Oracle Maximum
Availability Architecture

データベース統合のための高可用性ベスト・プラクティス

Database-as-a-Serviceの基盤

Oracle ホワイト・ペーパー | 2016 年 4 月

ORACLE

目次

概要

企業は、少ないコストでより大きな成果を上げ、リスクを軽減し、さらに俊敏性を強化することを強く迫られています。これらの目標を達成するために多くの企業が追求している戦略として、情報技術（IT）インフラストラクチャと、パブリックまたはプライベート・クラウド上での Database as a Service（DBaaS）の展開との積極的な統合があります。

データベース統合および DBaaS を通してコストを可能な限り削減するには、いくつかの重要な要素が必要です。ハードウェア・コストと管理コストを最大限削減するには、高密度の統合と管理の簡素化が必要です。次に、これらの要素を、可用性、パフォーマンス、およびデータ保護の品質保証契約（SLA）を達成できるインテリジェントなソフトウェア・インフラストラクチャと組み合わせる必要があります。

Oracle Database 12c に付属の Oracle Multitenant は、Oracle Database を根本的に再構築します。そのため Oracle Database 12c は、データベース統合に最適なプラットフォームとなります。Oracle Multitenant は斬新的であり、既存の Oracle データベースの統合を簡素化してコストを削減できます。また、Oracle Multitenant は革新的でもあります。統合密度を最大化するとともに、統合データベース環境の管理を以前の統合戦略と比べて劇的に簡素化できます。

Oracle Maximum Availability Architecture（Oracle MAA）を使用した Oracle Multitenant は、停止があればその影響が何倍にも拡大されるような統合データベース環境に高可用性とデータ保護を提供します。Oracle Multitenant および Oracle MAA を使用すれば、基盤となるハードウェア・プラットフォームやオペレーティング・システム環境には関係なく固有の利点が提供されますが、最大の利点は Oracle エンジニアド・システム上に展開された場合に得られます。Oracle 統合ハードウェア/ソフトウェア・ソリューションは、統合環境でのパフォーマンス、信頼性、管理性、サポートなどの複数の側面にわたる規模の経済利益を実現する標準の高性能プラットフォームを使用することによってライフ・サイクルの総コストを削減します。たとえば、Oracle Exadata Database Machine では、統合密度において従来のシステムと比べて最大 5 倍の優位性が実証されています。

このホワイト・ペーパーでは、Oracle Multitenant を使用したデータベース統合のための Oracle MAA のベスト・プラクティスについて説明します。ここでは、DBaaS の基盤である標準の HA アーキテクチャについて説明しています。このホワイト・ペーパーは、従来のデータベース展開から DBaaS への統合および移行を担当しているアーキテクト、IT ディレクター、データベース管理者などの専門家を対象とした内容になっています。推奨されるベスト・プラクティスは、Oracle エンジニアド・システムにのみ適用される最適化や例であると明示的に示されている場合を除き、Oracle Database によってサポートされるすべてのプラットフォームに等しく関連しています。

はじめに

このホワイト・ペーパーではまず、データベース統合に対する従来のアプローチとそのトレードオフについて考察し、Oracle Multitenant によってもたらされる高い価値について詳細を説明します。残りの部分では、データベース統合のための HA ベスト・プラクティスを示し、高可用性とデータ保護のためのサービス・レベルで規定される標準の参照アーキテクチャであるブロンズ、シルバー、ゴールド、およびプラチナについて説明します。

オペレーティング・システムの仮想化 - 仮想マシン

最初の頃の統合作業では、オペレーティング・システムの仮想化を使用したサーバー統合に重点を置いていました。オペレーティング・システムの仮想化では、単純なテンプレート駆動型の展開や、1 台の物理マシンを複数の仮想マシン（VM）に分割することによってサーバーの利用効率の向上を可能にします。サーバーの利用効率の向上は、データベース統合の目的の 1 つです。それぞれ利用効率の異なる 10 台の個別の物理マシン上で 10 個のデータベースを実行するのではなく、これらの同じデータベースを 1 つの物理システム上の 10 個の仮想マシン（VM）内に展開することによってサーバー・フットプリントを削減できます。

ただし、データベース統合の観点から見ると、VM が単独で実現できる内容には次のようないくつかの欠点が存在します。

» オペレーティング・システムとデータベース・ソフトウェアの複数のインスタンスが実行され、それぞれがメモリ・フットプリントやシステム・フットプリントを伴うため非効率となり、統合密度が制限されます。これにより、特定の物理マシン上に展開される VM やデータベースの数が増えるため、パフォーマンスの低下を引き起こします。

» VM は、管理する必要のあるデータベースやオペレーティング・システム環境の総数を削減することができないため、管理コストを制御する機能に制限があります。

» VM は、停止時間やデータ損失を許容できないミッション・クリティカルなアプリケーションの HA 要件に対応していません。停止後の動作は、コールド・リスタートに限定されており、リアルタイムのデータ保護は、外部のレプリケーション・テクノロジーとの統合に依存しています。

» VMは、ビジネス・サイクルの結果、システム・リソース（CPUやIOなど）の消費がデータベース統合環境内のあるコンシューマ・グループまたはデータベースから別のコンシューマ・グループまたはデータベースへと 1 日中変動する可能性のある Oracle Database に対する動的なリソース管理に限界があります。

スキーマ統合

スキーマ統合とは、スタンドアロン・データベースを、1 つのデータベースに統合されたスキーマに変換するプロセスのことをいいます。スキーマ統合は、VM をデータベース層の統合戦略として使用する場合に特有の欠点である統合密度と管理の効率性に対応します。多くの状況で展開が成功している一方で、スキーマ統合には、固有の一連のトレードオフがあります。特に、スキーマ統合向けに設計されていない既存のアプリケーションやデータベース環境では実装が困難です。また、VM で実現できる内容と比較すると、スキーマ・レベルでの移植性と分離についての独自の簡便性も備えていません。

Oracle Database 11gとのデータベース統合

Oracle Real Application Clusters（Oracle RAC）および Oracle Resource Management を使用した Oracle Database は、Oracle Database 用に最適化された最初の統合プラットフォームであり、これ

が Oracle Database 11g のための Oracle MAA のベスト・プラクティスになっています。Oracle RAC を使用すると、複数の Oracle データベースを 1 つの Oracle RAC クラスタに容易に統合できます。Oracle Resource Management は、Oracle RAC クラスタ上で統合された複数のデータベースおよびコンシューマ・グループにわたるシステム・リソースの優先順位付けを可能とするため、応答時間のサービス・レベル目標が確実に達成されます。

Oracle Exadata Database Machine は、スケーラブルなパフォーマンスと固有のリソース管理機能の組合せによって統合環境に付加価値を提供します。詳しくは、『Exadata Database Machine 上でのデータベース統合に関するベスト・プラクティス』を参照してください。

Oracle RAC、Exadata、および Resource Management は、管理性、パフォーマンス、および HA の大きな利点を備えた統合の簡単な方法を提供しますが、スキーマ統合と同程度の効率性を実現することはできません。Oracle RAC クラスタ内の各データベース（ディスク上に存在するシステムおよびユーザー・データ）には、1 つ以上のデータベース・インスタンス（専用のバックグラウンド・プロセスおよびメモリ領域）が存在します。各データベース・インスタンスは、1 つのデータベースをマウントして開くことができます。1 つのクラスタ内で統合されたデータベースの数が増えるにつれ、各データベース・インスタンスのために専用に確保する必要のあるメモリと CPU の量によって、実現可能な統合密度に実質的な制限が課せられます。

Oracle Multitenantを使用したデータベース統合

Oracle Multitenant は、マルチテナント・コンテナ・データベース（CDB）とプラガブル・データベース（PDB）の概念の導入により、Oracle Database のアーキテクチャを根本的に変えます。既存のデータベースは、PDB に容易に変換できます。複数の PDB を 1 つの CDB に'プラグイン'することによって統合が可能です [1]。Oracle Multitenant を使用した Oracle Database 12c は、データベース統合のすべての側面でもっとも効率的なプラットフォームを提供するように設計されています。

CDB には、すべての PDB によって使用される 1 組のバックグラウンド・プロセスおよび共有メモリ領域（SGA）が存在します。このアーキテクチャでは、複数の独立したデータベースを 1 台の物理マシン、複数の VM、または Oracle RAC クラスタに統合する従来のアプローチと比べて、CPU とメモリの使用率が少なくなります。CDB は物理環境または仮想環境のどちらにも展開できますが、データベース層に対する管理とパフォーマンスの効率は、物理マシン上に展開された場合がもっとも高くなります。CDB 自体がデータベース層の仮想化テクノロジーになるため、複数の VM やゲスト・オペレーティング・システムのオーバーヘッドが解消されます。

Oracle Multitenant アーキテクチャの優位性は、Oracle Database 12c を実行している Sun T5-8 サーバー上で実行された一連のテストによって実証されています。使用した T5-8 は、128 個のコア、2TB のメモリ、および 8 台の Exadata Storage Server で構成されており、テストは、252 個の統合された OLTP データベース（33%が 1GB での小規模、33%が 5GB での中規模、33%が 15GB での大規模）を使用して実行されました。テストは、マルチテナント・アーキテクチャでのプラガブル・データベースの展開をシングル・インスタンス・データベースと比較するように設計されていました。

テストでは、同じシステム・リソース（CPU、メモリ、I/O）を使用した場合、Oracle Multitenant はシングル・インスタンス・データベースと比べて統合密度を向上できることが実証されました。Oracle Multitenant が達成した内容は次のとおりです。

1 Oracle Database 12c Release 1（12.1）では、CDB に最大 252 個の PDB を含めることができます。将来のリリースでは、この上限を増やす予定です。

» データベースあたり同じスループットを実現しながら、統合密度（統合されたデータベースの数）を 50%向上

» 同じ数のデータベースを統合した場合、集計スループットを 80%向上

テストを別の観点から見ることで、Oracle Multitenant によってハードウェアおよびソフトウェアのライセンス・コストの削減がどのように実現されたのかがわかりました。

» 同数のデータベース（252）を同じスループット・レベルで統合するために、マルチテナント・アーキテクチャに必要な CPU コアは 64 個少なく、IOPS（1 秒間あたりの I/O 操作回数）は 1/3

図 1 は、これらのテスト結果のまとめたものです。詳しくは、Oracle Technology Network に公開されている Oracle Multitenant の統合に関する調査を参照してください。

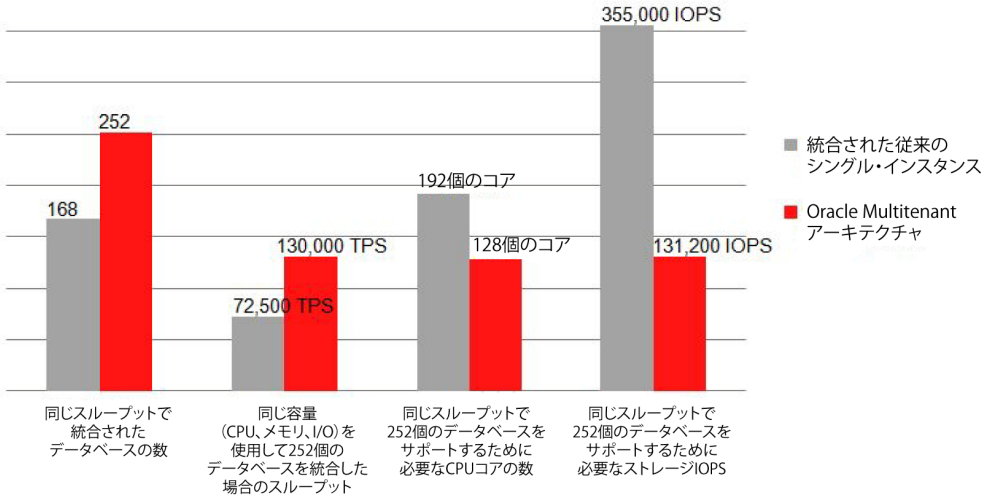

図 1：Oracle Multitenant アーキテクチャの統合の優位性

Oracle Multitenant はまた、一元管理の簡便性と効率性も提供します。図 2 に、これを説明するための簡単な図を示します。CDB 内で統合されている PDB の数には関係なく、管理対象のデータベース（マルチテナント・コンテナ・データベース）は 1 つだけです（1 つのバックアップ、ディザスタ・リカバリ（DR）のための 1 つのコピー、およびアップグレードやパッチ適用の対象となる 1 つのデータベース）。この原則を図 2 に示されている比較的小規模な統合環境に適用すると、Oracle Multitenant は、管理に必要な個別のデータベースの数を削減することによって 4 対 1 の優位性を実現しています。1 つの CDB 内で統合されるデータベースの数が増えるにつれ、この利点は大きくなります。

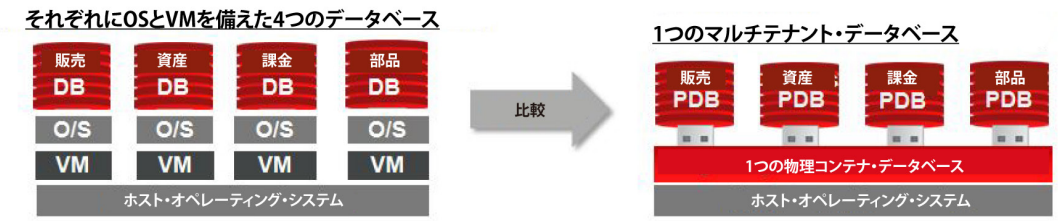

図 2：仮想マシンを使用した統合と、Oracle Multitenant を使用したデータベース仮想化の比較

Oracle Multitenant はまた、高いレベルでの分離も提供します。PDB をある CDB から容易にアンプラグして別の CDB にプラグインできるため、データベース管理者は、必要に応じて個々の PDB 上で保守を実行できます。同じ CDB 内の他の PDB に影響を与えることなく、個々の PDB に対してプロビジョニング、パッチ適用、クローン化、統合、リストア、移動を実行できます。

Oracle Multitenant は、代替の統合手法の各デメリットを回避しながら、優れた点は実現するという点において比類ない存在です。Oracle Multitenant が達成する内容は次のとおりです。

» VM の簡便性と柔軟性。統合密度やパフォーマンスが制限されたり、管理の複雑さが増したりすることはありません。

» スキーマ統合の高い統合密度。実装が複雑になったり、柔軟性や分離が制限されたりすることはありません。

» Oracle RAC と Oracle Database 12c を使用した単純なデータベース統合の HA、スケーラビリティ、および自動化されたワークロード管理。統合密度が制限されたり、アプリケーションごとの（それぞれ独自の運用オーバーヘッドを抱える）個別のデータベースの管理が複雑になったりすることはありません。

Oracle Multitenant は、Oracle Database の HA およびデータ保護機能とシームレスに統合します。この統合と Oracle Maximum Availability Architecture（Oracle MAA）のベスト・プラクティスが組み合わされて、データベース統合のための革新的なテクノロジーへの発展的なアップグレード・パスが提供されます。

Oracle Exadata - 統合およびDBaaS用に最適化済み

Oracle エンジニアド・システムは、Oracle Database 用にさまざまな最適化を行うシステムのファミリーです。これには、Oracle Exadata Database Machine、Oracle SuperCluster、Oracle Database Appliance、および Oracle Virtual Compute Appliance が含まれます。Oracle エンジニアド・システムは、Oracle Database や、Oracle によってサポートされるハードウェアおよびソフトウェア用に事前に統合および最適化されたプラットフォームを標準化することによって、ライフ・サイクル・コストを削減します。

Oracle Exadata Database Machine は、Oracle Database に最適なパフォーマンスと管理性を提供する目的で構築されたエンジニアド・システムです。そのスケーラブルなアーキテクチャと高度なソフトウェア機能により、統合および DBaaS のための標準のデータベース・プラットフォームとして最適な製品になっています。

Exadata Storage ソフトウェアの独自機能には、次のものが含まれます。

» ストレージ内でのデータベース処理のオフロード

  データベースの式の評価をストレージ・セルにプッシュして処理させることにより、問合せが大幅に高速化されます。条件を満たす行、指定された列、および条件付列のみがデータベース・サーバーに返されるため、データベース・サーバーへの非生産的なデータ転送が解消されます。これにより、完全に構成された Oracle Exadata Database Machine では、SQL データ・スループットが 1 秒あたり 100GB になる場合があります。

» データベース用に最適化されたストレージ

Hybrid Columnar Compression では、読取りが多いアーカイブ・データが 10 倍以上に圧縮されるため、全体的なストレージ・フットプリント要件が削減されます。

» データベース用に最適化された PCI フラッシュ

Exadata Smart Flash Cache では、頻繁にアクセスされるデータが高速なソリッド・ステート・ストレージに透過的にキャッシュされるため、問合せの応答時間やスループットが改善されます。ディスクではなく、フラッシュによって処理される書込み操作は、"ライトバック・フラッシュ・キャッシュ"と呼ばれます。このキャッシュは、バックアップや頻度の低い表スキャンがキャッシュされないという点でインテリジェントです。Exadata 11.2.3.3 より、Oracle Exadata Storage Server Software は、表やパーティションのスキャン・ワークロードによって読み取られるオブジェクトを、オンライン・トランザクション処理（OLTP）に影響を与えたり、フラッシュ・キャッシュを過負荷にしたりすることなく、しかも同時にオブジェクトを圧縮しながら、それらのオブジェクトが読み取られる頻度に基づいてフラッシュ・キャッシュ内に自動的にキャッシュするようになりました。PCI フラッシュにより、ディスク・コントローラのボトルネックが解消されます。

Exadata Smart Flash Log は、各フラッシュ・ディスク上の 512MB の領域のみを使用して、待機時間の短い一貫性のあるログ書込みを提供します。待機時間の短い一貫性のあるログ書込みにより、複合ワークロードの場合であっても、OLTP ワークロードの安定化と改善が可能になります。

Exadata Smart Flash Cache Compression（Exadata 11.2.3.3 以降）は、フラッシュ・キャッシュにロードされるユーザー・データを透過的に圧縮することによって、フラッシュ・キャッシュの論理容量を動的に向上させます。これにより、フラッシュ内に大量のデータを保持できるため、ディスク・ドライブ上のデータにアクセスする必要性が減少します。フラッシュ内のデータへの I/O は、ディスク上のデータへの I/O と比べて数桁早い実行速度です。圧縮および圧縮解除操作はアプリケーションやデータベースに対して完全に透過的に実行されます。また、1 秒あたり数百万 I/O の速度で動作している場合であってもパフォーマンス・オーバーヘッドは発生しません。

» データベース用に最適化された包括的なリソース管理

Oracle Database のリソース管理は、Oracle Database によってサポートされるすべてのプラットフォーム上で使用できます。「Oracle Database Resource Manager を使用したリソースの管理」で説明されているように、機能の例としては、それぞれの優先順位に基づいた PDB やデータベース・ワークロードへの CPU の保証、パラレル操作の管理およびペーシング、リソース集中型の問合せの検出および制御などがあります。ただし、Exadata では、DBaaS のサポートに不可欠な固有のリソース管理機能を追加で使用できます。これらの機能には、I/O リソース管理（IORM）とネットワーク・リソース管理が含まれます。

IORM は、複数のデータベース、PDB、およびデータベース・ワークロードで同じストレージを共有できるようにするだけでなく、I/O リソースをそれぞれの優先順位に従って利用できるようにします。Oracle Exadata Storage Server Software は IORM および Oracle Database Resource Manager と連携して、複数のデータベース、CDB、または PDB で同じデータベース・クラスタやストレージ・グリッドを共有している場合でも、顧客が定義したポリシーが満たされることを保証します。その結果、あるデータベースまたはアプリケーション・サービスが I/O 帯域幅を独占し、他のデータベースのパフォーマンスを予測不可能な方法で低下させることは起こり得

ません。Exadata 11.2.3.2.1 より、IORM は、Exadata 上で最大 1024 個のデータベースをサポートしています。Exadata 12.1.1.1 より、IORM は、Oracle Database 12c でマルチテナント・アーキテクチャ内の個々の PDB をサポートするように拡張されました。

Exadata 11.2.3.3 および 12.1.1.1 以降より、ネットワーク・リソース管理は、重要なデータベース・ネットワーク・メッセージに自動的かつ透過的に優先順位を付け、待機時間の影響を受けやすい操作での短い応答時間を保証するようになりました。優先順位付けが InfiniBand ファブリック全体を通して確実に実行されるようにするために、優先順位付けはデータベース・ノード、データベースの InfiniBand アダプタ、Oracle Exadata Storage Server Software、Exadata ストレージ・セルの InfiniBand アダプタ、および InfiniBand スイッチで実施されます。Oracle RAC キャッシュ・フュージョン・メッセージや、重要なバックグラウンド・ノード間通信などの待機時間の影響を受けやすいメッセージは、バッチ、レポート、およびバックアップ・メッセージより優先されます。ネットワークを飽和させる可能性のある大規模なバッチ・ロード、バックアップ、リカバリの読取り/書込み、集中的なレポートや問合せが存在する場合でもトランザクション処理に対して短い待機時間を保証するために、ログ・ファイルの書込み操作にはもっとも高い優先順位が与えられます。

Exadata は、その高度な機能と独自のサービス品質機能により、データベース統合環境内および DBaaS のワークロードをインテリジェントに管理するためにアプリケーションからデータベース、ネットワーク、ストレージに至るエンド・ツー・エンドの優先順位付けを提供する唯一のプラットフォームです。

論証 - Exadataを使用した高い統合密度

オラクルは、同様に構成された非 Exadata システムよりはるかに高い統合密度を実現する Exadata の機能を検証するためのテストを実施しました。同一のワークロードと Oracle Database 構成を使用して、Oracle Exadata Database Machine 上で 2 つの一連のテストを実行しました。

» 最初の一連のテストでは、比較のため、Exadata 固有のすべての機能を無効にしてベースラインを提供しました。この目的は非 Exadata システムをシミュレートすることにありましたが、Exadata 機能を有効にしていない場合でも Exadata システムの高い I/O およびネットワーク帯域幅や安定したパフォーマンス特性が得られるため、このアプローチでは比較用の保守的なベースラインが提供されました。この'非 Exadata'システムのパフォーマンスは、汎用的な Linux システムのパフォーマンスをはるかに超えています。

» 2 番目の一連のテストでは最初のテストと同じワークロードを同じマシン上で実行しましたが、今回は、Exadata Smart Flash Logging、Smart Flash Cache、Smart Scan、Smart Flash Cache Compression、ストレージ・インデックス、ネットワーク・リソース・マネージャ、I/O リソース・マネージャなど、Exadata のみの機能をすべて有効にしました。これにより、データベース・ワークロードの統合密度を向上させる Exadata の独自の機能を測定することが可能になりました。

» Oracle Multitenant は、データベース統合のための Exadata の独自機能の価値を分離するために、どちらのテストでも使用されませんでした。

図 3 に示されている概要は、次の 2 つの基本的な質問の答えを示しています。

» Exadata システム上では、同様に構成された非 Exadata x86 システムと比べてどれだけ多くのデータベースを統合できるか。

最初のテストでは、統合されるデータベースの数がボトルネックに達するまで増加する OLTP ワークロードをシミュレートしました。Exadata は、同等の X86 ハードウェアでは 40 個のデータベース（I/O バウンド）しかサポートされていないのに対して 160 個のデータベース（CPU バウンド）をサポートしているため、より短い応答時間とともに 4 倍の統合密度を示しました。パフォーマンスの品質保証契約でオーバーサブスクリプションが許可される統合環境では、Exadata システムの統合密度の優位性は同様に構成された x86 システムの統合密度の 5 倍に拡大しました。

» Exadata は、統合環境に典型的な複合ワークロードを、同様に構成された x86 システムと比べてどれだけ高速なのか。

このテストでシミュレートされたワークロードには、OLTP ワークロードとレポート・データウェアハウスが含まれていました。テストでは、Exadata が 15 倍のトランザクション応答時間と 2 倍の統合密度、および 6 倍を超えるトランザクション量の優位性を備えていることが示されました。このテストの興味深いもう一つの側面として、非 Exadata システムでは 40 個のデータベースで I/O ボトルネックが発生したのに対して、Exadata システムにはワークロード内のスパイクに対応したり、データベースをさらに統合したりするための I/O および CPU 容量がまだに残っていました。

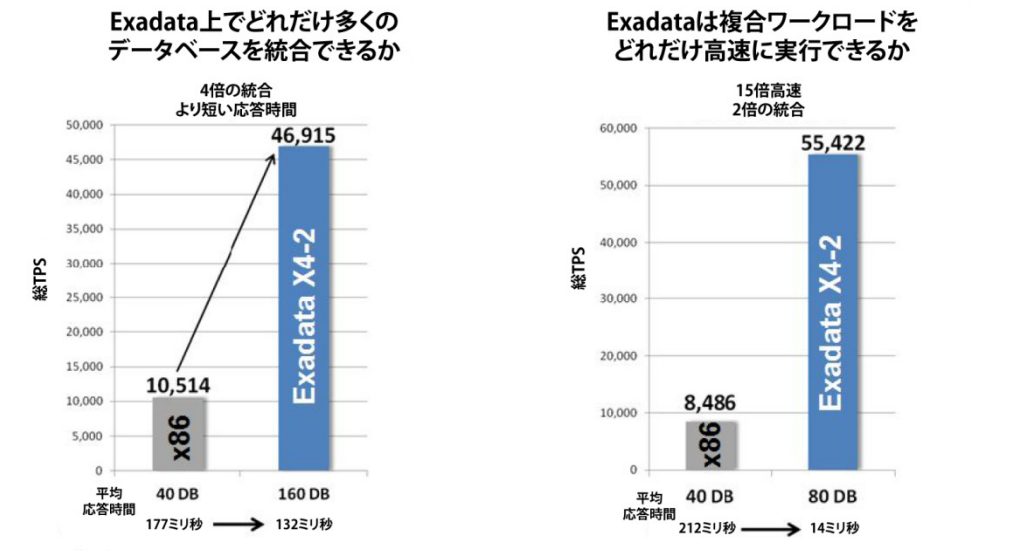

図 3：Exadata の統合密度およびパフォーマンス

統合密度の向上により、Exadata は、購入するハードウェアの削減、管理するシステムの削減、電力消費量の削減、必要な Oracle Database ライセンスの削減といったコスト上の大きな利点を生み出すことができます。

この項で説明したテストの方法、構成、および結果の詳細は、付録 A に記載されています。

高可用性参照アーキテクチャ

Oracle MAA のベスト・プラクティスでは、あらゆる業種のあらゆる規模の企業で必要となるすべての範囲の可用性とデータ保護に対応する、4 つの HA 参照アーキテクチャを定義しています。これらのアーキテクチャ（HA 層）は、プラチナ、ゴールド、シルバー、およびブロンズと呼ばれます。これらはそれぞれ、図 4 で説明されているサービス・レベルを提供します。

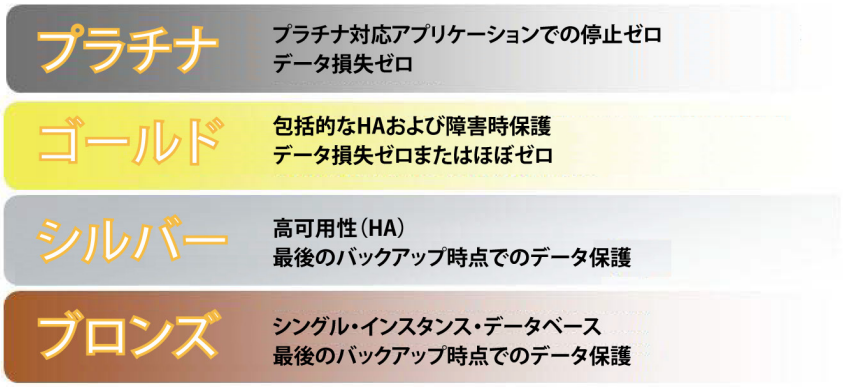

図 4：HA およびデータ保護のサービス・レベル

層ごとに異なる MAA 参照アーキテクチャを使用して、一連の最適な Oracle HA 機能を展開することで、コストと複雑さを最小限に抑えて特定のサービス・レベルを確実に実現します。各層は、データの破損、コンポーネント障害、システムおよびサイトの停止を含むすべての計画外停止に加えて、保守や移行などを目的とした計画停止にも明確に対応します。図 5 に、それぞれのアーキテクチャの概要を示します。

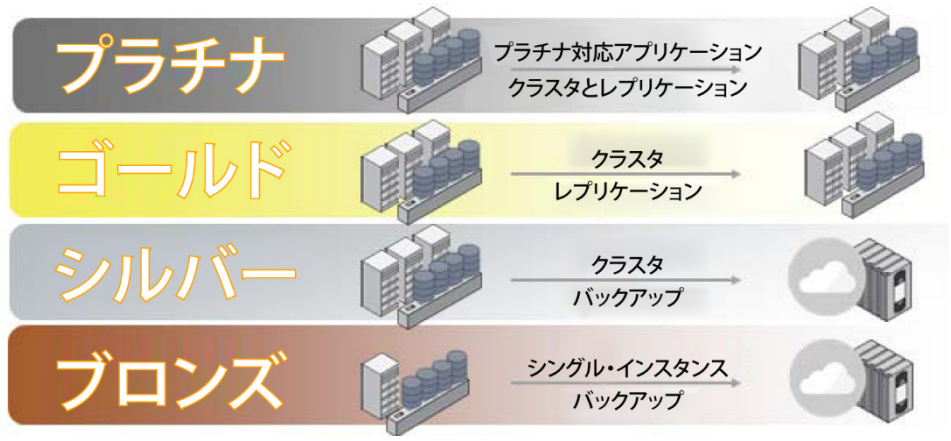

図 5：HA およびデータ保護の参照アーキテクチャ

ブロンズ層が適しているのは、単純な再起動やバックアップからのリストアで'十分な HA'を達成できるデータベースです。ブロンズはシングル・インスタンスの Oracle データベースをベースとしており、Oracle Enterprise Edition ライセンスに含まれる数多くのデータ保護および HA 機能を組み込んだ MAA ベスト・プラクティスを使用しています。停止時にデータベースを再起動できない場合に、データを保護して可用性をリストアするために、Oracle Recovery Manager（Oracle RMAN）を使用して Oracle に最適化されたバックアップを実行します。

シルバー層は、データベース・インスタンスまたはサーバーの障害とさまざまな計画保守の発生時に、停止時間を最小限またはゼロにする必要のあるデータベース向けに、追加の HA レベルを提供します。Oracle RAC または Oracle RAC One Node を使用したクラスタ化テクノロジーが追加されています。Oracle RMAN はデータベース向けに最適化されたバックアップを通じて、データ保護を提供し、機能停止によってクラスタの再起動が妨げられた場合に可用性を回復します。

ゴールド層では、シングル・ポイント障害に対する脆弱性が許容されないビジネス・クリティカルなアプリケーション向けに、サービス・レベルが大幅に引き上げられています。ゴールド層に追加されたデータベース認識型のレプリケーション・テクノロジーである Oracle Active Data Guard および Oracle GoldenGate は、本番データベースの 1 つまたは複数のレプリカを同期し、リアルタイムのデータ保護および可用性を提供します。データベース認識型のレプリケーションは、ストレージ・レプリケーション・テクノロジーを大幅に上回る HA およびデータ保護を提供します。また、常にすべてのレプリカを積極的に活用することで、投資収益率の改善とコスト削減を実現します。

プラチナ層には、いくつかの Oracle Database 12c 新機能と最新リリースで拡張された従来製品が採用されています。これには、実行中トランザクションを確実に再生することで、ユーザーがシステム停止に気付かないようにする Application Continuity、距離に関係なくデータ損失ゼロの保護を実現する Active Data Guard Far Sync、停止時間ゼロのアップグレードと移行を可能にする新しい GoldenGate 拡張機能、レプリケートされたデータベース環境に自動化されたサービス管理とワークロード・バランシングを提供する Global Data Services が含まれます。テクノロジーごとに追加の実装作業が必要になりますが、停止時間とデータ損失が許されないもっともクリティカルなアプリケーションにとって、非常に意味のある価値が提供されます。

それぞれの参照アーキテクチャが持つ HA 特性とデータ保護特性を、表 1 にまとめます。

表 1：高可用性とデータ保護

停止クラス/HA層	計画外停止（ローカル・サイト）	計画保守	データ保護	リカバリ不能なローカル停止およびディザスタ・リカバリ
プラチナ	プラチナ対応アプリケーションでのアプリケーション停止ゼロ	アプリケーション停止ゼロ	包括的な実行時検証と手動チェックの組合せ	プラチナ対応アプリケーションでのアプリケーション停止ゼロ、実行中のトランザクションの保持、データ損失ゼロ
ゴールド	包括的なHA/DR	すべてローリングまたはオンライン	包括的な実行時検証と手動チェックの組合せ	リアルタイムのフェイルオーバー、データ損失ゼロまたはほぼゼロ
シルバー	自動フェイルオーバーを備えたHA	一部ローリング、一部オンライン、一部オフライン	基本的な実行時検証と手動チェックの組合せ	バックアップからのリストア、最後のバックアップ以降に生成されたデータを失う可能性あり
ブロンズ	リカバリ可能なインスタンス障害とサーバー障害に自動再起動を提供するシングル・インスタンス	一部オンライン、ほとんどオフライン	基本的な実行時検証と手動チェックの組合せ	バックアップからのリストア、最後のバックアップ以降に生成されたデータを失う可能性あり

MAA 参照アーキテクチャは、Oracle データベース向けに最適化された標準インフラストラクチャを提供することで、企業が各種のサービス・レベル要件に適した HA レベルを活用できるようにします。標準化によってコストが削減されるだけでなく、ビジネス要件が変わった場合に、次の HA 層や別のハードウェア・プラットフォームへのデータベース移行が簡単になります。

Oracle 機能と参照アーキテクチャが実現するサービス・レベルについて、詳しくは MAA ベスト・プラクティスに関するホワイト・ペーパー『Oracle MAA リファレンス・アーキテクチャ』を参照してください。

データベース統合の計画

統合データベースの HA の計画における最初のステップは、すべてのデータベースの HA の計画と同じ場所から始まります。統合の候補である各データベースのデータ損失に対する許容性（リカバリ・ポイント目標、つまり RPO）および停止時間に対する許容性（リカバリ時間目標、つまり RTO）を評価するためのビジネス影響分析が実行されます。この分析ではまた、あるデータベースの使用不可状態またはデータ損失が、サポート対象のアプリケーションに効果的にサービスを提供する他のデータベースの機能に影響を与えるという、2 つ以上のデータベース間に存在する依存関係もすべて識別されます。

HA層への統合のグループ候補

2 番目のステップでは、各データベースの RTO および RPO に基づいて、互いに統合の対象になるデータベースのセットを特定します。Oracle MAA のベスト・プラクティスでは、前の項で説明されている HA 層の標準セットに従って、同様の RTO および RPO を持つデータベースを統合します。データベース間に依存関係が存在する場合は、各データベースが、もっとも厳格な HA 要件を持つデータベースに適した HA 層に割り当てられることに注意してください。

RTO および RPO の要件に従ってデータベースを標準の HA 層にグループ化するプロセスでは、次の 3 つの目標が達成されます。

» 制限された HA 層のセットに基づいた標準化による複雑さの軽減と規模の経済利益の実現。

» HA のインフラストラクチャやプロセスにおける過剰な投資または不必要な複雑さを避けることによる効率的な統合。たとえば、バックアップのリストアによって実現できる RTO および RPO を持つデータベースを、リアルタイムの HA および DR のために追加の HA インフラストラクチャが必要な RTO および RPO を持つ別のデータベースと統合することは非効率的です。

» DBaaS のためのサービス・カタログの HA およびデータ保護コンポーネントの確立。サービス・カタログでは、IT 組織がそのユーザー・コミュニティ（開発者、アーキテクト、およびエンドユーザー）に

  - データベース・サービスの配信および管理方法に関する詳細とともに提供するサービスを記述します。

統合手法の選択

各 HA 層の参照アーキテクチャでは、すべての統合手法と展開モデルがサポートされます。ただし、スタンドアロン・データベースを統合環境に移行するための詳細なベスト・プラクティスは、使用されている統合手法によって影響を受けます。

» VM の展開モデルを使用したサーバー統合は、データベースの観点から見ると、複数の独立した Oracle Database の異種のマシンから 1 台の物理マシンへの移行に似ています。この点で、このホワイト・ペーパーに含まれている HA ベスト・プラクティスは物理環境と仮想環境に等しく適用されます。ただし、VM では容量計画、パフォーマンス、システム管理、および HA に対する考慮事項が追加されるため、複雑さのレベルが増します。これらの考慮事項は、使用されている仮想化テクノロジーのベンダーに固有のものです。VM の HA を提供するため、または VM を統合手法として使用するための詳細なベスト・プラクティスは、このホワイト・ペーパーの対象範囲外です。

» スキーマ統合は、既存のアプリケーションでは実施が困難です。課題として、名前の衝突、セキュリティ上の懸念、スキーマごとのポイント・イン・タイム・リカバリでの困難さ、パッチ適用、クローニングなどがあります。スタンドアロン・データベースをスキーマ統合環境に移行するためのベスト・プラクティスは、このホワイト・ペーパーの対象範囲外です。

» Oracle RAC を Oracle Resource Manager と組み合わせて使用した単純なデータベース統合は、Oracle Database 11g のための Oracle MAA のベスト・プラクティスです。Oracle Database 12c の機能が明示的に示されている場合を除き、このホワイト・ペーパーに記載されている HA 層は Oracle Database 11g と Oracle Database 12c の両方に等しく関連しています。

» Oracle Database 11g のための Exadata 統合のベスト・プラクティスはこのホワイト・ペーパーの対象範囲外であり、『Exadata Database Machine 上でのデータベース統合に関するベスト・プラクティス』で詳細に説明されています。

» Oracle Multitenant は、Oracle Database 12c 以降の最適なデータベース統合手法です。マルチテナント・アーキテクチャは、以前の各統合手法の優れた特徴を併せもち、それぞれに付随するトレードオフはありません。

このホワイト・ペーパーでは、Oracle Multitenant がデータベース統合手法であり、Oracle Database 12c 以降で利用されることを前提にしています。Oracle MAA のベスト・プラクティスは物理環境または仮想環境が等しく関連していますが、VM に固有の追加の考慮事項は考慮されていません。

ハードウェア・プールを使用したHA層の調整

Oracle Database を仮想化し、Oracle Multitenant を使用して統合するプロセスは、大まかに言うと、各データベースを PDB に変換し、それらを特定の HA 層の 1 つ以上の CDB に統合するという単純な操作です。最終的には、各層内で統合されるデータベースに必要となるサービス・レベルに対して十分な容量をプロビジョニングするために必要な分析が必要です。容量計画を実施する一環として、各層内に展開される CDB の個数という点から最終的な統合密度を予測する必要があります。特定の層内のすべての PDB が同様のサービス・レベル要件を持っている場合でも、HA 層内に複数の CDB を展開することにはいくつかの理由があります。

» 基盤となるハードウェアに、特定の HA 層に割り当てられたすべての PDB を 1 台のサーバーまたはサーバーのクラスタに統合することを妨げる容量制限が存在する場合があります。パフォーマンス要件を評価し、データベース統合の候補として、あるデータベースの適合性を判定したり、統合しようとしている既存の管理対象サーバーを統合先のサーバーに適合させたりする必要があります。

» 多くの場合は、Oracle Database の複数のリリースやパッチセットを一度にサポートすることが必要になります。たとえば、ある CDB が Oracle Database 12c Release 1（12.1.0.1）で、2 番目の CDB は Oracle Database 12.1.0.2 である場合があります。これにより、他の 12.1 の PDB をアップグレードできない場合は、アップグレードを実施するために、個々の PDB を Oracle 12.1.0.1 の CDB から'アンプラグ'して Oracle 12.1.0.2 の CDB に'プラグイン'することができます。この状況は、たとえば、新しいリリースをサポートするベンダー提供アプリケーションの遅延が原因で発生する場合があります。

» マルチテナント・アーキテクチャの経験を積むために、慎重に、より限られた数の PDB で B を展開する方がよい場合があります。PDB をある CDB からアンプラグして別の CDB にプラグインするだけで統合密度を容易に向上させることができることを考えれば、最終的に各 HA 層内の CDB の数が削減されるため、この戦略に将来の不都合はまったくありません。

また、どのデータベースを最初に Oracle Multitenant に移行するべきかという従来からの疑問もあります。ブロンズ層では、もっとも多数のデータベースが統合の対象になります。オラクルは、少ないリスクで迅速な投資収益率がもたらされることを前提として、ブロンズ層から始めることを推奨します。

表 2 に、統合の計画プロセスの概要を示します。

表 2：統合の計画

統合のためのおもなステップ	推奨事項	利点
1.HAのSLAを標準化し、アプリケーションとそのデータベースを適切なHA層に割り当てる	Oracle MAAのHA層（プラチナ、ゴールド、シルバー、ブロンズ）に従います。	標準化によって運用コスト（OPEX）が削減され、リスクが低減されます。
2.SLAを満たすためのHAアーキテクチャを設計し、構成および運用のベスト・プラクティスを決定する	各HA層の推奨されるHA参照アーキテクチャに従います。 Oracle DatabaseのためのOracle MAAの構成および運用のベスト・プラクティス（www.oracle.com/goto/maa）を使用します。	OPEXが削減されます。 全体的な安定性と可用性が向上します。 Oracle MAAで検証された構成のベスト・プラクティスによってリスクが軽減されます。
3.ハードウェア・プラットフォームの数、およびOracle Databaseプラットフォームのオペレーティング・システムの数を削減する	LinuxまたはSolaris x86-64には、Exadata Database Machineが、最大の統合密度を実現するための標準のスケーラブルなデータベース・プラットフォームとなります。 より小規模な環境には、Oracle Database Applianceが、標準プラットフォームとなり、高い統合密度が提供されます。 SPARCおよびSolaris、または汎用データベースやアプリケーション処理には、Oracle SuperCluster（ExadataストレージおよびExadata機能を含む）を使用します。 Oracle Virtual Compute Applianceは、すべてのLinux、Oracle Solaris、またはMicrosoft Windowsアプリケーションのための仮想インフラストラクチャの展開用に最適化されています。	Oracle MAAのベスト・プラクティスは、エンジニアド・システム・プラットフォーム用に最適化され、事前に構成されています。 設定時間およびOPEXが削減されます。 システムとソフトウェアの両方の統合されたサポートによってOPEXが削減されます。 ハードウェア・プラットフォーム数が少ないため、CAPEXが削減されます。 オラクルのリソース管理および圧縮テクノロジーと組み合わされたExadataソフトウェアは、データベース統合用に最適化されています。 高い統合密度によって、総ライフ・サイクル・コストが削減されます。
4.データベース統合のターゲットであるハードウェア・プールを確立する	適切なサイズのハードウェア・プールを作成します。標準的なハードウェア・プールとして、ハーフ・ラックから、最大2つのフル・ラック間システムを推奨します。 各ハードウェア・プールをHA層に割り当てます。各層はHAアーキテクチャが異なるため、ハードウェア・プールごとに存在するHA層は1つだけです。	システムのスプロール化や多様性に伴う複雑さが軽減されます。 OPEXが削減されます。
5.標準イメージ・ソフトウェア・スタックを選択する	内部の検証の後、最新のOraclepatchセットを2つまたは3つ選択します。例として、推奨されるPSUまたはバンドル・パッチを含む11.2.0.3、11.2.0.4、12.1.0.1などがあります。 その他のバリアントは、データベース・パッチセット・リリースあたり1つしか選択しできません。 1つのOracle Grid Infrastructureと最大5つのOracleホームが推奨されます。	ソフトウェアのスプロール化が削減され、リスクが低減されます。OPEXが削減され、全体的な安定性と可用性が向上します。
6.データベース統合手法を選択する	データベース統合のもっとも効率的な形式として、Oracle Database 12cのマルチテナント・アーキテクチャを使用します。 厳密な分離のために専用リソースが必要であり、その結果としてのOPEXおよびCAPEXコストの増加が許容可能な場合は、Oracle VMやOracle Solaris Zonesなど、オペレーティング・システムの仮想化を使用します。	OPEXおよび資本支出が最大限節減されます。

7.システムのサイジング、リソース要件、およびパフォーマンス予測	各アプリケーションの現在と将来のCPU、I/O、メモリ、およびストレージ容量の消費を評価します。 EMCC Consolidation Planner（ドキュメントおよびデモンストレーションを参照）を使用するか、またはサイジング・サービスについてOracle ConsultingまたはOracle Advanced Customer Support Services（Oracle ACS）にお問い合わせください。	移行後のCAPEXおよびOPEXが削減されます。 データは、データベースをハードウェア・プールに割り当てたり、リソース・マネージャの割当ておよび制限を構成したりするために使用されます。
8.データベースを、HA層が同じである適切なハードウェア・プールに割り当てる	アクティブなデータベースをすでにホストしているハードウェア・プールに統合する場合は、現在のCPU、I/O、メモリ、およびストレージの消費を監視して、その使用可能な容量を把握します。 ハードウェア・プールに予備容量がない場合は、次のデータベースを新しいハードウェア・プールに割り当てます。その重要度とリソース消費の理由から、ハードウェア・プール内に1つのデータベースしか存在しない場合もあります。 プロビジョニングのためにEM12cを選択します。	
9.リソース割当ておよび制限を構成する	リソース管理のベスト・プラクティスを使用して、各データベースやPDBのリソースを保証および制限します。 分離のために専用リソースが必要なデータベースの場合は、PROCESSOR_GROUP_NAMEパラメータを使用して、そのインスタンスを専用のCPUまたはNUMAノードにバインドすることを検討してください。 詳しくは、このホワイト・ペーパーの「リソース管理」の項を参照してください。	効率的にリソースが使用されます。
10.監視およびセルフサービス・インフラストラクチャを展開する	Oracle Enterprise Manager Cloud Control 12c	OPEXが削減され、サービスが向上します。

マルチテナント・アーキテクチャへの移行

統合の計画を完了し、データベース統合の候補を適切なハードウェア・プールおよび HA 層に割り当てたら、次のステップでは既存のデータベースをマルチテナント・アーキテクチャに移行します。使用される方法は、次のものに依存します。

» ソース・データベースのタイプ（非 CDB、CDB、または PDB）

» ソース・データベースのバージョン

» ソースとターゲットのハードウェア・プラットフォームおよびデータベースの HA 要件

この項では、データベースをマルチテナント・アーキテクチャに移行するための戦略について説明します。汎用的な移行オプションについては「データベースのプラットフォームまたはロケーションの移行」を参照してください。また、マルチテナント以外のデータベースへの Exadata の移行のベスト・プラクティスについては、『Exadata MAA ベスト・プラクティス Oracle Database の移行』を参照してください。

Oracle Database 12c のリリースには、次の 3 つのデータベース・タイプがあります。

» 非コンテナ・データベース（非 CDB）。これは、マルチテナント・コンテナ・データベース（CDB）またはプラガブル・データベース（PDB）以外のすべてのデータベースです。これには、Oracle 12c リリースより前のすべてのデータベース、Oracle 12c にアップグレードされたがまだ PDB に移行されていない、Oracle 12c より前のすべてのデータベース、および CDB または PDB のどちらとしても作成されていない、Oracle 12c で作成されたすべてのデータベースが含まれます。

» マルチテナント・コンテナ・データベース（CDB）。CDB は、Oracle Database 12c で ENABLE_PLUGGABLE_DATABASE 初期化パラメータを TRUE に設定して作成されたデータベースです。CDB は、0 個、1 個、または多数の PDB のコンテナとして作成されます。CDB を使用してスキーマ・オブジェクトを作成したり、ユーザーSQL を処理したりすることはできないため、非 CDB を CDB に、またはその逆方向に変換することはできません。オラクルでは、新しい Oracle Database 12c データベースをすべて CDB として作成し、アプリケーション・データを PDB に格納することを推奨しています。

» プラガブル・データベース（PDB）。これは、アプリケーションには個別のデータベースとして見える移植可能なスキーマ、スキーマ・オブジェクト、および非スキーマ・オブジェクトの集合です。非 CDB は PDB に変換できますが、PDB を非 CDB に変換することはできません。

推奨される移行方法

マルチテナント・アーキテクチャへの移行オプションは、ソース・データベースのバージョンや、このホワイト・ペーパーで先に説明した、必要とする HA 層に大きく左右されます。表 3 に、移行オプションの概要を示します。My Oracle Support Note 1576755.1『Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage』も参照してください。

表 3：マルチテナント・アーキテクチャへの移行戦略

HA層/ソース	移行方法	考慮事項	Data Guardスタンバイの影響	予測停止時間
シルバー、ゴールド、またはプラチナ 任意のデータベース・タイプ、データベース・バージョン、プラットフォーム	GoldenGate	GoldenGateのインスタンス化の前に、PDBをシードから事前に作成する必要があります。 Data Pumpまたは論理的なインスタンスを使用している場合は、オブジェクトを（圧縮などのために）再編成および再最適化できます。 高速フォールバック・オプションを使用できます。	プライマリ上でのOracle GoldenGateオブジェクトのインスタンス化には、スタンバイCDBを維持するための追加の手順が必要になることがあります。	ほぼゼロ（手動のスイッチオーバー操作が前提）
シルバー Oracle Database 12c 非CDBおよび エンディアンの一致するプラットフォーム	Oracle RMAN Active Database Duplication HA/DR	describeプラグイン/非CDBプラグインの実行。増分ファイル移行がサポートされます。ステージング領域は必要ありません。 これは物理的な移行であるため、オブジェクトは（圧縮などのために）再編成または再最適化されません。	ファイルをスタンバイCDBにコピーします。 MRPがファイルを見つけると、MRPは処理を続行します。	45分（最後の増分の時間や、noncdb_to_pdb.sqlの実行に必要な時間によって影響を受ける）
ブロンズまたはシルバー Oracle Database 11.2.0.3以降のすべて すべてのプラットフォーム	全データベースTTS/Data Pump	Data Pumpのインポートの前に、PDBをシードから事前に作成する必要があります。 増分ファイル移行がサポートされます。 増分バックアップのためのステージング領域が必要です。 これは物理的な移行であるため、オブジェクトは（圧縮などのために）再編成または再最適化されません。	ASMCMD cpまたはホストベースのコピー・コマンドを使用して、ファイルをスタンバイにコピーします。 データファイルの処理については、Data Guardのドキュメントに従います。	2時間（Data Pumpのメタデータ・インポートに必要な時間によって影響を受ける）
ブロンズ すべてのOracle Databaseリリース すべてのプラットフォーム	トランスポータブル表領域（TTS）およびDataPumpのエクスポート/インポート データをコピーまたは抽出してロードするカスタム・スクリプト	Data Pumpのインポートの前に、シードのPDBを事前に作成する必要があります。 Data Pumpまたは論理的なインスタンスを使用している場合は、オブジェクトを（圧縮などのために）再編成および再最適化できます。	N/A	エクスポートおよびインポートするサイズと時間に依存 数時間～数日

トランスポータブル表領域および Data Pump を使用して、Oracle Database 12c より前の Oracle Database リリースから PDB に直接移行することは可能ですが、最初にデータベースを Oracle Database 12c にアップグレードする方法を推奨します。Oracle Database 12c ソースからマルチテナント・アーキテクチャに変換することによる移行の利点には、次のものが含まれます。

» 簡便性。手順が少なくなり、また多くのツールを使用できるため、プロセスが簡素化されます。

» マルチテナント・アーキテクチャに移行する前に、Oracle Database 12c データベースのパフォーマンスと機能を検証できます。

最初にデータベースをアップグレードすることを選択する場合は、Oracle Database のバージョン 10.2.0.5、11.1.0.7、および Oracle 11g Release 2 のバージョン 11.2.0.2 以降を Oracle Database 12c に直接アップグレードできることに注意してください。その他のすべてのバージョンでは、最初に

データベースを上に示した最小暫定リリースのいずれかにアップグレードすることにより、2 回のアップグレードが必要です。任意のプラットフォームでの旧リリースから Oracle Database 12c へのアップグレードについて詳しくは、My Oracle Support Note 1462240.1『Oracle Database 12c Upgrade Companion』を参照してください。Exadata システムで Grid Infrastructure と Oracle Database を 12.1.0.2 にアップグレードする場合、My Oracle Support Note 1681467.1 を参照してください。

ファイル移動

データベースのあるデータセンターから別のデータセンターへの移動、またはあるシステムから別のシステムへの移動は、データベース・サイズ、ネットワーク帯域幅、ソースとターゲットの両方で使用可能なシステム・リソースなどを考慮すると、難しい場合があります。可能な場合は、次のいずれかのオプションを使用することにより、ファイル移動のオペレータや停止時間の影響を緩和してください。

» ファイルをプラグインの前にターゲットの場所にコピーします。Oracle Database 12c より、Oracle RMAN の複製コマンドは CDB と PDB に対応しています。付録 B には、Oracle RMAN の複製を移行に使用した例が示されています。その他の移行のユースケースの詳細および例については、My Oracle Support Note 1576755.1『Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage』を参照してください。

  さらに、最初の複製に Oracle RMAN バックアップ増分を適用して、全体的な停止時間を最小限に抑えることもできます。一部のオプションには、Oracle RMAN バックアップ増分の使用、および同じソースやターゲット・プラットフォームへの適用や、クロス・プラットフォーム・トランスポートのための『非一貫性バックアップを使用した表領域のクラス・プラットフォーム・ トランスポートの実行』で説明されている手順の使用が含まれています。Oracle Database 12c より前のリリースでは、My Oracle Support Note 1389592.1『Reduce Transportable Tablespace Downtime using Cross Platform Incremental Backups』の手順を使用できます。最後の増分を適用するには、アンプラグ/プラグ操作のために追加の時間が少し必要です。ただし、ソース・ファイルは変更されないまま残っているため、問題が発生した場合はすぐにフォールバックが可能です。

» ソース環境と宛先の環境の両方からデータファイルにアクセスできるようにストレージを構成します。ソース・データファイルが、宛先の CDB からアクセスできる共有ストレージ上に存在する場合は、ファイルを移動する必要がないため、プラグイン・コマンドを簡素化できます。所定の場所にあるソース・ファイルの使用により、もっとも高速なアンプラグ/プラグ操作が可能になります。ただし、これによってソース・ファイルが変更されます。以前の環境へのフォールバックが必要な場合は、データベースのリストアとリカバリか、または少なくとも、データ損失なしの Oracle RMAN の SWITCH TO COPY を使用する必要があります。

これらのいずれのオプションでも、プラグイン文で NOCOPY 句を指定できます。

MultitenantとData Guard

運用

Data Guard のフィジカル・スタンバイ・データベースと REDO Apply では通常、新しい PDB のデータファイルがスタンバイ・サイトに事前にコピーされており、プライマリ・データベースから受信する REDO をすぐに適用できる状態であることが前提となっています。またスタンバイ・データベースでは、CREATE PLUGGABLE DATABASE 文のファイル名変換仕様がすべて無視されます。場所とファイル名の指定には、スタンバイ・データベースの DB_CREATE_FILE_DEST および

DB_FILE_NAME_CONVERT の初期パラメータ設定のみが使用されます。

これらの機能の微妙な差違については、次の 2 つの My Oracle Support Note を参照してください。発生する可能性のある停止時間を短縮したり、新しい PDB のディザスタ・リカバリ保護に対する影響を軽減したりすることができます。

- My Oracle Support Note 1576755.1：Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage

  前述のとおり、この Support Note では、実質的に最小限の停止時間で非 CDB からマルチテナントに移行する方法を説明しています。また、Data Guard 環境のプライマリ CDB で新しい PDB が作成された直後にフィジカル・スタンバイ・データベースでも PDB が完全に作成され、ディザスタ・リカバリ機能をすぐに利用できることを確認する方法についても、詳しく説明しています。

  ここに記載されている方法は、ディザスタ・リカバリ保護がすぐに必要であり、次のいずれかの条件に当てはまる場合に最適です。

  - ソース・データベースが非 CDB であり、12c データベースか 12c より前のリリースである。
  - ソース・データベースが、あるコンテナ・データベースからアンプラグされ、別のコンテナ・データベースにプラグインされた PDB である。

- My Oracle Support Note 1916648.1：Making Use of the STANDBYS=NONE Feature with Oracle Multitenant

  この Support Note では、CREATE PLUGGABLE DATABASE 文での STANDBYS=NONE 句の使用（PDB のリカバリの延期）について説明しています。CREATE PLUGGABLE DATABASE 文で STANDBYS=NONE 句を使用すると、すべてのスタンバイで PDB が作成されますが、REDO Apply を実行できるだけの PDB しか作成されません。新しい PDB のファイルは OFFLINE/RECOVER とマークされ、その PDB でさらに REDO を適用しようとしても無視されます。将来的には、PDB ファイルが任意またはすべてのフィジカル・スタンバイ・データベースにコピーされ、PDB のリカバリが有効になり、それ以降は REDO が PDB に適用されるようになると期待されています。PDB の作成時には、スタンバイ・データベースのサブセットに PDB を作成するように指定できません。

  CREATE PLUGGABLE DATABASE 文に STANDBYS=NONE が含まれると、すべてのフィジカル・スタンバイで新しい PDB のリカバリが遅れ、リカバリを有効にしたいスタンバイごとにリカバリ・プロセスの実行を有効にする必要があります。スタンバイ・データベース上の既存の PDB はすべて、現在のリカバリ・ステータス（DEFERRED または ENABLED）のままとなります。

  この Support Note では、PDB のリカバリを有効にする手順と、PDB のリカバリ遅延が発生した場合に Data Guard のロール移行がどのように動作するかについて説明しています。

ここに記載されている方法は、迅速なディザスタ・リカバリ保護が不要であるか、次のいずれかの条件に当てはまる場合に最適です。

- リモート・クローニング（CREATE PLUGGABLE DATABASE PDBCLONE FROM PDBSOURCE@DB_LINK）またはマニフェスト・ファイル(CREATE PLUGGABLE DATABASE PDBCLONE AS CLONE USING 'manifest.xml'）の使用により、ソース PDB が別のコンテナ・データベースからクローニングされている。
- ソース PDB とクローン PDB が同じコンテナ・データベースに存在するが、Active Data Guard のライセンスがない。

PDB の作成、クローニング、プラグイン時のスタンバイ・データベースの動作

次の項目では、新しい PDB、プラグイン PDB、またはクローン PDB として PDB を作成する際に、スタンバイ・データベースで発生するさまざまな動作について説明します。詳しくは、My Oracle Support Note 2049127.1『Data Guard Impact on Oracle Multitenant Environments』を参照してください。

- シードからの新しい PDB の作成

  空のプラガブル・データベースを新規作成すると、各フィジカル・スタンバイ・データベースに自動的にレプリケートされ、REDO Apply によってスタンバイ・サイトの PDB$SEED ファイルがコピーされて、新しい PDB が作成されます。

  このような場合は、リカバリを延期しないことを推奨します。

- 同一コンテナ・データベース内での既存の PDB から新しい PDB へのクローニング

  スタンバイ・データベースで Active Data Guard を実行している場合の動作（REDO Apply の実行時にはスタンバイ・データベースが読取り専用で開きます）は、Active Data Guard を使用していない場合と異なります。

  - プライマリでの同一コンテナ内の PDB のクローニング：

    - Active Data Guard を実行するフィジカル・スタンバイ・データベースでは、スタンバイ・サイトのソース・ファイルがコピーされ、スタンバイに自動的に PDB が作成されます。

      このような場合は、リカバリを延期しないことを推奨します。

    - Active Data Guard を実行しないフィジカル・スタンバイ・データベースでは現在のところ、REDO Apply プロセスによってソース PDB ファイルを直接コピーすることはできません。ファイルの整合性を保証できないためです。

このような場合は、CREATE PLUGGABLE DATABASE 文で STANDBYS=NONE 句を使用して、PDB のリカバリを延期することを推奨します。Note 1916648.1 のような方法で、新しいプラグイン PDB のデータファイルをプライマリ CDB からスタンバイ・データベースにコピーしておけば、将来のいずれかの時点で新しい PDB をリカバリできます。

- 別の CDB からの PDB のプラグイン（クローニングではない）

REDO Apply では"所定の場所"（DB_FILE_CREATE_DEST または DB_FILE_NAME_CONVERT に由来する場所）での検索はできますが、その対象は REDO の適用先の結果ファイルのみであり、ソース・ファイルは対象外です。これらのファイルは、プラグイン時にプライマリで確認されたときと完全に同じ状態であることが必要です。つまりこれらのファイルは、プライマリでの PDB プラグインより前に、ソース・データベースからスタンバイ・データベースの所定の場所にコピーされている必要があります。REDO Apply では、データファイルの所定の場所の調査とファイル・ヘッダー情報の検証によって、ファイルが正しいことが確認されます。Oracle Managed Files の場合、REDO Apply は所定の場所のディレクトリを検索し、（ファイル・ヘッダー情報を照合して）ファイル名は無視するようにコーディングされています。

このような場合は、リカバリを延期しないことを推奨します。新しい PDB のデータファイルを所定の場所に事前にコピーしてから、プライマリ CDB でプラグイン文を実行する必要があります。こうしておけば、REDO Apply の実行時に Note 1576755.1 のようなプロセスでデータファイルをスタンバイ・データベースに取り込むことができます。

- マニフェスト・ファイルを使った別の CDB からの PDB のクローニング

マニフェスト・ファイルを使った PDB のクローニングは（前項の）PDB へのプラグインと似ていますが、違いもあります。このプロセスでは新しい PDB が作成され、その PDB の新しい GUID が生成されます。新しい PDB の作成前に、その GUID を決めることはできません。つまり、Oracle Managed Files/Oracle Automatic Storage Management（Oracle ASM）の構成で、前述のように PDB のデータファイルをスタンバイの"所定の場所"に事前にコピーしておくことはできません。GUID は、所定の場所のディレクトリ構造の一部であるためです。

このような場合は、CREATE PLUGGABLE DATABASE 文で STANDBYS=NONE 句を使用して、PDB のリカバリを延期することを推奨します。Note 1916648.1 のような方法で、新しい PDB のデータファイルをプライマリ・データベースからスタンバイ・データベースにコピーしておけば、将来のいずれかの時点で PDB をリカバリできます。

- データベース・リンクによるリモート PDB クローンの実行

データベース・リンクによるリモート PDB クローンの実行は、マニフェストによるクローニングと似ています。Oracle Managed Files/Oracle ASM 構成の場合、新しい PDB の GUID

を事前に決めることはできないため、データファイルを"所定の場所"に事前にコピーしておくことはできません。

このような場合は、CREATE PLUGGABLE DATABASE 文で STANDBYS=NONE 句を使用して、PDB のリカバリを延期することを推奨します。Note 1916648.1 のような方法で、PDB のデータファイルをプライマリ・データベースからスタンバイ・データベースにコピーしておけば、将来のいずれかの時点で PDB をリカバリできます。

移行

Data Guard を含む事前構成されたゴールドまたはプラチナ・アーキテクチャへの移行を実行する場合は、プラグイン・プロセスの一部として、データファイルのコピーがスタンバイの場所で確実に使用できるようにすることも必要です。メディア・リカバリでファイルを自動的に見つけ、中断なしで REDO の適用を続行できるようにするには、プラグイン操作を実行する前に、これらのファイルがスタンバイ上の場所に存在する必要があります。Oracle Database は、DB_FILE_NAME_CONVERT の設定を使用するか、または DB_FILE_CREATE_DEST ディレクトリでファイルを自動的に検索します。

同じ CDB への移行を複数回実行する予定がある場合は、プロセスを簡素化するために、スタンバイ・サイトへの REDO 転送を遅延させ、すべての移行が完了した後にスタンバイを再インスタンス化することを推奨します。これにより、プラグイン操作のたびにコピーを実行するのではなく、すべてのファイルをスタンバイ・サイトに 1 回のステップでコピーできるようになります。このオプションのデメリットは、PDB スタンバイのインスタンス化が完了するまでプライマリ・データベースが完全には保護されないことです。

Oracle Resource Management

ハードウェア・プール内の CPU、I/O、およびメモリ・リソースを取得するために、すべてのデータベース、CDB、および PDB が競合します。この項では、統合されたマルチテナント・アーキテクチャ環境内のリソースを計画、構成、および管理する方法について説明します。Oracle Database の既存の管理ツールはすべてマルチテナント・アーキテクチャに対応しているため、これを使用して、データベースや PDB に十分なリソースが確実に保証されるようにしてください。また、これらのツールが他のアプリケーションに悪影響を与えることはありません。これは、顧客が決まった量のシステム・リソースへの特定のサービスまたはアクセスに対して対価を支払っている DBaaS 環境で特に関係があります。

リソース管理ツールは、Exadata I/O リソース・マネージャとネットワーク・リソース・マネージャを除き、非 Exadata システムと Exadata システムの両方で使用できます。最新の推奨されるパッチ、監視スクリプト、およびステップ・バイ・ステップの実装ガイドについては、MOS Note 1339769.1 『Master Resource Manager』を参照してください。

計画

パフォーマンスに関する統合計画の最初のステップは、データベースのパフォーマンス要件を決定することです。図 5 に、プラチナ、ゴールド、シルバー、およびブロンズの各パフォーマンス分離層を示します。リソース管理を使用したパフォーマンス分離のレベルが低くなるに従って、統合密度は向上します。

図 6：統合とパフォーマンス分離層

一般に、ゴールド層のパフォーマンス要件を持つデータベースまたは PDB は HA サービス要件もゴールド層になりますが、これを厳密に適用する必要はありません。より高いパフォーマンス分離層にあるデータベースは、保証されたリソースのより高いシェアを要求します。CPU 用の CDB Resource Management や Exadata IORM などのリソース管理ツールでは、指定された量のリソースを各データベースに保証しながらリソースを共有できます。これらのツールの場合、使用可能なリソースが絶えず変化するシステムでは、リソース割当てが少ない、低いパフォーマンス分離層にあるデータベースや PDB のパフォーマンスが変動する可能性があります。より一貫性のあるパフォーマンスを提供するには、リソース・マネージャの制限を使用してリソースに上限を設定します。リソースの制限は、支払が発生する DBaaS 環境においては、顧客により高いパフォーマンス層へのアップグレードを促す効果的な方法でもあります。

もっとも単純な計画が最適な計画になる傾向があります。

» 同じハードウェア・プール内で実行されているすべてのデータベースと CDB を、同じ HA サービスおよびパフォーマンス分離層（ブロンズ、シルバー、ゴールドまたはプラチナ）に含めてください。さらに、CDB 内の各 PDB にも、同じ HA サービス・レベルおよびパフォーマンス分離層を割り当ててください。

» インスタンス・ケージング、メモリ・パラメータ、および Exadata IORM を使用して、ハードウェア・プールを共有するデータベースと CDB を管理します。

» 異なるデータベース・バージョンが必要な場合は個別の CDB を作成しますが、運用コストを削減するために、Oracle データベース・ホームの数を最大 5 個に制限します。

» OLTP やデータウェアハウスなどのアプリケーション・タイプごとに個別の CDB を作成します。OLTP とデータウェアハウスのリソース要件は大きく異なる傾向があるため、このグループ化によって CDB の管理が容易になります。たとえば、OLTP データベースが大きなバッファ・キャッシュやフラッシュ・キャッシュからメリットが得られるのに対して、データウェアハウスは高いディスク I/O スループットからメリットが得られます。

» Oracle Exadata Database Machine を使用している場合は、各アプリケーションをフラッシュ・キャッシュからメリットが得られる CDB と、それなしでも許容可能な性能が得られる CDB に分離します。これにより、Exadata Smart Flash Cache または Smart Flash Log を利用する必要のあ

る OLTP データベース（PDB）のフラッシュ・ヒット率が向上します。たとえば、より速い応答時間を許容できる多くのテストおよび開発データベースを、Exadata IORM プランによってフラッシュ・キャッシュが無効になっている CDB にプラグインし、I/O 集中型アプリケーションの重要なデータベースを、Exadata IORM プランによってフラッシュが有効になっている個別の CDB にプラグインできます。もう 1 つの例として、フラッシュ・キャッシュは必要としない可能性があるが、Exadata のスマート・オフロード機能やストレージ・インデックスからもっとも大きなメリットが得られる純粋なデータウェアハウスまたは分析データベースがあります。

構成に関するこの追加の推奨事項は、データベースやアプリケーションのパフォーマンス要件が大きく変動する可能性のあるブロンズ層にもっとも関連しています。すべてのアプリケーションが重要であり、かつパフォーマンスの影響を受けやすい上位の層の場合、Exadata Smart Flash Cache は、すべてのアプリケーションの I/O リクエスト（ログ書込み、OLTP または小さな I/O 読取り/書込み、および表スキャン）を動的に調整します。

» CDB を共有する PDB を管理するには、最初から CDB Resource Management を使用します。これにより、ハードウェア・プール内で数十または数百のデータベースが実行されている場合でも、顧客が同じリソース・セットにアクセスし、一貫性のあるパフォーマンスが得られることが保証されます。最初から使用することにより、CDB リソース・プランで、各 PDB に保証された CPU およびディスク I/O リソースを割り当てることができます。さらに、CDB リソース・プランでは、PDB が使用できる CPU またはディスク I/O の量に強い制限を設定できます。これは、顧客がパフォーマンスに対して対価を支払うマルチテナント・アーキテクチャ環境で有効です。

» マルチテナント・アーキテクチャでの体験と専門知識を得るには、潜在的な統合密度がもっとも高いブロンズ層に重点を置き、1 回につきデータベースの小さなバッチを段階的に導入します。

リソースの制御

表 4 に、統合環境またはマルチテナント・アーキテクチャ環境で共有リソースを制御するための推奨事項の概要を示します。これらの推奨事項は、すべての HA サービス・レベルおよびパフォーマンス層で適用できます。各層に固有の推奨事項は、その後の表に示されています。

表 4：リソース制御のためのガイドライン

リソース	ガイドライン
メモリ	» HA層（ブロンズ、シルバー、ゴールド、プラチナ）には関係なく、メモリが決してオーバーサブスクライブされないようにしてください。データベースをハードウェア・プールに移行する前に、容量計画と傾向分析を実行するようにしてください。 » 目標または現在の共有メモリ要件に基づいてHugePagesを設定します。 » PGA_AGGREGATE_LIMIT初期化パラメータを無効にしないでください。PGA_AGGREGATE_LIMITは、PGAメモリ使用量に強い制限を適用するものであり、パフォーマンス低下やインスタンス排除につながる可能性のある余分なページングを回避するために重要です。この値を減らす場合は、必ずPGA_AGGREGATE_TARGETも減らしてください。このパラメータは動的に調整できます。 » 次の計算式を使用して、データベースの初期フットプリントを計算します。表5の各パフォーマンス層のベスト・プラクティスに従い、メモリがオーバーサブスクライブされないようにしてください。実際のピーク・メモリ使用量に対するPGAおよびプロセスごとのメモリ統計を取得できたら、それらの数値をこの初期の近似の代わりに使用できます。 » OLTPデータベース：SGA_TARGET + PGA_AGGREGATE_TARGET + 4MB * （最大のPROCESSES） » DW/BIデータベース：SGA_TARGET + 3 * PGA_AGGREGATE_TARGET » LOG_BUFFERを64ビット・システムでの最大値である256MBに設定します。 » 参考資料： » HugePagesについて詳しくは、MOS 361468.1およびMOS 401749.1を参照してください。

CPU	インスタンス間のCPU競合を回避するために、同じサーバー上で実行されているデータベースとCDBをインスタンス・ケージングします。インスタンス・ケージングは、リソース集中型の問合せやその他のワークロードの急増によって、使用可能なすべてのシステム・リソースが消費されてしまわないようにします。 » 表5の各パフォーマンス層のベスト・プラクティスに従って、インスタンス・ケージングを構成します。 » インスタンス・ケージングを構成するには、'spfile'内のCPU_COUNTを、インスタンスが使用できるCPUスレッドの最大数に設定します（最小は2）。次に、有効なRESOURCE_MANAGER_PLAN（非CDBの場合のDEFAULT_PLANや、CDBの場合のDEFAULT_CDB_PLANなど）を設定することによってインスタンス・ケージングを有効にします。リソース要件が変更された場合は、CPU_COUNTパラメータを動的に変更できます。これにより、インスタンス・ケージングのサイズがただちに調整されます。 » 参考資料 » My Oracle Support Note 1362445.1 – 『Configuring and Monitoring Instance Caging』 » OSセマフォの構成に関連した一般的なベスト・プラクティスについては、『Exadata Database Machine上でのデータベース統合に関するベスト・プラクティス』を参照してください（Exadataのみ）。 » CDBを共有するPDB間のCPU使用量や競合を管理するには、CDBリソース・マネージャを使用します。CDBリソース・プランでは、PDBごとのSHARESおよびUTILIZATION_LIMITディレクティブを次のように構成できます。 » SHARESを使用して'公平性'を構成します。シェアは、PDBの相対的な重要性を表します。シェアが多いPDBは、より多くのCPUを使用することを許可されます。すべてのPDBが等しい場合は、それらのシェアを同じ値に設定するか、またはDEFAULT_CDB_PLANを使用します。あるPDBが別のPDBより2倍重要である場合は、そのシェアを2倍にします。"デフォルト・ディレクティブ"を使用すると、各PDBにデフォルトのシェア数を与えることができます。 既存のデータベースをCDBに移行する場合は、同等のCDBリソース・プランを作成するために、個々のデータベースのCPU_COUNTをSHARESに変換します。 » 'パフォーマンスに対する支払い'を構成するには、UTILIZATION_LIMITSを使用します。パブリック・クラウドでは、PDBが使用できるCPUの量をサブスクリプション率に基づいて制限します。インスタンス・ケージングによって、あるデータベースがアイドル・システム上のすべてのCPUを利用してしまわないようにするのと同様に、この制限は"ブロンズ"のPDBによってアイドル・システム上のすべてのCPUが利用されてしまわないようにします。PDBがプラグインされた時点からDBRMの'使用率制限'を設定すると、予測が最初から正しく設定されるようにするのに役立ちます。 » CDBに対してインスタンス・ケージングが有効になっている場合は、インスタンス・ケージングのサイズ（つまり、CPU_COUNT）に基づいて使用率制限を適用することにより、PDBが利用できるCPUの最大量を決定します。 » Oracle RACデータベースでは、すべてのインスタンスに対して同じCDBリソース・プランを設定する必要があります。そうしないと、パラレル・ステートメント・キューイングとExadata IORMの両方が予測不可能な方法で動作します。 » 参考資料 » My Oracle Support Note 1567141.1 - 『Migrating Databases using Instance Caging to a CDB』
プロセッサとセッション	ほとんどの場合、PDBはCDBレベルで設定された初期化ファイル・パラメータ設定を使用しますが、いくつかのパラメータはPDBレベルで設定できます。これらの設定はパラメータ・ファイル内には格納されず、CDB内にある内部の表に格納されることに注意してください。変更可能なPDB固有の設定の完全なリストを表示するには、次の文を実行します。 SQL> SELECT NAME FROM V$SYSTEM_PARAMETER WHERE ISPDB_MODIFIABLE='TRUE' ORDER BY NAME; 次のリストは、CDBにPDBを追加するときに変更が必要になるPDBとCDBの両方の設定について説明しています。 » SESSIONS： CDBとそのPDBで使用可能なセッションの数を制御するには、ルート・レベルにある初期化パラメータ・ファイル内にそのCDBのセッションの総数を設定します。各PDBについて、そのPDBが開始できるセッションの数に強い制限を適用するためにSESSIONSパラメータの値を設定できます。初期化パラメータのSESSIONSはCDBの最大値を設定し、PDB内のSESSIONS設定はそのPDBに固有の値です。 » PARALLEL_MAX_SERVERS： PARALLEL_MAX_SERVERSでは、インスタンスのパラレル実行プロセスとパラレル・リカバリ・プロセスの最大数を指定します。要求が増えると、OracleデータベースはPARALLEL_MAX_SERVERSで定義された数まで、パラレル実行プロセス数を増加します。この値を小さくした状態で、各アプリケーションがパフォーマンス要件を満たすことができるようにしてください。PARALLEL_MAX_SERVERSが高すぎる値に設定されていると、ピーク期間中にメモリ・リソースが不足する可能性があり、それによってパフォーマンスが低下し、さらにデータベース・ノードが不安定になる場合があります。 Exadataの使用時： » X2-2、X3-2、X4-2、X5-2、X6-2の場合：すべてのインスタンスおよびPDB <= 240ではsum(PARALLEL_MAX_SERVERS) » X2-8、X3-8、X4-8、X5-8、X6-8の場合：すべてのインスタンスおよびPDB <= 1280ではsum(PARALLEL_MAX_SERVERS) CDBでは、パラレル・サーバーはすべてのPDBによって共有されます。PDBがいつでも使用できるパラレル・サーバーの最大数を制限するには、CDBリソース・プランのPARALLEL_SERVER_LIMITディレクティブを、PDBがいつでも使用できるPARALLEL_MAX_SERVERSの最大の割合に設定します。 » プロセス数およびデータベース・サーバーへの接続数の制限： 適切な数のプロセスを維持することには、メモリ、CPU、およびデータベースのラッチ競合、ログ・ファイル同期の待機、全体的なアプリケーション・フェイルオーバー時間の回避または削減などの多くの利点があります。また、プロセス数や接続数を削減すると、パフォーマンスとスループット、さらにもっとも重要な点としてシステムの安定性の向上にもつながる場合があります。 CPUコア数の最大2倍という控えめなアクティブ・プロセス数を使用し、データベース・ノード全体の合計プロセス数をCPUコア数の最大10～12倍にします。次に示す手法を1つ以上使用することで、全体的なプロセス数を削減してパフォーマンスと安定性を向上させることができます。 » 接続プールを使用し、接続の最大数をアクティブな作業セッションの見積り数を若干上回る値に設定することによって、プロセス数およびデータベース・サーバーへの接続数を制限します。最小接続数と最大接続数を同じ値に設定して動的な接続プールを解消することによって、プロセスのコストのかかる割当ておよび割当て解除を回避します。 注：Exadata上で実行している場合は、共有サーバー/MTS、接続プールなどの使用に関する提案については、『Exadata Database Machine上でのデータベース統合に関するベスト・プラクティス』を参照してください。 » データベース・ノードまたはインスタンス障害の後のログオン・ストームを回避するために、着信接続要求を抑制するようにOracleリスナーを構成します。 » これがActive Data Guardである場合は、REDO Applyのパラレル化を制限します。 » 参考資料 » プロセス数の削減について詳しくは、『Exadata Database Machine上でのデータベース統合に関するベスト・プラクティス』を参照してください（Exadataのみ）。

IORM （Exadataのみ）	» すべてのCDBおよびPDBにわたるディスクへの公平なアクセスを保証するために、Exadata IORMを有効にします。デフォルトでは、IORMは'basic'のIORM目標で動作し、重要なディスクI/Oの待機時間が極端なレベルに達しないようにするために、ほんのわずかな量のスケジューリングしか実行しません。完全に有効にするには、'IORM目標'をデフォルトの'basic'ではなく'auto'に設定します。IORMは目標を使用して、データベース間IORMプランや、すべてのCDBおよびデータベース・リソース・プランを実施します。 » OLTPデータベースとデータウェアハウスが同じExadataストレージ・セルを共有しているときは、ほとんどのOLTP I/Oがフラッシュで処理され、ほとんどのデータウェアハウスI/Oがディスクで処理されます。OLTPデータベースがディスクへのI/Oを発行している場合は、データウェアハウス・ワークロードからの競合によって、ディスク待機時間が許容できないほど長くなる可能性があります。IORMを有効にすると、データベース間IORMプランおよびCDBリソース・プランでOLTPデータベースとPDBに大きなリソース割当てを提供することにより、OLTP I/Oに対するディスク待機時間を短縮できます。ディスク待機時間が引き続き長すぎる、または一貫性がない場合は、IORM目標を"balanced"または"low latency"に変更できます。 » スマート・スキャンを実行する多数のアプリケーションを統合する場合は、データベース間、CDB、またはデータベース・リソース・プランでUTILIZATION_LIMITを構成します。 » さまざまなワークロードを異なるCDBに分離すると、データウェアハウス・アプリケーションが重要なOLTPデータベースのフラッシュ・リソースを消費しないようにして、OLTPアプリケーションが必要なSLAを維持できるようになります。 » 参考資料：My Oracle Support 1363188.1- 『Configuring Exadata I/O Resource Manager for Common Scenarios』を参照してください。
ネットワーク	» ネットワーク・リソース管理（Exadataのみ）は、重要なデータベース・ネットワーク・メッセージに自動的かつ透過的に優先順位を付け、待機時間の影響を受けやすい操作での短い応答時間を保証します。優先順位付けがInfiniBandファブリック全体を通して確実に実行されるようにするために、優先順位付けはデータベース、データベースのInfiniBandアダプタ、Oracle Exadata Storage Server Software、ExadataストレージのInfiniBandアダプタ、およびInfiniBandスイッチで実装されます。Oracle RACキャッシュ・フュージョン・メッセージなどの待機時間の影響を受けやすいメッセージは、バッチ、レポート、およびバックアップ・メッセージより優先されます。トランザクション処理の待機時間を短くするために、ログ・ファイルの書込み操作にもっとも高い優先順位が与えられます。ネットワーク・リソース管理は常に有効化されており、DBバージョン11.2.0.4/12cのIBスイッチ・ソフトウェア・リリース2.1.3-4を含むExadata 11.2.3.3.0から使用できます。 » Oracle RACデータベース内の最小限のインスタンス上でPDBをアクティブ化することによって、過剰なキャッシュ・フュージョン・ネットワーク・トラフィックを回避します。不必要なデータの配信を回避するために、PDBサービスを使用して、特定のPDBがアクティブであるインスタンスを特定します。さらに、CDB RACデータベースのすべてのインスタンスにわたって負荷が分散されていることを確認します。
ストレージ・グリッド	» ハードウェア・プールごとに1つの共有ストレージ・グリッドを構成します。1つの共有ストレージ・グリッドの管理は簡単で、管理コストも削減されます。ストレージを共有することで、領域や帯域幅の使用効率も向上します。 » これがゴールドまたはプラチナのハードウェア・プールである場合は、Exadataセルのローリング・アップグレード中やストレージ・タイプ障害からの最適なデータ保護および冗長性を実現するために、DATAおよびRECOディスク・グループにはASMの高冗長性を使用します（Exadataのみ） » ハードウェア・プールは1つの層に対して指定されるため、共有ストレージ・グリッドは1つの層だけを対象にします。
クラスタ	» ハードウェア・プールごとに1つのクラスタを使用します。すべてのデータベース・サービスが、さらにロードバランシングしたり、アプリケーションをクラスタ内の特定のデータベース・インスタンスまたはPDBにルーティングしたりするために使用される1つのOracle Clusterwareインストールによって管理されます。 » Oracle Multitenantでは、CDBの数をクラスタあたり最大10個（つまり、データベース・ノードあたり最大10インスタンス）に維持することが推奨されます。各CDBインスタンスに最大252個のPDBを割り当てることができます。 » データベース・ノードまたはクラスタあたりに作成するデータベース・インスタンスまたはPDBの実際の数は、アプリケーション・ワークロードや、各インスタンスまたはPDBのシステム・リソース消費によって異なります。

表5に、パフォーマンス層に基づくデータベースの推奨ガイドラインを示します。

表5：パフォーマンス層別のリソース管理のガイドライン

パフォーマンス層	CPU	メモリ	Exadata I/O
プラチナ	オーバーサブスクリプションのないインスタンス・ケージング。すべてのデータベースにわたるCPU_COUNTの合計が、サーバーのCPUスレッド数の75%を超えないようにしてください。 LinuxのcgroupまたはSolarisリソース・プールを構成し、PROCESSOR_GROUP_NAMEパラメータを設定することによって、各インスタンスを専用のCPUまたはNUMAノードにバインディングすることを考慮して ください。MOS Note 1585184.1を参照してください。	すべてのデータベースでSGA_TARGETおよびPGA_AGGREGATE_TARGETを構成します。 PGA、SGA、およびクライアント・プロセスのためのメモリの合計が、合計システム・メモリの75%を超えないようにしてください。 他の11gデータベースとの統合に注意してください。12cでのみ使用可能なPGA_AGGREGATE_LIMITパラメータを使用しないと、 データベースのPGA使用量を厳密に制限することはできません。	データベース間IORMを構成します。ストレージ・セルが複数のパフォーマンス層のデータベースをホストしている場合は、上位の層にあるデータベースにより大きな割当てまたはシェアを与えます。
ゴールド	オーバーサブスクリプションのないインスタンス・ケージング。すべてのデータベースにわたるCPU_COUNTの合計が、サーバーのCPUスレッド数の90%を超えないようにしてください。	すべてのデータベースでSGA_TARGETおよびPGA_AGGREGATE_TARGETを構成します。 PGA、SGA、およびクライアント・プロセスのためのメモリの合計が、合計システム・メモリの75%を超えないようにしてください。	データベース間IORMを構成します。ストレージ・セルが複数のパフォーマンス層のデータベースをホストしている場合は、上位の層にあるデータベースにより大きな割当てまたはシェアを与えます。
シルバー	インスタンス・ケージング。データベースのワークロードが異なる時間にピークに達する場合は、適度な量のオーバーサブスクリプションは許容できます。	すべてのデータベースでSGA_TARGETおよびPGA_AGGREGATE_TARGETを構成します。 PGA、SGA、およびクライアント・プロセスのためのメモリの合計が、合計システム・メモリの80%を超えないようにしてください。	データベース間IORMを構成します。ストレージ・セルが複数のパフォーマンス層のデータベースをホストしている場合は、上位の層にあるデータベースにより大きな割当てまたはシェアを与えます。

ブロンズ	インスタンス・ケージング。オーバーサブスクライブ。	すべてのデータベースでSGA_TARGETおよびPGA_AGGREGATE_TARGETを構成します。 PGA、SGA、およびクライアント・プロセスのためのメモリの合計が、合計システム・メモリの90%を超えないようにしてください。	データベース間IORMを構成します。ストレージ・セルが複数のパフォーマンス層のデータベースをホストしている場合は、上位の層にあるデータベースにより大きな割当てまたはシェアを与えます。 より一貫性のあるパフォーマンスを得るため、およびそのスマート・スキャンを制約するために、ブロンズ・データベースに"制限"を設定することを検討します。 フラッシュ・キャッシュがすべてのデータベースを収容できるほど大きくない場合は、テストや開発などの重要でないデータベースに対するデータベース間計画でフラッシュ・キャッシュを無効にすることを検討します。フラッシュ・キャッシュ・ミス率の監視については、次の項を参照してください。

リソースの監視

データベースがハードウェア・プールに移行される前に、データベース・パフォーマンスの監視および分析が開始されます。データベースが統合の適切な候補であるかどうかを判定する前に、ソースにおけるそのデータベースの平均およびピーク時のリソース消費を把握することが重要です。

ターゲットの統合環境に移行した後も、統合の計画が有効であったかどうかを確認するために、パフォーマンス・インディケータの綿密な監視が引き続き重要です。移行される各データベースのプロセスと影響を本番環境への移行の前に検証できるように、最初にテスト環境を使用することが常に推奨されます。

監視は、次の 3 つのレベルで実行してください。

» システムの監視。サーバーとストレージを監視して、CPU、メモリ、およびストレージ使用率が許容可能なレベルにあるかどうかを確認します。さらに多くのデータベースをサーバーまたはストレージに統合する予定がある場合は、使用可能な容量や余裕も監視します。

» CDB またはデータベースの監視。データベースを監視して、そのデータベースが実際に使用しているリソースの量を確認します。この確認は、PGA などの、リソース・マネージャを使用して制約できないリソースでは特に重要です。また、そのデータベースがリソース・マネージャによってどれだけ制限または抑制されているかを確認するためにも監視する必要があります。リソース・マネージャのために CPU または I/O の待機が発生し、データベースのパフォーマンスが許容できない場合は、リソース・マネージャをチューニングするか、またはデータベースをより多くのリソースを持つシステムに移動することができます。

» PDB の監視。各 PDB をデータベースと同様に監視します。その実際のリソース使用量が現在の CDB リソース・プランで保証されているリソースを超えている場合は、この CDB に PDB が追加されるとともに、そのパフォーマンスはおそらく低下します。パフォーマンスの低下が許容できない場合は、この CDB インスタンスに PDB を追加しないでください。

次の表では、キー・パフォーマンス・インディケータについて説明します。

表6：キー・パフォーマンス・インディケータ

監視/管理	ガイドライン
メモリ	システムの場合： » ページングが発生しないようにします。ゼロまたは非常に低いページインおよびページアウト率を監視するには、vmstatコマンドを使用します。 » メモリが決してオーバーサブスクライブされないようにします。Linux上で/proc/meminfoを使用して、合計システム・メモリを取得します。それを、さまざまなOracleデータベースやクライアント・プロセスに実際に割り当てられているメモリと比較します。SGA使用量は、各データベースのSGA_TARGETパラメータから計算できます。PGA使用量は、下の統計を使用して確認できます。 データベースまたはCDBの場合： » v$pgastatを使用して、インスタンスあたりの実際のPGA使用量を監視します。"maximum PGA allocated"の統計は、インスタンスがその起動以降に割り当てた最大PGAです。"total PGA allocated"の統計は、インスタンスが現在割り当てているPGAの量です。これらの値を監視すると、データベース管理者は、インスタンスによって実際に使用されているPGAの量を知ることができます。これらの値をPGA_AGGREGATE_TARGETと比較して、データベースがその値を超えているかどうかや、どの程度超えているかを確認します。
CPU	システムの場合： » OSツールまたはv$sysmetric_historyの"ホストCPU使用率"メトリックを使用して、システムのCPU使用率を監視します。CPU使用率がほぼ100%である場合は、OSツールまたはv$sysmetric_historyの"OS負荷"メトリックを使用して、システムがどれだけオーバーサブスクライブされているかを判定します。表5「パフォーマンス層別のリソース管理のガイドライン」で示されているように、過剰なオーバーサブスクリプションを回避します。過剰な負荷を回避するには、インスタンス・ケージングを使用します。 データベースまたはCDBの場合： » インスタンス・ケージングまたはCDB リソース・マネージャが有効になっている場合は、インスタンスが実際に使用したCPUの量と、必要であるにもかかわらず、使用を妨げられたCPUの量を監視します。MOS Note 1338988.1の説明に従って、v$rsrcmgrmetric_historyを使用します。すべてのコンシューマ・グループおよびPDBにわたる'avg_running_sessions'の合計は、実際に使用されるCPUの数を指定します。すべてのコンシューマ・グループおよびPDBにわたる'avg_waiting_sessions'の合計は、CPUが不足したためにリソース・マネージャによって実行された抑制を指定します。これは、追加の必要なCPU数に対応します。 » CDBの場合は、使用可能なCPU容量や余裕を監視して、さらに多くのPDBを追加できるかどうかを判定します。CDBリソース・マネージャが有効になっている場合は、すべてのコンシューマ・グループおよびPDBにわたるv$rsrcmgrmetric_historyの'avg_running_sessions'と'avg_waiting_sessions'を合計することによって、必要なCPUの総量を計算します。この合計がCPU_COUNT未満である場合、この合計とCPU_COUNTの差はこのCDB上のCPUの余裕であるため、PDBを追加することを検討できます。この合計がCPU_COUNTを超えている場合、このCDBはすでに容量に応じて動作しており、余裕はありません。PDBを追加しても、CPUの競合を増やすだけです。そのため、追加のPDBは、既存のPDBとそのアプリケーションがある程度のパフォーマンス低下を許容できる場合に、ブロンズのCDBに対してのみ追加してください。監視はピークの時間帯に実行するようにします。詳しくは、MOS Note 1338988.1を参照してください。 PDBの場合： » 各PDBの実際のCPU使用量とCPU待機時間を監視します。v$rsrcmgrmetric_historyの'avg_running_sessions'メトリックを使用して、PDBが実際に使用したCPUの数を確認します。実際のCPU使用量をPDBの保証されたCPUと比較します。これは、インスタンス上で開かれているすべてのPDBにわたるシェアの総数でそのPDBのシェアを割った値に基づきます。たとえば、PDBのシェアが1であり、シェアの合計が5である場合、そのPDBにはCPU_COUNTの5分の1が保証されます。PDBの実際のCPU使用量がその保証されたCPU使用量を超えている場合は、PDBを追加すると、PDBのパフォーマンスが低下する可能性があります。v$rsrcmgrmetric_historyの'avg_waiting_sessions'メトリックを使用して、PDBのCPU待機時間を表示します。ゼロ以外である場合、PDBのパフォーマンスは、CDBリソース・プランのそのシェアまたは使用率制限を増やすことによって改善する可能性があります。詳しくは、MOS Note 1338988.1を参照してください。
ディスク/フラッシュ	Exadataストレージ・セルの場合。MOS Note 1337265.1のスクリプトを使用して、Exadata I/Oメトリックを監視します。ほとんどの場合、これらのメトリックはEnterprise Manager 12cでも表示できます。 » CD_IO_ST_RQメトリックを使用して、バッファ・キャッシュ読取りなどのOLTP I/Oのディスク待機時間を監視します。待機時間が長く、OLTPデータベースでパフォーマンスの問題を引き起こしている場合は、Exadata IORMを有効にします。データベース間IORMプランで、高パフォーマンス層のOLTPデータベースに多く割り当てます。待機時間が依然として許容できない場合は、IORM目標を"auto"から"balanced"または"low latency"に変更することを検討してください。 » CD_IO_LOADメトリックを使用して、ディスクの負荷を監視します。この値が大きいと待機時間が長くなるため、OLTPデータベースを含むシステムでは、5を超える値は大きいと見なされます。さらに待機時間が長くなることが許容できない場合は、データベースを追加しないでください。データウェアハウスでは、負荷が高いとディスクのスループットが向上するため、大きな値は問題ありません。 » ただし、負荷のメトリックが常に20を超えている場合は、データベースを追加しないでください。 » DB_IO_UTIL_SMおよびDB_IO_UTIL_LGメトリックを使用して、複数のデータベースにわたる実際のディスクI/O使用率を監視します。この比較は、どのデータベースがディスクをもっとも高い頻度で使用しているかを示します。 » FC_IO_RQ_DISK_WRITEを使用して、フラッシュ書込みIOPSを監視します。合計の書込みIOPSが公開された最大値に達しており、フラッシュ・パフォーマンスの低下が受け入れられない場合は、フラッシュ・キャッシュを使用する新しいデータベースまたはCDBを追加しないでください。データベースまたはCDBに対するフラッシュ・キャッシュは、データベース間IORMプランの"flashCache=off"ディレクティブを使用して無効にすることができます。 » OLTPのフラッシュ・ヒット率を計算するために、FC_IO_RQ_RをFC_IO_RQ_R_MISSと比較します。現在のパフォーマンスを維持する場合は、フラッシュを使用するデータベースを追加する前に、そのフラッシュ・ヒット率を監視してフラッシュに余裕があることを確認してください。たとえば、フラッシュ・ヒット率が80%以上のときにパフォーマンスが許容可能な場合は、データベースを追加してフラッシュ・ヒット率が80%未満になると、パフォーマンスが低下する可能性があります。

	データベースまたはCDBの場合： » AWRを使用して、I/O待機イベントを監視します。問題のある待機イベントが"db file sequential read"である場合は、データベースのフラッシュ・キャッシュ・ヒット率（上記を参照）、ストレージ・セルのディスク待機時間（上記を参照）、および小さなリクエストに対するデータベースのIORM低下時間（以下を参照）を監視およびチューニングします。問題のある待機イベントが"cell smart table scan"である場合は、データベースの低下時間（以下を参照）を監視およびチューニングします。 » IORMが有効になっている場合は、DB_IO_WT_SM_RQおよびDB_IO_WT_LG_RQメトリックを使用して、リクエストあたりの平均IORM低下時間を監視します。待機時間が長く、データベース・パフォーマンスに満足できない場合は、データベース間計画でそのデータベースの割当て/シェアまたは使用率制限のどちらかを増やす必要があります。すべてのデータベースにわたって待機時間が長い場合は、ディスクがその最大容量に達しています。この場合は、既存のデータベースがパフォーマンスの低下を許容できない限り、新しいデータベースを追加しないでください。 » CDBの場合は、PDB_IO_UTIL_SMおよびPDB_IO_UTIL_LGメトリックを使用して、複数のPDBにわたる実際のディスクI/O使用率を監視します。この比較は、CDB内のどのPDBがもっとも高い頻度でディスクを使用しているかを示します。 PDBの場合： » IORMが有効になっている場合は、PDB_IO_WT_SM_RQおよびPDB_IO_WT_LG_RQメトリックを使用して、リクエストあたりの平均IORM低下時間を監視します。待機時間が長く、PDBのパフォーマンスに満足できない場合は、CDBリソース・プランでそのPDBのシェアまたは使用率制限のどちらかを増やす必要があります。すべてのPDBにわたって待機時間が長い場合は、データベース間IORMプランでそのCDBの割当てまたは制限、あるいはその両方を増やす必要があります。それまでは、既存のPDBがパフォーマンスの低下を許容できない限り、新しいPDBを追加しないでください。
データベースのパフォーマンス・インディケータ	» Automatic Workload RepositoryまたはEnterprise Manager 12c、あるいはその両方を使用してデータベース負荷のプロファイルを表示し、ドリルダウンして上位の待機イベントを確認します。ここから、特定の待機にドリルダウンします。 » ASH分析を使用して、コンシューマ・グループまたはPDB別のCPU使用量の内訳を表示します。
アプリケーションのパフォーマンス・インディケータ	» Automatic Workload RepositoryまたはEnterprise Manager 12c、あるいはその両方を使用してデータベース負荷のプロファイルを表示し、ドリルダウンして上位の待機イベントを確認します。ここから、特定の待機にドリルダウンします。 » ASH分析を使用して、コンシューマ・グループまたはPDB別のCPU使用量の内訳を表示します。
全般	» MOS Note 1070954.1のexachk（Exadataのみ）

高可用性とデータ保護

非統合環境におけるサービス・レベルの予測は、統合環境の場合とはまったく異なります。たとえば、開発者または部門で使用するスタンドアロン・データベースについて考えてみます。データベース停止のイベントによって引き起こされる中断のレベルは、停止が解決されている間に、生産性を維持するために他の作業を見つけることができる、より小規模なユーザー・コミュニティに限定されます。次に、この同じデータベースが、それぞれ異なる部門やユーザー・コミュニティをサポートしている他の 100 個のデータベースと統合されていた場合はどうなるかを考えてみます。統合データベースに影響を与える停止の企業に対する中断のレベルは 100 倍に拡大されるため、HA およびデータ保護がはるかに高い優先事項になります。

Oracle Multitenant は、Oracle Maximum Availability Architecture（Oracle MAA）を使用して、統合環境の HA およびデータ保護の要件に対応します。HA およびデータ保護に対する Oracle MAA の従来の目標に加えて、マルチテナント・アーキテクチャのコンテキストに固有の目標が存在します。

» 一元管理による簡素化。Oracle MAA のベスト・プラクティスは、HA およびデータ保護の目標がコスト上の最大の利点（資本コストおよび運用コスト）を実現しながら達成されるように、大規模な統合環境の管理を容易に拡張できることが必要です。

» 分離。Oracle MAA のベスト・プラクティスでは、1 つの PDB に影響を与える問題が、CDB 内の他の PDB の可用性に影響を与えないようにする必要があります。

表面的には、最初の 2 つの目標は矛盾するように見えます。"n 個の"環境の分離によって、多くの場合は、個別に管理する必要のある"n 個の"異なる環境が生成されます。Oracle MAA はマルチテナント・アーキテクチャを利用できるため、一元管理が可能な大きな利点を実現しながら、分離における妥協を最小限に抑えて HA が達成されます。

» Oracle MAA を使用すると、CDB を、そこに含まれている PDB の数には関係なく 1 つのデータベースとして管理できます。100 個の PDB を含む CDB は、保守や運用に関する多くのタスクについて 100 分の 1 の削減を実現できます（1 つの Oracle RMAN バックアップ、ディザスタ・リカバリのための 1 つの Oracle Active Data Guard スタンバイ、およびアップグレードやパッチ適用の対象となる 1 つのデータベース）。Oracle MAA ではまた、必要に応じて、他の PDB とは分離して PDB を管理する柔軟性も提供されます。次に例を示します。

» 個々の PDB をリストアする必要がある場合、Oracle RMAN は、同じ CDB 内で開かれている他の PDB に影響を与えることなくそれを実行できます。プラグインされたばかりの PDB は、定期的にスケジュールされた次回の CDB バックアップの前に問題が発生した場合でも確実にリカバリできるように、プラグイン操作の後ただちにバックアップするようにしてください。

» 高速なポイント・イン・タイム・リカバリが必要な場合は、Oracle Multitenant の最初のリリースにより、フラッシュバック・データベースを CDB レベルで使用できます。Oracle Multitenant の将来のリリースでは、他の PDB の可用性に影響を与えることなく、フラッシュバック・データベースを個々の PDB 上で使用できるようになる予定です。

» 個々の PDB で、管理者が CDB 内の他の PDB に影響を与える可能性があると確信する問題が発生した場合は、疑わしい PDB を容易にアンプラグし、分離によって問題を解決できる別の CDB にプラグインできます。また、アンプラグ/プラグ操作には、元の CDB 内で実行されている他の PDB に影響を与えることなくパッチを適用する柔軟性もあります。問題が解決されたら、その PDB を新しい CDB 内に置いておくか、または新しいパッチがインストールされた後、その PDB をアンプラグして元の CDB にプラグインすることができます。

» プライマリ CDB データベースで PDB が破損し、その PDB が正常な既存のスタンバイ・データベースがある場合は、スタンバイから新しい CDB に PDB をアンプラグおよびプラグすることで、1 回の PDB フェイルオーバーを実行できます。My Oracle Support Note『Unplugging a Single Failed PDB from a Standby Database and Plugging into a New Container』（ドキュメント ID 2088201.1）を参照してください。

» Oracle Resource Manager は、ワークロードの急激な増加や、消費パターンを変化させるバグやその他の何らかのイベントが発生した場合に、PDB によってシステム・リソースの割当て済みのシェアを超える量が消費されないようにします。

» Oracle GoldenGate の論理レプリケーションを使用すると、CDB 内の 1 つまたは複数の PDB をレプリケートできます。これにより、最小限の停止時間または（双方向レプリケーションを使用して）停止時間ゼロで PDB を別の CDB に移行する柔軟性が提供されます。このプロセスは、計画保守の対象であるすべての操作（プラットフォーム移行、Oracle Database のアップグレード、バックエンド・データベース・オブジェクトを変更するアプリケーション・アップグレード、通常は停止時間を必要とするその他のデータベース保守など）をサポートする新しい CDB 内に PDB のクローンを作成することから始まります。クローン化された PDB が新しいバージョンで使用可能になったら、GoldenGate のレプリケーションを使用して、それを元のソースがクローン化された後に発生した新しいトランザクションと同期します。検証が完了すると、ユーザーが新しいバージョンに移行されます。この移行を処理するには、次の 2 つのオプションがあります。

» ユーザー・セッションを終了することによって、最小限の停止時間を実現できます。これにより、Oracle GoldenGate は、コミットされたすべてのトランザクションをレプリケートし、ソース・データベースを停止した後、ユーザーが新しいバージョンに接続できるようにする処理を完了できます。

» 双方向レプリケーションを使用すると、停止時間ゼロを実現できます。これにより、自然に接続を解除し、次に再接続したユーザーから徐々に移行できるようになるため、停止時間ゼロのユーザー・エクスペリエンスが得られます。また、ワークロードが増加して新しいバージョンで予期しない問題が発生した場合は、即時フェイルバックも可能になります。双方向レプリケーションでは、Oracle GoldenGate で提供される基本的な機能（最初の変更を優先）を使用するか、またはより高度な競合解消方式を実装するように Oracle GoldenGate を拡張することによって、管理者は競合の検出と解決を実施する必要があります。アプリケーションに関する知識が必要になります。「Multitenant と GoldenGate の考慮事項」を参照してください。

Multitenant と GoldenGate の考慮事項

GoldenGate Release 12.1.2 以降、PDB 間でデータをレプリケートできるようになりました。たとえば、複数の PDB からデータを抽出して非マルチテナント・データベースにレプリケートしたり、非マルチテナント・データベースからデータを抽出して複数の PDB にレプリケートしたりすることができます。

マルチテナント・データベースを使った GoldenGate の構成について詳しくは、『Installation and Configuring Oracle GoldenGate for Oracle Database』ガイドを参照してください。

http://docs.oracle.com/goldengate/c1221/gg-winux/GIORA/config_containerdb.htm#GIORA942

GoldenGate とマルチテナント・データベースについては、他にもパフォーマンス上の推奨事項があります。

- 単一の統合キャプチャ・モードである Extract を使用して、PDB からデータを抽出します。Extract プロセスごとに、すべての PDB のデータが含まれる CDB 全体の REDO ストリームが読み込まれるため、単一の Extract を使用すれば、Extract プロセスの追加時の I/O 量の増加を抑制できます。
- 複数のターゲット・データベースにデータをレプリケートする必要がある場合は、複数の Data Pump を構成して、証跡ファイル・コンテンツのサブセットを各ターゲット・データベース・ホストに送信します。
- Replicat を使用すれば、SQL*Net 接続文字列で指定される 1 つの PDB のみにデータを適用できます。1 つの Replicat で、証跡ファイルに含まれる複数の PDB からデータを適用できます。
- 追加の統合 Replicat を構成する前に、サーバーで使用可能な CPU 帯域幅を監視し、追加プロセスを実行できるヘッドルームが十分あることを確認することが重要です。統合 Replicat では自動並列度機能が使用されるため、適用を待つ別の処理がある場合は、適用サーバー・プロセスの数が増えます。Replicat で作成される適用サーバーの最大数は、GoldenGate のパラメータである INTEGRATEDPARAMS（MAX_PARALLELISM n）によって制御されます。デフォルト値は 50 です。1 つの統合 Replicat による過大な CPU の消費を防ぎ、他の Replicat に必要なリソースが不足しないようにするには、この値を減らすことが必要な場合があります。

計画外停止の管理

表 7 に、マルチテナント・アーキテクチャ内のデータベースに影響を与える可能性のあるさまざまな計画外停止を示します。また、このホワイト・ペーパーで先に説明した各 HA 層で発生する停止に対応するための Oracle HA ソリューションも示しています。

表7：マルチテナント・アーキテクチャでの計画外停止マトリックス

イベント	ブロンズ、シルバー、ゴールド、およびプラチナのためのソリューション	リカバリにかかる時間（RTO）	データ損失（RPO）
インスタンス障害	ブロンズ：Oracle Restart シルバー：Oracle RACまたはオプションで Oracle RAC One Node ゴールド：Oracle RAC プラチナ：Application Continuityパラメータが 指定されたOracle RACMOS Note 1585184.1を参照してください。	サーバーを再起動できる場合は数分 Oracle RACでは数秒 Oracle RAC One Nodeでは数分 数秒 アプリケーション停止ゼロ	ゼロ ゼロ ゼロ ゼロ ゼロ
永続的なノード障害 （ただし、ストレージは使用可能）	ブロンズ：リストアとリカバリ シルバー：Oracle RAC シルバー：Oracle RAC One Node ゴールド：Oracle RAC プラチナ：Oracle RACおよび Application Continuity	数時間～数日 数秒 数分 数秒 アプリケーション停止ゼロ	ゼロ ゼロ ゼロ ゼロ ゼロ
ストレージ障害	すべての層：Oracle Automatic Storage Management	停止時間ゼロ	ゼロ
データ破損	ブロンズ/シルバー：基本的な保護 一部の破損にはPDBまたはCDB全体のリストアとリカバリが必要 ゴールド/プラチナ：Oracle Active Data Guardによる破損に対する包括的な保護および自動ブロック修復	数時間～数日 自動ブロック修復によりゼロ 書込み損失による破損であり、Data Guardのファスト・スタート・フェイルオーバーを使用した場合は数秒～数分	リカバリ不能の場合は最後のバックアップ以降 書込み損失による破損でない限りゼロ
人為的エラー	すべて：フラッシュバック・ドロップ、フラッシュバック表、フラッシュバック・トランザクション、フラッシュバック問合せ、およびUNDOによって解決される論理的な障害 すべて：データベース全体、およびOracle RMANのポイント・イン・タイム・リカバリ（PDB）またはフラッシュバック・データベースを必要とするPDBに影響を与える包括的な論理的な障害 ゴールド/プラチナ：Oracle GoldenGateを使用すると、1つのPDBだけをフェイルオーバー可能	検出時間に依存するが、これらのオブジェクトを使用するPDBおよびアプリケーションに分離される 検出時間に依存する 検出時間に依存するが、実際のフェイルオーバーには数秒かかる場合がある	論理障害に依存する 論理障害に依存する 論理障害に依存する
データベースが使用不可、システム、サイト、またはストレージ障害、幅広く分散した破損または災害	ブロンズ/シルバー：リストアとリカバリ ゴールド：セカンダリへのフェイルオーバー（Oracle Active Data GuardまたはOracle GoldenGate） プラチナ：Application Continuityを使用したActive Data Guardのフェイルオーバー	数時間～数日 数秒 アプリケーション停止ゼロ	最後のバックアップ以降 ゼロ～ほぼゼロ ゼロ
単一PDBが使用不可	ブロンズ/シルバー：PDBのリストアとリカバリ ゴールド：CDBセカンダリへのフェイルオーバーまたはPDBフェイルオーバーの実行（Note 2088201.1） （Oracle Active Data GuardまたはOracle GoldenGate） プラチナ：Application ContinuityによるActive Data Guardのフェイルオーバーまたは PDBフェイルオーバーの実行（Note 2088201.1） （Oracle Active Data GuardまたはOracle GoldenGate）	数時間～数日 数分 アプリケーション停止ゼロ Oracle GoldenGateとPDBのフェイルオーバーの場合は、Application Continuityがサポートされていないため数分かかります。	最後のバックアップ以降 ゼロ～ほぼゼロ Oracle GoldenGateの場合はほぼゼロ Data GuardとPDBのフェイルオーバーの場合はゼロ Oracle GoldenGateの場合はほぼゼロ
パフォーマンス低下	すべての層：データベース・リソース・マネージャおよびチューニング	停止時間はないが、サービスは低下	ゼロ

注：PDB を確実にリカバリできるようにするために、PDB または CDB のバックアップは、PDB のプラグイン操作の後ただちに実行します。PDB をリストアするには、関連付けられた CDB が使用可能で、かつ動作している必要があります。

個々の PDB に関する多くの問題を他の PDB に影響を与えることなく解決することは可能ですが、分離が必要な状況も存在します。たとえば、CDB によって使用されている Oracle ホームにパッチを適用することが必要になる場合があります。このシナリオでは、新しい Oracle ホームと CDB を作成した後、問題のある PDB をアンプラグして新しい CDB にプラグインすることを推奨します。それにより、元の CDB 内の他の PDB にまったく影響を与えないという 100%の確実性で問題を解決できます。問題が解決されたら、その PDB をアンプラグして元の CDB にプラグインするか、または将来のある時点まで新しい CDB 内に置いておくことができます。

Data Guard 環境のプライマリ・サイトでいずれかの PDB に問題があり、フェイルオーバーが必要な場合は、通常、CDB 全体とそのすべての PDB をスタンバイ・サイトにフェイルオーバーします。個々の PDB をスタンバイに直接フェイルオーバーできなくても、プライマリですぐに修復できない障害が発生した単一の PDB を迅速に移動することはできます。そのためには、この PDB をスタンバイ・サイトから抽出し、スタンバイ・サイトに共存する CDB に最小限の停止時間でプラグインします。このプロセスの実行手順については、Note 2088201.1 を参照してください。この手順を実行すれば、影響を受ける PDB を分離して移動する際に、その他の PDB を引き続き正常に運用できます。

計画保守の管理

計画保守の観点から見ると、非 CDB で使用可能な従来のソリューションはすべて、マルチテナント・アーキテクチャ環境で動作できます。さらに、保守を 1 つの PDB に対してのみ実行するか、または同じコンテナ内のすべての PDB に対して実行するかを管理者が決定できる場合があります。オラクルでは、どちらの状況にも対応できる柔軟性を提供しています。

表 8 に、計画保守に対応した Oracle HA ソリューションを示します。

表8：マルチテナント・アーキテクチャでの計画保守マトリックス

イベント	ブロンズ、シルバー、ゴールド、およびプラチナのためのソリューション	予測される停止時間
移行	「マルチテナント・アーキテクチャへ の移行」を参照してください。	状況に応じて変化
動的なオンライン・リソース・プロビジョニング または オンライン再編成および再定義	すべて：各PDB内の選択されたオブジェクトのオンライン再編成および再定義 ドキュメント：「動的およびオンライン・リソース・プロビジョニング」および「オンライン再編成および再定義」	ゼロ
オンライン・パッチ データベースとGrid Infrastructureのパッチおよびワンオフ・パッチ	すべて：CDB全体へのオンライン・パッチ適用が可能（該当する場合） すべて：PDBのアンプラグと、対象のソフトウェア・リリースで動作する個別のCDBへのプラグインが可能 シルバー：CDB全体でOracle RAC One Nodeのローリング・アップグレードの利用が可能（該当する場合） ゴールド/プラチナ：CDB全体でOracle RACのローリング・アップグレードの利用が可能（該当する場合）。プラチナ層ではApplication Continuityが補完 ゴールド：オプションで、CDB全体でData GuardのStandby-First Patch適用の利用、およびData Guardのスイッチオーバーの発行が可能 プラチナ：オプションで、CDB全体でData GuardのStandby-First Patch適用の利用、およびData Guardのスイッチオーバーと Application Continuityの発行が可能	ゼロ 数秒～データファイル・コピー・オプションなしの場合は1時間（推定値） サービスの再配置によりゼロ サービスの再配置によりゼロ アプリケーション停止ゼロ 数秒～数分 アプリケーション停止ゼロ

データベース・パッチセット	すべて：PDBのアンプラグと、対象のソフトウェア・リリースで動作する個別のCDBへのプラグインが可能 ゴールド/プラチナ：CDB全体でパッチセットおよびメジャー・データベース・リリースに対するData Guardのデータベース・ローリング・アップグレードの利用が可能 プラチナ：対象とするソフトウェア・バージョンで動作するセカンダリGoldenGateレプリカへのCDBまたはPDBによるフェイルオーバーが可能	数秒～データファイル・コピー・オプションなしの場合は1時間（推定値） 数秒～数分 停止時間ゼロ
アプリケーション・アップグレード	プラチナ：エディションベースの再定義では、開発者がこの機能を利用するように設計する必要がある プラチナ：対象とするアプリケーション変更が適用されたGoldenGateレプリカへのPDBによるスイッチオーバーが可能 ドキュメント：「オンライン・アプリケーションのメンテナンスおよびアップグレード」	ソフトウェアのスプロール化が削減され、リスクが低減されます。OPEXが削減され、全体的な安定性と可用性が向上します。

パッチ適用およびアップグレード

パッチ適用およびアップグレードは、マルチテナント・アーキテクチャによってもっとも大きな影響を受ける領域です。マルチテナント・アーキテクチャによって提供される"一元管理"機能を使用して、1 回のアップグレードで、CDB 内のすべての PDB を新しいバージョンにアップグレードできます。また、Grid Infrastructure のローリング・パッチ適用、Real Application Cluster のローリング・パッチ適用 =、Data Guard の Standby-First Patch 適用、Data Guard の一時ロジカル・ローリング・アップグレード、Oracle GoldenGate などの既存のツールを使用して、CDB 全体を 1 回の操作でアップグレードするための停止時間をほぼゼロまたはゼロに短縮することもできます。

CDB 内の PDB のサブセットのみをアップグレードする場合は、アップグレードされたバージョンで完全に新しい CDB を作成し、PDB をアンプラグしてプラグインすることができます。これにより、特定のアプリケーション・ニーズに対処するために必要となる柔軟性が提供されます。この操作は、個々のデータベースの通常のアップグレードより短い時間で済みます。Oracle GoldenGate ではまた、異なる CDB 間でレプリケートすることにより、最小限の停止時間または停止時間ゼロで PDB をアップグレードするオプションも提供されます。

DBaaSのライフ・サイクル管理

Oracle Enterprise Manager 12c は、マルチテナント・アーキテクチャおよび Database as a Service（DBaaS）でのデータベースのライフ・サイクルのさまざまなフェーズを管理するうえで重要な役割を果たします。Oracle Enterprise Manager 12c では、ビジネス・ユーザーや技術系ユーザーのための IT リソースのセルフサービス展開のほか、各種のマルチテナント・アーキテクチャに対応したリソース・プーリング・モデルが可能になります。DBaaS は、エンドユーザー（データベース管理者、開発者、QA エンジニア、プロジェクト・リーダーなど）がデータベース・サービスをリクエストし、それをプロジェクトのライフ・タイムを通して消費した後、自動的にプロビジョニング解除されてリソース・プールに返されるようにすることができるパラダイムです。

DBaaS は、次のものを提供します。

» データベース・サービスをプロビジョニングする共有および統合されたプラットフォーム

» これらのリソースをプロビジョニングするためのセルフサービス・モデル

» データベース・リソースをスケールアウトおよびスケールバックする順応性

» データベース使用量に基づくチャージバック

顧客がすでにデータベースを稼働させている既存の実装の場合、Oracle Enterprise Manager 12c はこれらのデータベースを自動的に検出し、その使用状況のベースラインを特定して、マルチテナント・アーキテクチャに移行するための統合に関するアドバイスを提供できます。次に、非 CDB のデータベースをマルチテナント・アーキテクチャに移行するためのガイド付きフローを提供できます。また、移行の過程で、以前のバージョン（つまり、Oracle Database 10g または Oracle Database11g）からデータベースをアップグレードすることもできます。

新しいデータベースのロールアウトの場合、Oracle Enterprise Manager 12c はネイティブなプラグ/アンプラグのメカニズムを利用した、PDB をプロビジョニングするための標準のワークフローを提供します。セルフサービスの DBaaS ユーザーが使用できるように、セルフサービス・インタフェースにも同じ手順が公開されています。

Oracle Enterprise Manager は、多数の PDB を管理するために必要な自動化機能を提供します。インベントリ管理やレポートなどの構成管理機能は、あらゆる不要なスプロール化を防止するのに役立ちます。スタック全体を比較し、最適なベースラインに対して一貫性があるかどうかをチェックできるように、Oracle Enterprise Manager には、業界をリードする構成のずれを管理する機能が用意されています。このコンテキストで特筆すべきもう 1 つの機能として、コンプライアンス管理があります。Oracle Enterprise Manager には、データベースの構成を既知の業界標準やベスト・プラクティスに対してチェックするためのルールが標準で含まれています。

最後に、データベースへのパッチ適用やアップグレードは、特に管理者が数百および数千の PDB や基盤となる CDB を処理する必要がある場合、データセンター内の非常に時間がかかる、労働集約的なアクティビティになることがあります。Oracle Enterprise Manager 12c には、事前のチェックとパッチ適用後のレポートで補完された、パッチ適用およびアップグレードの自動化機能が標準で含まれています。

要約すると、マルチテナント・アーキテクチャ内のデータベースのライフ・サイクル管理には、次の機能が含まれています。

- 非 CDB からの移行
- PDB の初期のプロビジョニングおよびクローニング
- インベントリ追跡、構成のずれの追跡、およびトポロジ・マッピング
- 構成のコンプライアンス管理

パッチ適用およびアップグレードの自動化 詳しくは、次のドキュメントを参照してください。

- データベース管理
- データベースのライフ・サイクル管理
- データベース・クラウドの管理

結論

ある一連の目標（たとえば、システム・フットプリントの削減）に対応するように設計された戦略によって、統合環境のその他の側面（たとえば、パフォーマンス、HA、データ保護、管理コスト）に関する新しい課題が生成されないように、データベース統合および DBaaS の実装には総合的なアプローチが必要です。

Oracle Multitenant を使用した Oracle Database 12c、Oracle MAA のベスト・プラクティス、および Oracle Enterprise Manager は、効率的で、信頼性に優れたデータベース統合および DBaaS に必要な総合的なソリューションを提供します。これらのソリューションによってすべてのプラットフォーム上での統合および DBaaS が可能になりますが、最適なサービスを提供しながら、総ライフ・サイクル・コストを最小限に抑えるという目標の完全な実現は、Oracle エンジニアド・システムを使用して達成されます。

付録A：Exadataの統合密度およびパフォーマンス

オラクルは、Oracle Exadata Database Machine 上で高い統合密度を実現する機能を検証するための一連のテストを実施しました。同一のワークロードと Oracle Database 構成を使用して、Oracle Exadata Database Machine 上で 2 つの一連のテストを実行しました。最初の一連のテストでは、比較のため、Exadata 固有のすべての機能を無効にしてベースラインを提供しました。この目的は非 Exadata システムをシミュレートすることにありましたが、Exadata 機能を有効にしていない場合でも Exadata システムの高い I/O およびネットワーク帯域幅や安定したパフォーマンス特性が得られるため、このアプローチでは比較のための積極的なベースラインが提供されました。2 番目の一連のテストでは最初のテストと同じワークロードを同じマシン上で実行しましたが、今回は、Exadata のみの機能をすべて有効にしました。これにより、データベース・ワークロードの統合密度を向上させる Exadata の独自の機能を測定することが可能になりました。

システム構成：

» Oracle Exadata Database Machine X4-2 フル・ラック

» 合計 192 個の CPU コアと 4TB のメモリを備えた 8 台のデータベース・サーバー

» ストレージ階層内の SQL 処理専用の 168 個のコアを備えた 14 台の Exadata Storage Server

» 44TB の Exadata Smart Flash Cache

» 40Gb/秒の内蔵 InfiniBand ネットワーク

データベース構成：

» Oracle RDBMS 12.1.0.1（PSU 2）および Oracle Grid Infrastructure 12.1.0.1（PSU 2）

» DBFS_DG は標準冗長性（二重ミラー）、DATA および RECO ASM ディスク・グループは高冗長性（三重ミラー）

» Oracle シングル・インスタンス・データベース（Oracle RAC 以外）、Oracle RAC One Node、および Oracle RAC を試行

» Oracle Database のための標準の Exadata 構成のベスト・プラクティスを使用（HugePages、適切なシステムおよびデータベース設定、Oracle MAA 設定の使用など）

» 各 OLTP データベースで 4GB SGA を使用

» 8 つのすべてのノード上で DW/レポート・データベースを実行、7GB SGA を装備

非 Exadata と Exadata の統合密度を比較するために分離された Exadata 機能

» Smart Flash Logging

» Smart Flash Cache

» Flash Cache Compression

» ネットワーク・リソース管理

» オフロード・スキャン

» I/O リソース管理

» Exadata ストレージ・インデックス

テスト実行およびワークロード

各テスト実行の長さは30分であり、これにはデータベースの再起動とウォーム・アップ期間が含まれていました。OLTP ワークロードは、Swingbench の注文入力アプリケーションを使用して生成されました。混在したユースケースを示すために各種の問合せを含むデータウェアハウス・レポート作成アプリケーションを使用しましたが、データウェアハウスやレポートの問合せを抑制するために、どの場合もデータウェアハウス・データベース上でインスタンス・ケージングを使用しました。

OLTP アプリケーション・ワークロードを起動するために Exalogic サーバーを使用しました。各 Swingbench アプリケーションは、独自の個別の接続プールを備えていました。

テスト結果

最大のスループットを得るために、すべてのテスト・ケースに同じ Oracle MAA 統合のベスト・プラクティスを組み込みました。非 Exadata のテストでは、リソース・バウンドになるまでデータベースを追加し続け、200 ミリ秒未満の平均応答時間が維持されました。平均応答時間は 177 ミリ秒で、I/O バウンドになる前に、40 個の OLTP データベースを統合することができました。上位の待機イベントは、55%の DB 時間での 47 ミリ秒のセル単一ブロック物理読取り（たとえば、db_file_sequential 読取りと同等）と、31%の DB 時間での 122 ミリ秒のログ・ファイル同期でした。平均ログ・ファイル・パラレル書込みは 23 ミリ秒であり、32 ミリ秒を超える異常値が 20%、最大 1 秒でした。待機プロファイルと I/O 待機時間が、OLTP データベースが 1 個だけか、または最大 5 個かでまったく異なっていたことに注意してください。OLTP データベースが少数しか存在せず、システム・リソースのボトルネックがない場合、データベース待機プロファイルは、4 ミリ秒の平均待機時間でのセル単一ブロック物理読取りの 60%の DB 時間と、DB CPU および 1 ミリ秒の非常に短いログ・ファイル同期での 30%の DB 時間で構成されていました。IO、CPU、およびネットワークのために競合する個別のデータベースが多数存在する場合は、待機時間が急速に増加します。

Exadata 機能を有効にした後、同じテストを実行して、応答時間や安定性を犠牲にすることなく同数の個別のデータベースを統合しました。次の例は、Exadata 上で 160 個の個々の OLTP データベースを統合した結果を示しています。

Exadata 統合テスト（OLTP ワークロード）：ブロンズ層

メトリック	非ExadataのOLTPワークロード	ExadataのOLTPワークロード	結果
累積TPS	10514	46915	4.4倍の改善
平均応答時間	177ミリ秒	132ミリ秒	25%の削減
データベース数	40	160	4倍に増加
上位の待機イベントと平均待機時間	セル単一ブロック物理読取り：47ミリ秒、55%のDB時間 ログ・ファイル同期：122ミリ秒、31%のDB時間 ログ・ファイルのパラレル書込み：23ミリ秒	セル単一ブロック物理読取り：11ミリ秒、31%のDB時間 ログ・ファイル同期：28ミリ秒、10%のDB時間 ログ・ファイルのパラレル書込み：5ミリ秒	Exadataでは、短い待機時間を保証するために基本的なデータベースI/Oが優先されました。Exadata Smart Flashによって、I/O帯域幅が大幅に拡張されました。

Exadata Smart Flash Cache 機能およびネットワーク・リソース管理では、短い待機時間と高帯域幅を保証するために、データベース読取りおよび書込み操作（特に、ログ書込み操作）が優先されました。160 個のデータベースの場合でも、平均ログ・ファイル・パラレル書込みは依然として 5 ミリ秒であり、ログ・ファイル同期は 28 ミリ秒、セル単一ブロック物理読取りは 11 ミリ秒でした。この例では、ワークロードが CPU バウンドになり、システムは非常に安定していました。

データベース 200 個まで統合密度を高めることに成功し、非 Exadata 構成の 5 倍の密度を達成しましたが、次に示すとおり、応答時間の増加はわずか 11%にとどまりました。

ブロンズ層での統合密度の向上

メトリック	非ExadataのOLTPワークロード	Exadataの OLTPワークロード	結果
累積TPS	10514	48309	4.5倍の改善
平均応答時間	177ミリ秒	196ミリ秒	11%の増加
データベース数	40	200	5倍に増加
上位の待機イベントと平均待機時間	セル単一ブロック物理読取り：47ミリ秒、55%のDB時間 ログ・ファイル同期：122ミリ秒、31%のDB時間 ログ・ファイルのパラレル書込み：23ミリ秒	セル単一ブロック物理読取り：4ミリ秒、11%のDB時間 ログ・ファイル同期：31ミリ秒、10%のDB時間 ログ・ファイルのパラレル書込み：6ミリ秒 書込み完了待機などのDBWRタイプの待機での45%のDB時間によって、バッファの待機やバッファ・ビジー状態の待機が解放されますが、チューニングする時間が足りませんでした。	Exadataでは、短い待機時間を保証するために基本的なデータベースI/Oが優先されました。Exadata Smart Flashによって、I/O帯域幅が大幅に拡張されました。

どちらの例も、ブロンズ層の**純粋な OLTP ワークロードでの 4 倍～5 倍という非常に優れたデータベース統合密度**を示しています。

実際の環境の OLTP アプリケーションでは、レポート、バッチ、ETL などの各アクティビティがある程度混在しています。個々のデータベースやアプリケーションを 40 個以上含む大規模な統合環境では、OLTP 以外のアクティビティが見られることが一般的です。これらのその他のアクティビティが、OLTP の応答時間やスループットに悪影響を与える場合があります。この例では、長時間実行される各種の問合せを含むレポートのワークロード用にデータウェアハウスを 1 個追加しました。すべての実行で、データウェアハウスとレポートのワークロードのためにインスタンス・ケージングを使用して、使用可能な I/O および CPU リソースがこのワークロードでいっぱいにならないようにしました。非 Exadata の場合は、40 個の OLTP データベースに加えて 1 個のデータウェアハウス/レポート・データベースで IOPS を使い切りました。また、Exadata の場合は、160 個の OLTP データベースに加えて 1 個のデータウェアハウス/レポート・データベースで CPU を使い切りました。この場合も、Exadata 機能によって主要なデータベース操作が優先され、スループットの向上（6.3 倍）、応答時間の大幅な削減（50%）、および統合密度の向上（4 倍）が可能になりました。

Exadata 統合テスト（OLTP、データウェアハウス、レポート）：ブロンズ層

メトリック	非ExadataのOLTPワークロード	ExadataのOLTPワークロード	結果
累積TPS	8466	53662	6.3倍の改善
平均応答時間	212ミリ秒	104ミリ秒	50%の削減
データベース数	41	161	4倍に増加
上位の待機イベントと平均待機時間	セル単一ブロック物理読取り：54ミリ秒、59%のDB時間 ログ・ファイル同期：104ミリ秒、23%のDB時間 ログ・ファイルのパラレル書込み：21ミリ秒	セル単一ブロック物理読取り：4ミリ秒、13%のDB時間 ログ・ファイル同期：38ミリ秒、21%のDB時間 ログ・ファイルのパラレル書込み：4ミリ秒	ExadataはOLTP I/Oを吸収し、スキャンをより効率的にオフロードしました。

シルバー、ゴールド、およびプラチナ・アプリケーションの場合は、CPU とメモリに余裕を残しておくことを推奨します。この例では、Exadata 上の CPU とメモリは使い切られていませんでした。統合密度は低めですが、応答時間の削減とスループットの向上は大幅に増大しています。

Exadata 統合テスト（OLTP、データウェアハウス、レポート）：シルバー、ゴールド、およびプラチナ層

メトリック	非ExadataのOLTPワークロード	ExadataのOLTPワークロード	結果
累積TPS	8466	55422	6.5倍の改善
平均応答時間	212ミリ秒	14ミリ秒	93%の削減
データベース数	41	81	2倍に増加
上位の待機イベントと平均待機時間	セル単一ブロック物理読取り：54ミリ秒、59%のDB時間 ログ・ファイル同期：104ミリ秒、23%のDB時間 ログ・ファイルのパラレル書込み：21ミリ秒	DB CPUは42%のDB時間で上位待機 セル単一ブロック物理読取り：1ミリ秒、23%のDB時間 ログ・ファイル同期：3ミリ秒、12%のDB時間 ログ・ファイルのパラレル書込み：1ミリ秒	CPUバウンドでない場合は、パフォーマンス向上と予測可能な応答時間が増大します。 この例ではシルバー以上の層に対して2倍の統合密度

環境の安定性を維持するために、Exadata 統合のベスト・プラクティスを適用しました。ほとんどの部分について、これらの構成のベスト・プラクティスは固有のものであり、すでに Oracle エンジニアド・システムには適用されています。ただし、次の変更およびカスタマイズが最大の影響を与えました。

- HugePages を設定します（すべてのデータベースを収容できるように調整する必要があります）。
- プロセス数を最小限に抑え、接続プールを使用します。
- ASM プロセス数を設定します。
- semmsl および semmns 設定を調整します。
- データウェアハウス・ワークロードをインスタンス・ケージングします。
- iorm objective=AUTO を使用します。

付録B：RMAN Active Database Duplicationによる移行

この付録では、Oracle Database 12c の非 CDB を PDB に移行するための停止時間の最小化のアプローチを取る場合に使用される手順の概要について説明します。このプロセスでは、FROM SERVICE を使用した新しい 12c RMAN Duplicate Active Database 機能を利用して、バックアップ・ファイルをステージングするための追加の領域をまったく必要とせずに、アクティブなデータベース・バックアップやネットワーク全体にわたる増分適用を実行します。各ファイルは Oracle Net を使用してコピーされ、新しい CDB のターゲットの場所に格納されます。この例では、データファイルを格納するために ASM が使用されていることを前提にしています。この時点では、ソース環境と宛先の環境が同じエンディアンである場合にのみ、非 CDB から PDB へのプラグインを実行できます。クロスエンディアンの移行には、Oracle GoldenGate、またはエクスポート/インポートの何らかの方法（従来のエクスポート/インポート、クロスエンディアンのトランスポータブル表領域、クロスエンディアン・データベース全体のトランスポータブル表領域のいずれか）が必要です。

次の手順は、Oracle Database 12c の非 CDB を PDB として CDB にプラグインするためのおもな手順を示しています。次のプロセスの詳細な例と、その他の移行方法については、My Oracle Support Note 1576755.1『Step by Step Examples of Migrating non-CDBs and PDBs Using ASM for File Storage』を参照してください。

Oracle Database 12c の非 CDB のデータベースを PDB としてプラグインするには、次の手順を実行します。

» データベースを Oracle Database 12c にアップグレードします。このデータベースは非 CDB になります。

» CDB を作成するか、または移行のターゲットにする既存の CDB を選択します。CDB の作成は、DBCA を使用するか、または SQL*Plus 経由で CREATE DATABASE コマンドを使用して実行できます。PDB を事前に作成する必要はないことに注意してください。PDB はプラグイン操作中に作成されます。

» オプションで、フィジカル・スタンバイ・データベースを作成します。

» Oracle Database 12c RMAN を使用して、元の非 CDB と新しい CDB の両方に接続します。このどちらの接続でも Oracle Net を使用する必要があります。非 CDB はターゲットとして、CDB はクローン・データベースとして接続してください。

```
RMAN> connect target <user>@non-cdb as sysbkup

RMAN> connect clone <user>@cdb as sysbkup
```

» 初期のアクティブな複製を実行して、ソース・データベースのイメージ・ファイルを宛先のサイトにコピーします。OMF を使用している場合は、Oracle RMAN がこれらのファイルを正しい場所にリストアします。

» 宛先の CDB 上に一時インスタンスを作成します。

» 非 CDB の制御ファイルを宛先の環境の一時的な場所にリストアします。

» 今リストアしたデータファイルを一時インスタンス上の制御ファイルにカタログ化し、SWITCH DATAFILE TO NEW コマンドを発行します。

» 非 CDB を PDB としてプラグインするために使用されるプラグイン SQL 文を作成します。CREATE 文で SOURCE_FILE_DIRECTORY 句を使用し、手順 5 でファイルがリストアされたディレクトリの場所を指定します。

» 必要な回数だけ、非 CDB の増分バックアップを実行する FROM SERVICE 句を使用してリカバリし、それらの変更を宛先の CDB 上のファイルに適用します。

» アンプラグ操作を実行する準備ができたら、次の手順を実行します。

  » 非 CDB を閉じたことを確認し、読取り専用で開きます。

  » 非 CDB 上にマニフェストの XML ファイルを作成します。

  » 非 CDB から宛先のコピーへの最後の増分適用を実行します。

  » マニフェストを宛先のホストにコピーします。

  » マニフェストの XML ファイルを使用して、PDB を作成します。

  » 新しい PDB を開きます。

  » 新しい PDB をバックアップします。

  » その PDB へのアプリケーション・アクセスを設定します。

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

海外からのお問い合わせ窓口
電話：+1.650.506.7000
ファクシミリ：+1.650.506.7200

CONNECT WITH US

blogs.oracle.com/oracle
facebook.com/oracle
twitter.com/oracle
oracle.com

Hardware and Software, Engineered to Work Together

Oracle is committed to developing practices and products that help protect the environment