス評価に好んで使われるツールとなりました。開発者は Kevin Closson で、こちらから入手できます。インストールと設定に数分かかりますが、実際の Oracle データベースを使用して、ユーザ定義の表領域に I/O パターンを生成できます。オールフラッシュ アレイを I/O で満たすことが可能な、新しいテストの 1 つです。生成する I/O のレベルを大きく下げれば、IOPS は小さくてもレイテンシの影響を受けやすいストレージ ワークロードをシミュレーションするのにも役立ちます。

7.3 Swingbench

Swingbench はデータベースのパフォーマンス テストに有効ですが、ストレージに負荷を与えるようなテストには向いていません。ネットアップが知るかぎり、Swingbench を使ったテストで、AFF アレイに大きな負荷をかけられるほどの I/O が生成された例はありません。一部のケースでは、Order Entry Test（OET）を使用して、レイテンシの観点からストレージを検証することができます。データベースに特定のクエリに起因する既知のレイテンシがある場合は、この方法が有効かもしれません。オールフラッシュ アレイに見込まれるレイテンシを実現するよう、ホストとネットワークを適切に設定する必要があります。

7.4 HammerDB

HammerDB は、特に TPC-C と TPC-H のベンチマークをシミュレーションするツールです。テストを適切に実施するために十分な大きさのデータセットを構築するのにはかなりの時間がかかりますが、OLTP やデータ ウェアハウス アプリケーションの検証には有効なツールです。

7.5 Orion

Oracle Orion ツールは Oracle 9 で広く使用されていましたが、各種ホスト オペレーティング システムの変更に対応するような更新は実施されていませんでした。OS やストレージ設定との互換性がないことから Oracle 10 や Oracle 11 に使用されることはほとんどありませんでした。

Oracle はこのツールを書き直し、現在は Oracle 12c にデフォルトでインストールされています。ツールは改良され、実際の Oracle データベースと同じ呼び出しの多くを使用しますが、コード パスや I/O の動作は Oracle が使用するものとまったく同じではありません。例えば、大半の Oracle I/O は同時に実行されますが、I/O 処理はフォアグラウンドで実行されるため、これは I/O が完了するまでデータベースが停止することを意味します。ストレージ システムをランダム I/O でフラッディングさせるだけでは、実際の Oracle I/O は再現できず、ストレージ アレイを比較したり、構成変更の影響を評価できる直接的な方法は提供できません。

とは言え、例えば特定のホスト / ネットワーク / ストレージ構成の最大パフォーマンスの一般的な測定や、ストレージ システムの健常性評価には Orion を使用できます。注意深くテストすれば、Orion を使用して、ストレージ アレイの比較や構成変更による影響の検証に役立つテストを実施できるかもしれません。ただし、IOPS やスループット、レイテンシを考慮したパラメータで、実際のワークロードを忠実に再現することが必要です。

8 移行

Oracle データベースの移行オプションについて、詳細な解説は TR-4534『Migration of Oracle Databases to NetApp Storage Systems』を参照してください。

9 一般的な Oracle 設定

以下のパラメータは一般に、どの構成にも適用可能です。

9.1 Filesystemio_options

Oracle の初期化パラメータ `filesystemio_options` は、非同期 I/O とダイレクト I/O の使用を制御します。一般的な言説と異なり、非同期 I/O とダイレクト I/O はお互いに相反するものではありません。このパラメータを正しく設定しているお客様はあまり多くなく、誤った設定が、パフォーマンスに関する多くの問題の直接的原因になっています。

非同期 I/O とは、Oracle I/O の並行処理が可能ということです。様々な OS で非同期 I/O を使用できるようになる以前、ユーザは大量の dbwriter プロセスを設定し、サーバの処理設定を変更していました。非同期 I/O では、OS がデータベース ソフトウェアに代わって、効率性に優れた並行処理で I/O を実行します。この処理によってデータがリスクにさらされることはありません。また、Oracle の Redo ログ作成などの重要な処理は、引き続き同期 I/O で実行されます。

ダイレクト I/O は OS のバッファ キャッシュを迂回します。UNIX システムの I/O は通常、OS のバッファ キャッシュを経由します。これは内部キャッシュを持たないアプリケーションでは便利ですが、Oracle の場合、システム グローバル領域（SGA）内に独自のバッファ キャッシュを備えているので、ほぼすべてのケースで、ダイレクト I/O を有効にしてサーバの RAM を SGA に割り当てた方が、OS のバッファ キャッシュを使用するよりも適しています。Oracle の SGA はメモリをより効率よく使用します。また I/O が OS のバッファを経由した場合、追加の処理が必要になってレイテンシが増大します。このレイテンシの増大は、低レイテンシが重要な要件である大量の書き込み I/O で問題となります。

以下に、`filesystemio_options` のオプションを示します。

- **async :** Oracle から OS に I/O 要求が送信されて処理されます。Oracle は I/O の完了を待たずに別の作業を実行できるので、I/O の並行処理が促進されます。
- **directio :** Oracle は、ホスト OS のキャッシュに I/O をルーティングする代わりに、物理ファイルに対して直接 I/O を実行します。
- **none :** Oracle は同期 I/O とバッファ I/O を使用します。この設定では、サーバ処理を共有するか占有するかの選択と、データベースへの書き込みの数がより重要になります。
- **setall :** Oracle は非同期 I/O とダイレクト I/O を両方使用します。

ほとんどの場合、setall が最適な設定ですが、以下の問題を考慮してください。

- 一部のお客様から、特に、以前の Red Hat Enterprise Linux 4（RHEL4）リリースで、過去に非同期 I/O に関する問題が報告されていました。ただし現在この問題が報告されることはなく、非同期 I/O は最新のすべての OS で安定して機能しています。
- データベースがバッファ I/O を使用していた場合は、ダイレクト I/O に切り替えることで、SGA サイズの変更が必要になる場合もあります。バッファ I/O を無効にすると、ホスト OS のキャッシュがデーベースにもたらすパフォーマンス上のメリットがなくなります。ただしこの問題は、RAM を SGA に戻すと解決され、最終的には I/O パフォーマンスの向上という結果が得られます。
- RAM は、OS のバッファ キャッシュよりも Oracle SGA に使用した方がよい場合がほとんどですが、選択が難しい場合もあります。例えば、データベース サーバの SGA のサイズがごく小さく、断続的にアクティブになる Oracle インスタンスが多数ある場合は、バッファ I/O の使用が好ましいかもしれません。そうすることで、使用されずに空いている OS の RAM を、実行中のすべてのデータベース インスタンスに柔軟に使用できます。これはきわめて稀なケースですが、一部のお客さまでは実際にこのような状況が報告されています。

注： filesystemio_options パラメータは、DNFS と ASM 環境では効果はありません。DNFS または ASM を使用すると、自動的に非同期 I/O とダイレクト I/O の両方が使用されます。

以下にネットアップの推奨事項を記載します。

- filesystemio_options を setall に設定します。ただし一部の状況では、ホストのバッファ キャッシュがなくなった結果 Oracle SGA のサイズ拡張が必要になる場合があります。

9.2 db_file_multiblock_read_count

このパラメータは、シーケンシャル I/O で Oracle が単一の処理として読み取る Oracle データベース ブロックの最大数を制御します。ただし、読み取り処理の際に Oracle が読み取るブロックの数やランダム I/O には影響しません。影響するのはシーケンシャル I/O のみです。

Oracle は、このパラメータを未設定のままにしておくことを推奨しています。設定しないでおけば、データベース ソフトウェアが自動で最適な値を設定します。具体的には、このパラメータは I/O サイズが 1MB になるような値に設定されます。例えば、8KB ブロックを 1MB で読み取るには 128 個のブロックを読み取る必要があるため、デフォルト値は 128 に設定されます。

データベースのパフォーマンスに関して、お客様のサイトに見られる問題のほとんどには、このパラメータが正しく設定されていないことが関係しています。Oracle のバージョン 8 と 9 では、ユーザがこの値を変更する余地が残されていたためデータベースを使用したまま Oracle 10 以降にアップグレードすると、このパラメータが init.ora ファイルに含まれていることがありました。このパラメータが 8 や 16 に設定されていると、デフォルト値の 128 と比べてシーケンシャル I/O のパフォーマンスに多大な影響を及ぼします。

以下にネットアップの推奨事項を記載します。

- db_file_multiblock_read_count パラメータを init.ora ファイルに含めることはしないでください。ネットアップのお客さまでこのパラメータを変更することでパフォーマンスが向上した例はありませんが、シーケンシャル I/O のスループットに影響したことが明らかなケースは多数あります。

9.3 Redo ブロック サイズ

Oracle は、512 バイトまたは 4KB どちらかの Redo ブロック サイズに対応しています。デフォルトは 512 バイトです。Redo 処理の際に書き込まれるデータの量が最小限に抑えられるため、これが最良のオプションです。ただし、Redo ログが頻繁に作成される場合は 4KB のサイズを使用することでパフォーマンスが向上することがあります。例えば、50MBps で Redo ログが作成される単一のデータベースの場合、Redo ブロック サイズが大きい方が効率的な可能性があります。ストレージ システムが多数のデータベースをサポートしていて、Redo ログの合計が大容量になるケースでは、Redo ブロック サイズを 4KB にした方がメリットがあるかもしれません。この設定では、4KB ブロックの一部だけを更新しなければならないという、非効率な I/O の部分処理が発生しないことがその理由です。

すべての I/O 処理が Redo ログのブロック サイズ単位で実行されるわけではありません。Redo ログが頻繁に作成されるデータベースの場合、複数の Redo ブロックで構成された大容量の I/O 処理が実行されるのが一般的です。この場合、これらの Redo ブロックの実際のサイズがロギングの効率に影響することは通常ありません。

以下にネットアップの推奨事項を記載します。

- デフォルトのブロック サイズは、特定のアプリケーションに関して文書化された要件がある場合や、ネットアップまたは Oracle のカスタマー サポートから推奨された場合にのみ変更してください。

9.4 チェックサムとデータ整合性

ネットアップには、データベースのデータ整合性をどうやって確保すべきかという質問がよく寄せられます。特に、Oracle RMAN のストリーミング バックアップから Snapshot ベースのバックアップに移行するお客様から、この質問を多く受け取ります。RMAN には、バックアップ処理の際に整合性チェックを実行する機能が備わっています。この機能に意味がないわけではありませんが、主にメリットがあるのは、データベースが最新のストレージ アレイを使用していない場合です。Oracle データベースに物理ディスクを使用している場合、ディスクは老朽化に伴って最終的にはほぼ間違いなく破損します。ただし、本物のストレージ アレイなら、この問題はアレイベースのチェックサムで対処できます。

本物のストレージ アレイでは、複数のレベルでチェックサムが実行され、データ整合性が保護されます。IP ベースのネットワークでデータが破損すると、Transmission Control Protocol（TCP）レイヤはパケット データを拒否して再送信を要求します。FC プロトコルには、カプセル化された SCSI データと同じくチェックサムが含まれています。アレイに配置された Data ONTAP には RAID とチェックサムによる保護があるため、データの破損が避けられないとしても、ほとんどのエンタープライズ アレイ同様、検出されて修正されます。もっとも多いのはドライブ全体の破損で、この場合は RAID のリビルドが要求され、データベースの整合性が維持されます。ディスク上のデータが破損していることを示すチェックサム エラーが Data ONTAP で検出されることもあります。ディスクは障害状態となって RAID のリビルドが始まりますが、この場合もデータ整合性への影響はありません。

Oracle のデータファイルと Redo ログ アーキテクチャも、最悪の状況下でも可能な限り高水準のデータ整合性を提供するよう設計されています。最も基本的なレベルで言えば、Oracle のブロックにはチェックサムが含まれており、ほぼすべての I/O で基本的な論理チェックが実行されます。Oracle がクラッシュしたり表領域がオフラインにならないかぎり、データは維持されます。データ整合性のチェック レベルは調整可能で、また、書き込みを確認するよう Oracle を設定することもできるので、クラッシュや障害のシナリオはほぼすべてがリカバリ可能です。きわめて稀に、リカバリ不可能なイベントが発生することがありますが、その場合は即座に破損が検出されます。

Oracle データベースを使用しているネットアップのお客様は、一度 Snapshot ベースのバックアップに移行すると、RMAN などのバックアップ製品を使わなくなることがほとんどです。RMAN を使用できるオプションとしては、SMO によるブロックレベルのリカバリがまだあるにはありますが、日常レベルで言えば、RMAN や NetBackup などの製品は、1 カ月または四半期に一度アーカイブ コピーを作成するのに時々使用される程度です。

お客様によっては `dbv` を定期的に実行し、既存のデータベースに対して整合性チェックを実施しているケースもありますが、これは不要な I/O 負荷を生成するので、ネットアップでは推奨していません。前述のように、データベースに今まで問題が発生したことがないのなら、`dbv` が問題を検出する可能性はほぼゼロです。にもかかわらず、きわめて高いシーケンシャル I/O 負荷がネットワークとストレージ システムに生成されます。既知の Oracle バグが見つかったなど、破損を裏付ける理由がなければ、`dbv` を実行する意味はありません。

10 フラッシュ

フラッシュや SSD テクノロジを Oracle データベースに使用する場合の詳しい説明は、本ドキュメントの範囲ではありません。ただし、よくある疑問と誤解の一部は取り上げる必要があります。このセクションで説明する原則は、Oracle ASM を含むすべてのプロトコルとファイルシステムに該当します。

10.1 Flash Cache

NetApp Flash Cache インテリジェント データ キャッシングは、業界をリードするフラッシュベース テクノロジとして Oracle 環境に使用されてきました。理由はいたって単純で、ほとんどのデータベースがランダム リードのレイテンシによる制約を受ける現状において、Flash Cache はランダム リードのパフォーマンスをアクセラレートするシンプルな方法だからです。

ただし Flash Cache には、フラッシュ メモリを装備した PCIe カードをホストしている特定のノードに固定されるという制約があります。スピンドルはサイズが拡大するにつれて導入される本数が減っています。このため、コントローラ障害が発生した場合、テイクオーバー ノードの Flash Cache カードがウォーム アップする間、パフォーマンスが一時的に低下するリスクが高まっています。ウォームアップが完了するまでの間は、回転式メディアの I/O が大幅に増える可能性があります。この理由から、フラッシュ テクノロジとしては、テイクオーバー時にフラッシュ レイヤが回転式メディアに従う NetApp Flash Pool インテリジェント データ キャッシングを推奨します。ウォームアップは不要で、キャッシュはコールドになりません。だからといって、Flash Cache を使用した構成に欠点があるわけではありません。ほとんどのシステムには、コントローラのテイクオーバー中に発生する I/O の増加に耐えられるだけのスピンドルが搭載されています。

通常、データベース ワークロードにはデフォルトの設定が最も適しています。ただし、`flexscale.enable=on` を設定して Flash Cache を有効にする必要があります。

Flexscale.lopri_blocks

`flexscale.lopri_blocks` は、Flash Cache インテリジェント キャッシングの使用に関するパラメータです。デフォルト値は `off` で、ランダム オーバーライトやシーケンシャル I/O など、優先度の低いブロック処理の I/O はキャッシュされません。これは、ほとんどのデータベースはランダム リード処理のレイテンシによって制限されるためです。ランダム オーバーライトが発生すると、Oracle データベースはほぼ必ずそのブロックのコピーを作成しますが、ブロックがすぐに再度読み取られる可能性はほとんどありません。そのため、このオーバーライトをキャッシュすると、貴重な Flash Cache のスペースが無駄になってしまいます。またシーケンシャル読み取り I/O の場合、きわめて大きなブロック処理なので、ストレージ アレイ独自の効率化テクノロジで処理されます。基盤のディスクが SATA でも同様です。この種の I/O に Flash Cache は役立ちません。この I/O をキャッシュしようとすると、大抵は CPU に不要な負荷がかかり、本来ならランダム I/O のキャッシュに使用すべき貴重な Flash Cache のスペースが無駄になってしまいます。

以下にネットアップの推奨事項を記載します。

- このパラメータを変更する場合は、ネットアップのカスタマー サポートまたはプロフェッショナル サービスにご相談いただくか、または事前に徹底的にテストしてください。

Flexscale.read-ahead_blocks

`flexscale.read-head_blocks` は Data ONTAP 8.2 で追加されたオプションで、読み取りデータだけが対象であること以外は `flexscale.lopri_blocks` とほぼ同じです。通常の処理では、Flash Cache はランダム アクセス データのみを格納しますが、`flexscale.read-ahead_blocks` を有効にすると、シーケンシャル読み取りのキャッシュが有効になります。前述のように、シーケンシャル I/O はすでに十分効率よく処理されているため、フラッシュ メモリに格納しても、大抵の場合メリットはありません。

以下にネットアップの推奨事項を記載します。

- このパラメータを変更する場合は、ネットアップのカスタマー サポートまたはプロフェッショナル サービスにご相談いただくか、または事前に徹底的にテストしてください。

Flexscale.random_write_through_blocks

`flexscale.random_write_through_blocks` オプションは Data ONTAP 8.3 で追加されました。前の 2 つのオプションとは異なり、このパラメータは Oracle ワークロードに効果を発揮することがあります。ほとんどの場合、ランダム ライト データはデータベースがそのブロックのコピーを保持するので、キャッシュが必要ありません。ただし、Oracle キャッシュの負荷が高い場合、ランダム ライト ブロックの再読み取りが短期間で発生する可能性があるため、そのデータを Flash Cache にキャプチャするとパフォーマンスが向上します。

後述の Flash Pool 書き込みキャッシュの解説で書き込みキャッシュの効果を詳しく説明していますので参照してください。

以下にネットアップの推奨事項を記載します。

- デフォルトでは、このパラメータは `off` に設定されます。ただしネットアップは、別の設定を試すことを推奨しています。データベースが `db_file_sequential_read` のパフォーマンスによって制限を受ける場合は、`random_write_through_blocks` を有効にすることを検討してください。最近書き込まれたブロックをすぐに再読み取りするデータベースでは、レイテンシが低減します。

10.2 SSD アグリゲート

SSD やフラッシュ メディアを Redo ログに使用することには、多くの誤解があります。Redo ロギングのパフォーマンスを向上させるためには、データを SSD に書き込むことが必要です。SSD は、直接接続されたデバイスに使用するとロギングのパフォーマンス向上に有効ですが、ネットアップのストレージ アレイには、ミラーリングされた不揮発性の NVRAM ベースまたは NVMEM ベースのソリッドステートストレージが必ず搭載されています。Oracle データベースが書き込み処理を実行すると、NVRAM や NVMEM に書き込みがジャーナルされ次第、確認応答が返されます。最終的に書き込みを受信するドライブのタイプが、書き込みパフォーマンスに直接影響することはありません。

SSD アグリゲートや AFF プラットフォームを、Redo ロギングなどのシーケンシャル ライトのホスティングに使用したり、一時的なデータファイル I/O に使用しても効果はありません。ただし AFF を選択することで、書き込みパフォーマンスが間接的に向上する場合があります。例えば、処理するランダム I/O が大量に発生するシステムで、回転式メディアが過負荷になると、受信する書き込みをドライブがタイムリーに処理しきれなくなり、NVMEM や NVRAM がいっぱいになることがあります。このような場合、SSD アグリゲートや AFF プラットフォームに変更すれば Redo ログのパフォーマンスが向上します。ただしこれはあくまで間接的なメリットで、ランダム I/O が適切に処理されることで書き込みパフォーマンスの問題も解決されるということです。結果として、受信する書き込みが NVMEM や NVRAM に遅延なくすべて格納できるようになり、書き込みは正常な状態に戻ります。

時には計画ミスが原因で、SSD アグリゲートを使用していてもパフォーマンスが低下することがあります。SSD は回転式メディアをはるかに上回る高速パフォーマンスを提供しますが、SSD アグリゲートは、アグリゲートを構成するデバイスの数が SAS や SATA のアグリゲートより著しく少ないことがたまにあります。例えば、サーバのパフォーマンスに問題がある環境は、その原因が、Redo ログを含めた高負荷のシーケンシャルライト ワークロードを、100 本のドライブで構成された大規模 SAS アグリゲートから、わずか 4～5 台のデバイスで構成された小規模の SSD アグリゲートに移したことにある場合があります。SSD は SAS より高速かもしれませんが、限界がないわけではありません。

SSD アグリゲートに最も適しているのは、ランダム IO ワークロードの処理です。インデックス作成は特に SSD ドライブに適した処理です。他のタイプの IO も SSD に移して損にはなりませんが、アグリゲートを構成するドライブの本数が少なすぎないことが条件です。ただし、かなり過負荷のシステムから移動したのでないかぎりパフォーマンスの向上は期待できません。

10.3 Flash Pool

Flash Pool インテリジェント キャッシングにも、Flash Cache と SSD アグリゲートの根底にある原則が当てはまります。Flash Pool を使用すると、Oracle データベースにとって、大抵パフォーマンス上一番のボトルネックになるランダム リードのレイテンシが改善されます。このテクノロジはコスト削減効果にも優れています。Oracle の基盤にあるストレージ システムの多くは、大量のスピンドルを使用することで、大量のランダム リード アクティビティを最小限のレイテンシで処理しています。少量の Flash Pool を割り当てることで、多数の回転式ドライブに置き換えることができます。

Flash Pool には、書き込みキャッシュ、特にオーバーライト キャッシュという、Flash Cache では実現できない別のメリットもあります。書き込みはまず NVRAM や NVMEM に送信されるので、書き込みキャッシュに Flash Pool を使用しても、書き込みパフォーマンスへの直接的な効果はありません。レイテンシの観点で言えば、I/O はデータが NVRAM や NVMEM にジャーナルされた時点で完了します。また、受信した書き込みがその後どのタイプのメディアに格納されるかでパフォーマンスに影響が出ることもありません。ただし、Flash Pool の書き込みキャッシュを使って回転式メディアにかかる負荷を減らせば、アレイ全体の I/O パフォーマンスが総体的に向上するので、書き込みパフォーマンスに間接的にメリットがもたらされます。

Flash Pool の書き込みキャッシュは、ランダム オーバーライト ブロックをすばやく再読み取りする際の読み取りレイテンシも改善します。通常、データベースは書き込まれたブロックのコピーを保持するので、このプロセスはすべてのデータベースに有効なわけではありません。Oracle バッファ キャッシュのサイズが拡大してキャッシュされる書き込みが増えれば、ブロックをディスクから再度読み取る必要性は低下します。その場合は書き込みキャッシュを無効にして、フラッシュの貴重なスペースをランダム リード処理に残した方がよいかもしれません。

一方、同じブロックに対して繰り返されるオーバーライトを SSD レイヤにキャッシュすれば、回転式メディアの負荷が低減するというメリットが得られます。対象となるのは、Oracle のバッファ キャッシュに負荷がかかり、即座に再読み取りされるためだけにブロックがキャッシュから外されるといった状況です。

以下にネットアップの推奨事項を記載します。

- ランダムリードとランダムライトの両方のキャッシュを定義したデフォルトの Flash Pool ポリシーをそのまま使用します。
- 書き込みキャッシュにはメリットはないかもしれません。ほとんどの Oracle データベースにおいて、SSD のスペースが過度に使用されるほどのランダムライトは発生しませんが、デフォルトのポリシーでは必要に応じて書き込みキャッシュを使用できます。

主な例外は、データベース ワークロードに次の特性がある場合です。

- ワークロードがアグリゲートを占有し、そのため Flash Pool のキャッシングのほとんどを占めている。
- ワークロードがランダム リードのレイテンシによって制限されることがわかっている。
- 書き込みアクティビティが比較的少ない。
- Oracle のバッファ キャッシュが比較的大きい。

このような場合は、Flash Pool の書き込みキャッシュポリシーを `none` に変更することを検討してください。変更すると、SSD で読み取りキャッシュに使用できるスペースが最大になります。

Flash Pool は、例えば Oracle DataGuard を使用する場合など、Oracle スタンバイ データベースに対してしばしば効果があります。これは、スタンバイ データベースには普通バッファ キャッシュがなく、そのため、同じブロックに対して読み取り、更新、書き込みが何度も実行されるという厳しい I/O パターンが発生するためです。Flash Pool は、集中的なオーバーライト アクティビティを SSD レイヤにキャプチャすることで、回転式メディアへの負荷を軽減します。Flash Pool のようなテクノロジが登場するまでは、プライマリ データベースよりも、レプリケーション先のスタンバイ データベースの方に多くの回転式ディスクが必要となることも珍しくありませんでした。

10.4 All Flash FAS（AFF）プラットフォーム

NetApp AFF は、SSD アグリゲートの価値を拡大する製品として、パフォーマンスを向上させ、オールフラッシュ プラットフォーム向けに調整された動作をデフォルトで提供します。関連する各種ドキュメントは、ネットアップ サポートサイトで入手できます。

特に考慮すべきは、フラッシュの効果は IOPS に限定されるわけではない点です。ほかにも、パフォーマンスの一貫性や予測性、消費電力の削減、放出熱の低減などの個別のメリット、および将来的にも有効なソリューションという総合的なメリットがあります。

多くの場合、オールフラッシュ プラットフォームを使用すると、レイテンシを低減するためだけに回転式メディアを次々と導入する必要がなくなるため、コストが削減されます。導入コストが大幅に下がり続けていることから、最初から AFF を選択するお客様も増えています。

11　イーサネット構成

Oracle データベース ソフトウェアのインストールに必要な TCP / IP 設定は、通常、NFS や iSCSI のストレージ リソースすべてに優れたパフォーマンスを提供するのに十分です。ただし一部のケースでは、ネットワーク アダプタ メーカー固有の推奨事項を実装したところ、10Gb 環境でパフォーマンスの向上が認められました。

11.1 イーサネット フロー制御

このテクノロジを使用すると、クライアントは送信元に対し、データの送信を一時停止するよう要求できます。通常は、送られてくるデータの処理が受信側で間に合わない場合にこれを実行します。かつては、送信元にデータ転送の中止を要求した方が、バッファがフルになってから受信側でパケットを廃棄するよりもシステムへの影響を抑えることができました。OS に TCP スタックが使用されている現在、これは当てはまりません。実際フロー制御では、問題が解決されるよりも、問題が起きる例の方が多く見られます。

特に近年は、イーサネット フロー制御に起因するパフォーマンスの問題が増加しています。これは、イーサネット フロー制御が物理層で動作することに理由があります。すべてのデータベース サーバがイーサネット フロー制御要求をストレージ システムに送信できるようネットワークが構成されていたら、接続しているクライアントすべてに対する I/O が停止する結果になります。1 台のストレージ コントローラで対応するクライアントの数がますます増えていることから、そのうちのいくつかのクライアントがフロー制御要求を送信する確率は高まっています。OS の仮想化を幅広く導入しているお客様のサイトでは、頻繁にこの問題が起きるようになりました。

ネットアップ システムに搭載されているネットワーク インターフェイス カード（NIC）では、フロー制御要求を受信しないようにしてください。その方法は、ネットワーク スイッチのメーカーによって異なります。ほとんどの場合、イーサネット スイッチのフロー制御は `receive desired` または `receive on` に設定できます。これらの設定では、フロー制御要求はストレージ コントローラに転送されません。ただし、ストレージ コントローラのネットワーク接続によっては、フロー制御の無効化が許可されない場合があります。この場合は、クライアントがフロー制御要求を送信しないよう、データベース サーバの NIC の設定を変更するか、データベース サーバが接続しているスイッチ ポートの NIC の設定を変更してください。

以下にネットアップの推奨事項を記載します。

- ネットアップのストレージ コントローラがイーサネット フロー制御パケットを受信しないようにします。通常は、コントローラが接続されているスイッチ ポートを設定しますが、スイッチによってはクライアント側の変更が必要な場合があります。

11.2 ジャンボ フレーム

1Gb ネットワークにジャンボ フレームを使用すると、CPU とネットワークのオーバーヘッドが低減してパフォーマンスがいくらか向上しますが、通常、顕著な効果は得られません。こうした状況でも、可能であればジャンボ フレームを実装するようネットアップでは推奨しています。それは、パフォーマンスの向上と将来のニーズに応えるソリューションの 2 つを同時に実現できるからです。

10Gb ネットワークにはジャンボ フレームの使用がほぼ必須条件です。ほとんどの 10Gb 環境の場合、ジャンボ フレームを使用しないと、10Gb に達する前に 1 秒あたりのパケット数が最大値に達してしまいます。ジャンボ フレームを使用すると、データベース サーバ、NIC、ストレージ システムが処理する大容量のパケットの数が減るため、TCP / IP 処理が効率化されます。NIC によって差はありますが、パフォーマンスは大幅に向上します。

ジャンボ フレームの実装に関しては、接続されたデバイスすべてがジャンボ フレームをサポートし、MTU サイズがエンドツーエンドで同じでなければいけないという、誤った考えが広く共有されています。実際は、ネットワークの 2 つのエンド ポイントが、接続を確立する際に、相互に許容可能な最大のフレーム サイズをネゴシエートします。一般的な環境では、ネットワーク スイッチの MTU サイズは 9,216、ネットアップ コントローラは 9,000 にそれぞれ設定されており、クライアントは 9,000 と 1,514 が混在しています。MTU9,000 に対応できるクライアントにはジャンボ フレームを使用でき、1,514 しか対応できないクライアントはより低い値でネゴシエートできます。

スイッチだけで接続された環境では、この設定で問題が生じることはまずありません。ただしルーティングされた環境では、ルータによってジャンボ フレームが断片化されないよう注意が必要です。

以下にネットアップの推奨事項を記載します。

- ジャンボ フレームの使用を推奨しますが、1Gb イーサネット（GbE）の場合は必須ではありません。
- ジャンボ フレームは 10GbE のパフォーマンスを最大化するために必要です。

11.3 TCP パラメータ

TCP タイムスタンプ、選択的確認（SACK）、TCP ウィンドウの拡張の 3 つは、設定ミスの多いパラメータです。インターネット上にある古いドキュメントの多くは、この 3 つのパラメータの 1 つまたは複数を無効にしてパフォーマンスを高めるよう推奨しています。CPU 機能が今よりもはるかに低く、TCP 処理のオーバーヘッドをできるだけ軽減することに効果があった何年も前であれば、この推奨事項にもいくらかメリットがありました。

しかし最新の OS では、これらの TCP 機能を無効にしても通常は何もメリットもないだけでなく、場合によってはパフォーマンスが低下します。特に仮想ネットワーク環境では、この 3 つの機能が有効でないと、パケット損失やネットワーク品質の変化に効率よく対処できず、パフォーマンスが低下する可能性が高まります。

以下にネットアップの推奨事項を示します。

- ホストの TCP タイムスタンプ、SACK、TCP ウィンドウ拡張を有効にします。

12 一般的な NFS 構成

12.1 NFS のバージョン

Oracle で現在サポートされている NFS は NFS バージョン 3 のみです。このため、Oracle データベースに関して、ネットアップは NFSv4、NFSv4.1、pNFS をサポートしていません。本ドキュメントは、Oracle のサポート状況が変わった時点で更新される予定です。

以下にネットアップの推奨事項を記載します。

- 現時点では NFSv3 が必須です。

12.2 TCP スロット テーブル

TCP スロット テーブルは、ホスト バス アダプタ（HBA）キュー深度に相当する NFS 環境の機能で、同時に未処理となることのできる NFS 処理の数を制御します。デフォルト値は通常 16 ですが、最適なパフォーマンスを実現するには少なすぎます。その一方で、新しい Linux カーネルでは、TCP スロット テーブルの上限を NFS サーバが要求でいっぱいになるレベルにまで自動で引き上げることが可能なため、逆の問題が起きています。

パフォーマンスを最適化し、問題を回避するには、TCP スロット テーブルを制御するカーネルのパラメータを調整する必要があります。

`sysctl -a | grep tcp.*.slot_table` コマンドを実行し、以下のパラメータを確認します。

```
# sysctl -a | grep tcp.*.slot_table
sunrpc.tcp_max_slot_table_entries = 128
sunrpc.tcp_slot_table_entries = 128
```

`sunrpc.tcp_slot_table_entries` はすべての Linux システムで表示されるはずですが、`sunrpc.tcp_max_slot_table_entries` が表示されるのは一部のシステムのみです。どちらも 128 に設定します。

12.3 インストールとパッチの適用

以下のマウント オプションが `ORACLE_HOME` にあると、ホストのキャッシングが無効になります。

```
cio, actimeo=0, noac, forcedirectio.
```

これは、Oracle ソフトウェアのインストールとパッチ適用の速度に深刻な悪影響をもたらす恐れがあります。お客様の多くは、Oracle バイナリのインストール時やパッチ適用時に、このマウント オプションを一時的に削除します。インストールやパッチ適用中に、ターゲットの `ORACLE_HOME` を使用している他のアクティブな処理がないことを確認すれば、このマウント オプションは削除しても安全です。

12.4 clustered Data ONTAP と NFS フロー制御

状況によっては、clustered Data ONTAP を使用するためには、Oracle または Linux のカーネル パラメータを変更しなければならないことがあります。これは NFS フロー制御に関連した変更で、イーサネット フロー制御と混同しないでください。NFS フロー制御を使用すると、Data ONTAP などの NFS サーバは、データ受信の応答確認を送信していない NFSN クライアントとのネットワーク通信を制限できます。この機能により、正常に動作していない NFS クライアントが、自身の応答処理能力を超えてデータを要求してきた場合に NFS サーバを保護します。保護しないと、NFS サーバのネットワーク バッファが確認応答のないパケットでいっぱいになってしまいます。

稀に、Oracle DNFS クライアントと最新の Linux NFS クライアントの両方で発生した大量の I/O が、clustered Data ONTAP NFS サーバが自身を保護できる限界を超えてしまうことがあります。この場合、NFS クライアントは受信データの処理を遅らせる一方で、より多くのデータを引き続き要求しますが、この遅れが、NFS 接続でのパフォーマンスや安定性の問題を招く可能性があります。

実際に問題が発生することは稀ですが、ネットアップでは以下の保護対策をベストプラクティスとして推奨しています。この方法を適用できるのは clustered Data ONTAP のみです。また、この変更によってパフォーマンスに悪影響が及ぶことはありません。

clustered Data ONTAP に関するネットアップの推奨事項は次のとおりです。

- Oracle DNFS を使用する場合は、`DNFS_BATCH_SIZE` パラメータを `128` に設定します。このパラメータを使用できるのは Oracle 11.2.0.4 以降です。パラメータを設定できない場合は、DNFS を使用しないでください。
- 前述した TCP スロット テーブル パラメータを、どちらも必ず 128 に設定します。

12.5 Direct NFS

Oracle の DNFS クライアントは、ホスト NFS クライアントを迂回し、NFS ファイル処理を NFS サーバで直接実行するように設計されています。そのために必要となるのは、Oracle Disk Manager ライブラリの変更だけです。処理の手順は Oracle のドキュメントに記載されています。

DNFS を使用すると I/O が最も効率的な方法で処理されるため、I/O パフォーマンスが全体的に向上し、ホストとストレージ システムの負荷が低減します。さらに、DNFS はマルチパスとフォールト トレランスも提供します。例えば、2 つの 10Gb インターフェイスをバインドして、20Gb の帯域幅を提供することが可能です。一方のインターフェイスに障害が発生すると、もう一方のインターフェイスで I/O が再送信されます。全体的な処理は FC マルチパスとほとんど同じです。

DNFS を使用する場合は、Oracle のドキュメント 1495104.1 に記載のパッチがすべてインストールされていることが重要です。パッチをインストールできない場合は、環境を検証して、Oracle のドキュメント 1495104.1 に記されているバグによって問題が発生しないことを確認する必要があります。場合によっては、必要なパッチをインストールできないために DNFS を使用できないこともあります。

注意

- DNFS を使用する場合は、その前に、Oracle ドキュメント 1495104.1 に記載のパッチがインストールされていることを確認してください。
- Oracle 12c 以降の DNFS は、NFSv3、NFSv4、NFSv4.1 をサポートしています。ネットアップのサポート ポリシーはすべてのクライアントについて v3 と v4 をカバーしていますが、現時点では Oracle DNFS での NFSv4.1 の使用はサポートされていません。

12.6 Direct NFS とホスト ファイルシステム アクセス

アプリケーションやユーザのアクティビティが、ホストにマウントされた参照可能なファイルシステムを使用する場合、DNFS を使用すると問題が生じることがあります。これは DNFS クライアントが、ホスト OS の帯域外でファイルシステムにアクセスし、DNFS クライアントがファイルの作成、削除、修正を行っても OS は認識できないことが原因です。

シングルインスタンス データベースのマウント オプションを使用すると、ファイルやディレクトリの属性のキャッシュが有効になり、結果としてディレクトリの内容もキャッシュされます。そのため、DNFS がファイルを作成すると、OS がディレクトリの内容を再度読み取ってファイルが参照可能になるまでに、若干の遅延が生じます。これは普通は問題にはなりませんが、稀に、SAP BR*Tools などのユーティリティで問題になることがあります。その場合はマウント オプションを変更して、Oracle RAC に推奨される設定に変更することで、問題に対処できます。ただしこの変更によってホストのキャッシングがすべて無効になります。

マウント オプションは、（a）DNFS を使用していて、（b）問題がファイルが参照可能になるまでの遅延に起因している場合にのみ変更してください。DNFS を使用していない場合に、シングルインスタンス データベースで Oracle Real Application Cluster（RAC）マウント オプションを使用すると、パフォーマンスが低下します。

注：「Linux NFSv3 のマウント オプション」セクションの `nosharecache` に関する注を参照してください。異常な結果につながりかねない Linux 固有の DNFS の問題を取り上げています。

12.7 ADR_HOME と NFS

一部のお客様から、`ADR_HOME` にあるデータに対して大量の I/O が発生し、それが原因でパフォーマンスに問題が生じるという報告を受けています。この問題は一般に、高いパフォーマンスを必要とするデータが大量に蓄積されて初めて発生します。I/O が過剰に発生する理由は不明ですが、どうやら、Oracle が対象のディレクトリを何度もスキャンして変更しようとしていることに原因があるようです。

`noac` または `actimeo=0`（あるいは両方の）マウント オプションを削除すると、ホスト OS のキャッシングが実行可能になるため、ストレージの I/O レベルを下げることができます。

以下にネットアップの推奨事項を記載します。

- パフォーマンスに問題が生じる恐れがあるので、`noac` や `actimeo=0` が設定されたファイルシステムには `ADR_HOME` データを配置しないでください。`ADR_HOME` データは、必要に応じて別のマウント ポイントに分離するようにします。

13 一般的な SAN 構成

13.1 ゾーニング

FC ゾーンに複数のイニシエータを配置することは絶対に避けてください。イニシエータを複数配置すると、最初は問題なく動作しているように見えても、最終的にはイニシエータ間のクロストークがパフォーマンスや安定性を妨げます。

マルチターゲット ゾーンは一般に安全と考えられていますが、稀に、ベンダーの異なる FC ターゲット ポートの動作が原因で問題が生じることがあります。例えば、ネットアップ製と EMC 製のストレージ アレイのターゲット ポートを、どちらも同じゾーンに配置することは避けてください。また、ネットアップ ストレージ システムとテープ デバイスを同じゾーンに配置することは、問題が生じる可能性をさらに高めます。

13.2 LUN アライメント

LUN アライメントとは、基盤となるファイルシステムのレイアウトに合わせて I/O を最適化することです。ネットアップ システムの場合、ストレージのまとまりは 4KB 単位なので、Oracle データファイルの 8KB ブロックを、4KB ブロック 2 つに正確に揃えます。LUN の構成エラーのためにアライメントがどちらかの方向に 1KB ずれると、8KB の Oracle ブロック 1 つが、4KB のストレージ ブロック 2 つではなく 3 つにまたがってしまいます。この状態は、レイテンシの増大やストレージ システム内で実行される I/O の増大につながります。

論理ボリューム マネージャを使用しない場合、一般に LUN アライメントは単なる考慮事項でしかありません。実際に問題となるのは、Linux と Solaris です。論理ボリューム グループを構成する物理ボリュームを、ディスク デバイス全体に対して定義した場合（パーティションなし）、LUN の最初の 4KB ブロックとストレージ システムの最初の 4KB ブロックが揃います。これは正しいアライメントです。問題が起きるのはパーティションありの場合で、パーティションによって、OS が LUN を使用し始める場所がずれます。オフセットが 4KB 単位であれば、LUN はアライメントされます。

Linux 環境では、ディスク デバイス全体を使って論理ボリューム グループを構築してください。パーティショニングが必要な場合は、アライメントをチェックするために `fdisk -u` を実行し、各パーティションが 8 の倍数で始まっていることを確認します。具体的には、パーティションが 512 バイトの倍数、つまり 4KB のセクターで始まるということです。

「圧縮」セクションで取り上げている、圧縮ブロックのアライメントに関する解説も参照してください。圧縮ブロックの 8KB の境界にアラインされたレイアウトであれば、4KB の境界にもアラインされます。

Solaris 環境のアライメントは、より複雑です。詳細については、該当する Host Utilities のガイドを参照してください。

注意

Solaris x86 環境では、ほとんどの構成に複数のパーティション レイヤがあるため、アライメントを適切に行うには、より注意が必要です。Solaris x86 のパーティション スライスは、通常、標準のマスター ブート レコードのパーティション テーブルの上にあります。

13.3 LUN のミスアライメントの警告

Oracle の Redo ロギングは通常アライメントされていない I/O を生成するため、Data ONTAP で LUN がミスアライメントされているという警告が誤って発行されることがあります。Oracle Redo ロギングは、Redo ログ ファイルのシーケンシャル オーバーライトを様々なサイズの書き込みで実行します。4KB の境界にアラインしていないログの書き込み処理があっても、次のログ書き込み処理が 4KB ブロックを埋めるため、一般にパフォーマンスの問題が起きることはありません。つまり、Data ONTAP は、一部の 4KB ブロックが 2 つの異なる処理によって書き込まれていたとしても、ほぼすべての書き込みを完全な 4KB ブロックとして処理することができます。

アライメントを確認するには、`sio` または `dd` などのユーティリティを使用し、定義したブロック サイズで I/O を生成します。I/O を生成したら、`stats` コマンドを使用してストレージ システムの I/O アライメントの統計を確認できます。詳細については、「付録 3：WAFL アライメントの検証」を参照してください。

13.4 LUN のサイジング

LUN は、Data ONTAP 上の仮想オブジェクトで、ホストしているアグリゲートのすべてのスピンドルにわたって配置されます。そのため、LUN にはそのサイズに関係なくアグリゲートの能力がフルに活用されるので、サイズによって LUN のパフォーマンスに違いが出ることはありません。

使用に際しては便宜上、LUN を特定のサイズに設定できます。例えば、データベースの基盤になる ASM ディスクグループを 2 つの 1TB LUN で構築した場合、その ASM ディスクグループは 1TB 単位で拡張しなければなりません。8 個の 500GB LUN で ASM ディスクグループを構築した方が、ディスクグループの増分単位が小さくなって便利だと思われるかもしれませんが、

汎用性の高い標準的なサイズで LUN を構成することは、管理が複雑になる恐れがあるので推奨できません。例えば、標準サイズの 100GB LUN は、1～2TB 規模のデータベースには使いやすいかもしれませんが、20TB のデータベースの場合は LUN が 200 個必要になります。これは、サーバのリブートに長い時間がかかり、様々な UI で管理しなければならないオブジェクトが増え、SMO などの製品で検出が必要なオブジェクトが多数になることを意味します。LUN のサイズを大きくして数を減らせば、こうした問題は避けられます。

注：

- LUN の数は、サイズ以上に重要です。
- LUN のサイズは、必要な LUN の数で決まることがほとんどです。
- 必要以上の数の LUN を作成することは避けてください。

13.5 LUN のサイズ変更と LVM ベースのサイズ変更

SAN ベースのファイルシステムが最大容量に達した場合、次の 2 つの方法で使用可能なスペースを増やすことができます。

- LUN のサイズを拡張する。
- 既存のボリューム グループに LUN を追加し、グループ内の論理ボリュームを拡張する。

どちらの方法もサポート可能ですが、一般に LUN のサイズ拡張の方が難しく、リスクがあります。以下に考慮すべき点をいくつか挙げます。

- ONTAP で作成した LUN は、元のサイズの約 10 倍まで拡張可能です。この最大値は、ディスク ジオメトリ固有の構造に基づいています。10 倍以上に拡張できるケースも時々ありますが、それにはパーティション テーブルの変更が必要であり、ディスク構成をホスト レベルで詳しく理解していることが求められます。
- LUN のサイズを変更する際には、まずデータベースをシャットダウンし、代わりの手段としてフォールバック用に Snapshot コピーを作成することを推奨します。これはすべてのケースの要件ではありませんが、この手順を踏まなかった場合、サイズが拡大された LUN をホスト OS レベルで再検出する際に、システムの停止やユーザ エラーが発生するリスクがあります。
- LUN のサイズ変更がスムーズな唯一の例外は、Microsoft Windows です。Windows では NetApp SnapDrive for Windows を使用して、LUN のサイズを安全かつ無停止で拡張できます。

LUN のサイズ変更は容量を拡張するための 1 つの選択肢ですが、通常は Oracle ASM などの論理ボリューム マネージャ（LVM）を使用することをお勧めします。LVM の主な目的の一つは、LUN のサイズ変更を回避することです。LVM を使用すると、複数の LUN を 1 つの仮想ストレージ プールにまとめることができます。このディスク プールから切り出した論理ボリュームは LVM で管理され、簡単にサイズを変更できます。ほかにも、使用可能な LUN すべてに論理ボリュームを分散することで、特定のディスクにホットスポットが発生しないようにできるというメリットもあります。透過的な移行は、一般に、論理ボリューム マネージャを使用して、論理ボリュームの基盤であるエクステントを新しい LUN に再配置することで実行できます。

以上の理由から、LUN のサイズ変更を伴うオプションではなく、LVM の使用を推奨します。

13.6 LUN の数

LUN のサイズと異なり、LUN の数はパフォーマンスに大きく影響します。Oracle データベースのパフォーマンスは、SCSI レイヤを介して並列 I/O を実行する能力によって左右されます。そのため、LUN は 1 つよりも 2 つ使用した方がパフォーマンスが向上します。Veritas VxVM や Linux LVM2、Oracle ASM などの論理ボリューム マネージャを使用すると、並行処理を簡単に促進できます。

100%SSD の環境にランダム I/O で高い負荷をかけて実施したテストでは、LUN の数が 64 個まではパフォーマンスの向上がみられましたが、通常、ネットアップのお客様が LUN の数を 9 個以上に増やすことで得られるメリットはごくわずかです。ネットアップは、ボリューム グループを構築する際には、I/O を均一に分散できるサイズにエクステントを設定するよう推奨しています。例えば、10 個の 100GB LUN で構成された 1TB のボリューム グループで、エクステントのサイズを 100MB に設定すると、エクステントは合計で 10,000 個になります（LUN 1 つにつき 1,000 個のエクステント）。その結果、この 1TB のボリューム グループに配置されたデータベースの I/O は、10 個の LUN すべてに均等に分散されます。

ストライピングは普通必要ありません。むしろ避けるべきです。ほとんどのデータベースは、シーケンシャル パフォーマンスよりもランダム I/O のパフォーマンスによって制限されます。多数のエクステントにまたがってデータファイルを配置すれば、大量のランダム I/O を多数のエクステントにわたってランダムに分散できます。ボリューム グループに含まれるすべての LUN が均等に使用され、いずれか 1 つの LUN のためにパフォーマンスが制限されることはありません。

以下にネットアップの推奨事項を記載します。

- 通常データファイルの I/O は、4～8 個の LUN で十分にサポートできます。LUN を 3 個以下にすると、ホストの SCSI 環境の制限が原因でパフォーマンスが制限される場合があります。
- ストライピングは使用しないでください。代わりに LVM ポリシーを使用し、エクステント全体、つまり LUN 全体にデータを分散するようにします。

13.7 データファイルのブロック サイズ

OS の中には、ファイルシステムのブロック サイズを選択できるものがあります。ファイルシステムがサポートするデータファイルに関しては、圧縮を使用する場合、ブロック サイズを 8KB にしてください。圧縮が不要な場合、ブロック サイズは 8KB と 4KB のいずれでも構いません。

OS の中には、ファイルシステムのブロック サイズを選択できるものがあります。データファイルをサポートしているファイルシステムに関しては、ブロック サイズを 4KB にしてください。512 バイトのブロックを使用するファイルシステムにデータファイルが配置されていると、ファイルのミスアライメントが発生する可能性があります。LUN とファイルシステムがネットアップの推奨事項を基に適切にアライメントされていても、ファイルの I/O は適切にアライメントされないかもしれません。その場合は、パフォーマンスに深刻な問題が発生します。

ブロック サイズと圧縮の関係については、「Data ONTAP 8.3.10」セクションで詳細を参照してください。

13.8 Redo ブロック サイズ

Redo ログをサポートしているファイルシステムには、Redo ブロック サイズの倍数のブロック サイズを使用する必要があります。そのためには、通常、Redo ログのファイルシステムと Redo ログ自体がどちらも 512 バイトのブロック サイズを使用することが必要です。Redo 率が非常に高い場合、少ない処理数で効率よく I/O を実行可能なため、4KB のブロック サイズの方がパフォーマンスが良いことがあります。Redo 率が 50MBps を超える場合は、4KB のブロック サイズを試すことを検討してください。

一部のお客さまで、ブロック サイズが 512 バイトの Redo ログをブロック サイズが 4KB のファイルシステムで使用するデータベースで、ごく小さなトランザクションが大量に発生する場合、問題が報告されています。このパフォーマンスの問題は、複数の 512 バイト単位の変更を 4KB の 1 つのファイルシステム ブロックに適用する際のオーバーヘッドが原因ですが、ファイルシステムのブロック サイズを 512 バイトに変更することで解決されました。

以下にネットアップの推奨事項を記載します。

- Redo ブロック サイズの変更は、関連するカスタマー サポートまたはプロフェッショナル サービス部門からアドバイスされた場合、あるいは正式な製品マニュアルに基づく場合を除いて、実行しないでください。

14 仮想化

14.1 概要

データベースを VMware ESX や Oracle OVM、KVM を使用して仮想化することは、ネットアップのお客様の間でますます一般化しており、最もミッションクリティカルなデータベースにも仮想化が選択されています。

仮想化のサポート ポリシーに関しては、特に VMware 製品で多くの誤解があります。実際、Oracle はいっさい仮想化をサポートしないという話を耳にすることも珍しくありません。しかしこの見解は正しくなく、これを信じては仮想化の機会を逃してしまいます。Oracle はドキュメント ID 249212.1 で、Oracle 環境の既知の問題を解説するとともに、RAC のサポートを明記しています。

Oracle が把握していない問題に遭遇した場合、物理ハードウェアで問題を再現するよう要求されることがあります。また、最新バージョンの製品を使用しているお客様のなかには、新しいバグが見つかる可能性があるので仮想化は導入したくないと考えるお客さまもいます。しかし、一般に提供されているバージョンの製品を使用しているお客さまで、仮想化に際してこのような状況が問題になったことはありません。

14.2 ストレージの提供

データベースの仮想化を検討する場合、ビジネス ニーズを基にストレージを決めることが必要です。これは一般に IT に関するすべての決定に当てはまりますが、規模や範囲がプロジェクトごとに大きく異なる仮想化においては特に重要です。

ストレージの提供に関しては、VM ゲストでストレージ リソースを直接管理してください。そのためには、以下のいずれかのストレージ構成を使用します。

- iSCSI LUN を、ハイパーバイザーではなく VM の iSCSI イニシエータで管理します。
- NFS ファイルシステムを、仮想マシン ディスク（VMDK）ではなく VM でマウントします。
- VM ゲストがファイルシステムを管理する場合は、FC raw デバイス マッピング（RDM）を使用します。

一般的なルールとして、Oracle ファイルにデータストアを使用することは避けてください。これには次のように多くの理由があります。

- **透明性：**VM が自身のファイルシステムを所有していると、データベース管理者やシステム管理者がデータについてファイルシステムのソースを特定しやすくなります。
- **パフォーマンス：**テスト結果から、すべての I/O をハイパーバイザー データストア経由でチャネリングすると、パフォーマンスに影響することがわかっています。
- **管理性：**VM が自身のファイルシステムを所有していると、ハイパーバイザー レイヤを使用するかどうかが管理性に影響します。ハイパーバイザー レイヤを使用していれば、サイトに仮想環境と非仮想環境が混在していても、同じ手順でサイト全体のプロビジョニング、監視、データ保護などを実行できます。
- **安定性とトラブルシューティング：**VM が自身のファイルシステムを所有していると、ストレージ スタック全体が VM 上に存在するため、安定した高パフォーマンスが実現し、問題のトラブルシューティングも一段と簡単になります。ハイパーバイザーは、FC や IP のフレームを転送する役目だけを担います。構成にデータストアが含まれていると、タイムアウト、パラメータ、ログ ファイル、および潜在的なバグが新たに加わるため、設定が複雑になります。
- **移動性：**VM が自身のファイルシステムを所有していると、Oracle 環境を移動するプロセスはずっとシンプルになり、ファイルシステムを仮想ゲストと非仮想ゲストの間で簡単に移動できます。
- **ベンダー ロックイン：**データをデータストアに配置すると、別のハイパーバイザーを活用したり、仮想環境からデータを取り出すことがすべてにおいてきわめて困難になります。
- **Snapshot の有効化：**一部の仮想環境では、帯域幅が比較的限られているため、バックアップ作業が問題になることがあります。例えば、多数の仮想データベースで日々必要とされるパフォーマンスを満たすには、4 ポートの 10GbE トランクで十分だったとします。RMAN などのバックアップ製品を使用してバックアップを実行するとなると、データのフルサイズのコピーをストリーミングする必要があるため、このトランクでは不十分です。

  VM 所有のファイルシステムを使用すると、Snapshot ベースのバックアップとリストアを簡単に活用できます。VM 所有のファイルシステムはストレージ システムにバックアップ処理をオフロードするため、バックアップウィンドウの帯域幅と CPU の要件をみたすためだけに、本来なら不要なハイパーバイザー構成を構築する必要はありません。

以下にネットアップの推奨事項を記載します。

- パフォーマンスと管理性を最適化するためには、Oracle データをデータストアに配置することは避けてください。ゲストが所有するファイルシステム（ゲストまたは RDM で管理される NFS や iSCSI のファイルシステム）を使用します。

14.3 準仮想化ドライバ

最適なパフォーマンスを実現するには、準仮想化ネットワーク ドライバを使用することが重要です。データストアが使用されている場合、準仮想化 SCSI ドライバは必須です。準仮想化デバイス ドライバを使用すると、エミュレートされたドライバとは異なり、ゲストがより緊密にハイパーバイザーと統合されます。エミュレートされたドライバを使用した場合、ハイパーバイザーは物理ハードウェアの動作の模倣により多くの CPU 時間を消費します。

ほとんどのデータベースのパフォーマンスはストレージによって制限されます。そのため、ネットワーク ドライバや SCSI ドライバが原因のレイテンシが特に目立ちます。ネットアップのカスタマー サポートに寄せられるパフォーマンスに関する苦情の多くは、準仮想ドライバをインストールすることで解決されています。あるお客様の POC では、データベースに ESX を使用したところ、同じハードウェアをベア メタルとして使用した場合よりもパフォーマンスが向上しました。テストには大量の I/O を使用しましたが、パフォーマンスに違いが出たのは、ESX 準仮想ネットワーク ドライバを使用したことが理由です。

以下にネットアップの推奨事項を記載します。

- 準仮想化ネットワーク ドライバと SCSI ドライバを常に使用してください。

14.4 RAM のオーバーコミット

RAM のオーバーコミットとは、物理ハードウェアに存在する容量よりも多くの仮想 RAM を、様々なホストに構成することです。これは、パフォーマンスに予期せぬ問題を引き起こす可能性があります。データベースを仮想化する際は、Oracle SGA の基盤のブロックが、ハイパーバイザーによってディスクにスワップ アウトされないようにしてください。ブロックがスワップ アウトされると、パフォーマンスがきわめて不安定になります。

以下にネットアップの推奨事項を記載します。

- ハイパーバイザーは、Oracle SGA のブロックがスワップ アウトされないように設定します。

15 クラスタリング

15.1 Oracle Real Application Cluster

このセクションの内容が当てはまるのは、Oracle 10.2.0.2 以降です。それよりも前のバージョンについては、本ドキュメントに併せて Oracle のドキュメント ID 294430.1 を参照し、最適な設定を確認してください。

disktimeout

プライマリ ストレージに関連する RAC パラメータは `disktimeout` です。このパラメータは、Voting ファイル I/O がそれまでに完了しなければならないタイムアウトを制御します。`disktimeout` パラメータの値を超えると、その RAC ノードはクラスタから除外されます。デフォルト値は 200 です。標準的なストレージ テイクオーバーやギブバックには、この値で十分です。

テイクオーバーやギブバックには多くの要素が影響するため、ネットアップでは、RAC 構成を本番環境に導入する前に徹底的にテストすることを強く推奨しています。ストレージ フェイルオーバーにかかる時間に加えて、例えば Link Aggregation Control Protocol（LACP）の変更が伝搬する時間、SAN マルチパス ソフトウェアが I/O のタイムアウトを検出して代替パスでデータを再送信する時間、さらにデータベースへのアクセスが大量に発生している場合は、Voting ディスク I/O を処理する前に大量の I/O をキューして再試行する時間が必要です。

ストレージを実際にテイクオーバーしたりギブバックしたりできない場合は、データベース サーバのケーブルを外すテストで影響をシミュレーションできます。

以下にネットアップの推奨事項を記載します。

- `disktimeout` パラメータは、デフォルトの 200 から値を変更しないでください。
- RAC 構成は必ず徹底的にテストしてください。

misscount

`misscount` パラメータは通常、RAC ノード間のネットワーク ハートビートにのみ影響します。デフォルトは 30 秒です。Grid バイナリがストレージ アレイ上にある場合や OS のブート ディスクがローカルでない場合、このパラメータが重要になることがあります。これには、ブート ディスクが FC SAN、NFS ブート OS、および VMDK ファイルなどの仮想データストアにあるホストが該当します。

ブート ディスクへのアクセスがストレージのテイクオーバーやギブバックで中断すると、Grid バイナリの場所または OS 全体が一時的にハングする可能性があります。その場合、Data ONTAP がストレージ処理を完了するのに必要な時間、および OS がパスを変更して I/O を再開するのに必要な時間が、misscount のしきい値を超える場合があります。結果、ブート LUN や Grid バイナリへの接続がリストアされたあと、ノードは即座に削除されます。大抵の場合、削除とその後のリブートでは、リブートの理由についてのメッセージはログに記録されません。影響はすべての構成に及ぶわけではないので、RAC 環境内の SAN ブート、NFS ブート、データ ストアベースのホストをすべてテストし、ブート ディスクへの通信が中断しても RAC が不安定にならないことを確認します。

ブート ディスクがローカルでない場合や、ファイルシステムが grid バイナリをホストしている場合は、disktimeout と同じ値に misscount を変更してください。このパラメータを変更した場合は、ノードのフェイルオーバーの時間など、RAC の動作への影響がないかどうかをさらにテストしてください。

以下にネットアップの推奨事項を記載します。

- misscount パラメータはデフォルト値の 30 から変更しないでください。ただし、以下の状況のいずれかが当てはまる場合は別です。
  - grid バイナリが、ネットワークに接続されたディスク（NFS、iSCSI、FC、データストアベースのディスクを含む）にある場合
  - OS が SAN ブートの場合
- 上記の場合は、ネットワークの中断が OS や GRID_HOME ファイルシステムへのアクセスにどのように影響するかを検証します。場合によっては、中断によって Oracle RAC デーモンが停止し、misscount に基づくタイムアウトや削除につながる可能性があります。タイムアウトのデフォルトは 27 秒で、misscount から reboottime を引いた値に相当します。この場合、misscount を 200 に増やして disktimeout と同じにします。

15.2 Solaris Cluster

アクティブ / パッシブ クラスタリング テクノロジである Solaris Cluster は、他のクラスタウェアよりもはるかに高度に統合されています。ほぼプラグアンドプレイに匹敵する操作で、データベースとアプリケーションをクラスタ リソースとして簡単に導入し、クラスタ内で簡単に移動することができます（関連する IP アドレス、構成ファイル、ストレージ リソースを含む）。この緊密な統合には、すべてのコンポーネントが適切に連携するよう、Solaris Cluster 向けの厳しい対応チェックが組み込まれています。

Data ONTAP は、SAN 環境の Solaris Cluster を幅広くサポートしています。詳細は、Interoperability Matrix Tool（IMT）を参照してください。

NFS 環境でのサポートには制限があります。一般に NFS にサポートへの障害になる要素はありませんが（自動マウントされた NFS ホーム ディレクトリの使用など）、データベースを Solaris Cluster の制御下に置くことはできません。以前は NFS エージェントを使用できましたが、この製品は 2012 年 10 月にサポートが終了しました。Solaris Cluster の標準機能を使用してクラスタ化できるカスタム サービスを構築することは可能ですが、この方法はおそらくほとんどの環境には適していません。ストレージなどのリソースを管理するスクリプトの作成に時間と手間がかかることがその理由です。

他のアクティブ / パッシブ クラスタリング テクノロジを探している場合は、Oracle RAC One Node を検討してください。Oracle RAC One Node は Oracle RAC よりも安いライセンス料で使用できますが、Solaris Cluster と同じ、アプリケーションに統合されたクラスタ制御機能が多数含まれています。

15.3 Veritas Cluster Server

Veritas Cluster Server（VCS）は Solaris Cluster とほぼ同じで、データベースやアプリケーションを導入サービスとしてパッケージ化し、アクティブ / パッシブ クラスタに導入できます。

VCS と SAN

Data ONTAP は、SAN 環境の VCS クラスタリングを幅広くサポートしています。詳細は、Interoperability Matrix Tool（IMT）を参照してください。

VCS と NFS

以前ネットアップでは、クォーラム、監視、管理、フェンシング、NFS ロック解除機能用に NFS クライアントを提供していましたが、主にこれらの機能が必要とされなくなったという理由から、NFS クライアントのサポートは 2012 年 10 月に終了しました。現在は VCS を使用して、標準で NFS ファイルシステムを管理、監視できます。NAS 環境のクォーラムの管理には、エージェントを必要としない複数の選択肢があります。

VCS と NFS フェンシング

アクティブ / パッシブ クラスタリングでは、フェンシングが一つの考慮事項です。フェンシングとは、ストレージ リソースを使用できるノードがクラスタの 1 つのノードに限定されることです。SAN 環境の場合、SCSI 永続的予約を使用してフェンシングを実装します。SCSI 永続的予約では、ノードが LUN を独占的に制御することを要求できます。NFS の場合は、ファイルシステムのエクスポート オプションを変更して、複数のノードのリソースにアクセスできないようにします。2 つの違いは、SAN 環境では、ストレージ リソースを要求しているクラスタ内のノードによってフェンシングが実行されるのに対し、NAS 環境では、ストレージ システム上でフェンシングを実行しなければならない点です。

NFS の場合、フェンシングは必須ではありません。SAN 環境の場合、SAN ファイルシステムを複数のサーバにマウントするという単純な処理によってデータが即座に破損してしまうことが珍しくないため、フェンシングの実装はより重要です。NFS はクラスタ化されたファイルシステムであるため、複数のサーバに 1 つのファイルシステムを問題なくマウントできます。

お客様の多くは、VCS、および HP ServiceGuard や IBM PowerHA などの同等の製品を使用してアクティブ / パッシブ クラスタリングを実装していますが、フェンシングは導入していません。1 つのリソースが 1 つのノードだけで動作することが、クラスタ ソフトウェアによって保証されるためです。フェンシングが必要な場合は、スクリプト作成に少々手間をかけるだけで、クラスタ リソースの一部として導入できます。

スクリプトの内容としては、サービスの開始時にストレージ システムに対してコマンドを実行し、（a）ターゲット ファイルシステムからすべてのノードへのアクセスを遮断し、続いて（b）サービスを開始する 1 つのノードへのアクセスを許可します。結果、このノードだけがターゲット ファイルシステムで I/O を実行できます。サービスのシャット ダウン時には、アクセスを削除するコマンドをストレージ システムに対して実行します。ほかにもバリエーションはありますが、これが最も包括的なアプローチです。

これらのシステムのサポートは、ネットアップ プロフェッショナル サービスで提供しています。詳細は、ネットアップの担当者にお問い合わせください。

VCS と NFS ロック解除

Oracle 環境の NFS ロックは、フェンシングの一種です。ターゲット ファイルに NFS ロックを検出した場合、Oracle データベースは起動しません。一般に、VCS 環境で NFS ロックを使用すると、VCS クラスタの正常な動作が妨げられます。ロックの解除が必要になるのは、正常にシャット ダウンしなかったノードのサービスを、別のノードがテイクオーバーするときのみです。Oracle データベースのクリーン シャットダウン時には、ロックは削除されます。ノードがクラッシュした場合、ロックは維持されるため、データベースを再起動するにはロックの解除が必要です。

ほとんどのお客様は、最初からロックが作成されないようにする NFS マウント オプションを指定することで、NFS ロックを無効にすることを選択しています。この方法が望ましくない場合は、スクリプトによってロックを解除できます。フェンシングの場合と同様、ロック解除のスクリプト作成にはネットアップ プロフェッショナル サービスのサポートを利用できます。場合によっては、Rapid Response Engineering を通じて完全にサポートされたオプションを利用できます。詳細は、ネットアップの担当者にお問い合わせください。

16 IBM AIX

このセクションでは、IBM AIX オペレーティング システム固有の構成について解説します。

16.1 同時 I/O

IBM AIX で最適なパフォーマンスを実現するには、同時 I/O を使用する必要があります。AIX は連続したアトミックな（不可分な）I/O を実行するため、大量のオーバーヘッドが発生します。そのため、同時 I/O を使用しないとパフォーマンスが制限される可能性があります。

従来ネットアップでは、`cio` マウント オプションを使用して、ファイルシステムで強制的に同時 I/O を実行することを推奨していました。ただしこの処理には欠点があるため、現在は推奨していません。AIX 5.2 と Oracle 10gR1 が登場して以来、AIX 上の Oracle では、個々のファイルを開いて同時 IO を実行できるようになり、ファイルシステム全体で同時 I/O を強制的に実行する必要がなくなりました。

同時 I/O を有効にする一番の方法は、`init.ora` のパラメータ `filesystemio_options` を `setall` に設定することです。これにより、Oracle が特定のファイルを開いて同時 I/O に使用できるようになります。

マウント オプションとして `cio` を使用すると、同時 I/O の使用が強制されるため、マイナスの影響が発生することがあります。例えば、同時 I/O を強制するとファイルシステムの先読みが無効になり、Oracle データベース ソフトウェアの外部で発生する I/O（ファイルの複製、テープ バックアップの実行など）のパフォーマンスが低下する恐れがあります。さらに、Oracle のバージョンによっては、Oracle GoldenGate や SAP BR*Tools などの製品と `cio` マウント オプションの使用には互換性がありません。

以下にネットアップの推奨事項を記載します。

- ファイルシステム レベルで `cio` マウント オプションを使用することは避け、`filesystemio_options=setall` を使用して同時 I/O を有効にしてください。
- `cio` マウント オプションは、`filesystemio_options=setall` を設定できない場合のみ使用してください。

16.2 AIX NFSv3 のマウント オプション

表 1 と表 2 に、AIX NFSv3 のマウント オプションをまとめました。

表 1）AIX NFSv3 のマウント オプション ― シングル インスタンス

ファイル タイプ	マウント オプション
`ADR_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
`ORACLE_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,intr

表 2）AIX NFSv3 のマウント オプション ― RAC

ファイル タイプ	マウント オプション
`ADR_HOME`	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=cp,timeo=600,rsize=65536,wsize=65536,nointr,noac

ファイル タイプ	マウント オプション
CRS / Voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac
専用 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
共有 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr

シングルインスタンスと RAC マウント オプションの主な違いは、RAC ではマウント オプションに noac がある点です。このオプションを使用するとホスト OS のキャッシングが無効になるため、データの状態について、RAC クラスタ内のすべてのインスタンスが一貫した情報を認識できるようになります。

cio マウント オプションと init.ora パラメータ filesystemio_options=setall を使用した場合も同様にホストのキャッシングが無効になりますが、noac は引き続き使用する必要があります。Noac は、共有 ORACLE_HOME 環境で、Oracle パスワード ファイルや spfile パラメータ ファイルなどのファイルの整合性を維持するために必要です。RAC クラスタの各インスタンスに専用の ORACLE_HOME がある場合、このパラメータは必要ありません。

16.3 AIX JFS / JFS2 のマウント オプション

表 3 に、AIX JFS / JFS2 のマウント オプションをまとめました。

表 3）AIX JFS / JFS2 のマウント オプション ― シングル インスタンス

ファイル タイプ	マウント オプション
ADR_HOME	デフォルト
制御ファイル データファイル Redo ログ	デフォルト
ORACLE_HOME	デフォルト

データベースを含むあらゆる環境で AIX hdisk デバイスを使用する場合は、パラメータ queue_depth をチェックします。このパラメータは HBA のキュー深度ではなく、個々の hdisk デバイスの SCSI キュー深度に関連するパラメータです。LUN の構成方法によっては、高いパフォーマンスを達成するには queue_depth の値が低すぎることがあります。テストでは、最適な値は 64 という結果が出ています。

17 HP-UX

このセクションでは、HP-UX オペレーティング システムに固有の構成について解説します。

17.1 HP-UX NFSv3 のマウント オプション

表 4 に、HP-UX NFSv3 のマウント オプションをまとめました。

表 4）HP-UX NFSv3 のマウント オプション ― シングル インスタンス

ファイル タイプ	マウント オプション
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,forcedirectio,nointr,suid
ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid

表 5）HP-UX NFSv3 のマウント オプション ― RAC

ファイル タイプ	マウント オプション
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,noac,suid
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac forcedirectio, suid
CRS / Voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac, forcedirectio,suid
専用 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid
共有 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac,suid

シングルインスタンスと RAC マウント オプションの主な違いは、RAC にはマウント オプションに noac と forcedirectio がある点です。このオプションを使用するとホスト OS のキャッシングが無効になるため、データの状態について、RAC クラスタ内のすべてのインスタンスが一貫した情報を認識できるようになります。init.ora パラメータ filesystemio_options=setall を使用した場合も同様にホストのキャッシングが無効になりますが、noac と forcedirectio は引き続き使用する必要があります。

noac は、共有 ORACLE_HOME 環境で、Oracle パスワード ファイルや spfile などのファイルの整合性を維持するために必要です。RAC クラスタの各インスタンスに専用の ORACLE_HOME がある場合、このパラメータは必要ありません。

17.2 HP-UX VxFS のマウント オプション

Oracle のバイナリをホストしているファイルシステムには、次のマウント オプションを使用します。

```
delaylog,nodatainlog
```

データファイル、Redo ログ、アーカイブ ログ、制御ファイルを格納しているファイルシステムで、HP-UX のバージョンが同時 I/O をサポートしていない場合は、次のマウント オプションを使用します。

```
nodatainlog,mincache=direct,convosync=direct
```

同時 I/O がサポートされている場合は（VxFS 5.0.1 以降、または ServiceGuard Storage Management Suite を使用している場合）、データファイル、Redo ログ、アーカイブ ログ、制御ファイルを格納しているファイルシステムで次のマウント オプションを使用します。

```
delaylog,cio
```

注：パラメータ db_file_multiblock_read_count は、VxFS 環境では特に重要です。Oracle は、別途指示された場合を除き、Oracle 10g R1 以降ではこのパラメータを設定しないままにするよう推奨しています。Oracle の 8KB ブロック サイズの場合、デフォルトは 128 です。このパラメータの値を強制的に 16 以下にする場合は、convosync=direct マウント オプションを削除し、シーケンシャル I/O のパフォーマンスが低下しないようにします。この処置はパフォーマンスの別の面にも影響するので、db_file_multiblock_read_count の値をデフォルト値から変更する必要があると判断した場合のみにしてください。

18 Linux

このセクションでは、Linux OS に固有の構成について解説します。

18.1 Linux NFS

スロット テーブル

Linux の NFS パフォーマンスは tcp_slot_table_entries というパラメータに依存します。このパラメータは、Linux OS で許容される未処理の NFS 処理の数を制御します。

2.6 の派生カーネル（RH5 と OL5 を含む）のほとんどではデフォルトは 16 ですが、このデフォルト値はパフォーマンス問題の原因となることがよくあります。逆にこれよりも新しいカーネルでは tcp_slot_table_entries の値に上限がないため、大量の要求によってシステムがフラッディングし、ストレージの問題が発生します。

解決策は、この値を静的に設定することです。NetApp NFS ストレージと Oracle データベースを使用する Linux OS では、値を 128 に設定してください。

RHEL 6.2 以前でこの値を設定するには、/etc/sysctl.conf に以下のエントリを追加します。

```
sunrpc.tcp_slot_table_entries = 128
```

また、2.6 カーネルを使用する Linux ディストリビューションのほとんどにはバグがあり、NFS クライアントがロードされる前に起動プロセスが/etc/sysctl.conf の内容を読み取ります。そのため NFS クライアントがロードされる時には、デフォルト値の 16 が使用されてしまいます。この問題を回避するには、/etc/init.d/netfs を編集してスクリプトの 1 行目で/sbin/sysctl -p を呼び出し、NFS がファイルシステムをマウントする前に tcp_slot_table_entries が 128 に設定されるようにします。

RHEL 6.3 以降でこの値を設定するには、RPC 構成ファイルに以下の変更を適用します。

```
echo "options sunrpc udp_slot_table_entries=64 tcp_slot_table_entries=128
tcp_max_slot_table_entries=128" >> /etc/modprobe.d/sunrpc.conf
```

18.2 Linux NFSv3 のマウント オプション

表 6 と表 7 に、NFSv3 のマウント オプションをまとめました。

表 6）Linux NFSv3 のマウント オプション ― シングル インスタンス

ファイル タイプ	マウント オプション
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr
ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr

表 7）Linux NFSv3 のマウント オプション ― RAC

ファイル タイプ	マウント オプション
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,actimeo=0
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,actimeo=0
CRS / voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,noac,actimeo=0
専用 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
共有 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr,actimeo=0

シングルインスタンスと RAC マウント オプションの主な違いは、RAC ではマウント オプションに actimeo=0 がある点です。このオプションを使用するとホスト OS のキャッシングが無効になるため、データの状態について、RAC クラスタ内のすべてのインスタンスが一貫した情報を認識できるようになります。init.ora パラメータ filesystemio_options=setall を使用した場合も同様にホストのキャッシングが無効になりますが、actimeo=0 は引き続き使用する必要があります。

actimeo=0 は、共有 ORACLE_HOME 環境で、Oracle パスワード ファイルや spfile などのファイルの整合性を維持するために必要です。RAC クラスタの各インスタンスに専用の ORACLE_HOME がある場合、このパラメータは必要ありません。

一般に、非データベース ファイルはシングルインスタンスのデータファイルと同じオプションでマウントします。ただしアプリケーションによっては、要件が異なる場合があります。マウント オプションの noac と actimeo=0 は、ファイルシステムレベルの先読みとバッファを無効にするため、できるだけ使用しないでください。抽出や変換、ロードなどのプロセスに深刻なパフォーマンス問題を引き起こす恐れがあります。

ACCESS と GETATTR

一部のお客様の間では、ACCESS や GETATTR などの IOPS が大量に発生し、ワークロードが占有される状況が観察されています。極端な例では、読み取りや書き込みなどの処理が、すべての処理の 10%にまで低下することもあります。Linux 上で actimeo=0 や noac を使用しているデータベースの場合、これは正常な動作です。この 2 つのオプションを指定すると、Linux OS はファイルのメタデータをストレージ システムから定期的にリロードします。ACCESS や GETATTR などの処理は影響力の低い処理で、データベース環境では Data ONTAP のキャッシュによって処理されます。読み取りや書き込みのように、ストレージ システムに本来の意味での要求を送信する純粋な IOPS とみなすことはできません。ただし、特に RAC 環境では、これらの IOPS はそれなりの負荷を生成します。この状況に対処するには、DNFS を有効にし、OS のバッファ キャッシュを迂回して不要なメタデータ処理を回避してください。

Linux Direct NFS

もう 1 つのマウント オプション nosharecache は、（a）DNFS が有効で、（b）1 つのソース ボリュームが 1 台のサーバに複数回マウントされていて、（c）ネストされた NFS マウントが使用されている場合に必要です。このような構成は、主に SAP アプリケーションをホストしている環境に見られます。例えば、ネットアップ システムの 1 つのボリュームが、/vol/oracle/base と /vol/oracle/home にディレクトリを持っているとします。/vol/oracle/base が/oracle でマウントされ、/vol/oracle/home が/oracle/home でマウントされている場合、同じソースに基づく NFS マウントがネストされる結果になります。

OS は、/oracle と/oracle/home が同じボリューム、つまり同じソース ファイルシステムに存在することを検出し、同じデバイス ハンドルを使用してデータにアクセスします。こうすることで、OS キャッシングなどの処理は改善されますが、一方で DNFS の妨げとなります。DNFS があるファイル（例えば/oracle/home の spfile）にアクセスしなければならない場合、誤ってデータへの間違ったパスを使用する可能性があります。この場合、I/O 処理は失敗します。このような場合は、ソースの FlexVol をそのホスト上の別の NFS ファイルシステムと共有するすべての NFS ファイルシステムに、nosharecache マウント オプションを追加してください。このオプションを指定したファイルシステムには、Linux OS によって個別のデバイス ハンドルが割り当てられます。

Linux Direct NFS と Oracle RAC

RAC のノード間で一貫性を保つにはダイレクト I/O が必要ですが、Linux にはその手段がありません。そのため、Linux OS で稼働する Oracle RAC には DNFS を使用することで特別なパフォーマンス メリットがあります。代わりに Linux では actimeo=0 マウント オプションを使用する必要があります。このオプションを使用すると、OS キャッシュのファイル データは即座に期限切れになります。一方でこのオプションを指定すると、Linux NFS クライアントによって属性データが定期的に再読み取りされるため、レイテンシに影響が及び、ストレージ コントローラの負荷が増大します。

DNFS を有効にするとホスト NFS クライアントが迂回されるため、この悪影響を回避できます。DNFS を有効にしたところ、RAC クラスタのパフォーマンスが著しく向上し、Data ONTAP の負荷が（特に特殊な IOPS に関して）大幅に低減したという報告が多数のお客様から寄せられています。

Linux Direct NFS と oranfstab ファイル

マルチパス オプションを設定して Linux に DNFS を使用する場合は、マルチパス サブネットを使用する必要があります。他の OS の場合、マルチパス DNFS チャネルを確立するには、LOCAL オプションと DONTROUTE オプションを使用して、1 つのサブネットに複数の DNFS チャネルを構成すればよいのですが、Linux ではこの方法が適切に機能せず、予期しないパフォーマンスの問題が起きることがあります。Linux では、DNFS トラフィックに使用している NIC を、それぞれ別のサブネットに割り当ててください。

18.3 一般的な Linux SAN 構成

圧縮のアライメント ー パーティショニング

圧縮で最適な効果をあげるには、8KB のディスク境界へのアライメントが必要です。アライメントをチェックするには、-u オプションを指定して fdisk ユーティリティを使用し、セクターに基づいてディスクを表示します。以下の例を参照してください。

```
[root@jfs0 etc]# fdisk -l -u /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
64 heads, 32 sectors/track, 10240 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 65536 bytes
Disk identifier: 0xb97f94c1

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1              36    20971519    10485742   83  Linux
Partition 1 does not start on physical sector boundary.
```

このパーティションは 8KB でアライメントされていません。パーティションのオフセットは 36 セクターで、4KB の境界にアラインされています。これは一般に優れたパフォーマンスに必要なアライメントですが、8KB の境界にはアライメントされていません。パーティションの開始位置を 16 セクターの倍数にして（512 バイト x16=8,192）、パーティションをアライメントする必要があります。

以下に、正しくアライメントされたパーティションの例を示します。

```
[root@jfs0 etc]# fdisk -l -u /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
64 heads, 32 sectors/track, 10240 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
```

```
I/O size (minimum/optimal): 4096 bytes / 65536 bytes
Disk identifier: 0xb97f94c1

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1              64    20971519    10485728   83  Linux
```

圧縮のアライメント ― ファイルシステム

パーティションに加えて、ファイルシステムも 8KB の境界にアライメントする必要があります。そのためには、ファイルシステムのブロック サイズが 8KB である必要があります。Oracle ASM を使用している場合、ASM が実行するエクステントの割り当てとストライピングにより、8KB のアライメントが保証されます。

他のファイルシステムを使用する場合は、ブロック サイズを 8KB に指定してください。ただしファイルシステムによっては指定できない場合もあります。

I/O スケジューラ

Linux カーネルでは、ブロック デバイスへの I/O のスケジューリングを低レベルで制御できます。デフォルト値は、Linux のディストリビューションによって大きく異なります。テストからは、通常は Deadline を使用すると最適な結果が得られることがわかっていますが、NOOP の方が多少よい結果になることもあります。パフォーマンスの違いはごくわずかですが、データベース構成から最大のパフォーマンスを引き出す必要がある場合は両方のオプションをテストしてください。多くの構成では CFQ がデフォルトですが、これは、データベース ワークロードのパフォーマンスに重大な問題を引き起こすことが実証されています。

I/O スケジューラの設定方法については、関連する Linux ベンダーのドキュメントを参照してください。

マルチパス

一部のお客様では、マルチパス デーモンがシステムで実行されていなかったために、ネットワークが停止した際にシステムがクラッシュしてしまうことがありました。Linux の最近のバージョンでは、OS とマルチパス デーモンのインストールの際に、OS がこの問題に対して脆弱なままに設定されることがあります。パッケージは正しくインストールされますが、リブート後に自動で起動するよう設定されません。

例えば RHEL5.5 の場合、マルチパス デーモンのデフォルトは次のようになっていることがあります。

```
[root@jfs0 iscsi]# chkconfig --list | grep multipath
multipathd      0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

これは、次のコマンドで修正できます。

```
[root@jfs0 iscsi]# chkconfig multipathd on
[root@jfs0 iscsi]# chkconfig --list | grep multipath
multipathd      0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

18.4 ASMlib のブロック サイズ

ASMlib は、オプションの ASM 管理ライブラリおよび関連するユーティリティです。ASMlib の第一の機能は、LUN や NFS ベースのファイルに、ASM リソースであるという人間が判読可能なラベルを付けることです。

ASMlib の最近のバージョンは、Logical Blocks Per Physical Block Exponent（LBPPBE）という LUN のパラメータを検出します。最近までこの値は Data ONTAP SCSI ターゲットでは報告されていませんでしたが、現在は、4KB のブロック サイズが推奨されることを示す値が返されます。これはブロック サイズの定義ではなく、LBPPBE を使用するアプリケーションに対して、所定のサイズの I/O は効率よく処理されるというヒントを与えるものです。ただし ASMlib は LBPPBE をブロック サイズとして解釈し、ASM デバイスが作成されるたびに ASM ヘッダーをスタンプします。

この処理は、さまざまな方法でアップグレードや移行に問題を引き起こします。すべては、同じ ASM ディスクグループに、ブロック サイズの異なる複数の ASMlib デバイスを混在させられないことが原因です。

例えば旧型のアレイでは、LBPPBE の値が 0 と報告されるか、まったく報告されません。ASMlib はこれを 512 バイトのブロック サイズと解釈します。新型のアレイは 4KB のブロック サイズと解釈されると考えられますが、512 バイトと 4KB のデバイスを同じ ASM ディスクグループに混在させることはできません。混在させると、2 台のアレイの LUN を使用して ASM ディスクグループのサイズを拡張したり、ASM を移行ツールとして活用することができなくなります。別のケースとしては、RMAN では、512 バイトのブロック サイズの ASM ディスクグループと、4KB のブロック サイズの ASM ディスク グループ間でファイルを複製しようとしても、許可されない可能性があります。

推奨される解決策は、ASMlib にパッチを適用することです。Oracle のバグ ID は 13999609 で、パッチは、oracleasm-support-2.1.8-1 以上で提供されています。このパッチを適用すると、`/etc/sysconfig/oracleasm` 構成ファイルのパラメータ `ORACLEASM_USE_LOGICAL_BLOCK_SIZE` を `FALSE` に設定できるようになります。このオプションを設定すると、ASMlib は LBPPBE パラメータを使用できなくなるため、新しいアレイの LUN が 512 バイトのブロック デバイスとして認識されるようになります。

注： このオプションを使用しても、以前に ASMlib でスタンプされた LUN のブロック サイズは変わりません。例えば、ブロック サイズが 512 バイトの ASM ディスクグループを、ブロック サイズを 4KB と報告する新しいストレージ システムに移行しなければならない場合、新しい LUN が ASMlib でスタンプされる前にオプション `ORACLEASM_USE_LOGICAL_BLOCK_SIZE` を設定する必要があります。

2 つ目のオプションは、Data ONTAP 7-Mode で使用できます。現行のすべてのリリースでは、コマンド `lun set report-physical-size <path> disable` で LBPPBE 設定を無効にし、各 LUN を 512 バイトのブロックの動作にリセットできます。

最後に、ASMlib にパッチを適用できない場合は、ASMlib の使用を中止します。この変更にはシステムの停止が伴い、ASM ディスクのスタンプを解除して、`asm_diskstring` パラメータが正しく設定されていることを確認する必要があります。ただし、データの移行は必要ありません。

18.5 Linux ext3 および ext4 のマウント オプション

デフォルトのマウント オプションを使用することを推奨します。

19 Microsoft Windows

このセクションでは、Microsoft Windows OS に固有の構成について解説します。

19.1 NFS

Oracle は、Direct NFS Client に Microsoft Windows をサポートしています。このことは、複数の環境のファイルの表示、ボリュームの動的なサイズ変更、安価な IP プロトコルの活用など、NFS を管理する上でのメリットをもたらします。DNFS を使用する Microsoft Windows にデータベースをインストールして設定する方法については、Oracle の公式ドキュメントを参照してください。特別なベストプラクティスはありません。

19.2 SAN

圧縮効率を最適化するには、NTFS ファイルシステムの割り当て単位が必ず 8,192 バイト以上になるようにしてください。一般的なデフォルトの 4,096 バイトの割り当て単位を使用すると、効率性が低下します。

20 Solaris

このセクションでは、Solaris OS に固有の構成について解説します。

20.1 Solaris NFSv3 のマウント オプション

表 8 に、Solaris NFSv3 のマウント オプションをまとめました。

表 8）Solaris NFSv3 のマウント オプション ― シングル インスタンス

ファイル タイプ	マウント オプション
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr, llock,suid
ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid

llock を使用すると、ストレージ システムでロックを取得およびリリースする際のレイテンシが取り除かれるため、パフォーマンスが劇的に向上することが実証されています。多数のサーバが同じファイルシステムをマウントするよう構成され、Oracle がそのデータベースをマウントするよう構成されている環境では、このオプションの使用に注意が必要です。このような構成はきわめて稀ですが、一部のお客様では実際に使用されています。1 つのインスタンスが誤って 2 回開始されると、Oracle は外部サーバのロック ファイルを検出できないため、データが破損します。NFS ロックには、NFS バージョン 3 同様に保護機能はなく、単なる推奨事項です。

llock と forcedirectio パラメータは同時に使用できないので、directio が使用されるように init.ora ファイルに filesystemio_options=setall を指定することが重要です。このパラメータを指定しないと、ホスト OS のバッファ キャッシュが使用され、パフォーマンスに悪影響が及びます。

表 9）Solaris NFSv3 のマウント オプション ― RAC

ファイル タイプ	マウント オプション
ADR_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,noac
制御ファイル データファイル Redo ログ	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,nointr, noac,forcedirectio
CRS / Voting	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536, nointr,noac,forcedirectio
専用 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536,suid
共有 ORACLE_HOME	rw,bg,hard,vers=3,proto=tcp,timeo=600,rsize=65536,wsize=65536, nointr,noac,suid

シングルインスタンスと RAC マウント オプションの主な違いは、RAC にはマウント オプションに noac と forcedirectio がある点です。このオプションを使用するとホスト OS のキャッシングが無効になるため、データの状態について、RAC クラスタ内のすべてのインスタンスが一貫した情報を認識できるようになります。init.ora パラメータ filesystemio_options=setall を使用した場合も同様にホストのキャッシングが無効になりますが、noac と forcedirectio は引き続き使用する必要があります。

actimeo=0 は、共有 ORACLE_HOME 環境で、Oracle パスワード ファイルや spfile などのファイルの整合性を維持するために必要です。RAC クラスタの各インスタンスに専用の ORACLE_HOME がある場合、このパラメータは必要ありません。

20.2 Solaris UFS のマウント オプション

ネットアップは、ロギング マウント オプションの使用を強く推奨しています。このオプションを使用すると、Solaris ホストがクラッシュしたり FC 接続が中断した場合にデータの整合性が維持されます。ロギング マウント オプションは、Snapshot バックアップの使用も維持します。

20.3 Solaris ZFS

優れたパフォーマンスを実現するためには、Solaris ZFS をインストールして注意深く設定してください。

カーネル

信頼できる ZFS パフォーマンスを実現するには、LUN のアライメント問題に対するパッチを Solaris カーネルに適用する必要があります。この修正は、Solaris 10 のパッチ 147440-19 と Solaris 11 の SRU 10.5 に含まれています。ZFS では Solaris 10 以降のみを使用してください。

LUN 構成

LUN を構成するには、次の手順を実行します。

1. タイプが `solaris` の LUN を作成します。
2. IMT で指定されているホスト ユーティリティ キット（HUK）をインストールします。
3. HUK の指示に忠実に従います。基本の手順についてはこのセクションで大まかに説明しますが、詳しい手順は最新のドキュメントを参照してください。
   a. `host_config` ユーティリティを実行して `sd.conf/sdd.conf` ファイルを更新します。SCSI ドライブが Data ONTAP の LUN を正しく検出できるようになります。
   b. `host_config` ユーティリティの指示に従い、マルチパス I/O（MPIO）を有効にします。
   c. リブートします。この手順は、変更をシステム全体に認識させるために必要です。
4. LUN をパーティショニングし、適切にアライメントされていることを確認します。アライメントを直接テストして確認する方法については、「付録 3：WAFL アライメントの検証」を参照してください。

zpool

zpool を作成する場合は、その前に「LUN 構成」セクションの手順を実行してください。この手順を正しく完了しないと、I/O のアライメントが原因で、深刻なパフォーマンス低下が起きる恐れがあります。Data ONTAP のパフォーマンスを最適化するためには、I/O がディスクの 4K の境界にアライメントされている必要があります。zpool 上に作成されたファイルシステムは、パラメータ `ashift` で制御される実効ブロック サイズを使用します。これは、コマンド `zdb -C` を実行することで確認できます。

`ashift` のデフォルト値は 9 で、これは 2^9、つまり 512 バイトを意味します。パフォーマンスを最適化するには、`ashift` の値が 12（2^12=4K）である必要があります。この値は zpool の作成時に設定され、変更はできません。つまり、`ashift` が 12 以外の zpool のデータは、新しく作成した zpool にデータを複製して移行する必要があります。

zpool を作成したら、次に進む前に `ashift` の値を確認してください。値が 12 以外の場合は LUN が正しく検出されていないので、作成した zpool を削除し、関連するホスト ユーティリティのドキュメントに記載された手順をすべて正しく実行したかどうかを確認し、再度 zpool を作成します。

zpool と Solaris LDOM

Solaris LDOM には、I/O のアライメントが正しくなければならないもう 1 つの要件があります。LUN が 4K デバイスとして正しく検出されても、LDOM 上の仮想 vdsk デバイスには I/O ドメインの設定が引き継がれません。その LUN を基盤にした vdsk は、デフォルトの 512 バイト ブロックに戻ります。

追加の構成ファイルが必要です。まず個々の LDOM に Oracle のバグ 15824910 のパッチを適用し、設定オプションを追加できるようにします。このパッチは、現在使用されているすべての Solaris バージョンに組み込み済みです。LDOM にパッチを適用すると、適切にアライメントされた新しい LUN を設定できるようになります。手順は次のとおりです。

1. まず、新しい zpool で使用する LUN を指定します。この例では c2d1 デバイスです。

```
root@LDOM1 # echo | format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
  0. c2d0 <Unknown-Unknown-0001-100.00GB>
     /virtual-devices@100/channel-devices@200/disk@0
  1. c2d1 <SUN-ZFS Storage 7330-1.0 cyl 1623 alt 2 hd 254 sec 254>
     /virtual-devices@100/channel-devices@200/disk@1
```

2. ZFS pool に使用するデバイスの vdc インスタンスを取得します。

```
root@LDOM1 # cat /etc/path_to_inst
#
# Caution! This file contains critical kernel state
#
"/fcoe" 0 "fcoe"
"/iscsi" 0 "iscsi"
"/pseudo" 0 "pseudo"
"/scsi_vhci" 0 "scsi_vhci"
"/options" 0 "options"
"/virtual-devices@100" 0 "vnex"
"/virtual-devices@100/channel-devices@200" 0 "cnex"
"/virtual-devices@100/channel-devices@200/disk@0" 0 "vdc"
"/virtual-devices@100/channel-devices@200/pciv-communication@0" 0 "vpci"
"/virtual-devices@100/channel-devices@200/network@0" 0 "vnet"
"/virtual-devices@100/channel-devices@200/network@1" 1 "vnet"
"/virtual-devices@100/channel-devices@200/network@2" 2 "vnet"
"/virtual-devices@100/channel-devices@200/network@3" 3 "vnet"
"/virtual-devices@100/channel-devices@200/disk@1" 1 "vdc" << We want this one
```

3. /platform/sun4v/kernel/drv/vdc.conf を編集します。

```
block-size-list="1:4096";
```

デバイス インスタンス 1 には、4,096 のブロック サイズが割り当てられます。

もう 1 つの例として、vdsk のインスタンス 1～6 を 4K のブロック サイズに構成する必要があり、`/etc/path_to_inst` が以下のように設定されているとします。

```
"/virtual-devices@100/channel-devices@200/disk@1" 1 "vdc"
"/virtual-devices@100/channel-devices@200/disk@2" 2 "vdc"
"/virtual-devices@100/channel-devices@200/disk@3" 3 "vdc"
"/virtual-devices@100/channel-devices@200/disk@4" 4 "vdc"
"/virtual-devices@100/channel-devices@200/disk@5" 5 "vdc"
"/virtual-devices@100/channel-devices@200/disk@6" 6 "vdc"
```

4. 編集後の vdc.conf ファイルは次のようになります。

```
block-size-list="1:8192","2:8192","3:8192","4:8192","5:8192","6:8192";
```

注意

vdc.conf を設定および vdsk を作成した**あと**、LDOM をリブートする必要があります。この手順は省略しないでください。ブロック サイズの変更が有効になるのはリブートの**あと**です。zpool の設定を続行し、前述のように、`ashift` が正しく 12 に設定されていることを確認してください。

ZIL

一般に、ZFS Intent Log（ZIL）を別のデバイスに配置する理由はありません。このログはメインのプールとスペースを共有できます。ZIL を別に使用する主な理由は、最新のストレージ アレイで書き込みキャッシュ機能を備えていない物理ドライブを使用する場合です。

logbias

Oracle データをホストする ZFS ファイルシステムでは、`logbias` パラメータを設定します。

```
zfs set logbias=throughput <filesystem>
```

このパラメータを使用すると、全体的な書き込み量が削減されます。デフォルトでは、書き込みデータはまず ZIL にコミットされ、その後メインのストレージ プールにコミットされます。この手法では、最も低レイテンシのメディア上の単一の I/O トランザクションでデータがコミットされるため、SSD ベースの ZIL デバイスおよびメインのストレージ プール用の回転式メディアなど、シンプルなディスク構成に適しています。

独自のキャッシュ機能を備えた最新のストレージ アレイを使用する場合、この手法は通常必要ありません。例えばレイテンシの影響を受けやすいランダム ライトが大量に発生するワークロードなど、ごくまれな状況においては、単一のトランザクションで書き込みをログにコミットした方が望ましい場合があります。ログに書き込まれたデータは最終的にメインのストレージ プールに書き込まれるため、書き込み処理が二重に発生し、ライト アンプリフィケーションという状況を招きます。

ダイレクト I/O

Oracle 製品を含め、多くのアプリケーションでは、ダイレクト I/O を有効にすることでホストのバッファ キャッシュを迂回できます。ただし ZFS ファイルシステムでは想定する結果が得られず、ホストのバッファ キャッシュが迂回されても ZFS 自体がデータをキャッシュし続けます。この場合、I/O がストレージ システムに達しているかどうか、または I/O が OS にローカルにキャッシュされているかどうかの予測が難しく、fio や sio などのツールを使用してパフォーマンス テストを実行すると誤った結果になることがあります。また、こうした総合的なテストで、ZFS と他のファイルシステムのパフォーマンスを比較することもきわめて難しくなります。現実には、実際のユーザ ワークロードでファイルシステムのパフォーマンスにほとんど違いはありません。

複数の zpool

ZFS ベースのデータを、Snapshot をベースにバックアップ、リストア、クローニング、アーカイブする処理は、zpool レベルで実行する必要があり、通常は複数の zpool が必要です。zpool は LVM ディスク プールに似ており、同じルールで構成する必要があります。例えばデータベースの場合、データファイルを `zpool1` に、アーカイブ ログ、制御ファイル、Redo ログを `zpool2` に配置するレイアウトが最も適しています。このレイアウトでは、データベースがホット バックアップ モードに設定される標準のホット バックアップを実行し、続けて `zpool1` の Snapshot コピーを実行できます。次にデータベースのホット バックアップ モードが解除され、ログがアーカイブされ、`zpool2` の Snapshot コピーが作成されます。リストア処理では、ZFS ファイルシステムをアンマウントして zpool を完全にオフラインにし、続けて SnapRestore のリストア処理を実行します。その後、zpool を再度オンラインにしてデータベースをリカバリできます。

filesystemio_options

Oracle パラメータ `filesystemio_options` は、ZFS では異なる動作をします。`setall` または `directio` を使用した場合、書き込み処理は同期されて OS のバッファ キャッシュを迂回しますが、読み取りは ZFS によってバッファされます。この場合、I/O が ZFS キャッシュによって代わりに処理されることがあるため、ストレージのレイテンシと総 I/O が表示される値よりも低くなり、パフォーマンス分析が困難になります。

21　まとめ

本ドキュメントの冒頭で述べたように、Oracle 環境のストレージ構成は、環境による違いがあまりにも大きいため、本当の意味でのベストプラクティスはほとんどありません。データベース プロジェクト 1 つ取ってみても、ミッションクリティカルなデータベース 1 つで構成されていることもあれば、5,000 のレガシー データベースで構成されていることもあり、規模も数ギガバイトから数百テラバイトと様々です。クラスタウェアや仮想化などのオプションを使用すると、さらに多様になります。

言いかえれば、設計上の考慮事項やストレージの実装計画の際に考慮すべき問題点がいくつもあるということです。最適なソリューションは、環境の技術的な詳細と、プロジェクトの背景にあるビジネス要件の 2 つによって決まります。ネットアップとパートナーのプロフェッショナル サービス エキスパートは、複雑なプロジェクトのサポートを提供しています。プロジェクト中はサポートが不要な場合も、ネットアップが初めてのお客様には、プロフェッショナル サービスを利用して高度なアプローチの設計に役立てていただくことを強く推奨します。

付録 1：ファイルシステムのレイアウト

以下のレイアウトはベストプラクティスではありません。データベース プロジェクトは、そのサイズ、数、コスト、バックアップウィンドウ、保持要件、セキュリティのニーズ、ディザスタ リカバリ計画の点で要件が様々に異なるため、最適なファイルシステム レイアウトを紹介することは不可能です。本ドキュメントでは、ファイルシステムのレイアウト設計の基本原則を確認します。以下に紹介する例のいくつかは、組み合わせることでより効果を発揮します。

お客様の環境に関係がないと思われる例も含め、すべてを確認してください。すべての例を参照することで、データ保護、リカバリ、パフォーマンスの概念と、それらがファイルシステムのレイアウトにどう関係しているかを理解することができます。

目標復旧時点と目標復旧時間

ファイルシステム レイアウトで考慮すべき最も重要な要素は、目標復旧時点（RPO）と目標復旧時間（RTO）です。RPO はどの程度リカバリ可能か、つまりリカバリが必要な状況で許容されるデータ損失の最大量です。失ってもよいデータが 15 分前までなら、RPO は 15 分になります。RTO は、データベース サーバが停止したあとに許容される最大停止時間です。リカバリ時間には様々なタスクが含まれると考えられるため、RPO は、RPO ほどは明確に定義されません。

例えば、単純にデータベースをリカバリすると決定してから実際にリカバリするまでの時間の場合もあれば、データベース サーバに障害が発生した瞬間からの時間の場合もあります。あるいは、データベースをオンラインに戻すのに必要な時間だけが含まれる場合もあれば、キャッシュをウォームアップしてアプリケーションをリスタートするのに必要な時間が含まれる場合もあります。データベースだけで考えれば、RPO はアーカイブ ログのバックアップ頻度に左右され、RTO はデータファイルのバックアップ頻度に左右されます。

RPO と RTO には、この他にも考慮が必要な要素があります。一般に RPO と RTO には、通常運用と災害時運用があります。例えば、通常運用の RPO はゼロで、通常のリカバリではデータ損失が絶対に発生せず、Redo ログのコピーが常に 1 つ存在する必要があります。災害時の RPO と RTO はここまで厳しくありません。

災害時の RPO は、お客様のインフラ面での制約やビジネス ニーズによって様々です。例えば、MetroCluster は RPO=0 を提供する製品ですが、多くのお客様がこれを導入して複数のサイト間で同期ミラーリングを実現しています。一方で、サイト同士が数千キロ離れているお客様もいます。この場合、同期ミラーリングは不可能で、RPO は使用可能な帯域幅によって決まります。お客様によっては 15 分という短い RPO を実現している場合もありますが、一般的な RPO は 1～2 時間です。

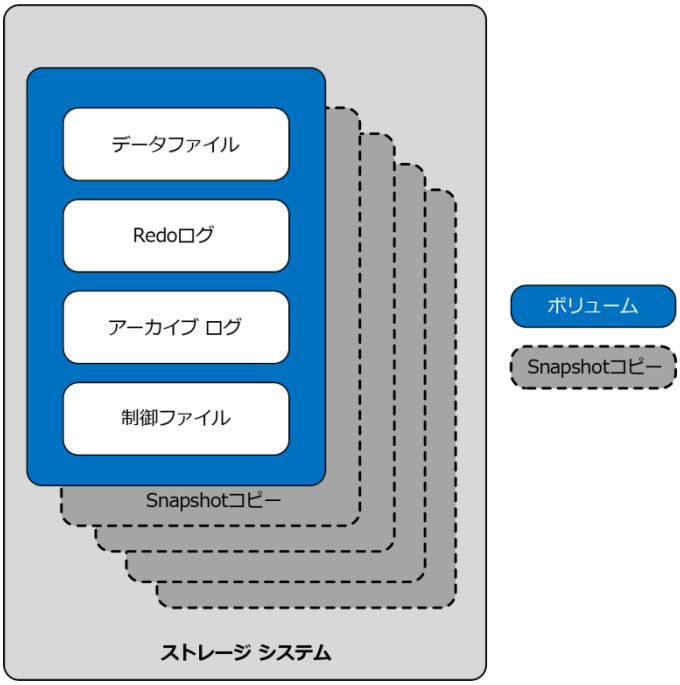
災害時の RTO には、「リカバリ」という言葉の定義が必要です。多くの場合、アプリケーション サービスやホスト名、さらには IP アドレスを別のサイトに移行するためには多くの社員の協力が必要です。災害の発生を発表するかどうかの判断に、数時間かかる場合もあります。これらすべてがサービスの RTO に関係し、一般にデータベースの RTO はディザスタ リカバリ プロセスのごく一部に過ぎません。

要するに、RPO と RTO の定義では、通常の RPO および RTO と、ディザスタ リカバリの RPO および RTO とを区別し、さらにリカバリの要件を定義する必要があります。RPO と RTO のニーズはストレージ設計の最も重要な部分を左右するため、ストレージの設計に着手する前に必ずこの 2 つを定義し、理解しておく必要があります。

小規模のシングルインスタンス データベース

もっともシンプルなファイルシステム レイアウトは、1 つのボリュームにデータベース全体を格納するレイアウトです（図 1 を参照）。

図 1）小規模のシングルインスタンス データベース



このレイアウトは、Oracle のバックアップとリカバリでサポートされています。詳細は、Oracle のドキュメント ID 221779.1 を参照してください。

この手順の鍵は、従属書き込みの順序を保持することです。Oracle のほとんどの書き込み処理は重要でなく、ファイルシステムや OS の非同期 I/O デバイスによって順序を変更することが可能です。ただし一部の書き込み処理はデータベースの正しい運用に重要なため、別の書き込みが実行される前にストレージ システムから確認応答を受け取る必要があります。つまり Oracle では、確認応答のあった前の書き込みが正常に完了したかどうかで、次の書き込みが実行されるわけではないということです。

NetApp Snapshot コピーでは、Snapshot コピーの作成時にストレージ システムから確認応答を受け取っている書き込みも、Snapshot コピーに含まれるようになっています。これにより、1 つ以上のボリュームやストレージ システムにわたって、従属書き込み順序に準拠した Snapshot コピーを作成できます。データベースの場合、データベースの Snapshot コピーを作成する際に、データベースのすべてのファイルが従属書き込み順序で含まれるため、リカバリ可能な一連の Snapshot コピーが実現します。このプロセスは電源障害時をシミュレーションしたものです。

また、あらゆるベンダーの同期レプリケーション ソリューションの基になっている原理でもあります。プライマリ サイトが失われた場合、セカンダリ サイトにはデータベースのコピーが従属書き込み順序で保管されています。データベースを起動すると、Redo ログが再生され、データベースを開くことができます。

Snapshot コピーを活用する最も簡単な手法は、データベース全体を 1 つのボリュームに配置することです。このバックアップ方法は、バックアップ時点までリカバリ可能な小規模なデータベースに最適です。Snapshot コピーのリストアを実行すると、その Snapshot コピーが作成されたときの状態にデータベースが戻ります。通常、Snapshot コピーは 6～24 時間おきに作成します。作成頻度を増やすと、Snapshot コピーの数が増えて管理が難しくなったり、ボリュームあたり 255 個という Snapshot コピーの上限に達してしまう恐れがあります。

注： この方法が適しているのは、バックアップ時点へのリカバリが求められるデータベースで、バックアップの間隔が 6 時間以上です。バックアップの頻度をこれ以上に増やすと、ボリュームの Snapshot コピーがすぐに最大値に達してしまいます。

「小規模」の定義は多くの要因によって変わります。この方法は、最大ボリューム サイズが 128TB までのデータベースで有効です。ただし、通常このサイズのデータベースでは、RPO と RTO の要件を満たすために各種ファイルシステムの分離が必要になります。

注： 一般的な目安として、サイズが 1TB 未満のデータベースに最も適しています。

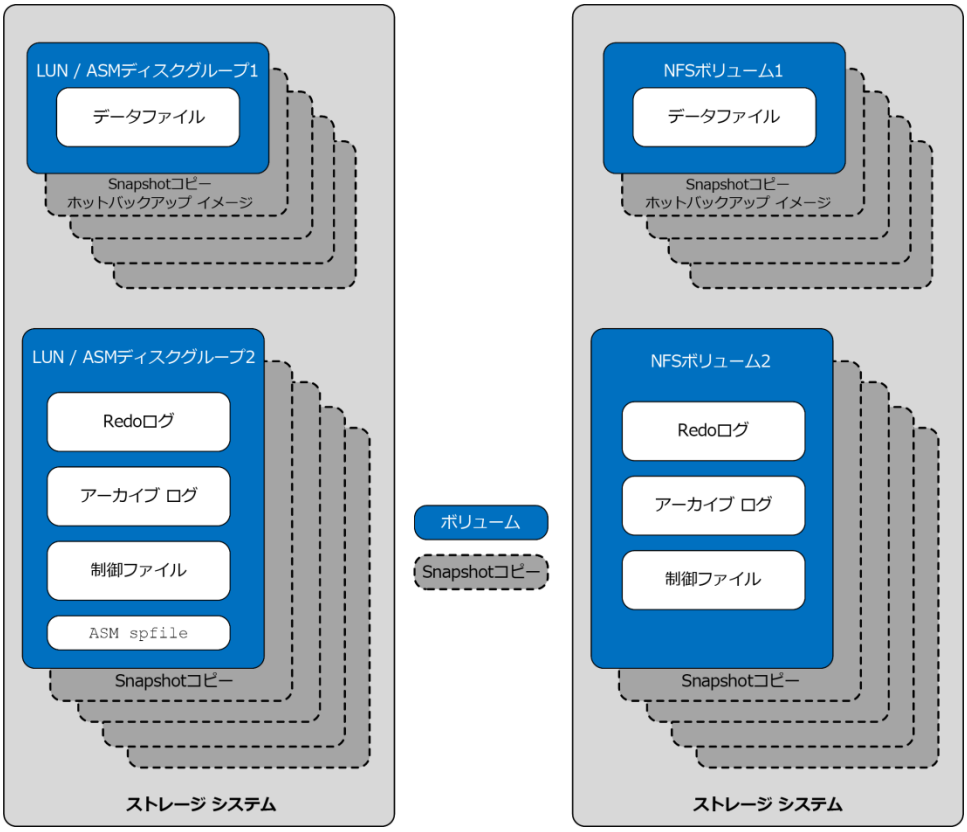
RPO と RTO

このケースでの RPO は Snapshot コピーの作成頻度によって異なります。リカバリ手順は、主にデータベースをシャット ダウンして、1 つの Snapshot コピーから SnapRestore でリストア処理を実行するだけのため、RTO はおよそ 1 分です。

Snapshot ベースのホット バックアップ

図 2 に、ホット バックアップに使用されるシンプルなレイアウトを示します。左の図は、ASM を使用した LUN の例、右の図は、同じ原則を NFS に適用した例を示しています。

図 2）シンプルなホット バックアップ



このレイアウトには 2 つのボリュームが含まれています。ASM の例では、各ボリュームに特定の ASM ディスクグループの LUN が含まれています。1 つ目のディスクグループは、データファイルのディスクグループです。柔軟性を最大限に高めるために、データファイルだけが含まれています。2 つ目のディスクグループには、それ以外のファイルがすべて含まれています。

注： 最初に作成された ASM ディスクグループには、その ASM に使用する spfile が含まれます。このグループがデータファイルとスペースを共有するとリカバリ オプションの妨げとなるため、共有は避けてください。データファイル以外のディスクグループを最初に作成するか、データファイルのディスクグループ以外の場所に spfile を移動してください。

NFS の例は、ASM の例とほぼ同じです。これは、ユニファイド ストレージ システムとしての clustered Data ONTAP の柔軟性を表しています。どの機能もプロトコルに依存しません。

Snapshot の作成

ホットバックアップ Snapshot コピーは、SnapManager for Oracle（SMO）、SnapManager for SAP、NetApp SnapProtect® ソフトウェア、Snap Creator、または簡単なスクリプトを使用して作成できます。データファイルがホットバックアップ モードにある間に Snapshot コピーを作成することが第一の目標です。このホットバックアップ イメージが、データベース リカバリのベースになります。

リカバリ手順 — ASM

この手順は、手動または SMO などのアプリケーションを使用して実行できます。通常のケースでは、以下のリカバリ手順を実行してください。

1. データベースをシャットダウンします。
2. ASM ディスクグループ 1 をディスマウントします。ASM 全体をシャット ダウンする方が簡単かもしれません。
3. データファイルのディスクグループをリカバリします。
4. ASM ディスクグループを再マウントするか、ディスクグループをシャット ダウンした場合は ASM を起動します。
5. 使用可能なアーカイブ ログをすべて再生します。
6. Redo ログをすべて再生します。

リカバリ手順 — NFS

この手順は、手動または SMO などのアプリケーションを使用して実行できます。通常のケースでは、以下のリカバリ手順を実行してください。

1. データベースをシャットダウンします。
2. データファイルのボリュームをリカバリします。
3. 使用可能なアーカイブ ログをすべて再生します。
4. Redo ログをすべて再生します。

NFS ベースのファイルは、自動ツールやストレージ管理者のサポートなしでエンドユーザが `.snapshot` ディレクトリからリカバリすることも可能です。

RPO と RTO

この構成では一般に RPO はゼロです。データベースは破損する可能性がありますが、アーカイブ ログと Redo ログは引き続き使用可能です。例外は、ASM ディスクグループが完全に破壊された場合、または NFS ファイルが完全に削除された場合です。どちらの場合も、アーカイブ ログを格納しているボリュームをリカバリしなければなりません。したがって極端なケースでは、アーカイブ ログ ボリュームの Snapshot コピーの頻度によって RPO が決まります。

RTO は、データファイルの Snapshot コピーの頻度によって決まります。このモデルを使用するほとんどの環境で、バックアップの実行頻度は 1 日に 1 回です。最悪のシナリオは、前回のバックアップから 23 時間 59 分後にデータベースが破損した場合で、リカバリでは、前回のバックアップをリストアし、23 時間 59 分間アーカイブ ログを再生することが必要になります。一般に、ユーザはこの状況を許容可能と判断します。ただし、ログの再生時間が RTO の要件を超える場合は、データファイルのホット バックアップをより頻繁に実行し、アーカイブ ログの再生に必要な時間を短縮する必要があります。

コントローラのストライピング

最大限のパフォーマンスが求められるデータベースは、コントローラにまたがってストライピングすることができます。ほとんどのデータベースに求められるパフォーマンスは 1 台のコントローラで実現されるため、通常この操作は不要ですが、重要なワークロードにはストライピングが必要になることがあります。ストライピングを実施することで、各コントローラの CPU、キャッシュ、フラッシュ、ディスクといったリソースがすべてかつ均等に使用されます。

SAN

図 3 に、最もシンプルな、Oracle ASM でコントローラをストライピングする方法を示します。

図 3）ASM を使用したコントローラのストライピング

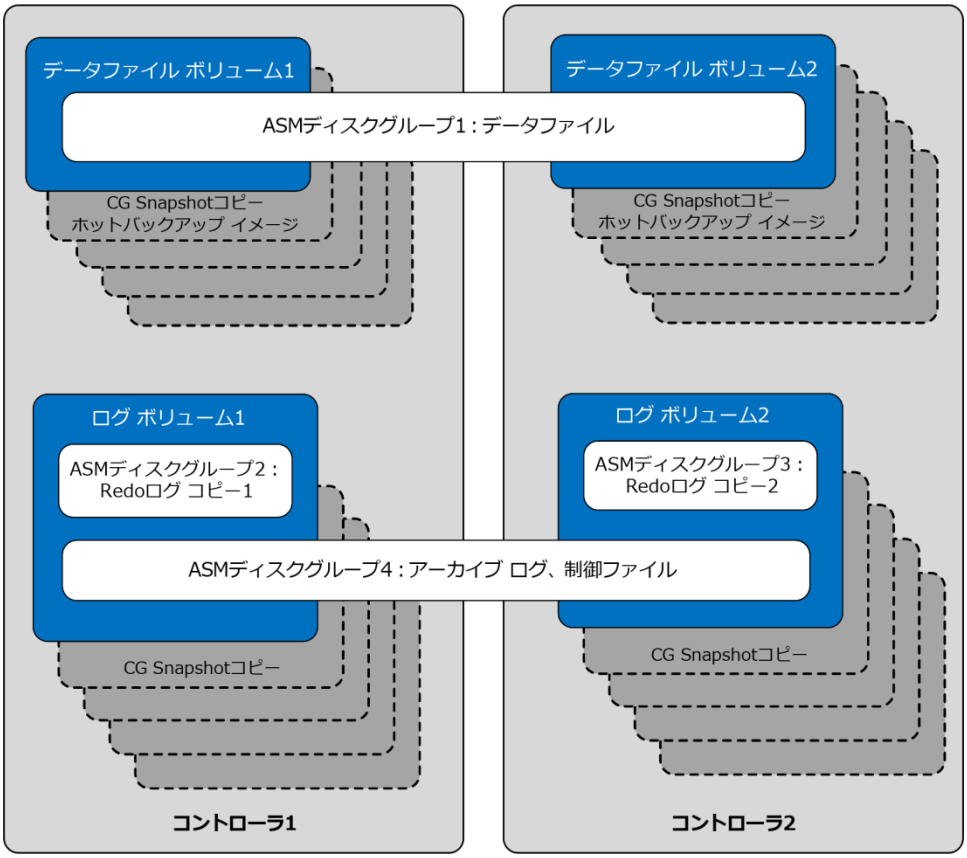


図 3 のモデルは前の 2 つの図とほぼ同じですが、この例では ASM を使用してコントローラ間でデータを分散しています。他の論理ボリューム マネージャ（LVM)も使用できますが、最も広く使用されているのは ASM です。ASM の特長は、ストライピングのきめ細かさにあります。他の LVM 製品を使用した場合、論理ボリュームのベースをストライプではなくエクステントにする必要があります。例えば AIX LVM を使用する場合、可能なかぎり小さいサイズのエクステントを使用して論理ボリュームを構築してください。論理ボリュームをストライプ セットとして構成することはしないでください。この方法は主にシーケンシャル I/O 向けに設計されたもので、アプリケーションの I/O サイズとストライプ サイズを慎重に一致させる必要があります。この構成は Oracle には推奨されません。

Oracle の場合、重要なランダム I/O をできるだけ多数の LUN に確実に分散することが求められます。分散型エクステント ポリシーを使用すると、個々のデータファイルがファイルシステムに書き込まれる際に分割され、それぞれが別々の LUN の別々のエクステントに配置されます。その結果、ランダム I/O の実行時に I/O が均等に分散されます。

図 3 では、Redo ログがコントローラ間でミラーリングされるように明示的に構成されています。複数のコントローラにまたがる 1 つの ASM ディスクグループに、Redo ログをまとめるという方法もあります。ただしこの方法では、Redo ログの複数の書き込みが一度に同じコントローラに対して実行される可能性があります。Redo ログのコピーをそれぞれ別のコントローラに分離しておけば、最小限の競合で最大限のパフォーマンスを実現できます。

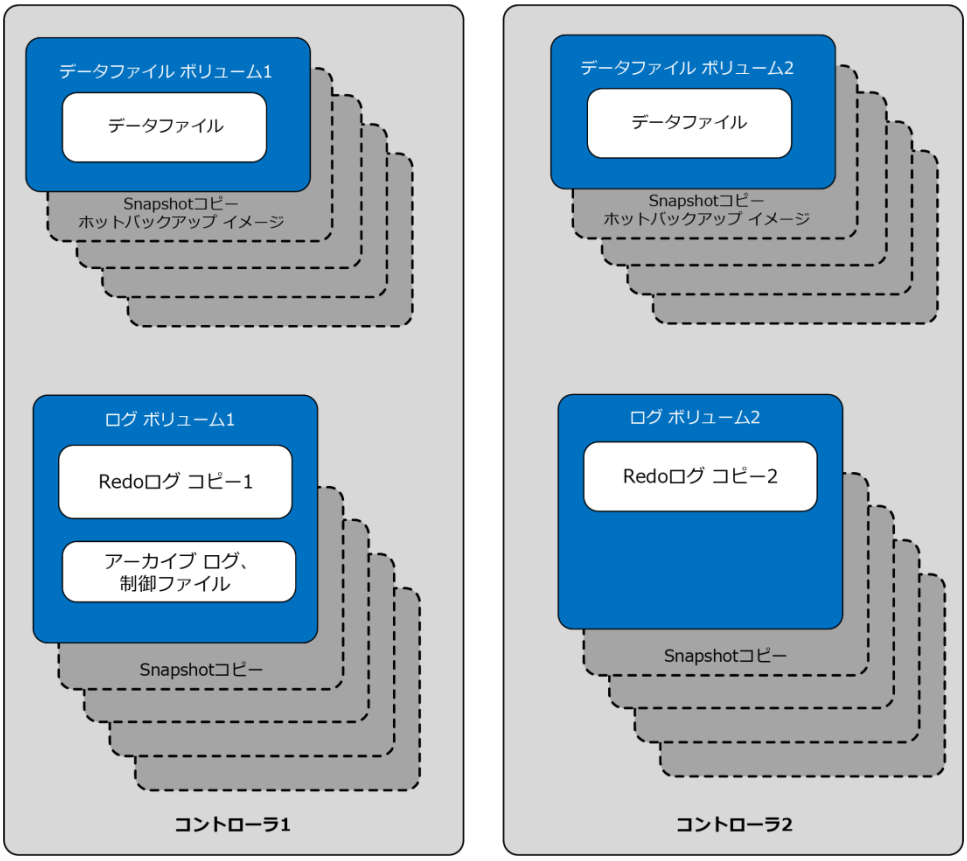
このバックアップ方法には、CG Snapshot コピーが必要です。ASM ディスクグループをコントローラ間で分散する場合、Snapshot コピーに従属書き込み順序が維持されている必要があります。コントローラ上で直接スケジュールされている Snapshot コピーの場合、従属書き込み順序を維持することはできません。Snap Creator および SnapManager for Oracle では CG Snapshot コピーを作成でき、さらにその処理をスクリプト化できます。

NFS

コントローラをストライピングする方法は NFS にも使用できますが、NFS ファイルの分散は手動で実行する必要があります。アーカイブ ログ ボリュームは分散できませんが、このボリュームが処理するのはアーカイブ ログのシーケンシャル ライト アクティビティのみです。一般にこの種の I/O は、Redo ロギングに比べて優先度が低く、レイテンシの影響も受けません。そのため、データファイルのランダム I/O アクティビティのような負荷がストレージ システムにかかることはありません。

対象となるファイルは常に 1 台のコントローラのみに配置されるので、CG Snapshot コピーを使用する必要はありません。データファイル ボリュームの Snapshot は、データベースがホットバックアップ モードにある間に順次作成可能です（図 4 を参照）。

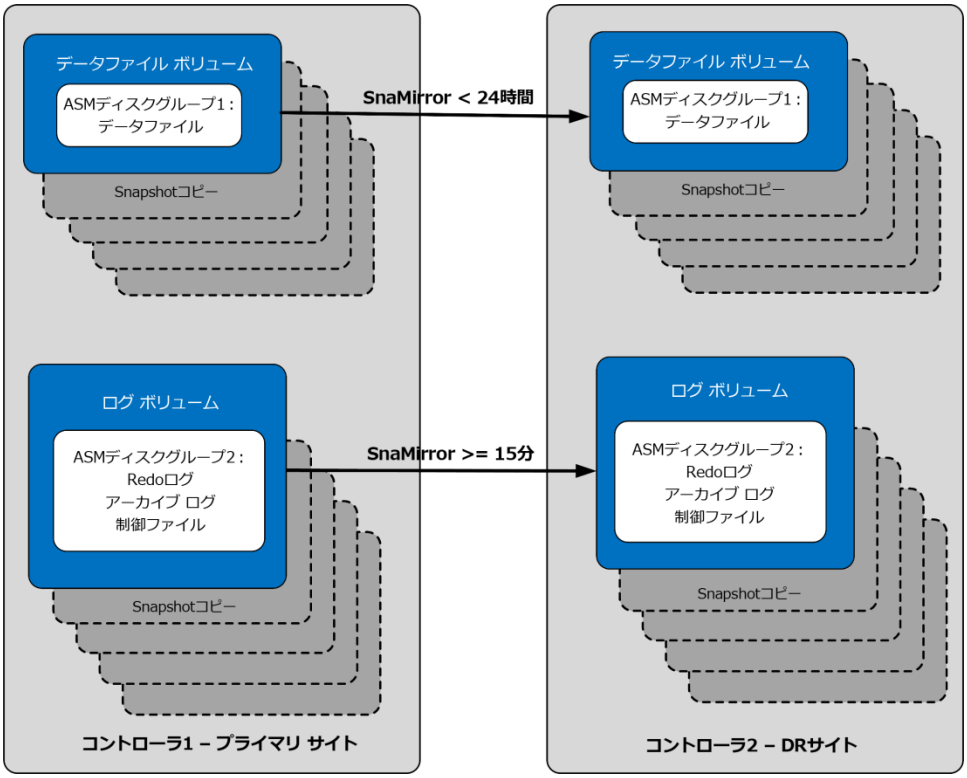
図 4）NFS を使用したコントローラのストライピング



SnapMirror ベースのディザスタ リカバリ

図 5 は、Snapshot ベースのシンプルなディザスタ リカバリに SnapMirror を活用する方法を示しています。この例では ASM を使用していますが、他のファイルシステム オプションにも同じ手順を適用できます。

図 5）SnapMirror ベースのディザスタ リカバリ



データファイルのバックアップ ― ホット バックアップ

大半のデータベース管理者は、リカバリの最初の手順としてホット バックアップを使用しようとしますが、これは必ずしも必須ではありません。詳細は次のセクションを参照してください。

ホット バックアップ Snapshot コピーの作成プロセスはいたってシンプルで、たとえば、cron などのスケジューラを使用して 23 時 58 分にデータベースをホット バックアップ モードに切り替え、Data ONTAP への直接 Snapshot コピーのスケジュールを 0 時 0 分に設定し、0 時 2 分にデータベースをホット バックアップ モードから通常モードに戻すだけです。

この手法はアプリケーション ソフトウェアを必要としません。また、拡張性にきわめて優れており、数千ものデータベースを保護できることが実際の環境で証明されています。監視が必要な場合は、ボリュームの Snapshot コピーのタイムラグが 25 時間を超過した場合に（前回の Snapshot コピー作成から 25 時間以上経過した場合に）アラートが送信されるよう、NetApp OnCommand®ソフトウェアを設定するだけです。アラートは、Snapshot コピーを作成できなかったこと、つまりバックアップに失敗したことを示しています。

データファイルのバックアップ ― Snapshot 最適化

Oracle 12c を使用している場合、Snapshot コピーの作成プロセスはさらに簡単で、Data ONTAP 上で直接 Snapshot コピーのスケジュールを設定できます。データベース管理者にとってはホット バックアップを使用したリカバリ手順の方がなじみがありますが、Oracle 12c では不要です。

Oracle 10g および 11g では、データベースがホットバックアップ モードにないときに作成された Snapshot コピーを使用できましたが、データベースの一貫性を保つために、リカバリ時に手動での操作が別に必要でした。そのため、この方法が Oracle 10g や 11g で用いられることはほとんどありませんでした。Oracle 12c ではこの手順が自動化されています。

ホット バックアップの主なメリットとして、データファイルがホット バックアップ モードに切り替わった時点がデータファイルに記録され、ホット バックアップ モードが終了した時点がアーカイブ ログに記録されることがあります。データファイルの一貫性を維持するためには、この 2 つの時点の間のデータを再生する必要があります。

Oracle 12c では、新しい機能として、Snapshot コピーが作成された日時を指定できるようになりました。次に例を示します。

```
RECOVER DATABASE UNTIL TIME '10/13/2015 15:00:00' SNAPSHOT TIME '10/13/2015 12:00:00';
```

`snapshot time` 引数は、2015 年 10 月 13 日 12:00 以降に作成された Snapshot コピーをリカバリに使用することを指定しています。これにより、sqlplus は、再生する正しいアーカイブ ログを選択できます。プロセス全体は、ホットバックアップ リカバリと基本的に同じです。

詳細については、Oracle 12c の各種ドキュメントで、「Recovery Using Storage Snapshot Optimization」のトピックを参照してください。また、Oracle のサードパーティ スナップショットのサポートについて、Oracle のドキュメント ID Doc ID 604683.1 の「Data ONTAP とサードパーティのスナップショット」セクションも併せて参照してください。

ログのバックアップ

アーカイブ、Redo、制御ファイルのボリュームは、コントローラでの定期的な Snapshot コピーなど、任意の方法でバックアップできます。

レプリケーション間隔

データファイル ボリュームのレプリケーション間隔は RTO を左右します。一般的な間隔は 24 時間で、最大で 24 時間分のアーカイブ ログを再生する必要があるということです。

ログ ボリュームのレプリケーション間隔は RPO を左右し、通常は使用可能な帯域幅によって制限されます。ボリュームの変更率が高い場合はより多くの帯域幅を必要とします。また、同時に実行可能な SnapMirror 更新の回数にも制限があります。ほとんどの環境では、最大 15 分おきの更新が可能です。

ディザスタ リカバリの手順 ― ホット バックアップ

データベースがホットバックアップ モードにある間に Snapshot コピーが作成された場合、リカバリ手順は次のようになります。

1. データファイル ボリュームとログ ボリュームのミラーを解除します。
2. データファイル ボリュームの SnapRestore リストア処理を実行し、ホットバックアップ モードで作成された最新の Snapshot コピーにリストアします。
3. アーカイブ ログを再生します。
4. Redo ログを再生します。

注： この手順は容易にスクリプト化できます。

ディザスタ リカバリ手順 ― Snapshot 最適化

この Snapshot 最適化の手法を使用すると、リカバリ手順を簡易化できます。

1. データファイル ボリュームとログ ボリュームのミラーを解除します。
2. 最新のデータファイル ボリュームの更新に Snapshot コピーが使用されています。この Snapshot コピーのタイムスタンプを特定します。このタイムスタンプは、clustered Data ONTAP で `snapshot show -time` を使用して取得できます。
3. sqlplus を使用してログを再生し、`SNAPSHOT TIME` を Snapshot コピーのタイムスタンプの数分あとに指定します。

注： `SNAPSHOT TIME` は正確に指定してください。そうしないと、デーベースが修復不可能な状態に破損する可能性があります。ストレージ システムとデータベース サーバのクロックが同期していることも重要です。

SnapManager for Oracle と Snap Creator のレイアウト

SMO でデータベースを管理する場合、唯一のベストプラクティスというものはありません。Data ONTAP は大幅に強化されており、この点に関する柔軟性が向上しました。

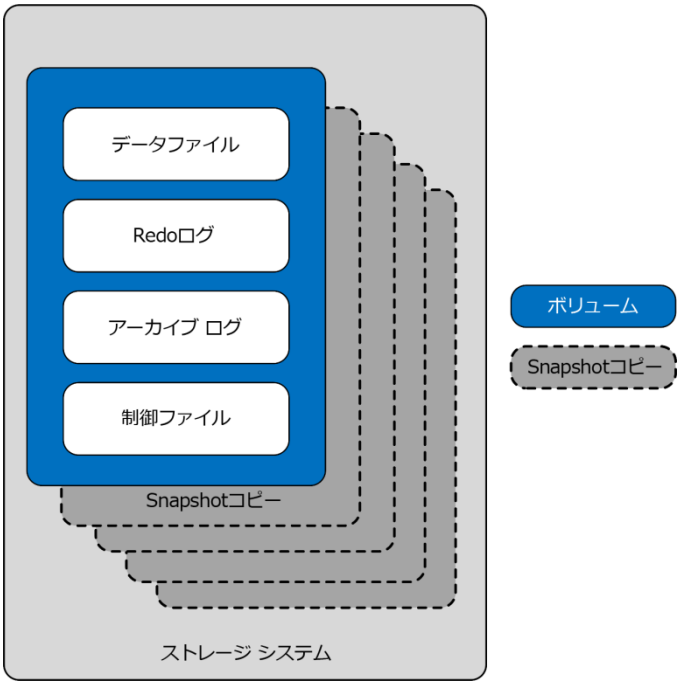
Data ONTAP 8.2 よりも前は、SMO と Snap Creator を使用する環境は VBSR に依存していることがほとんどでした。このプロセスにより、ボリュームを以前の状態に数秒でリストアできます。SMO ファイルシステムの最も重要な側面は、他のデータ タイプの影響を受けないボリュームにデータファイルを分離することでした。SMO は、他のデータ タイプが存在するために VBSR を使用できない構成でも機能しますが、リストア速度が遅くなっていました。データファイルが分離されていれば、ファイルを瞬時にリストアして、即座にアーカイブ ログの再生を実行できます。

「Snapshot ベースのリカバリ」セクションの説明にあるように、clustered Data ONTAP は、よりきめ細やかなリストア機能で最適化されています。そのため、データファイルを専用ボリュームに分離する必要はほとんどありません。NFS が使用されている場合、個々のデータファイルを Snapshot コピーからすばやくリストアできます。LUN の場合は、今までどおりデータファイルを専用のディスクグループに分離する必要がありますが、専用のボリュームは必要ありません。LUN は Snapshot コピーからすばやくリストアできます。

データの混在

図 6 は、すべてのデータ タイプが 1 つのボリュームに混在している構成を示しています。

図 6）データが混在している SMO と Snap Creator のレイアウト



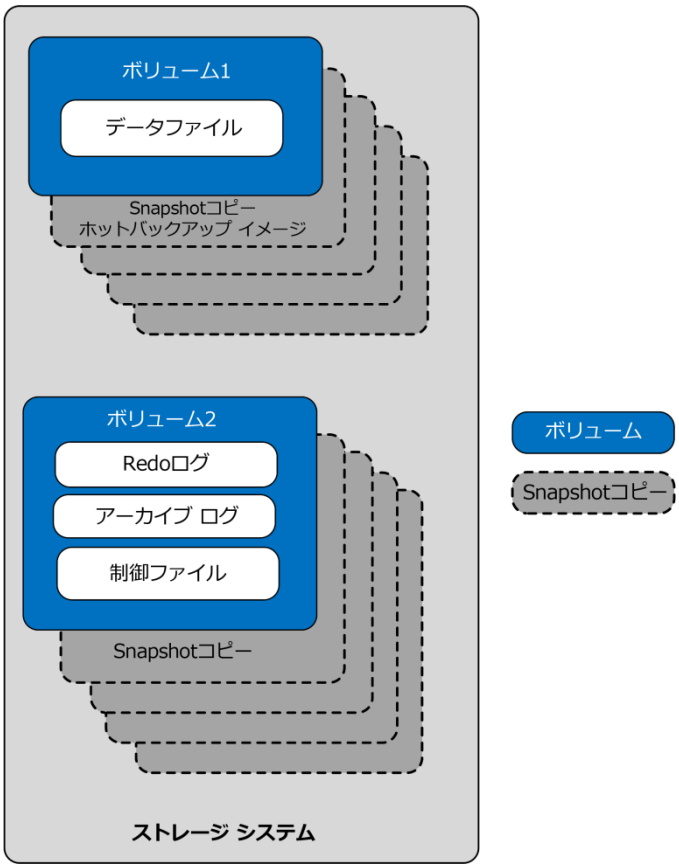
このモデルの主なメリットは、シンプルさです。管理対象のボリュームが 1 つと 1 セットの Snapshot コピーだけで構成されているため、バックアップ時点へのリカバリがきわめて容易です。ただし、考慮が必要な問題がいくつかあります。

- ホット バックアップの手順には、データファイル用とログ用に 2 つの Snapshot コピーが必要です。この 2 つの Snapshot コピーは、ボリュームあたり 255 個という Snapshot コピーの最大数にカウントされます。
- NFS 環境では、データファイルは個々に高速でリストアできます。
- SAN 環境では、ボリューム内の特定の LUN または複数の LUN にデータファイルを分離することで、他のファイルに影響を与えることなく、ひとまとまりでリストアできます。
- データの書き替え率がきわめて高いと、容量管理に影響が及ぶ恐れがあります。例えば、データファイルの各 Snapshot コピーは、アーカイブ ログ内のデータもキャプチャします。アーカイブ ログのデータ量が増えすぎてパージが必要となった場合、古いアーカイブ ログは Snapshot コピーに残っているため、アーカイブ ログの容量を別途管理することはできません。

2 ボリューム モデル

このモデルを使用すると、より柔軟な容量管理が可能となり、データファイルのリカバリ速度が向上します。図 7 は、2 つのボリュームを使用した SMO と Snap Creator のレイアウトを示しています。

図 7）2 つのボリュームを使用した SMO と Snap Creator のレイアウト

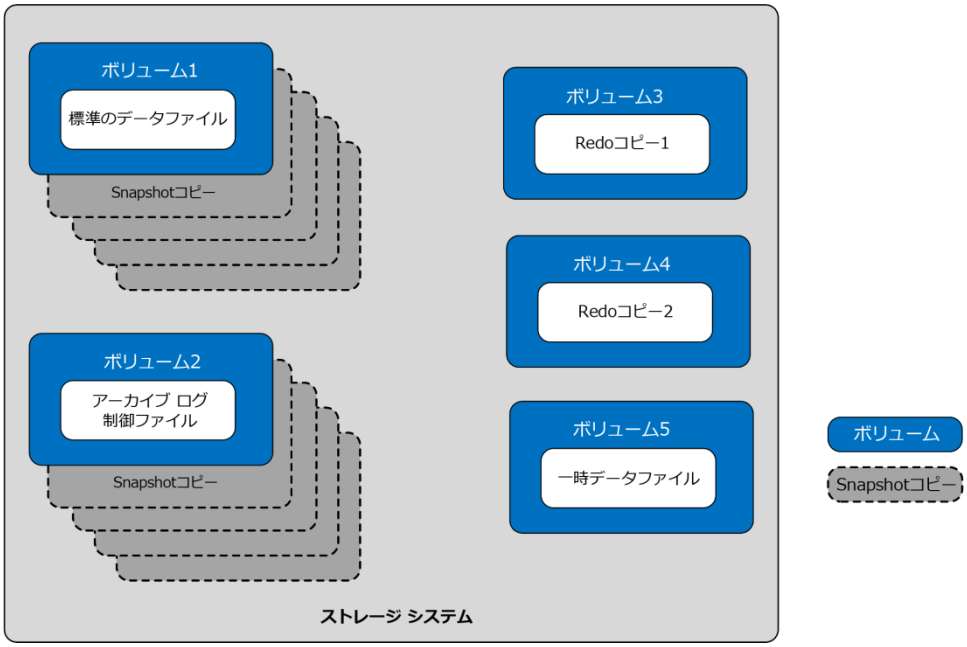


データファイルが特定のボリュームに分離されているため、他のファイルに影響を与えることなく、ボリュームベースの高速リストアを実行することができます。アーカイブ ログと制御ファイルのデータを 1 つの Snapshot コピーで保護できるため、アーカイブ ログ データのボリュームで管理上の問題が生じた場合は、データをより簡単にパージできます。Redo ログのデータはアーカイブ ログの Snapshot コピーにもキャプチャされますが、Redo ログのサイズは、通常アーカイブ ログに必要なスペースに比べればわずかです。そのため、Redo ログの Snapshot コピーによるディスク スペースの無駄な消費は気になるほどの量ではありません。

データの完全分離

非常に大規模なデータベースでは、すべてのデータ タイプを完全に分離しなければならない場合があります。図 8 は、SMO と Snap Creator を使用したレイアウトで、データを完全に分離した状態を示しています。

図 8）データを完全に分離した SMO と Snap Creator のレイアウト



このレイアウトでは、プロビジョニング時に追加の手順が必要ですが、一方で管理面に多数のメリットをもたらします。

具体的には、次のようなメリットがあります。

- Snapshot コピーに不要なデータがキャプチャされることはありません。
- 容量をさらにきめ細かく管理できます。例えば、アーカイブ ログの場所に想定以上の容量が必要になった場合、`volume move` 処理によって、ログの場所を別のコントローラに無停止で変更できます。
- パフォーマンスをさらにきめ細かく管理できます。例えば、Redo ログが想定以上に作成されていることが確認された場合、`volume move` 処理によって、Redo ログ ボリュームの 1 つを別のコントローラに無停止で移動できます。
- 一時データファイルが大量のスペースを消費することは普通ありませんが、一時データファイルが大量に使用されている場合、書き替え率がきわめて高くなる恐れがあります。リカバリ処理に一時データファイルは必要ないため、Snapshot コピーにキャプチャすることはスペースの無駄です。

ハイブリッド EF / FAS を使用したデータ保護

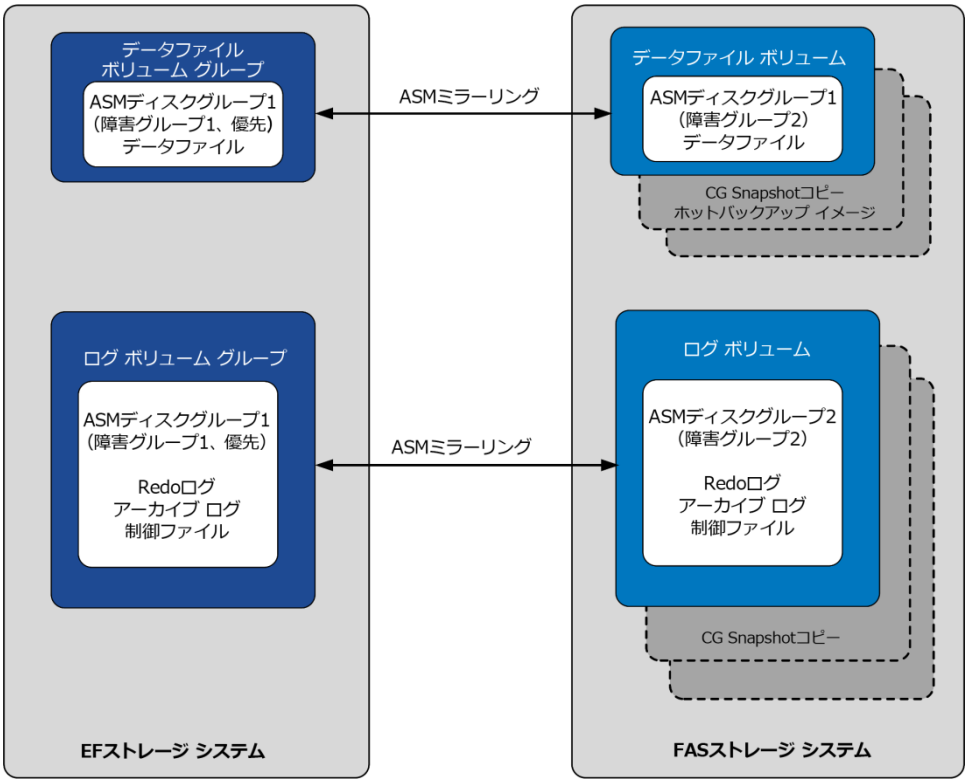
このセクションでは、ハイブリッド アーキテクチャに使用可能なデータ保護戦略を説明します（TR-4145『All Flash Accelerator for Oracle』を参照）。この戦略を使用すると、Data ONTAP のデータ管理機能と、NetApp Santricity® OS を搭載した EF シリーズ アレイの、きわめて高い物理パフォーマンスを活用できます。

TR-4145 に詳述されているハイブリッド アクセラレータは、定評のある Oracle ASM の機能を活用して、ミラーのどちらか優先される一方を使用してミラーリングを実行します。ミラーのどちらか一方のことを、一般に「プレックス」と言います。優先プレックスを設定する主な目的は、処理が速いプレックスから優先的に読み取ることです。書き込みについては、引き続き両方のプレックスに同期でミラーリングする必要があります。ただし、データベースのほとんどは、ランダム リードのレイテンシが主要なボトルネックになっているので、ミラー関係のうち処理が速い方のプレックスを選択することでパフォーマンスが向上します。大抵の場合、距離の関係で一方のプレックスの方が処理が速くなります。例えば、Oracle ASM ミラーリング環境の多くは DR を目的に構築されており、1 つのプレックスはローカルに、もう 1 つはリモート サイトにあるため、後者のレイテンシが高くなります。ローカルのプレックスを優先プレックスに選択すれば、パフォーマンスが向上します。

他の論理ボリューム マネージャにも同じような機能がありますが、Oracle データベースには Oracle ASM を LVM として使用するのが最も一般的です。Oracle ASM では、プレックスは `failgroup` と呼ばれます。

図 9 は、EF と FAS のコンポーネントを使用したレイアウトの概要を示しています。

図 9）ハイブリッド EF / FAS を使用したデータ保護



バックアップ

このセクションで説明するバックアップ手順は、本ドキュメントで取り上げている他のモデルの手順と同じです。その方法は次のようにきわめてシンプルです。

1. cron などのスケジューラを使用し、23 時 58 分にデータベースをホットバックアップ モードに切り替えます。
2. Data ONTAP への直接 Snapshot コピーのスケジュールを 0 時 0 分に設定します。
3. 0 時 2 分にデータベースをホットバックアップ モードから通常モードに戻します。

保護を強化するには、アーカイブログ ボリュームへの直接 Snapshot コピーをスケジュール設定します。

このプロセスでは、パフォーマンスや容量が制約となる場合があります。読み取り負荷の高い作業は EF 側のミラーで実行されますが、このようなアーキテクチャが導入されている大半の環境では、両方のプレックスで大量の書き込み I/O が発生します。さらにこうしたデータベースは平均以上のサイズである可能性があるため、複数の FAS ボリューム、あるいは複数のコントローラが必要になる場合があります。その場合、スケジュール設定された Snapshot コピーは一貫性が維持されないため使用できません。

Snap Creator と Snap Creator Oracle プラグインの組み合わせが最適なオプションです。Snap Creator には、マルチボリューム環境に必要な整合グループ Snapshot コピーを作成する機能が標準で備わっており、Snap Creator Oracle プラグインはハイブリッド EF / FAS 環境で機能します。Snap Creator のドキュメントは、ネットアップ サポート サイトで入手できます。

リカバリ

リカバリ手順では、ミラーの正しいプレックスが検出され、リカバリに使用されていることを手動で確認する必要があります。リカバリの手順は以下のとおりです。

1. Oracle を、ASM を含めて完全にシャット ダウンします。
2. EF LUN をフェンシングします。フェンシングの方法は、FC ゾーニングを変更してアレイを分離する、ASM_DISKSTRING をアラートで送信する、または単純に EF アレイの電源を切るなど様々です。
3. 必要なデータファイルの Snapshot コピーを FAS システムにリストアします。これにより、ホットバックアップ モード中に作成されたデータファイルの Snapshot コピーの状態に ASM ディスクグループが戻ります。
4. 必要に応じて、ログの Snapshot コピーをリストアします。通常は、稼働中の ASM ディスクグループにアーカイブ ログが残っているので、この処理は不要です。
5. ASM を再起動します。障害グループはないため、FORCE オプションを使用する必要があります。
6. データベースを起動し、必要に応じてリカバリを実行します。
7. EF LUN のフェンシングを解除し、ミラーを再構築します。パフォーマンスへの影響を最小限にするために再構築率を抑える必要があることがありますが、再構築処理は単純なシーケンシャル I/O 処理なので、ストレージ システムに大きな負荷がかかることは一般にありません。

Snap Creator を使用すると、このリカバリ手順を完全自動化できます。ただし手順そのものは、環境の構成とリカバリ要件によって異なるので、ネットアップ プロフェッショナル サービスに支援を要請してください。

付録 2：古い NFS ロック

Oracle データベース サーバがクラッシュすると、再起動時に古い NFS ロックで問題が生じる場合があります。この問題は、サーバの名前解決を注意深く設定することで回避できます。

この問題は、ロックの作成と解除に使用される 2 つの名前解決手法が微妙に異なることによって発生し、ネットワーク ロック マネージャ（NLM）と NFS クライアントという 2 つのプロセスが関係しています。NLM はホスト名の確認に uname –n を使用し、rpc.statd プロセスは gethostbyname() を使用します。OS が古いロックを適切に解除するためには、この 2 つのホスト名が一致していなければなりません。例えば、dbserver5 によって所有されるロックを検索している場合に、dbserver5.mydomain.org という名前のホストでロックが登録されていたとします。gethostbyname() が uname –a と同じ値を返さないと、ロックの解除プロセスは成功しません。

以下のスクリプト例では、名前解決が完全に一貫しているかどうかを検証しています。

```
#! /usr/bin/perl
$uname=`uname -n`;
chomp($uname);
($name, $aliases, $addrtype, $length, @addrs) = gethostbyname $uname;
print "uname -n yields: $uname\n";
print "gethostbyname yields: $name\n";
```

gethostbyname が uname に一致しない場合はロックが古い可能性があります。以下の結果は潜在的な問題を示しています。

```
uname -n yields: dbserver5
gethostbyname yields: dbserver5.mydomain.org
```

通常この問題は、/etc/hosts に記述されているホストの順序を変更することで解決できます。例えば、hosts ファイルに次のエントリが含まれているとします。

```
10.156.110.201  dbserver5.mydomain.org dbserver5 loghost
```

問題を解決するには、完全修飾ドメイン名と短いホスト名の記述順序を次のように変更します。

```
10.156.110.201  dbserver5 dbserver5.mydomain.org loghost
```

gethostbyname() は短いホスト名である dbserver5 を返すようになり、これは uname の出力に一致します。このため、サーバがクラッシュするとロックは自動的に解除されます。

付録 3：WAFL アライメントの検証

優れたパフォーマンスを達成するには、WAFL を正しくアライメントすることが重要です。Data ONTAP はブロックを 4KB 単位で管理しますが、だからと言ってすべての処理が 4KB 単位で実行されるわけではありません。実際、Data ONTAP は様々なサイズのブロック処理に対応しますが、基盤となる計算処理は WAFL によって 4KB 単位で管理されます。

「アライメント」という用語は、Oracle の I/O がこの 4KB 単位にどう対応しているかを意味します。パフォーマンスを最適化するには、Oracle の 8KB ブロックが、ディスク上で、WAFL の 4KB の物理ブロック 2 つに配置されている必要があります。Oracle のブロックが 2KB ずれて配置されている場合、このブロックは、4KB ブロックの半分と次の 4KB ブロック全体、さらに 3 つ目の 4KB ブロックに配置されます。この状態は、パフォーマンスの低下を招きます。

NAS ファイルシステムの場合、アライメントは問題となりません。Oracle データファイルは Oracle ブロックのサイズを基にファイルの開始位置にアライメントされるため、常に 8KB、16KB、32KB のブロック サイズでアライメントされます。すべてのブロック処理は、ファイルの開始位置から 4 キロバイト単位でオフセットされます。

一方 LUN の場合は、一般に何らかのディスク ヘッダーやファイルシステムのメタデータが開始位置に含まれるため、オフセットが発生します。最新の OS でアライメントが問題になることは、まずありません。最新 OS は、標準の 4KB セクターを使用する物理ディスク向けに設計されており、パフォーマンスを最適化するには I/O も 4KB の境界に合わせる必要があるからです。

ただし、例外がいくつかあります。4KB I/O 向けに最適化されていない古い OS からデータベースを移行した場合や、パーティション作成時のユーザ エラーにより、4KB 単位以外のサイズでオフセットされている場合です。

以下は Linux 固有の例ですが、手順はどの OS にも該当します。

アライメント

以下に、パーティションが 1 つの単一の LUN でアライメントをチェックする例を示します。

まず、ディスクで使用可能なパーティションをすべて使用するパーティションを作成します。

```
[root@jfs0 iscsi]# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0xb97f94c1.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

The device presents a logical sector size that is smaller than
the physical sector size. Aligning to a physical sector (or optimal
I/O) size boundary is recommended, or performance may be impacted.

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-10240, default 1):
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-10240, default 10240):
Using default value 10240

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@jfs0 iscsi]#
```

以下のコマンドを使用して、アライメントを正確にチェックできます。

```
[root@jfs0 iscsi]# fdisk -u -l /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
64 heads, 32 sectors/track, 10240 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 65536 bytes
Disk identifier: 0xb97f94c1

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1              32    20971519    10485744   83  Linux
```

出力からは、セクターが 512 バイト単位で、パーティションの開始位置がこのセクター32 個分であることがわかります。つまり 32 x 512 = 16,384 バイトとなり、これは 4KB の WAFL ブロックの倍数なので、このパーティションは正しくアライメントされています。

アライメントが正しいことを確認するには、次の手順を実行します。

1. LUN の Universally Unique Identifier（UUID）を確認します。

```
FAS8040SAP::> lun show -v /vol/jfs_luns/lun0
              Vserver Name: jfs
                  LUN UUID: ed95d953-1560-4f74-9006-85b352f58fcd
                    Mapped: mapped
```

2. Data ONTAP コントローラでノード シェルを開始します。

```
FAS8040SAP::> node run -node FAS8040SAP-02
Type 'exit' or 'Ctrl-D' to return to the CLI
FAS8040SAP-02> set advanced
set not found.  Type '?' for a list of commands
FAS8040SAP-02> priv set advanced
Warning: These advanced commands are potentially dangerous; use
         them only when directed to do so by NetApp
         personnel.
```

3. 手順 1 で確認したターゲットの UUID に対して統計データを収集します。

```
FAS8040SAP-02*> stats start lun:ed95d953-1560-4f74-9006-85b352f58fcd
Stats identifier name is 'Ind0xffffff08b9536188'
FAS8040SAP-02*>
```

4. I/O をいくつか実行します。`iflag` 引数を使用し、I/O がバッファされずに同期されるようにする必要があります。

注： このコマンドは慎重に使用してください。`if` と `of` の引数の順序を逆に指定すると、データが破棄されてしまいます。

```
[root@jfs0 iscsi]# dd if=/dev/sdb1 of=/dev/null iflag=dsync count=1000 bs=4096
1000+0 records in
1000+0 records out
4096000 bytes (4.1 MB) copied, 0.0186706 s, 219 MB/s
```

5. 統計を終了し、アライメントのヒストグラムを表示します。すべての I/O が `.0` バケットにあることを確認します。これは、I/O が 4KB のブロック境界でアライメントされていることを意味します。

```
FAS8040SAP-02*> stats stop
StatisticsID: Ind0xffffff08b9536188
lun:ed95d953-1560-4f74-9006-85b352f58fcd:instance_uuid:ed95d953-1560-4f74-9006-85b352f58fcd
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.0:186%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.1:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.2:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.3:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.4:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.5:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.6:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.7:0%
```

ミスアライメント

以下に、ミスアライメント I/O の例を示します。

1. 4KB の境界に合っていないパーティションを作成します。OS が最新の場合、これはデフォルトの動作ではありません。

```
[root@jfs0 iscsi]# fdisk -u /dev/sdb
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First sector (32-20971519, default 32): 33
Last sector, +sectors or +size{K,M,G} (33-20971519, default 20971519):
Using default value 20971519

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

2. パーティションは、デフォルトの 32 ではなく、33 セクターのオフセットで作成されています。「アライメント」セクションの手順を繰り返します。次のようなヒストグラムが表示されます。

```
FAS8040SAP-02*> stats stop
StatisticsID: Ind0xffffff0468242e78
lun:ed95d953-1560-4f74-9006-85b352f58fcd:instance_uuid:ed95d953-1560-4f74-9006-85b352f58fcd
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.0:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.1:136%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.2:4%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.3:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.4:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.5:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.6:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_align_histo.7:0%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:read_partial_blocks:31%
```

明らかにミスアライメントであることがわかります。I/O のほとんどが .1 バケットに収まり、予想したオフセットに一致しています。パーティションの作成時に、最適なデフォルトから 512 バイトうしろに作成したので、ヒストグラムは 512 バイトでオフセットされています。

また、read_partial_blocks 統計がゼロ以外の値を示していますが、これは、実行された I/O が 4KB ブロック全体を満たしていないことを意味します。

Redo ロギング

ここでは、データファイルを対象とした手順を取り上げます。Oracle の Redo ログとアーカイブ ログは、I/O パターンが異なります。例えば、Redo ロギングでは 1 つのファイルが繰り返し上書きされます。デフォルトの 512 バイトのブロック サイズを使用している場合、書き込み統計は次のようになります。

```
FAS8040SAP-02*> stats stop
StatisticsID: Ind0xffffff0468242e78
lun:ed95d953-1560-4f74-9006-85b352f58fcd:instance_uuid:ed95d953-1560-4f74-9006-85b352f58fcd
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.0:12%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.1:8%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.2:4%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.3:10%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.4:13%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.5:6%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.6:8%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_align_histo.7:10%
lun:ed95d953-1560-4f74-9006-85b352f58fcd:write_partial_blocks:85%
```

I/O はヒストグラムのすべてのバケットに分散されますが、これはパフォーマンス上の問題とはなりません。ただし、Redo ロギング率がきわめて高い場合は、4KB のブロック サイズを使用するとパフォーマンスが向上します。この場合は、Redo ロギング LUN を正しくアライメントされていることを推奨しますが、優れたパフォーマンスにとってはデータファイルのアライメントほど重要ではありません。

本ドキュメントに記載されている、特定バージョンの製品と機能がお客様の環境でサポートされるかどうかは、ネットアップ サポート サイトにある Interoperability Matrix Tool（IMT）で確認してください。NetApp IMT には、ネットアップがサポートする構成を構築するために使用できる製品コンポーネントやバージョンが定義されています。サポートの可否は、お客様の実際のインストール環境が公表されている仕様に従っているかどうかによって異なります。

著作権に関する情報

商標に関する情報

TR-3633-0916-JP