ORACLE

データベース統合のための ベスト・プラクティス

Oracle MAAリファレンス・アーキテクチャを含む
実装ガイド

2020年7月 | バージョン[3.0]

免責事項

本文書には、ソフトウェアや印刷物など、いかなる形式のものも含め、オラクルの独占的な所有物である占有情報が含まれます。この機密文書へのアクセスと使用は、締結および遵守に同意したOracle Software License and Service Agreementの諸条件に従うものとします。本文書と本文書に含まれる情報は、オラクルの事前の書面による同意なしに、公開、複製、再作成、またはオラクルの外部に配布することはできません。本文書は、ライセンス契約の一部ではありません。また、オラクル、オラクルの子会社または関連会社との契約に組み込むことはできません。

本書は情報提供のみを目的としており、記載した製品機能の実装およびアップグレードの計画を支援することのみを意図しています。マテリアルやコード、機能の提供をコミットメント（確約）するものではなく、購買を決定する際の判断材料になさらないでください。本書に記載されている機能の開発、リリース、および時期については、弊社の裁量により決定されます。

製品アーキテクチャの性質上、コードが大幅に不安定化するリスクなしに、本書に記載されているすべての機能を安全に含めることができない場合があります。

目次

見出し画像1：AutoTextドロップダウンからご自身のドキュメントに追加してください。この画像はプレースホルダであり、置き換えるか削除する必要があります。

概要

IT組織はコストを管理し、俊敏性（アジリティ）を向上させ、可用性を確保し、セキュリティ・リスクを削減し、ビジネス目標達成のため適度なシステム・パフォーマンスを提供するよう強く求められています。IT組織がこれらの目標を達成するために利用する一般的な戦略として、データベースと他のシステムの統合が登場しました。本書では、Oracleデータベース統合のベスト・プラクティスについて説明します。これには、O/S仮想化、リソース管理、Oracle Multitenantのデータベース機能、Oracle Exadata Database Machine、Oracle Exadata Cloud@Customer、Oracle Exadata Cloud Service、Exadataにも常駐するOracle Autonomous Databaseなどの主要テクノロジーの利用が含まれます。

オンプレミスおよびクラウドのOracle Exadata Databaseプラットフォームは、低コストのデータベース統合を実現できる理想的なプラットフォームです。ビジネス目標を達成するために、簡潔な管理、業界屈指の可用性、エンタープライズグレードのセキュリティ、優れたパフォーマンス能力を導入し、管理できます。ExadataにはOracle Maximum Availability Architecture（MAA）のベスト・プラクティスが全般的に組み込まれ、標準化された運用プラクティスも統合されているため、統合されたデータベース環境における最大限の信頼性を確保できます。

Oracle Exadata Database Machineは、従来のオンプレミス環境、Oracle Public Cloud、およびOracle Cloud@Customerモデルにデプロイできます。データベース統合は、本書で説明するように、このすべてのデプロイメント・モデルに適用されます。さらに、Oracle Autonomous Databaseには、本書で説明する原則が自動的に組み込まれるため、極めて容易なデータベース統合が可能です。

本書ではこれらのトピックについて掘り下げることで、IT組織に対してOracleデータベース統合のために考えられる最良のガイダンスを提供します。

データベース統合の概要

本書では、コスト管理、データベースの管理性向上、可用性の確保、セキュリティ強化、Oracleデータベースを利用するビジネス・プロセスに対する適度なパフォーマンスの提供というおもなビジネス目標を達成するための、Oracleデータベース統合のベスト・プラクティスについて説明します。本書に記載されている推奨事項の多くはどのプラットフォームにも適用できるものですが、本書では特に、Oracle Exadata Database Machineおよびこの製品によるオンプレミスおよびクラウド内でのビジネス目標の達成方法について重点的に説明します。

データベース統合の定義

データベース統合では、オンプレミスまたはOracle Cloud内にある単一のコンピューティング・インフラストラクチャー式に、複数のデータベースを配置します。データベース統合はサーバー統合と似て非なるもので、サーバー統合は、仮想マシンを実行する単一の物理サーバーに複数の物理サーバーを統合することです。データベース統合のおもな目的は、単一のサーバーに多数のプロセッサ・コアが搭載された市販の高性能サーバーを利用することで、データベース・インフラストラクチャのコストを削減することです。データベース統合には以下のいずれかの構成が関わります。

- 単一の物理サーバーで複数のデータベース
- 単一の仮想マシンで複数のデータベース
- 一つの物理サーバー・クラスタで複数のデータベース
- 一つの仮想マシン・クラスタで複数のデータベース

データベース統合によって、少ないサーバー数で多くのデータベースを運用できるため、必要なインフラストラクチャが減少しますが、さらに運用コストも減少することになります。各物理サーバーは1つの情報技術資産、つまり“構成アイテム”（CI）を表すものであり、データセンターのラック内に設置して、他の機器と接続し、セキュリティを確保し、保守する必要があります。各仮想マシンも別の構成アイテム（CI）を表すものであり、同じくデプロイし、（仮想的に）接続し、セキュリティを確保し、保守する必要があります。データベース統合は、Oracle Database Cloud ServiceやOracle Exadata Cloud Serviceなどのクラウド環境にも適用されます。

また、Oracleデータベースは、物理サーバーまたは仮想マシンから成るクラスタに統合することもできます。Oracle Databaseのクラスタ機能は、アクティブ-パッシブ・クラスタリングのほか、Oracle Real Application Clustersを利用したアクティブ-アクティブ・クラスタリングにも対応します。クラスタリングは、Oracleデータベースの可用性向上とパッチのローリング適用のため、および複数の物理サーバーまたは仮想マシンにわたるスケーラビリティ確保のために利用されます。

Oracleデータベースはさらに、Oracle Database 12cで初めて導入されたOracle Multitenantオプションを使用して、いわゆるコンテナ・データベースに統合することも可能です。Multitenantオプションによって、複数のデータベースを単一のコンテナ・データベース下で管理でき、バックアップ、ディザスタ・リカバリ構成、クラスタ構成がすべて単一で済むようになります。

データベース統合の目的

データベース統合は、コンピューティング・インフラストラクチャのコストやIT運用コストの削減を目指したものですが、ビジネス要件を満たすために、以下の要因についても考慮する必要があります。

- コストの削減
- 管理の簡素化
- セキュリティと独立性の向上
- 可用性目標の達成
- 必要とされるパフォーマンスの提供

コストの削減はビジネス側でのデータベース統合のおもな推進要因になります。統合によるコスト削減効果は、従来型のオンプレミス・データセンターだけでなく、クラウド・サービスを利用する場合にも適用されます。データベースの統合によって、オンプレミスのコンピューティング資産や、クラウド・サービスを利用してデプロイされているシステムの利用率を引き上げることができます。また、統合によって、Oracleデータベースのリソース管理機能を使用したコンピューティング・リソースの動的な自動再割当ても可能になります（オンプレミス、クラウドを問わず）。

統合環境におけるデータベースの**管理の簡素化**は、まず、保守を必要とする構成の標準化と種類の削減から始まります。データベース統合によって多数のデータベースを一元管理できるようになるため、物理サーバー、仮想マシンの管理業務が減少し、さらにOracle Multitenantのコンテナ・データベースを使用すればデータベースの数自体も減少します。管理の簡素化によって、データベースやビジネス・アプリケーションの可用性確保に企業の労力を集中させることができます。

統合環境におけるデータベースの**セキュリティと独立性の向上**は、脆弱性の数を削減すること、データベース間の標準化を進めること、およびそれらのデータベースの保護に必要となる管理上の負荷を軽減することで達成できます。データベース統合によってセキュリティ面の危険性が高まる可能性もあります。攻撃者が非統合環境よりも多くのデータにより高いアクセス権でアクセスできる可能性があるからです。Oracle Databaseはそれ自体に幅広いセキュリティ機能が搭載されており、Exadata Database Machineの一部となることでそれらの機能が強化され、セキュリティの脆弱性がさらに緩和されます。

データベースおよびアプリケーションの可用性目標はビジネス要件によって決められますが、その**可用性目標の達成**は、複数のデータベースが同じインフラストラクチャに存在しているという事実だけからも、統合環境において必要不可欠です。本書では、システムの障害に対する耐久性を高め、複数の独立性レイヤーを導入して障害の影響範囲（“爆発半径”）を狭めるための適切な構成について説明します。また、フォルト・トレランスの確保や、保守作業中、アップグレード中、あるいはインフラストラクチャ・コンポーネント障害への対応中にも継続的な可用性を実現できる適切な運用プラクティスについても取り上げます。

データベースおよびアプリケーションで**必要とされるパフォーマンスを提供**することは、あらゆる統合環境の大きな目標の1つです。あるデータベースのパフォーマンスが、別のデータベースで起きている問題の影響を受けないようにする必要があります。データベース、そしてそれらのデータベースによって支えられるビジネス・アプリケーションには適度なコンピューティング・リソースを割り当てる必要があります。Oracle Database Resource Manager（DBRM）とExadata I/O Resource Manager（IORM）のリソース管理ツールは、データベースとアプリケーションの適度なパフォーマンスを確保するために必要となるレベルの制御機能を提供します。

データベース統合における仮想マシンの役割

仮想マシン（VM）はサーバー統合で広範に使用されていますが、Oracleデータベースの統合においても特有の役割を果たします。仮想マシン（一部のプラットフォームでは論理パーティションまたは論理ドメインとも呼ばれます）によって、単一の物理サーバーで複数のオペレーティング・システム（O/S）イメージを、それらのO/Sイメージの内容を互いに分離しながら同時に実行できます。各O/Sには専用のディスク、メモリ、CPU、仮想ネットワーク・インタフェースが割り当てられ、各O/Sに個別の専用システム・プロセスがあり、各O/Sは同じ物理サーバー上で実行される他のO/Sイメージからは分離されます。仮想マシンのおもな利点は、複数のシステムを同じ物理サーバー上で統合した場合に、O/Sレイヤーでの独立性を確保できることです。

仮想マシン（VM）テクノロジーは、セキュリティ上必要となるところで独立性を確保するという、データベース統合における重要な役割を果たします。たとえば、開発環境とテスト環境の分離、あるいは複数の本番データベースの分離のために仮想マシンを使用できます。また、PCI（Payment Card Industry）データやHIPAA（Healthcare Insurance Portability and Accountability Act）データなどのデータ・セキュリティ要件に合わせて独立性を確保するためにも仮想マシンを使用できます。この独立性には、各VMクラスタ用に個別のVLANタグ付きネットワークを使用することも含まれます。
仮想マシンには、アップグレード中にデータベースのグループを分離できる、O/Sスナップショットなどのより柔軟なフォールバック方式を利用できるなど、システム・ソフトウェアの保守における利点もあります。一方で、それぞれの仮想マシンが、インストール、パッチ適用、セキュリティ、管理を含む管理上のオーバーヘッドをもたらします。オラクルは、この管理上のオーバーヘッド増加という理由で、VMごとにデータベースを1つデプロイするという方法は推奨していません。オラクルが推奨しているのは、物理マシンあたり少数のVMをデプロイし、各VMで複数のデータベース、コンテナ・データベース、プラガブル・データベースのいずれかまたはすべてをホストするという、より賢明な仮想マシンの使い方です。また、最高レベルのパフォーマンスが必要なケースや、数百テラバイトのデータや数百人から数千人のユーザーを擁する極めて大規模なデータベースの場合には、ベア・メタル・マシン（VMなし）が推奨されます。

データベース統合におけるReal Application Clustersの役割

Oracle Real Application Clusters（Oracle RAC）は、データベース統合環境において高可用性とスケーラビリティを実現するための重要な機能を提供します。サーバーのクラスタリングでは、クラスタウェアと呼ばれるソフトウェア（Oracle Clusterwareなど）の制御下で協調的に動作する2台以上のサーバー（または仮想マシン）がグループ化されます。Oracle Real Application Clustersによって、Oracleデータベースは、クラスタの複数のサーバーにわたって展開し、一般にアクティブ-アクティブ・クラスタリングと呼ばれる構成で、クラスタ内のすべてのサーバー上で同時にアクティブに稼働できるようになります。Oracle RACでは、クラスタにサーバー（またはVM）を追加することによるデータベースの“水平”スケーリングが可能です。Oracle RACには、サーバー（またはVM）の保守作業中や、サーバー（またはVM）の障害発生時にも継続して運用できる機能もあります。Oracle RACによって、以下の状況でもデータベースを継続して運用できます。

- サーバー・ハードウェアの保守
- O/Sの保守
- VMハイパーバイザの保守
- サーバー・ハードウェア障害
- O/S障害（例：カーネル・パニック）
- データベース・ソフトウェアの更新
- Oracle Grid Infrastructureの更新

Oracle RACは、四半期ごとのセキュリティ・パッチ適用、データベース・ソフトウェア修正、Oracle Grid Infrastructureの更新などのハードウェアおよびソフトウェアの保守を容易にします。Oracle RACは、Oracle Application Continuityとの併用によって高可用性が確保されます。Oracle Application Continuityは、サーバー障害の発生時にOracle RACクラスタ内部の他の稼働中サーバーにアクティブなセッションをすべて移行することで、（処理中のトランザクションを失うことなく）ユーザー・セッションの継続的な運用を可能にする機能です。
スケーラビリティ確保のためにOracle RACを利用する必要のない、性能が低くクリティカルでないデータベースについては、データベース統合環境でOracle RACなしでデータベースを使用できます。ただし、これらのデータベースは上記の状況では保護されず、クラスタリングによる利点がないために管理者の運用上の負担が大きくなる恐れもあります。クリティカルでないデータベースでも、可能ならば管理上の負担を軽減するためだけにOracle Real Application Clustersを使用するべきです。

統合におけるリソース管理の重要性

Oracle Database Resource Manager（DBRM）とExadataシステムのI/O Resource Manager（IORM）は、あらゆるOracleデータベースの運用で重要なコンポーネントですが、データベース統合においてはいっそうその重要度が増します。Oracleリソース・マネージャ（DBRMとIORM）が提供する制御機能は、リソースを各データベースに分配し、あるデータベースのパフォーマンス問題が同じシステム上の他のデータベースに影響するようなデータベース間のリソース競合を防ぐために必要になります。DBRMやIORMがない場合は、管理者は、より管理が難しく、Oracleデータベースのリソース割当てについて同程度の制御機能のないツールを組み合わせて使用することになります。Oracleリソース・マネージャ（DBRMとIORM）を使用すれば、管理者は以下を実行できます。

- データベースへのシステム・リソースの割当てと管理
- データベースおよびビジネス・アプリケーションの適度なパフォーマンスの確保
- データベースおよびシステムの安定性の確保

Oracle Database Resource ManagerとI/O Resource Manager（Exadataシステム用）によって、データベース管理者はデータベースにシステム・リソース（CPU、メモリ、プロセス、I/O、ネットワーク）を割り当てて、ビジネスの需要の変化に応じて長期間その割当てを管理することができます。これらのOracleリソース・マネージャでは、システム・リソースの再割当てを簡単に、しかも動的にも実行できるため、ビジネス・アプリケーションに必要となる適度なパフォーマンスを実現できます。リソース管理を適切に構成することで、アプリケーション、データベース、またはシステムの不正動作によるリソース不足を防ぎ、システムの安定性を高めることもできます。

Oracleデータベースも非Exadataシステム用のI/Oリソース管理機能を提供しています。ただし、その場合はExadataのような緊密な統合はされません。このトピックについて詳しくは、本書のリソース管理に関するセクションを参照してください。

データベース統合の手段

以下で説明するとおり、Oracleデータベースを物理マシンまたは仮想マシンに統合する方法は3つあります。仮想マシンはO/Sレイヤーでの独立性を確保するために使用されますが、ここでの"統合"は1つの仮想マシン内、1つの仮想化されていない物理サーバー内、または複数の物理サーバーまたは仮想マシンで構成される1つのクラスタ内で複数のデータベースを実行することを表します。

以下のいずれかの手段によって複数のデータベースを統合できます。

- 物理サーバー、仮想マシン、またはクラスタごとの複数のデータベース
- Oracle Multitenant
- スキーマ統合

これらの手段は、一連の物理マシンまたは仮想マシンにデータベースを統合するために、個別に使用することも組み合わせて使用することもできます。

物理マシンまたは仮想マシンごとの複数のデータベース

Oracleデータベースは単一の物理マシンまたは仮想マシンに統合でき、複数のOracle RACデータベースは物理マシンから成るクラスタまたは仮想マシンから成るクラスタに統合できます。統合されるOracleデータベースは同じバージョンでも異なるバージョンでもよく、Oracleソフトウェアは共有しても専用のものを使用してもかまいません（共有または専用のORACLE_HOMEを使用可能）。Oracle Database Resource Manager（DBRM）のインスタンス・ケージング機能を使用すれば、単一のサーバーまたはサーバー・クラスタに統合された複数のデータベースにわたってCPUを割り当てることができます。ExadataのI/O Resource Manager（IORM）機能はこれらのリソース管理機能をExadataシステムのストレージ・レイヤーにまで拡張したものです。Oracleリソース・マネージャについて詳しくは、本書の関連するセクションを参照してください。

Oracle Multitenantを使用したデータベース統合

Oracle Multitenantは、以前はスキーマ統合（詳しくは以下を参照）により達成していた効率性を可能にしながら、アプリケーション間で必要とされる独立性も確保します。Oracle Multitenantでは、コンテナ・データベース（CDB）内部に存在するプラガブル・データベース（PDB）という概念が導入されました。プラガブル・データベースは、1つのコンテナ・データベースに接続し、その後は"切断（アンプラグ）"と"接続（プラグ）"の操作によって別のコンテナ・データベースに移動できます。

このアーキテクチャによって、以下に示すとおり、単一サーバー上に展開できるデータベースが大幅に増加しました。

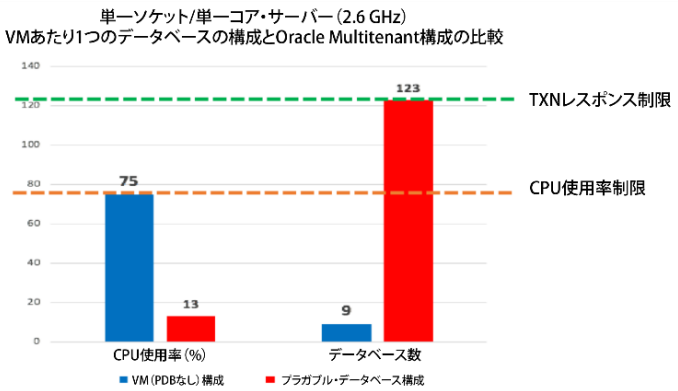

図1：仮想マシンを使用した統合とOracle Multitenantを使用したデータベース仮想化の比較

オラクルの内部的なテストでは、シングル・インスタンスのデータベースを専用の仮想マシン上で実行する場合と比較して、Oracle Multitenantでは統合密度が高くなりました。図1では、Oracle Multitenantを使用した場合に123個のデータベースが同じサーバー上に統合された一方で、仮想マシンごとに1つのデータベースを使用するアプローチの場合は同じサーバーで9つのデータベースしか展開できませんでした。VMごとに1つのデータベースを使用するアプローチではすぐにCPU使用率制限（75 %）に達しましたが、Oracle Multitenantを使用した場合はトランザクションのレスポンス・タイム制限に達するまでに123個のデータベースが統合されました。Oracle Multitenantは、多数のデータベースを1つのデータベースとして管理できるため、管理上の負担を軽減することもできます。

スキーマ統合

スキーマ統合は、Oracle Multitenantが登場する前に使用されていたアプローチであり、複数のデータベースのスキーマを1つのデータベースに統合して、システム・リソースの利用を効率化し、より高密度の統合を実現するものです。スキーマ統合ではコンピューティング・リソースを効率化できるものの、多くの場合はアプリケーションの変更が必要になります。スキーマ統合は、Oracle Multitenantでは可能な、アプリケーション間で必要となる程度の独立性は確保できず、アップグレードなどの保守作業のための独立性の確保も容易ではないためです。Oracle Multitenantは、スキーマ統合に見られた複雑さに対処できるため、高密度の統合を目指す場合に優先して使用すべきソリューションになっています。スキーマ統合は、単一アプリケーションに属する複数のデータベースを組み合わせる場合など、一部の状況では有用なアプローチですが、本書では詳細については割愛します。

Exadataが統合に適している理由

オンプレミスおよびOracle Cloud向けのOracle Exadata Database Machineは、最適なパフォーマンス、可用性、管理性をOracle Databaseにもたらすことを目的に設計されたエンジニアド・システムです。スケーラブル・アーキテクチャと高度なソフトウェア機能を備えているため、Oracleデータベース統合のための標準的なデータベース・プラットフォームとして適しています。Exadataは前述した以下の各ビジネス目標に対応できます。

- コストの削減
- 管理の簡素化
- セキュリティと独立性の向上
- 可用性目標の達成
- 必要とされるパフォーマンスの提供

Oracleデータベースは非Exadataプラットフォーム上でも統合できますが、Exadataを統合に利用することには多大な利点があります。

高密度の統合によるコスト削減

オラクルのベンチマーク・テストによると、Exadataではほぼ同じ規模の非Exadataシステムよりもきわめて高い密度の統合が可能です。テストのワークロードには、150個から300個のOLTPデータベースが含まれており、これらのデータベースでは、Oracle Multitenantを搭載したExadata X8-2クォーター・ラック上でOracle Real Application Clustersが実行されました。このテストでは、図2に示すとおり、データベースの数を150個から300個まで増やしても、Exadataのトランザクション・レスポンス・タイムは定義済みの上限値3ミリ秒以内に収まりました。

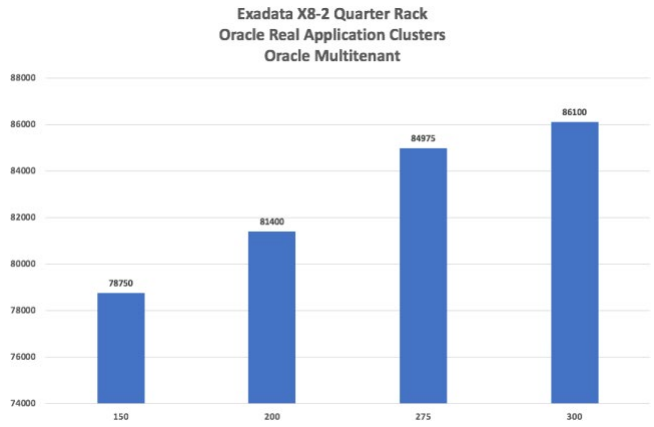

図2：Exadata統合の密度とパフォーマンス

Exadataは高いI/O帯域幅、高いIOPS（1秒あたりの入力/出力操作数）、高いスループット率を、データベース処理のストレージへのオフロードと組み合わせることで、従来のプラットフォームよりも高密度の統合を実現します。Exadata Elastic Expansionを使用すれば、システムへのデータベースの追加時やデータ量の増加時に、停止時間ゼロでストレージを追加できます。

標準的な構成と管理プラクティスによる簡素化

非常に多くのOracleユーザーがExadata Database Machinesを運用しており、Exadataを管理するための標準化されたプラクティスが開発されてきました。標準化によって運用環境の複雑さが軽減し、セキュリティと可用性が向上し、コストが削減されます。Exadataは単一のプラットフォームとしてどのような規模や複雑さのワークロードでも実行できるため、すべてのワークロードに対して使用可能な唯一の単一プラットフォームとなっています。

Exadataによるセキュリティ強化

Exadataプラットフォームには、高いセキュリティを確保するための多くの標準機能が搭載されており、システム管理者に余計な負担はかかりません。データベースの統合によって保護対象のシステム数が減少します。そして、Exadataはデフォルトで高セキュリティのプラットフォームです。Exadataには以下のような機能が搭載されています。

- 多層防御
- 最小限の権限付与の原則

- 更新およびセキュリティ修正の事前統合
- 最小限の攻撃対象範囲
- ソフトウェアの事前スキャン
- 自動侵入検知環境（AIDE）

オラクルはExadataのセキュリティ設計において多層防御アプローチを採用しています。この防御機能には、厳密認証、暗号化機能、データベースのユーザーとロール、システム・レベル（O/S）セキュリティなどがあります。また、オラクルは“最小限の権限付与”の原則を導入しており、インフラストラクチャ管理などに個別の管理ロールを割り当てたり、データベース管理者ロールを下位で複数に分割したりできます。

Exadataでは、四半期ごとのリリースに統合されたセキュリティ修正、月1回のセキュリティ・リリース、およびゼロデイ脆弱性に対処するための緊急の修正があります。これらの更新は、O/S、仮想マシン、クラスタウェア、論理ボリューム・マネージャ、ドライバ、ストレージ・ソフトウェア、ネットワーク、データベース・ソフトウェアそれ自体を含むスタック全体にわたって事前統合されています。ExadataにはOracleデータベースの実行に特化した必須ソフトウェア・コンポーネントのみが含まれているため、攻撃対象範囲は最小限に抑えられています。オラクルは業界屈指のセキュリティ・スキャナを使用して、Exadataソフトウェア・リリースのそれぞれに対して複数のセキュリティ・スキャンを実施しています。これらのスキャンには、マルウェア・スキャン、脆弱性スキャン、構成の適合性評価が含まれます。ユーザーも、デプロイ済みのシステムに対してスキャンを独自に実行することで、納品された構成から一切逸脱していないことを確認する必要があります。

また、Exadataには、19.1 Exadataソフトウェア・リリースより、Advanced Intrusion Detection Environment（AIDE）が搭載されています。AIDEは、SHA256ハッシュ値のシグネチャを期待値データベースに照合することで、重要なファイルの変更を自動的に検出する機能です。詳しくは、『Exadata Security Guide』（https://docs.oracle.com/en/engineered-systems/exadata-database-machine/dbmsq/security-guide-exadata-database-machine.pdf）を参照してください。

ExadataとMAAによる可用性の確保

Exadataは、Maximum Availability Architecture（MAA）向けの代表的なプラットフォームです。これは、最高レベルの可用性とデータ保護を達成し、Data GuardとOracle GoldenGateで最適なパフォーマンスを確保するために、Exadataプラットフォームにこれまで多大な開発投資が行われてきた結果です。そのような可用性機能には、ノード停止の高速検出、ストレージ障害の自動検出とリバランス、ネットワーク障害の自動検出、インスタンス障害発生中の高速検出と再構成などがあります。オラクルは、Exadataソフトウェア・リリースのたびに、可用性の全領域に対して継続的な改善を行っています。Exadataの高いパフォーマンスは、高密度の統合に加え、RACインスタンスの高速リカバリ、DRスタンバイ・リカバリなどのHA機能やDR機能にも貢献します。詳しくは、Exadata MAAのホワイト・ペーパー（https://www.oracle.com/technetwork/jp/database/features/availability/exadata-maa-131903-ja.pdf）を参照してください。

Exadataのパフォーマンス

Exadataの高パフォーマンス機能は、優れたパフォーマンスを達成するために使用されることが多いですが、その同じパフォーマンス上の利点により、他のプラットフォームでは実現できない高密度の統合を行うことができます。Exadataでは、Smart Flash、スマートな広帯域幅の内部InfiniBandネットワーク、Oracleデータベースのプロセスや SQLの処理をExadataストレージ・レイヤーにオフロードするスマート・ストレージによって、一般的なボトルネックの多くが削減または解消され、データベース・トランザクションが最適化されます。Exadataの高度なリソース管理機能（DBRM、IORMの両方を含む）を使用すると、各データベースに適正量のリソースを割り当てることができます。そのため、Exadataでは、パフォーマンスを管理して、ビジネス側から求められる理想的なレベルのパフォーマンスを提供できるのです。Exadataは最高レベルのパフォーマンスを提供することで知られていますが、リソース管理の利用によって、構成済みのサービス・レベルに準拠したパフォーマンスを提供することもできます。

スタンドアロンのデータベースまたはMultitenantデータベースを使用してより高密度の統合を実現するのに加え、Exadataプラットフォームは、統一された制御の下で包括的なリソース管理機能を提供しています。CPU、メモリ、プロセス、I/Oなどのシステム・リソースを単純に割り当てて管理するだけで、Exadataプラットフォーム上のデータベースのパフォーマンスを容易に管理できます。ExadataはOracleデータベース・レベルのリソース管理機能をExadataストレージ・ネットワークやI/Oリソースにまで拡張したものになっています。一般的なリソース管理機能やExadata環境特有のリソース管理機能については、本書のリソース管理に関するセクションを参照してください。

Exadataのデプロイメント・モデル

Exadata Database Machineは、以下の3つのデプロイメント・モデルを使用してデプロイできます。いずれのモデルでも、同じExadataテクノロジーが基盤として使用されます。

- オンプレミスのExadata
- Exadata Cloud@Customer
- Exadata Cloud Service
- Oracle Autonomous Database

これらのデプロイメント・モデルすべてで、コストの管理、操作の簡素化、セキュリティの確保、可用性目標の達成、必要なレベルのデータベース・パフォーマンスの提供に関して、同じ利点がもたらされます。Oracle Cloud ServiceおよびCloud@Customerでのデプロイメントでは、オラクルのクラウド・ツールやCloud Service製品によってさらに操作を簡素化できる追加機能が提供されます。またOracle Cloudでは、必須のデータ暗号化とクラウド・ネットワークのセキュリティを使用して、セキュリティもさらに強化されています。

統合の基盤構築

データベースの統合は、コストの抑制、簡潔な運用、セキュリティ強化、および必要なレベルのアプリケーション・パフォーマンス達成というビジネス目標を満たすための標準化されたシステム構成と運用プラクティスによる強固な基盤の上に成り立ちます。標準化された構成には、ビジネスのニーズに合わせられる十分な柔軟性も必要です。
このセクションでは、これらの目標を達成するための多数の重要な標準的構成について説明します。ただし、これらは、各コンポーネントのすべての構成に対して包括的に推奨される構成ではありません。各領域の推奨事項全般については、追加情報のセクションを参照してください。

Exadata CloudとExadata Cloud@Customer

Exadata Cloud Service、Exadata Cloud@Customer、およびAutonomous Databaseは、このセクションで説明するベストプラクティスの推奨事項が組み込まれているため、統合に必要な基盤が整っています。これらの製品のいずれにも、インフラストラクチャ管理サービスのほか、データベース統合のビジネス目標を達成するために必要な自動化が搭載されています。

可用性の高いストレージ

データベースの統合によって、それらのシステムに関連付けられるストレージのパフォーマンスと可用性に関する要件が引き上げられます。ストレージで停止時間が発生すると、多くのデータベースに同時に影響を及ぼす可能性があります。Exadataシステムの場合、どのような統合環境でも、ハードウェア障害に対する保護を強化し、保守作業中にも運用を継続できるように、高冗長性ストレージ（三重ミラー化）を使用する必要があります。Oracle CloudのすべてのExadataシステムは、高冗長性ストレージ（ディスク・グループ）を使用して構成されています。

仮想マシンの構成

仮想マシン（VM）を使用する場合、各物理サーバーに2～4つの仮想マシンを構成することで、過度な複雑さや管理上のオーバーヘッドを加えることなく、必要な場所で独立性を確保できます。仮想マシンはオンプレミスのExadata構成ではオプションですが、Exadata Cloud ServiceとExadata Cloud@Customerの構成では、オラクルのチームと顧客の業務チームの間で管理機能の独立性を確保する上で不可欠です。複数のVM構成の場合、各デプロイメント・タイプで構成できる仮想マシンの最大数を把握しておくことが重要です。Exadataでの仮想マシンの使用について詳しくは、次のドキュメントを参照してください。
OVM（Oracle仮想マシン）：https://www.oracle.com/technetwork/database/availability/exadata-ovm-2795225.pdf
KVM（カーネル仮想マシン）：https://www.oracle.com/a/tech/docs/exadata-kvm-overview.pdf

メモリ構成

このセクションではサーバー構成を包括的に取り扱うことはしませんが、データベース統合環境においては以下のトピックに特に注意する必要があります。データベース統合環境におけるメモリ管理は、複数のデータベースにわたって影響を及ぼす可能性があることから、特に重要になります。LinuxおよびOracleデータベースの以下のメモリ管理設定を使用してください。

- Linux Huge Pages
- Oracle Database USE_LARGE_PAGESパラメータ

Linuxでは特定のワークロードの種類（OLTPまたはデータウェアハウス）向けに設計された割合で、システム・メモリの一部をHuge Pagesに割り当てる必要があり、そのHuge Pages領域内にOracle SGAを配置する必要があります。Oracle DatabaseでUSE_LARGE_PAGES=ONLYと指定することで、SGAがHuge Pages領域に配置されるようになります。そうすることで、影響を受けるデータベース・インスタンスが起動しなくなりますが、この設定では、システム上の他のデータベースに影響を及ぼしうるメモリ不足状態を防ぐことを目的としています。Linux Huge Pagesの使用について詳しくは、MOS 361468.1とMOS 401749.1を参照してください。

Multitenantのコンテナ・データベース

Oracle Multitenantデータベースを実行する物理マシンまたは仮想マシンのそれぞれに、アップグレードを容易にする目的で、少なくとも2つのコンテナ・データベース（CDB）を用意する必要があります。アップグレードの実行中はCDB間でプラガブル・データベース（PDB）を移動できます。または、管理の複雑さを抑えるために、CDB全体をアップグレードすることもできます。各サーバー（物理または仮想）の1つのCDBを現在のデータベース・バージョン用に使用して、アップグレード実行中に2つ目のCDBを次のデータベース・バージョン用に使用します。データベースは、再配置操作を使用してアップグレードの実行中にコンテナ間で移動できます。

構成チェック・ツール

オラクルは多数の構成チェック・ツールを提供しており、これらのツールは最新のベスト・プラクティスの構成チェックに合わせて頻繁に更新されます。構成チェック・ツールには、以下のものがあります。

- DB Security Assessment Tool（DBSAT）
- exachk

Oracle Database Security Assessment Tool（DBSAT）は、Oracleデータベースのセキュリティ脆弱性のチェックに使用するツールです。exachk（EXA Check）ツールは、Exadataシステム全体の包括的なヘルス・チェックであり、ストレージ、ネットワーク・オペレーティング・システム、Grid Infrastructure、およびデータベースを評価します。重要なExadata MAA構成のプラクティスやソフトウェアの脆弱性がターゲット・システムで公開された場合に強調表示されます。

Data GuardとActive Data Guard（読取り用レプリカ）

Oracle Data GuardはOracle9i以降、20年以上にわたって業界屈指のディザスタ・リカバリ（つまり“スタンバイ”データベース）機能を提供してきました。Data Guardのスタンバイ・データベースは、REDOログに含まれるデータベース・トランザクションの自動適用によって、“プライマリ”ソース（本番）データベースと常に同期されます。Data Guardのスタンバイ・データベースが消費するリソースは、災害発生シナリオで有効化されるまでは、ソース（“プライマリ”）データベースで消費されるリソースのほんの一部です。
Data Guardのスタンバイ・データベースはおもにディザスタ・リカバリの間に使用することが想定されています。一方、Active Data Guard（ADG）では、スタンバイ・データベースがプライマリ・データベースによって生成されたREDO変更ストリームを基に更新されている間でも、そのスタンバイ・データベースを“読取り用レプリカ”としてアクティブに問い合わせることができます。また、Data Guardでは特に物理的なデータ破損の自動ブロック修復機能があるため、保護状態がいっそう強化されます。

Exadataのインフラストラクチャ とDBバージョンの相互運用性

Exadataのインフラストラクチャ・レイヤーは、Exadataシステム上で稼働するデータベースの基礎となるソフトウェア・レイヤーから構成されます。Oracle Database、コンテナ・データベース、プラガブル・データベースは、これらのインフラストラクチャ・レイヤーの上にインストールされます。基盤のExadataインフラストラクチャ・レイヤーのリリースごとに、複数のOracle Databaseバージョンがサポートされます。バージョンの相互運用性については、MOS Note 888828.1、およびExadata Cloud Serviceソフトウェア・バージョン2333222.1に記載されています。

MAAリファレンス・アーキテクチャ

ある共通のインフラストラクチャ上でデータベースを統合すれば、可用性の観点で危険性が高まります。Oracle Maximum Availability Architecture（MAA）のベスト・プラクティスでは、全範囲の可用性とデータ保護に対応した4つのHAリファレンス・アーキテクチャを定義しています。これらは、データベース統合環境における可用性とデータ保護のニーズに対応するために必要となります。このアーキテクチャ（4つのHA層）は、プラチナ、ゴールド、シルバー、ブロンズと呼ばれており、これらのサービス・レベルは図3のとおりです。

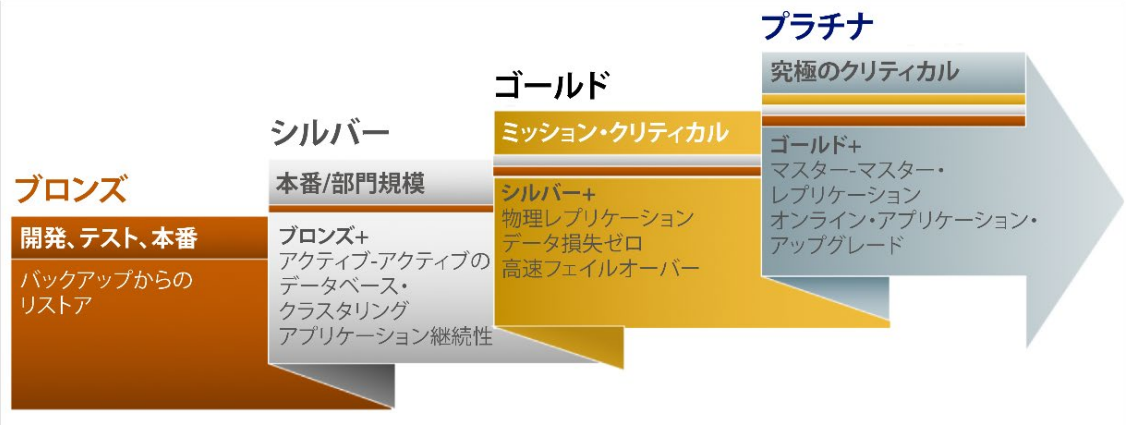

図3：HAとデータ保護のサービス・レベル

各層は異なるMAAリファレンス・アーキテクチャを使用して、最小限のコストと複雑さで所定のサービス・レベルを確実に達成する、最適なOracle HA機能をデプロイします。これらのアーキテクチャは、データ破損、コンポーネント障害、システムやサイトの停止といったあらゆる種類の計画外停止と、ソフトウェアの保守、移行、データベースのアップグレードなどを目的とする計画停止に対応しています。

ブロンズのMAAリファレンス・アーキテクチャでは、基本的なデータベース・サービスが必要最低限のコストで提供されます。コストと実装の複雑さが軽減されている代わりに、高可用性とデータ保護のレベルが低減されています。このアーキテクチャは、テストや開発で使用されるデータベース、重要度が低い本番用のアプリケーションやデータベースに適しています。ブロンズ・アーキテクチャでは、Oracle Enterprise Editionに含まれる高可用性機能が使用されます。ブロンズは、Oracle Databaseのシングル・インスタンスまたはマルチテナント・アーキテクチャをデフォルトとします。Oracle RestartまたはOracle Clusterwareの高可用性機能を使用して、障害が発生したインスタンス、データベース・サーバー、関連の管理サービスが再起動されます。サイトの完全な停止という最悪のシナリオでは、セカンダリ・ロケーションでこのようなタスクを実行するにはさらに時間が必要であり、停止時間が数日に及ぶ可能性もあります。

ブロンズのリファレンス・アーキテクチャでは、リカバリ速度を最大化するために、同じデータセンター内にローカル・バックアップを保持することが常に推奨されます。サイトの停止や災害に備えて、リモート・データセンターにバックアップの2つ目のコピーを保持することも推奨されます。Oracle Cloud Database Backup Serviceを使用すると、オンプレミス・データベースのクラウドベースのバックアップを保持できます。シルバーでは、データベースの可用性が一段階向上し、データベース・インスタンスやサーバーの障害時のほか、オペレーティング・システム、データベース、Grid Infrastructureソフトウェアのアップデート（必要なセキュリティ・アップデートを含む）をはじめとする様々な計画保守作業において、停止時間が最小限またはゼロに抑えられます。シルバーでは、高可用性の確保のために、Oracle Real Application Clustersを使用したクラスタリング・テクノロジーが追加されています。システムの停止時にクラスタを再起動できない場合に備え、Oracle Recovery Managerによるデータベース向けに最適化されたバックアップによって、データが保護され、可用性がリストアされます。アプリケーション・コンティニュイティによって、処理中のトランザクションを高い信頼性で再実行できるため、ユーザーがシステムの停止に気付くことはありません。

ゴールドは、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・レベルを大幅に引き上げたものであり、短い停止時間ですべての計画外停止と大規模なデータベース・ソフトウェア・アップグレードを実行できるようになっています。ゴールドには、Oracle Active Data Guardを使用したデータベース認識型のレプリケーション・テクノロジーが加わり、本番データベースの1つ以上のレプリカが同期されるため、データの保護と可用性が強化されています。Far Syncは、リアルタイムのデータ保護と可用性の範囲を拡大し、本番サイトとディザスタ・リカバリ・サイトが遠く離れている場合でもサポートします。データベース認識型のレプリケーションによって、ストレージ・レプリケーション・テクノロジーで可能となる程度を上回る、大幅に強化されたHA、ディザスタ・リカバリ、データ保護が実現されます。また、これによって常にすべてのレプリカをアクティブに利用できるようになるため、コスト削減と投資回収率の向上の両方が実現します。

プラチナはゴールド層の保護を土台とし、Oracle GoldenGateを加えることで、双方向機能による停止時間ゼロのアップグレードと移行を可能にしています。プラチナ層のGlobal Data Servicesによって、レプリケートされたデータベース環境のサービス管理とワークロード・バランシングが自動化されます。また、プラチナ層には、エディションベースの再定義（EBR）とシャーディングのための最適なオプション構成が含まれています。EBRは、アプリケーションのアップグレードをオンラインで実行できるようにする機能で、シャーディングは、独立したデータベースのファーム全体のデータを、共通キー（シャード・キー）を使用してインテリジェントに水平パーティション化する機能です。これらのテクノロジーはそれぞれ、プラチナ層に導入するには追加の作業が必要になりますが、停止時間やデータ損失が許容されないもっともクリティカルなアプリケーションにきわめて大きな価値をもたらします。

MAAリファレンス・アーキテクチャは、Oracle Database向けに最適化された標準的なインフラストラクチャを示すものであり、企業が異なるサービス・レベル要件に適したHAレベルを選択できるようになっています。標準化によってコストが削減され、ビジネス要件が変更された場合にも、データベースを今のHA層から次の層に、あるいは今のハードウェア・プラットフォームから別のハードウェア・プラットフォームに容易に移行できます。なお、これまで説明した各層特有のOracle HA機能に加えて、すべての層に、構成、監視、アラート、管理のためのOracle Enterprise Manager Cloud Controlも含まれています。

詳しくは、MAAアーキテクチャのドキュメント（https://www.oracle.com/webfolder/technetwork/tutorials/architecture-diagrams/high-availability-overview/high-availability-reference-architectures.html）、Exadata MAAアーキテクチャのプレゼンテーション（https://www.oracle.com/a/tech/docs/exadata-maa.pdf）、およびExadata Cloud MAAアーキテクチャのプレゼンテーション『Oracle Cloud: Maximum Availability Architecture Presentation』（https://www.oracle.com/a/tech/docs/cloud-maa-overview.pdf）を参照してください。

MAAのデータベース統合への適用

非統合環境に期待されるサービス・レベルは、統合環境とは大きく異なります。たとえば、開発者や部門が使用するスタンドアロン・データベースでは、システムの停止による中断のレベルは、小規模なユーザー・ベースに制限されます。しかし、この同じデータベースを他の100個のデータベースと統合し、各データベースで異なる部門やユーザー・コミュニティをサポートする場合には、影響ははるかに大きくなります。統合データベースに影響を及ぼすシステム停止による企業全体の中断のレベルは100倍に膨れ上がるため、HAやデータ保護の優先順位が高くなるのです。本書で説明する原則は、従来型ハードウェア・アーキテクチャ上のオンプレミスのデータベース・デプロイメントのほか、オンプレミス、Cloud@Customer、Exadata Cloud Serviceの各環境にデプロイされたExadataに同様に適用されます。

仮想マシンによる独立性の確保

オラクルは、仮想マシンを使用するシステムでの物理サーバーあたりの仮想マシンをおおよそ2〜4個までにすることを推奨しています。各仮想マシンに付随する管理上のオーバーヘッドの追加を最小限に抑えるためです。仮想マシンを使用して、環境に応じてデータベースのグループを分離したり（開発環境をテスト環境から分離する）、ガバナンス要件に合うようにデータベースを分離したり（PCIデータをPCIではないデータから分離するなど）することが可能です。また、仮想マシンを使用して、データベースのサブセットを他のサブセットから分離することで、アップグレードやその他のO/Sレベルのシステム保守タスクの影響を抑えることもできます。Oracle Real Application Clustersでは、ユーザー・セッションを仮想クラスタの別のノードに再配置することで、停止時間ゼロの保守が可能になります。そのため、Oracle RACでは、保守の独立性を確保するためにVMあたり1つのデータベースをデプロイするというアプローチが不要になり、大量のVMの存在による管理上のオーバーヘッドも発生しません。

統合環境でのパッチ適用とアップグレード

パッチ適用とアップグレードは、統合環境では特に難しい作業になります。複数のデータベースとビジネス・アプリケーションが共通の一連のハードウェアと共通のソフトウェア・スタックに依存することになるため、アップグレードを容易にするために適切な独立性を確保することが不可欠です。たとえば、CDBレベルでは1回のアップグレードでも、そのコンテナ内のすべてのPDBに影響します。この“多数の一元管理”ができることによって、管理者の作業時間は大幅に削減される一方で、アップグレード調整の必要性が高まります。そのため、調整しやすくするために、SLAとソフトウェア更新サイクルに従ってデータベースをグループ化する必要があります。また、Grid Infrastructureのパッチのローリング適用、Oracle Real Application Clustersのパッチのローリング適用、Data GuardのStandby Firstパッチ、Data Guardの一時ロジカル・ローリング・アップグレード、Oracle GoldenGateなどの機能を使用すれば、停止時間をゼロまたはほぼゼロに抑えながら、1回の操作でCDB全体をアップグレードできます。
CDB内部の個別のPDBをアップグレードするために、アップグレード後のバージョンで新しいCDBを作成し、PDBをその新しいCDBに再配置することができます。これにより、PDBの移行操作中は停止時間が発生しますが、固有のアプリケーション・ニーズに対応するために必要になると思われる柔軟性を確保できます。また、運用の複雑さは増しますが、Oracle GoldenGateにも停止時間を最小限またはゼロに抑えてPDBをアップグレードするオプションがあります。それは、異なるCDB間、および異なるバージョンのデータベース間でレプリケートするというものです。

アプリケーション・コンティニュイティによる可用性の確保

データベースの統合には、各物理サーバーまたは仮想マシン、あるいはその両方で実行される複数のデータベースが関わります。Oracle Real Application ClustersとActive Data Guardでは、計画停止、計画外停止の両方で高可用性を確保できます。Oracle Databaseでも、以下の機能により継続的なアプリケーション・サービスの可用性を確保できます。

- 透過的アプリケーション・フェイルオーバー（TAF）
- 透過的アプリケーション・コンティニュイティ（TAC）

計画保守では、RACクラスタのノード間で、またはActive Data Guardのスタンバイ・データベースに対してセッションがドレインされリバランスされます。一方、処理中の（アクティブな）セッションは、TAFまたはTACを使用して、RACクラスタのノード間で再配置できます。RACノード間でのセッションの自動再割当ては、物理サーバーまたは仮想マシンの障害発生時にも実行されます。高可用性が求められる統合環境のデータベースでは、これらの機能を有効化した構成にしてください。セッションのドレインとリバランス、およびTAFとTACの使用について詳しくは、以下の継続的な可用性に関するホワイト・ペーパーを参照してください。
https://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/adb-continuousavailability-5169724.pdf

サーバーあたり複数のデータベースを使用する場合の考慮事項

各物理サーバー、仮想マシン、または物理マシンや仮想マシンから成るクラスタ上に、非PDB（非マルチテナント）データベースを複数設置することで、統合を実現できます。これらのデータベースでは、Oracle Multitenantの利点、たとえばバックグラウンド・プロセス数の削減、共有メモリの使用量の削減、システム・リソースの動的共有、一元管理による管理上の利点などは得られません。非マルチテナントの複数のデータベースでは、Oracleソフトウェアの単一のインストールを共有できます。これは、共有Oracleホーム（$ORACLE_HOME）構成と呼ばれています。この場合は、アップグレードしやすくするために、少なくとも2つのOracleホームを用意してください。

Multitenantの考慮事項

Oracle MultitenantではOracle Maximum Availability Architecture（MAA）を使用して、統合環境のHAとデータ保護の要件に対応します。HAおよびデータ保護に関するMAAの一般的な目標に加えて、マルチテナント・アーキテクチャのコンテキスト固有の目標も存在します。

一元管理による簡潔性：MAAベスト・プラクティスでは、HAおよびデータ保護の目標を達成し、かつ最大限のコスト面の利益（資本コストと運用コストの面で）が得られるように、大規模な統合環境の管理を容易にスケーリングできる必要があります。

独立性：MAAベスト・プラクティスでは、単一のPDBに影響を及ぼす問題が、CDB内の他のPDBの可用性にまで影響を及ぼさないようにする必要があります。

表面的には、最初の2つの目標は矛盾しているように見えます。“n”個の環境を分離しようとすれば、通常は、個別管理の必要がある“n”個の異なる環境が生成されるためです。Oracle MAAでは、マルチテナント・アーキテクチャを利用することで、独立性ができる限り損なわれないようなHAを達成しながら、かつ一元管理による大きな利点を得ることもできます。

Oracle Multitenantでは、CDBにいくつのPDBを格納するかにかかわらず、CDBを単一のデータベースとして管理できます。100個のPDBがあるCDBでは、多くの保守タスクや運用タスクを100分の1に削減できます。Recovery Managerバックアップが1つになり、ディザスタ・リカバリ用のOracle Active Data Guardスタンバイが1つになり、アップグレードやパッチの対象のデータベースが1つになるためです。Oracle MAAには、適宜他のPDBから分離してPDBを管理できる柔軟性もあります。次に例を示します。

ある個別のPDBをリストアする必要がある場合、Recovery Managerでは同じCDB内で稼働中の他のPDBに影響を及ぼさずにそのリストアを実行できます。注意点として、接続したばかりのPDBは、次の定期CDBバックアップの実行前に問題が発生した場合でもリカバリできるように、その接続操作後すぐにバックアップするようにしてください。

高速ポイント・イン・タイム・リカバリが必要な場合、Oracle Multitenantの最初のリリース（12cR1）ではCDBレベルでフラッシュバック・データベースを利用できました。Oracle Database 12c Release 2では、フラッシュバック・データベースを他のPDBの可用性に影響を及ぼさずに個別のPDB上で利用できるようになっています。

個別のPDBでCDBの他のPDBに影響を及ぼすと思われる問題が発生した場合、その問題のPDBを容易に切断して別のCDBに接続し、そのCDB内で問題を分離して解決することができます。また、切断/接続操作を使用すれば、元のCDB内で実行中の他のPDBに影響を及ぼさずに柔軟にパッチを適用できます。問題の解決後、そのPDBは新しいCDBに残しておくことも、新しいパッチのインストール後に元のCDBに再配置することもできます。

プライマリCDBデータベース上の1つのPDBが破損し、そのPDBが健全な状態にあるスタンバイ・データベースが存在する場合は、そのスタンバイからPDBを切断して新しいCDBに接続することで、単一PDBのフェイルオーバーを実行できます。Oracle Database 12c Release 2では、この機能はData Brokerコマンドライン・インタフェースで自動化されます。詳しくは、My Oracle Support Note：Unplugging a Single Failed PDB from a Standby Database and Plugging into a New Container（Doc ID 2088201.1）を参照してください。

Oracleリソース・マネージャは、ワークロードの急激な増加や、リソース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・リソースの割当て分を超える量が消費されないようにします。

計画外停止の管理

表1に、マルチテナント・アーキテクチャでデータベースに影響を及ぼしうるさまざまな計画外停止について示します。この表では、前述のHA層のそれぞれで利用可能な、停止に対処するためのOracle HAソリューションも示しています。

表1：マルチテナント・アーキテクチャ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、プラチナのソリューション	リカバリ期間（RTO）	データ損失（RPO）
インスタンス障害	ブロンズ：データベースの再起動	サーバー再起動後、数分	ゼロ
	シルバー、ゴールド、プラチナ： Oracle RAC Application Continuity	Oracle RACの場合は数秒 Application Continuityの場合はゼロ	ゼロ
永久的なDBノード障害	ブロンズ：リストアおよびリカバリ	数時間～1日	ゼロ
	シルバー、ゴールド、プラチナ： Oracle RAC Application Continuity	Oracle RACおよびExadataのノード停止の高速検出を使用する場合は数秒 Application Continuityの場合はゼロ	ゼロ

ストレージ障害	すべての層：Exadata上のAutomatic Storage ManagementおよびExadata高冗長性（三重ミラー化）	データベースの停止時間ゼロ	ゼロ
物理的なデータ破損	ブロンズ、シルバー：基本的な保護、一部の破損ではPDBまたはCDB全体のリストアやリカバリが必要	リストアおよびリカバリが必要な場合は数時間	最後のバックアップの時間による
	ゴールド、プラチナ：Oracle Active Data Guardによる包括的な破損保護と自動ブロック修復	物理的な破損に対してActive Data Guardの自動ブロック修復を使用する場合はゼロ Data Guardのファスト・スタート・フェイルオーバー（FSFO）を使用して修復する場合に、書込み消失による破損が発生した場合は数秒～数分	物理的な破損に対して自動ブロック修復を使用する場合はゼロ SYNCまたはFAR SYNCを使用する場合はゼロ ASYNCを使用する場合、またはプライマリ上で書込み消失が発生した場合は数秒
論理的なデータ破損	すべての層：論理的な障害についてはフラッシュバック・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	検出時間による	障害の程度による
	ブロンズ、シルバー：一部の論理的なブロック破損では、PDBまたはCDB全体のリストアやリカバリが必要	数時間～1日	障害の程度による
	ゴールド、プラチナ：DB_BLOCK_CHECKINGが有効な場合、REDO Applyの実行中はスタンバイによって論理的な破損を防止、検出。検出後は、最小限の停止時間で、フェイルオーバーを数秒で完了可能	手動フェイルオーバーの実行後、数秒	数秒
クラスタ、データベース、またはサイト全体の障害	ブロンズ、シルバー：リストアおよびリカバリ	数時間～1日	最後のバックアップ以降。Oracle Cloudでは、最後のアーカイブ・バックアップから30分以内
	ゴールド、プラチナ：Oracle Active Data Guardのファスト・スタート・フェイルオーバーを使用してセカンダリにフェイルオーバー	数秒～2、3分	ゼロ～ほぼゼロ
単一PDB障害	ブロンズ、シルバー：PDBのリストアおよびリカバリ	数分～数時間	最後のバックアップ時点のリカバリ・ポイント。 Oracle Cloudでは、最後のアーカイブ・バックアップから30分以内
	ゴールド、プラチナ：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）	数秒～数分	Data GuardとPDBフェイルオーバーの場合はゼ
パフォーマンス低下	すべての層：Database Resource Managerによる影響範囲の分離	影響を受けたデータベースでのサービス品質の低下	ゼロ

個別のPDBに伴う問題の多くは、他のPDBに影響を及ぼさずに修復できますが、分離が必要になる状況もあります。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必要がある場合があります。このシナリオでは、新しいOracleホームとCDBを作成してから、PDBを切断して、そのPDBを新しいCDBに接続することが推奨されます。そうすれば、元のCDB内の他のPDBにまったく影響がないと100 %確信して、問題を解決できます。問題を解決した後は、そのPDBを切断して元のCDBに接続するか、何らかの時点までそのPDBを新しいCDBに残すことができます。

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も機能します。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。以下の表2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2：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、プラチナのソリューション	予測される停止時間
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	すべての層：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、オンライン再編成および再定義	ゼロ
データベースとGrid Infrastructureソフトウェアの更新	すべての層：CDB全体へのオンライン・パッチ適用が可能（該当する場合） シルバー、ゴールド、プラチナ：CDB全体でOracle RACのローリング・アップグレードとApplication Continuityの利用が可能 ゴールド、プラチナ：ゴールドおよびプラチナでは、まずStandby-Firstパッチの推奨事項に従って最初にスタンバイにソフトウェアの変更を適用して、そのソフトウェアの変更を追加の注意項目として評価してから、プライマリにOracle RACのローリング・アップグレードとアプリケーション・コンティニュイティを適用する	ゼロ 数秒～数分 ゼロ サービスの再配置の場合はゼロ アプリケーションの停止ゼロ
データベース・アップグレード（例：18cから19cへ）	すべての層：PDBを対象のソフトウェア・リリースのCDBに再配置することが可能 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：CDBを対象のリリースにアップグレードし、アップグレードした新しいCDB/PDBに接続をスイッチオーバーする	推定では数秒～1時間（データファイル・コピー・オプションなしの場合）。再配置後にPDBメタ・データをアップグレードする必要がある 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：PDBを、対象とするアプリケーション変更が適用されたGoldenGateレプリカにスイッチオーバーすることが可能 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を削減し、リスクを低減。運用費を削減し、全体的な安定性と可用性が向上

統合前の容量使用率の測定

このセクションでは、容量使用率の継続的な監視方法ではなく、統合環境への移行前に実施する容量使用率の測定方法について取り上げます。監視には、継続的な運用サポート、関連ツール（Oracle Enterprise Managerなど）、それらのツールの実装などが関係しますが、これらについては本書では割愛します。システム・リソース使用率の正しい測定は、リソース計画とリソース管理の構成にとって不可欠なものであり、詳しくは本書で後述します。

重要なこととして、データベース・パフォーマンスとデータベースに割り当てられるシステム容量は密接に関係していますが、結局のところ、この2つは異なるシステム管理の側面です。データベースに割り当てるリソースを追加することで、多くの場合パフォーマンスが改善する一方で、データベースのパフォーマンス・チューニングによって多くの場合リソース使用率が減少します。したがって、パフォーマンスと容量は互いに関連していますが、個別に取り扱う必要があります。このセクションでは、各データベースに割り当てられるリソースの量（つまり容量）について取り上げますが、データベースやアプリケーションのパフォーマンス・チューニングについては割愛します。

オラクルの自動ワークロード・リポジトリ（AWR）は、Oracle Diagnostic Packを使用している場合に、Oracleデータベースや、Oracleデータベースをホストするシステムについて、容量使用率を分析するための詳細データを提供します。AWRはこのような詳細分析に使用できる価値あるツールですが、このセクションでは、ほぼすべての環境で使用可能な、広範かつ最小限のアプローチを説明します。

容量使用率の測定を開始するタイミングは、各データベースを統合環境に移行する前です。統合環境への移行を計画するのと同時に、ソースにおけるデータベースの平均およびピーク時のリソース消費について把握することが重要になります。ターゲットの統合環境への移行が完了した後も、ワークロードが変化したときに必要に合わせて容量の割当てを調整できるように、容量使用率を追跡（監視）してください。データベースを本番環境に移行する前に、テスト環境でパフォーマンスと容量のニーズについて検証することも常に推奨されます。

容量使用率の測定値は、以下の3つのレベルで収集する必要があります。

システム・レベル。CPU、メモリ、ストレージの使用率（ストレージの領域とI/O）が許容できるレベルであるかを確認するために、サーバーおよびストレージを監視します。現行システムよりも多くのデータベースを一連のサーバーやストレージに統合する予定の場合は、空き容量、つまりヘッドルームを監視する必要もあります。

データベース（またはCDB）レベル。リソース使用パターンを見極めるためには、データベースを評価する必要があります。また、リソース・マネージャ（現在使用中の場合）によってリソース使用が制限またはスロットルされているかを判断するためにも、データベースの評価が必要です。リソース・マネージャの制約のためにCPUまたはI/Oの待ち時間が発生している場合は、データベースまたはコンテナ・データベースに割り当てられているリソースを増やす必要がある可能性があります。適切なリソース割当てを行うため、またはチューニングが必要な場合にも、データベース・パフォーマンスの監視は重要です。

PDBの監視（Multitenantのみ）。各PDBは、データベースと同じ方法で監視する必要があります。実際のリソース使用率がCDBのリソース・プランまたは個々のPDBの設定によって保証されているリソースを超過している場合は、PDBをCDBに追加するにつれてパフォーマンスが低下すると見込まれます。予測されるパフォーマンス低下が許容できないレベルである場合は、そのCDBにそれ以上PDBを追加しないでください。

容量に関する主要インジケータについて以下の表に示します。

表3：容量に関する主要インジケータ

監視/管理	ガイドライン
CPU	**システム（物理サーバーまたは仮想マシン）：** 入手可能なツールまたは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リプションされているかを判定します。表4「環境別のオーバーサブスクリプションに関するガイドライン」で示すとおり、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。 既存システムのデータベースのCPU使用率は、データベースを新しいシステムに統合するためのベースラインとして使用されます。古いハードウェアから移行したデータベースの場合、Exadataシステムでは必要なCPUリソースが減少することが多くあります。しかし、Exadataへの移行によりCPU使用率がどれほど減少するかは、各データベース固有のワークロードによって異なります。 そのため、初期デプロイメントのベースラインとしては、既存システムのCPU使用率が使用されます。リソース・プランを有効にしたデータベースでは、以下の問合せを使用して、そのデータベースのCPUリソース使用率（実行中セッション、待機中セッションを含む）を測定できます。 select to_char(begin_time, 'HH24:MI') time, sum(avg_running_sessions) avg_running_sessions,

	sum(avg_waiting_sessions) avg_waiting_sessions from v$rsrcmgrmetric_history group by begin_time order by begin_time; この問合せによる平均実行中セッション数から、必要なCPU_COUNTを決定できます。ただし、平均待機中セッション数（Average Waiting Sessions）が不定期にゼロより大きく、パフォーマンスが許容できるレベルではない場合、CPU_COUNTの値を大きくする必要があります。 **データベースまたはCDB：** インスタンス・ケージング、CDBリソース・マネージャ、またはPDBごとのCPU管理が有効になっている場合は、インスタンスが実際に使用したCPUの量と、必要であるにもかかわらず使用を妨げられたCPUの量を監視します。MOS Note 1338988.1の説明に従って、v$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、実際に使用されたCPUの数を示します。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スロットルを示します。この値は、追加の必要なCPU数に相当します。 CDBの場合は、CPUの空き容量（ヘッドルーム）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・マネージャまたはPDBごとの管理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$rsrcmgrmetric_history実行結果の'avg_running_sessions'と'avg_waiting_sessions'を合計することによって、必要なCPUの総量を計算します。この合計がCDBインスタンス全体のCPU_COUNT未満である場合、この合計とインスタンスのCPU_COUNTの差がこのCDB上のCPUヘッドルームとなるため、PDBの追加を検討できます。この合計がインスタンスのCPU_COUNTを超えている場合、このCDBはすでに容量いっぱいに動作しており、ヘッドルームはありません。PDBを追加しても、CPUの競合を増やすだけです。そのため、追加のPDBは、既存のPDBとそのアプリケーションがある程度のパフォーマンス低下を許容できる場合に、ブロンズのCDBに対してのみ追加してください。監視はピークの時間帯に実行するようにします。詳しくは、MOS Note 1338988.1を参照してください。 **PDB：** 各PDBの実際のCPU使用量とCPU待機時間を監視します。v$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にわたる共有値の総数でそのPDBの共有値を割った値に基づきます。たとえば、PDBの共有値が1であり、共有値の合計が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$rsrcmgrmetric_historyの'avg_waiting_sessions'メトリックを使用して、PDBのCPU待機時間を表示します。ゼロ以外である場合、PDBのパフォーマンスは、CDBリソース・プランのその共有値または使用率制限を増やすことによって改善する可能性があります。詳しくは、MOS Note 1338988.1を参照してください。
16コア有効	**システム（物理サーバーまたは仮想マシン）：** ページングが発生しないようにします。ページイン率およびページアウト率がゼロまたは非常に低い状態であることを監視するために、vmstatコマンドなどのツールを使用できます。 メモリについては、オーバーサブスクリプションを使用しないでください。Linuxシステムでは、/proc/meminfoを評価することによって、割当て済みの合計システム・メモリを把握できます。そのO/Sレベルで割り当てられたメモリと、Oracleデータベースやクライアント・プロセスに割り当てられたメモリを比較します。SGA使用量は、各データベースのSGA_TARGETパラメータから計算できます。PGA使用量は、下記の統計値を使用して確認できます。 **データベースまたはCDB：** SGAはデータベースやシステム上の他のデータベースのパフォーマンスに影響を及ぼさないように、Huge Page内に配置される必要があります。USE_LARGE_PAGES=ONLYを設定することでSGAが強制的にHuge Page内に配置され、Huge Page領域が十分でない場合はデータベースは起動されません。その他の場合、SGAのサイズを収集する目的は容量使用率の把握のみです。 PGAの使用量は、PGA_AGGREGATE_TARGETによって大きく変わり、v$pgastatまたは監視ツールを使用して監視できます。PGAのサイズは、PGA_AGGREGATE_LIMITを構成している場合はこれが強い制限値になります。または、“maximum PGA allocated”統計値が、インスタンスがその起動以降に割り当てた最大PGAを示します。“total PGA allocated”統計値は、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、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。PGA_AGGREGATE_LIMITは、可能な場合は中断せずにPGAを削減したり、超過したPGAを使用しているセッションを終了させたりすることで、PGA使用量に強い制限を適用するものです。
I/O	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。これらのメトリックは、Enterprise Managerで参照することもできます。 OLTP I/OについてExadata Flash Cacheのヒット率を監視し、ヒット率が許容できない場合は、そのデータベースのFlash Cache割当ての増加分を評価します。待機時間がまだ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。 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_RとFC_IO_RQ_R_MISSを比較して、OLTPフラッシュ・ヒット率を計算します。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ヘッドルーム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%以上のときの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スロットル時間（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、データベースのスロットル時間（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スロットル時間を監視します。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、データベース間プランでそのデータベースの割当て/共有値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でのパフォーマンス低下を許容できる場合を除き、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スロットル時間を監視します。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の共有値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、既存のデータベースでのパフォーマンス低下を許容できる場合を除き、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・インジケータ	自動ワークロード・リポジトリまたはEnterprise Manager、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・インジケータ	自動ワークロード・リポジトリまたはEnterprise Manager、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。

統合のためのリソース管理

データベース統合とは、おもにインフラストラクチャと運用にかかるコストを削減する目的で、複数のデータベースを共有のコンピューティング/ストレージ・インフラストラクチャに配置することです。しかし、共有インフラストラクチャに対しては通常、それらの共有リソースに依存するビジネス・アプリケーションの責任者の間でパフォーマンスへの懸念が生じることになります。許容可能なパフォーマンスを提供するのに必要な適正量のシステム・リソースが各アプリケーションに割り当てられているとビジネス・オーナーに納得してもらうには、どうすればいいのでしょうか。その答えになるものがリソース管理です。

Oracleデータベースには、リソース管理のための多数の強力なツールが搭載されています。これらのツールにより、重要なビジネス機能をサポートするために必要となる適正量のリソースを確保しながら、複数のデータベースで単一のコンピューティング・インフラストラクチャを共有できるようになります。これらの機能を利用すれば、IT組織は複数の物理サーバーや仮想マシンをデプロイすることによる管理上のオーバーヘッドを負うことなく、リソースを適切に管理できます。Exadataはこれらのリソース管理機能をExadataストレージ層にまで拡張して、データベースが依存するリソース一式を包括的に管理できるようになっています。この章で説明したリソース管理のアプローチは、以下のセクションで説明する設定と同じ設定を使用して、オンプレミスのExadata、Exadata Cloud Service、Exadata Cloud@Customerに適用されます。

リソース管理の定義

クラウド環境でも従来のデータセンター環境でも、コンピューティング・リソースはあらゆる組織にとって大きなコストとなります。それらのコンピューティング・リソースの制御コストを適正管理することでビジネス・アプリケーションのパフォーマンスを確保し、あるビジネス・アプリケーションで発生したパフォーマンスの問題が別のビジネス・アプリケーションに影響することを防ぐことができます。オラクルのリソース管理ツールには、各データベース（ひいては各ビジネス・アプリケーション）に対して、想定される最小限の量のシステム・リソースが割り当てられるように、また、各データベースがその割当て済みリソース量を超えてリソースを利用することのないように、システム・リソースを管理できる機能が含まれています。オラクルのツールでは以下のリソースについて制御できます。

- コンピューティング（CPU）
- メモリ
- プロセス（データベース・セッション、パラレル問合せプロセス）
- I/O操作と帯域幅
- Exadataストレージおよびインターコネクト・ネットワーク
- ストレージ領域（ディスク、フラッシュ）
- Exadata Smart Flash Cache

コンピューティング、メモリ、プロセスの制御機能はすべてのプラットフォームにわたってOracle Database Resource Manager（DBRM）によって提供されます。一方、I/Oおよびストレージ関連の制御機能はExadataプラットフォーム上の専用のExadata I/O Resource Manager（IORM）によって提供されます。オラクルは、非Exadata環境向けにI/O率制限のための制御機能も提供していますが、ExadataストレージはOracleデータベースと深く統合され、ストレージとI/Oをはるかに堅牢に制御できるようになっています。

リソース管理は、いわゆる“ノイジー・ネイバー”問題、つまりあるアプリケーション（またはデータベース）が想定を超える量のリソースを消費しうるという問題からデータベースを保護します。たとえば、あるデータベース内で復帰しない（Runaway）SQL文が実行されると、同じコンピューティング・リソースやストレージ・リソースを共有する他のデータベースにも影響を及ぼす可能性があります。Oracleリソース・マネージャは、1つのデータベース（そのデータベース内で実行される、復帰しないSQL文を含む）が同じシステム上の他のデータベースに影響を及ぼすことを防ぎます。リソース・マネージャのコンシューマ・グループも、ある個別のデータベース内で実行された復帰しないSQLの影響を抑えることができます。

Exadataネットワーク・リソース管理は、Exadataストレージおよびクラスタのインターコネクト・ネットワーク上のネットワーク・トラフィックを管理するための自動的な機能であり、管理者による介入は必要ありません。この機能では、Exadataシステムのパフォーマンスを確保するために、重要度の高いネットワーク・メッセージが、重要度の低いメッセージよりも自動的に優先されます。

リソース不足（パフォーマンス問題）

データベースがリソース・プランで許可された量を超えるリソースを使用しようとしたときに、リソース不足の状態になります。一般的なリソース不足の原因は、アプリケーションのコーディング・エラー（SQLのコーディング・エラーを含む）、誤った削除操作などによる索引の消失、不正確なオプティマイザ統計です。これらの問題から、いわゆる“復帰しない”SQL文が実行されることになります。その結果、データベースがリソース・プランで許可された量を超えるCPU、I/O、その他のシステム・リソースを消費する（または消費しようとする）ことになります。影響を受けるデータベースはその後リソース・マネージャによる制限を受けます。その結果、そのデータベースのすべてのユーザーが影響を受ける可能性はありますが、同じシステム上の他のデータベースが影響を受けることはありません。リソース・マネージャのコンシューマ・グループを作成することによって、あるデータベース内での復帰しないSQL文の影響を抑えることができます。

復帰しないSQL文やその他の予期しない高負荷のワークロードのためにリソース不足が発生したとき、DBRMとIORMはパフォーマンス問題の根本原因にはなりません。復帰しないアプリケーションのSQL文やワークロードこそがその原因であり、DBRMとIORMはそのデータベースが同じシステム上の他のデータベースのパフォーマンスに影響を及ぼさないように、定義済みのリソース制限を適用しているだけなのです。

リソース・マネージャによる可用性の向上

Oracleリソース・マネージャ（DBRMとIORM）を使用すれば、データベースがプロビジョニング済みのシステム・リソースの量を超えてリソースを消費することはありません。このセクションで説明したとおり、これらのリソース制御機能はCPU、メモリ、O/Sプロセス、I/Oの使用を制御します。リソース・マネージャは、アプリケーションやデータベースが過剰なリソースを消費することを防ぎます。過剰なリソースの消費は、これらのシステムの可用性の問題や停止時間の発生につながる可能性があるからです。このセクションで説明したとおりの適切なリソース管理によって、コネクション・ストーム、PL/SQLアプリケーションのメモリ・リーク、復帰しないアプリケーションSQL文などのアプリケーション障害から保護できます。

リソース割当ての階層

システムのリソースは、各アプリケーションのビジネス・ニーズに従って、1つの階層内の連続するレイヤーを通じて再分配されます。リソースは、コスト制御のため、およびデータベースを互いに分離して“ノイジー・レイバー”問題を防ぐために必要となる粒度まで細分化した上で、割り当てて管理することができます。ノイジー・レイバー問題とは、あるデータベース、アプリケーション・サービス、マイクロサービス、ジョブ・タスク、またはユーザーが、システム上の他の要素に影響を及ぼす問題です。リソース割当ての階層には以下の要素が含まれます。

- 物理サーバー
- 仮想マシン
- データベースまたはコンテナ・データベース
- プラガブル・データベース
- コンシューマ・グループ
- ユーザーとジョブ

OracleデータベースまたはOracleデータベースのグループが稼働するシステムは、場合によっては複数の物理サーバーで構成されます。各物理サーバーには一定量のリソース（CPU、メモリ、有効なプロセス制限）が含まれ、それらは複数の仮想マシンに再分配することが可能です。仮想マシンのリソースは、1つ以上のデータベースまたはコンテナ・データベースに再分配され、割り当てられます。コンテナ・データベースの場合、それぞれに割り当てられたリソースも再分配して、1つ以上のプラガブル・データベースに割り当てることが可能です。最後に、1つのデータベース内のリソースを各種コンシューマ・グループに割り当てることができます。コンシューマ・グループとは、実質的には個別ユーザー、ユーザーのグループ、ジョブ・クラスなどです。

リソースのオーバーサブスクリプション

管理者は、システム・リソースをデータベース間で慎重に分配しようとして、ある特定の時点で特定のデータベースが必要とするリソースよりも多くの量を割り当てる傾向にあります。リソースは多くの場合、ピーク時の処理期間に合わせて割り当てられますが、比較的需要の少ない期間には使用率が低くなります。たとえば、小売業では繁忙期に合わせてリソースが割り当てられることが多く、税務関連業では税の申告時期のユーザー・ニーズに基づいてリソースが割り当てられ、財務会計システムでは、月末、四半期末、会計年度末の処理ニーズに合わせてリソースが割り当てられます。

オーバーサブスクリプションは、処理の需要の浮き沈みがアプリケーション間で異なることを利用するためのテクニックです。オーバーサブスクリプションは、ユーザーやデータベースがリソースについて同時にピーク時の需要を迎えることはないという前提で、システム上に物理的に存在する量を超えたリソースを割り当てることを意味します。このセクションでは、各リソース・タイプのガイドライン、ワークロード別のガイドライン、および開発環境、テスト環境、本番環境、ディザスタ・リカバリ環境などの環境別のガイドラインを含めて、オーバーサブスクリプションについて掘り下げます。

プラガブル・データベース間のオーバーサブスクリプション

コンテナ・データベースは、システムの安定性の問題を引き起こすことなく、すべてのリソースについてそのコンテナ内のプラガブル・データベース間でオーバーサブスクリプションを実行するための手段を提供しています。データベースのパフォーマンスが低下する可能性はありますが、コンテナ内のデータベースがシステム・レベルでサポート可能な量を超えてリソースを消費することはないため、オペレーティング・システムが影響を受けることはありません。

メモリのオーバーサブスクリプション

VMレベル、データベース・レベル、またはコンテナ・データベース間でのメモリのオーバーサブスクリプションは決して実行しないでください。ノード排除、ノード障害、極度のページング/スワッピングなどにつながる重大なパフォーマンスおよび安定性の問題が発生する可能性があるためです。メモリは仮想マシン間で慎重に分配した上で、データベース間およびコンテナ・データベース間でさらに分配します。
メモリのページングによって、運用の安定性の問題や深刻なパフォーマンス問題が発生する可能性があります。コンテナ内のプラガブル・データベースでは、パフォーマンスにそのような影響を及ぼすことなく、メモリのオーバーサブスクリプションを利用できます。

プロセスのオーバーサブスクリプション

各オペレーティング・システムのイメージは（ベア・メタル上で実行している場合でも仮想マシン内で実行している場合でも）、割り当てられたコンピューティング・リソース（プロセッサ・コア）に基づいて、特定数のO/Sプロセスしかサポートできないようになっています。オラクルは、データベースに割り当てられるプロセッサ・コアあたりのデータベース・セッション数64の範囲内で最大数を割り当てることを推奨します。特定のデータベースでの最適なセッション数はワークロードによって異なります。プロセスのオーバーサブスクリプションでは、単一のサーバー、仮想マシン、またはクラスタ上に位置する複数のテスト用、QA用、または開発用データベースのように、アイドル状態のデータベースとアクティブなデータベースが同時に存在することが前提となります。

コンピューティングおよびI/Oのオーバーサブスクリプション

仮想マシン、データベース、またはコンテナ・データベースのレベルで割り当てられるコンピューティング・リソースやI/Oリソースについては、システムの安定性の問題を引き起こすことなくオーバーサブスクリプションを実行することが可能です。ただし、この場合は容量使用率と起こりうるアプリケーション・パフォーマンスへの影響の間にトレードオフが生じます。コンピューティング（プロセッサ・コア）に対して緩やかなオーバーサブスクリプションが使用されている場合には、安定性の問題を引き起こすことなく、オペレーティング・システムによってクリティカルな処理の優先順位付けを効果的に管理できます。

重大性別のオーバーサブスクリプション

リソースのオーバーサブスクリプションによって空きリソースを十分に活用できますが、リソース使用率を十分に制御できるわけではありません。たとえば、1台のベア・メタル・サーバーまたは仮想マシン内で使用可能なすべてのCPUを2つのデータベースに割り当てることができるとします。一方のデータベースのアクティビティ量が多く、他方のデータベースのアクティビティ量が少ない場合は、オーバープロビジョニングによってそのすべてのリソースを活用できます。しかし、両方のデータベースのアクティビティ量が同時に多くなると、これらのデータベースでリソースが競合状態になり、一方または両方のデータベースが制御困難なパフォーマンスの問題を被ります。極端な状況では、システム・リソースが過負荷になったために、サーバーが不安定または無応答の状態になります。

ミッション・クリティカルなシステム、あるいは“最重要”システムでは、システムのパフォーマンス問題を回避するために、オーバーサブスクリプションは慎重に検討の上使用してください。以下の表に示すとおり、メモリのオーバーサブスクリプションはプラガブル・データベースの場合に限り使用してください。コンテナによって、システム・レベルでのオーバーサブスクリプションから保護されるためです。CPU、プロセス、I/Oについては、特に重要度の低い環境において、ビジネス・ニーズに応じたオーバーサブスクリプションが可能です。表4（下記）に、5種類のシステムについて4つの分類レベルで示します。テスト環境とQA環境はこの例では同じ重大性レベルと見なしています。

表4：環境別のオーバーサブスクリプションに関するガイドライン

重大性	環境	CPU	PROCESSES	メモリ	I/O
1	クリティカルな本番環境	CPUスレッド数の75％を超過しない	コア数 x 64	なし	I/O割当て率 x 2に制限
2	クリティカルでない本番環境	CPUスレッド数と同一	コア数 x 64	なし	I/O割当て率 x 2に制限
3	テスト環境	CPUスレッド数 x 2	コア数 x 128	なし	I/O割当て率 x 2に制限
4	QA環境	CPUスレッド数 x 2	コア数 x 128	50％（PDBのみ）	I/O割当て率 x 4に制限
5	開発環境	CPUスレッド数 x 2	コア数 x 128	50％（PDBのみ）	I/O割当て率 x 4に制限

Database Resource Manager（DBRM）の有効化

DBRMを使用して詳細なリソース管理プランを作成できますが、非CDBのデータベース・デプロイメントの大半では、単純にデフォルト・プランを使用するようにしてください。デフォルトのリソース管理プランを有効にするには、非CDBデータベースのそれぞれで以下のコマンドを実行します。

```
SQL> ALTER SYSTEM SET
        RESOURCE_MANAGER_PLAN = ‘DEFAULT_PLAN
        ‘ SID = ‘*‘
        SCOPE=both;
```

コンテナ・データベースでは、以下のコマンドに示すとおり、DEFAULT_CDB_PLANを使用します。この設定では、コンテナ・データベース下に格納されるすべてのプラガブル・データベースが対象となります。

```
SQL> ALTER SYSTEM SET
        RESOURCE_MANAGER_PLAN = 'DEFAULT_CDB_PLAN'
        SID = '*'
        SCOPE=both;
```

上記のコマンドでは、SPFILEがすべてのデータベースで使用されていることが前提となっており（SCOPE=both句で示されている）、これにより各インスタンスのメモリに加え、spfile内でもリソース・マネージャ・プランが設定されます。SID='*'句によって、この設定が非RAC環境またはRAC環境のすべてのインスタンスに適用されます。SPFILEの使用は、Oracleデータベース管理の"ベスト・プラクティス"とされています。

PDBパフォーマンス・プロファイルの有効化

"割当て方法"がプラガブル・データベース・レベルで使用されている場合には、コンテナ・レベルのDEFAULT_CDB_PLANで十分です。しかし、プラガブル・データベースのリソース共有機能を最大限に活用するには、PDBパフォーマンス・プロファイルを使用する必要があります。

```
DECLARE
   v_plan VARCHAR2(30) := 'cdb_profile_plan'; BEGIN
   DBMS_RESOURCE_MANAGER.clear_pending_area; DBMS_RESOURCE_MANAGER.create_pending_area;

   DBMS_RESOURCE_MANAGER.create_cdb_profile_directive( plan
                                => v_plan,
      profile                   => 'xxsmall',
      shares                    => 1,
      utilization_limit         => 4,
      parallel_server_limit => 4);

   DBMS_RESOURCE_MANAGER.validate_pending_area; DBMS_RESOURCE_MANAGER.submit_pending_area;
END;
/
```

この例では、"xxsmall"という名前のPDBパフォーマンス・プロファイルが作成されます。以下のコマンドにより、このPDBパフォーマンス・プロファイルを使用するようにプラガブル・データベースを変更します。

```
ALTER SESSION SET CONTAINER=pdb1;
ALTER SYSTEM SET DB_PERFORMANCE_PROFILE=xxsmall SCOPE=BOTH;
```

コンテナ・データベース内のPDBで必要となるすべてのパフォーマンス・プロファイルを作成し、そのコンテナ内のプラガブル・データベースに対して有効にしてください。パフォーマンス・プロファイルは、PDBがCDB管理者以外の個別の管理者によって管理される場合に備え、ロックダウン・プロファイルに含めることができます（Oracle 12cR2以降のリリース）。

プラガブル・データベースのロックダウン・プロファイル

プラガブル・データベースの大規模デプロイメントでは多くの場合、管理者のレイヤーが2つ存在します。コンテナ・データベース・レベルを管理する"フリート管理者"と、各コンテナ・データベース内のプラガブル・データベースを個別に管理する1人または複数人の管理者です。ロックダウン・プロファイル（Oracle 12cR2以降のリリースで使用可能）によって、プラガブル・データベースに割り当てるリソースをCDB管理者が制御して、各データベースに対して承認された量を超えるリソースをPDB管理者が割り当てることを予防できます。

Exadata I/O Resource Manager（IORM）の有効化

デフォルトでは、Exadata I/O Resource Manager（IORM）はすべての環境で有効化されますが、"basic"のI/Oリソース・プランが使用されます。このプランでは、どのデータベースがI/Oを生成しているのかにかかわらず、サイズと重大性に基づいてI/Oの優先順位が決定されます。以下のコマンドはIORMオブジェクトを"auto"に設定するものです。この設定によって、IORMのプランとプロファイルが使用できるようになります。

```
dcli -g ~/cell_group
        -l root cellcli
        -e alter iormplan objective = auto
```

この例では、"dcli"コマンドを使用して、システム内のすべてのExadataストレージ・セルにわたって、IORMプランの目標を"auto"に設定しています。

CELLCLIユーティリティを使用して、各セル上でコマンドを実行することもできます。

Exadata IORMのプランとプロファイル

Exadata I/O Resource Manager（IORM）では個別のデータベースに対してプランを使用するか、（Oracle 12cR2以降のリリースで）プロファイルを使用して類似したリソース要件を複数のデータベースに割り当てることができます。IORMでは、Exadataシステムの以下のリソースを制御できます。

- 他のデータベースとの相対的なI/O割当て率
- システムで使用可能なI/Oの制限
- Flash Cacheの制限（上限）
- Flash Cacheのサイズ（領域の予約）

SHARE句では、そのシステム上のすべてのデータベースを基準として、各データベースに対してシステム上のI/Oリソースの最小割当て率を指定します。たとえば、10個のデータベースのそれぞれにshare=10と割り当てた場合、各データベースは全体のI/Oリソースの10 %以上を受け取ります。データベースは、他のデータベースがアイドル状態の場合には、指定されたリソース割当て率を超えるリソースを消費できます。

SHAREではデータベースで使用可能な最小のリソースを設定する一方、LIMIT句ではデータベースの最大I/Oリソースを設定します。各データベースにLIMITを割り当てることで、リソース割当ての一貫性が高まり、より一貫性のあるパフォーマンスを実現できます。ただし、LIMITを使用すればアイドル状態のリソースを使用できなくなり、それがITアセットの使用率低下につながります。

個別のデータベースのIORMプランでは、データベースのDB_UNIQUE_NAMEと同じプラン名を使用します。プロファイルは、type=profile句を含むDBPLANのことで、以下の例に示すように、このディレクティブを個別のデータベースのプランではなくプロファイルとして指定します。

以下のExadata dcliコマンドは、構成内のすべてのセルにわたって、ORCLという名前のデータベースに対してIORMを作成または変更するものです。IORMプランは、各データベースのDB_UNIQUE_NAMEに基づいています。

```
dcli -g ~/cell_group
        -l root cellcli
-e alter iormplan DBPLAN=((name=ORCL, share=10, limit=21))
```

以下のExadata dcliコマンドはtype=profileを使用して、構成内のすべてのセルにわたって“myprofile”という名前のIORMプロファイルを作成または変更するものです。

```
dcli -g ~/cell_group
        -l root cellcli
-e alter iormplan DBPLAN=((name=myprofile, share=10, limit=21, type=profile))
```

同じIORMプロファイルを使用するすべてのデータベースに、同じ割当て率、制限、Flash Cacheが割り当てられます。管理性向上のため、データベースに割り当てるプロファイル（“シェイプ”）は、SMALL、MEDIUM、LARGEなどの少数にしておくことが推奨されます。この考え方について詳しくは、リソース・マネージャのシェイプに関するセクションを参照してください。

データベース・パフォーマンス・プロファイルの設定（Exadata IORM）

Exadata環境では、データベース（非PDB、PDBの両方）でIORMパフォーマンス・プロファイルを使用できます（Oracle 12cR2以降）。プロファイルを使用することで、各データベースに対して個別にIORMプランを設定する必要がなくなります。同じプロファイルを使用するすべてのデータベースに、IORMプロファイル内に設定されている同じIORMリソースが割り当てられます（次のセクションを参照）。

```
SQL> ALTER SYSTEM SET
        DB_PERFORMANCE_PROFILE = ‘myprofile‘ SID = ‘*‘
        SCOPE=both;
```

データベースのDB_PERFORMANCE_PROFILE設定は、前のセクションで説明したとおり、Exadataストレージの IORMプランに設定されたプロファイル名を指定する必要があります。

非Exadataプラットフォームの I/Oリソース管理（PDBのみ）

Oracle Database 12c Release 2以降で、非ExadataプラットフォームでOracle Multitenantを使用する場合、以下の設定によって、コンテナ・データベース内のプラガブル・データベースで基本的なI/Oリソース管理機能を利用できるようになります。

- MAX_IOPS

- MAX_MBPS

これらのデータベース・パラメータ設定により、ある程度のI/Oリソース管理機能を利用できるようになりますが、非ExadataストレージにはExadata Smart Flash Cacheは含まれていないこと、および非Exadataシステムでは、Flash CacheのサイズとFlash Cacheの制限を制御するIORM設定の代わりとなる設定がないことに注意してください。

Exadataのストレージ・ネットワーク・リソース管理機能（“basic” I/Oリソース・プランにもあります）ではタイム・クリティカルなI/Oがクリティカルでないi/Oよりも優先されますが、非Exadataシステムにはこの機能がありません。タイム・クリティカルなI/Oイベントの例としては、ログ・ファイルの同期、ログ・ファイルのパラレル書込み、ファイル・ヘッダーの書込み、チェックポイントで実行されるデータ・ブロックの書込み、バッファ・キャッシュの空き領域に対して実行されるデータ・ブロックの書込みなどが挙げられます。I/Oの重大性は、書込み中のオブジェクトやデータベース内のアクティビティとの関連で決定されます。この機能は、Exadataでのみ可能となるデータベース・ソフトウェアとハードウェアの統合に依存するものです。

O/Sプロセスの制御

データベース・プロセスが過剰に生成されないように、データベース管理者は各データベースで、アクティブなセッション数をCPU_COUNTの数以下に設定する必要があります。それに加えて、管理者は1台の物理サーバーまたは仮想マシン内のすべてのデータベースにおける総セッション数の上限を、その物理サーバーまたは仮想マシン内の総コア数の64倍に設定する必要があります（コアあたり2つのハイパースレッドを使用するサーバーではCPU_COUNTの32倍）。どのオペレーティング・システムでもプロセス数が多ければ、メモリのページング、CPU競合、さらにはノードの不安定化につながります。データベース接続数が多いからと言って必ずしもアプリケーションのスループットが高くなるわけではなく、実際には競合が増加し、アプリケーションのパフォーマンスが低下する場合もあります。パラレル問合せを使用する場合は、並列処理に多くのリソースを割り当てるために、セッション数も削減する必要があります。

システム内のプロセス総数に加えて、いわゆる“ログイン・ストーム”と呼ばれる状態の接続到着率もシステムの安定性の問題につながります。O/Sがそれらの接続の処理を試みるためです。以下の手法を1つ以上使用することで、プロセス総数と接続到着率を削減して、システムのパフォーマンスと安定性を向上させることができます。

Oracleリスナーの構成によって着信接続要求をスロットルして、データベース・ノードやインスタンスの障害後にログオン・ストームが発生しないようにします。

プロセス数とデータベース・サーバーへの接続数を**制限**します。そのために、接続プールを使用し、最大接続数を、予測されるアクティブな処理中セッション数を少し上回る値に設定します。

コストのかかるプロセス割当ておよび割当て解除を**抑制**します。そのためには、最小接続数と最大接続数を同じ数に設定して、動的な接続プールを使用しないようにします。

Oracleリスナーは、着信接続要求をスロットルして、データベース・ノードやインスタンスの障害後にログオン・ストームが発生しないように構成できます。Oracle Net Listener内の接続速度リミッター機能により、データベース管理者（DBA）はリスナーが処理する新しい接続数を制限できます。この機能を有効にしておくと、Oracle Net Listenerはリスナーが毎秒処理する新規接続数に対して、ユーザーが指定した最大値を設定します。

構成によっては、一連のエンドポイント、または特定のエンドポイントに対して速度を指定できます。この機能は以下の2つのlistener.ora構成パラメータによって制御されます。

CONNECTION_RATE_<listener_name>=number_of_connections_per_secondでは、速度制限のあるすべてのリスニング・エンドポイント全体に適用されるグローバル速度が設定されます。このパラメータを指定すると、指定されているすべてのエンドポイント・レベルの速度数値より優先されます。

RATE_LIMITは、特定のリスナー・エンドポイントに速度制限が課されていることを示します。このパラメータは、リスナー・エンドポイント構成のADDRESSセクションで指定されます。RATE_LIMITパラメータが0より大きい値に設定されている場合、エンドポイント・レベルで速度制限が課されます。

例：新しい接続のスロットルによってログオン・ストームを防ぎます。

```
APP_LSNR= (ADDRESS_LIST= (ADDRESS=(PROTOCOL=tcp)(HOST=)(PORT=1521)(RATE_LIMIT= 10))
(ADDRESS=(PROTOCOL=tcp)(HOST=)(PORT=1522)(RAT E_LIMIT=20))
(ADDRESS=(PROTOCOL=tcp)(HOST=)(PORT=1523))
)
```

上記の例では、エンドポイント・レベルで接続速度が指定されています。毎秒、ポート1521で最大10の接続が処理されます。ポート1522の接続は毎秒20に制限されています。ポート1523の接続は制限されていません。速度が超過すると「TNS-01158: Internal connection limit reached（内部接続制限に到達）」とログに記録されます。詳しくは、『Oracle Database Net Servicesリファレンス』ガイドを参照してください。

アイドル・セッションとアイドル・ブロッキング・セッションのタイムアウト

Oracle Database Resource Manager（DBRM）を使用して、アイドル・セッションとアイドル・ブロッキング・セッションのタイムアウトを制御し、システム全体の安定性を確保できます。アイドル・セッション、および他のセッションをブロックしているアイドル・セッションを制御するには、以下の設定を使用します。

- MAX_IDLE_TIME：アイドル・セッション全般（ブロックしているかどうかにかかわらず）の制御に使用
- MAX_IDLE_BLOCKER_TIME：ブロッキング・セッションのタイムアウトの制御に使用

MAX_IDLE_TIME設定は、データベース・パラメータを使用して制御することも、リソース・プラン・ディレクティブとして制御することも可能です。MAX_IDLE_BLOCKER_TIMEは、リソース・プラン・ディレクティブとしての設定のみが可能です。アプリケーションがシステム内にアイドル・セッションを残すことが常態化している場合や、アプリケーションでアイドル・セッションの終了を簡単に許容できない場合は、MAX_IDLE_TIMEではなくMAX_IDLE_BLOCKER_TIMEを使用することを推奨します。

データベースおよびPDBのリソース制御

各データベースのリソースは、各プラガブル・データベース（PDB）またはスタンドアロン（非PDB）データベース内の中核的な制御機能を使用して制御します。DBRM（またはExadataのIORM）はこれらの設定を使用して、使用可能なシステム・リソースを適切に管理し、データベース・パフォーマンスを想定レベルで維持しながらシステムの安定性も確保します。表5では、データベース間リソース管理の中核的な制御機能と、12cR1から19cまでのそれらの制御機能に対する機能拡張について示します。

表5：リリース別のデータベース・リソース制御機能（12cR1 – 19c）

	12cR1	12cR2	18c	19c
RESOURCE_MANAGER_PLAN	☒	☒	☒	☒
CPU_COUNT	☒	☒	☒	☒
SGA_TARGET	☒	☒	☒	☒
PGA_AGGREGATE_TARGET	☒	☒	☒	☒
PARALLEL_MAX_SERVERS	☒	☒	☒	☒
SESSIONS	☒	☒	☒	☒
PGA_AGGREGATE_LIMIT	☒	☒	☒	☒
MAX_IDLE_BLOCKER（リソース・プラン・ディレクティブ）	☒	☒	☒	☒
DB_PERFORMANCE_PROFILE		☒	☒	☒
MAX_IDLE_TIME		☒	☒	☒

この表と以下の詳細説明からわかるように、新しいリソース管理設定がOracle 12c Release 1、12c Release 2、Oracle 19cで導入されています。これらの設定によってリソースの管理性が向上し、データベース・レベルまたはシステム・レベルのリソース不足を原因とする可用性の問題も改善されます。

PGA_AGGREGATE_LIMIT設定（12cR1で導入）はデータベース内のすべてのセッションによって割り当てられるPGAメモリ総量を制御するものです。この設定によって、PGA内の過度なメモリ割当て（PL/SQLコード内のメモリ割当て超過など、さまざまな条件で発生する可能性がある）を制限することでシステム全体の可用性が向上します。

DB_PERFORMANCE_PROFILE設定（12cR2で導入）は、IORMプロファイルの使用を有効化するもので、1つのプロファイルで多くのデータベースのリソースを制御できるようになります。12cR2より前のリリースでは、IORMプランはデータベースのDB_UNIQUE_NAMEに従って、個別のデータベースに対して設定されます。

MAX_IDLE_TIME設定（12cR2で導入）は、アイドル状態のデータベース・セッションを自動的に終了できるものであり、この設定によって、システム・リソースを保持しているアイドル・セッションを終了させることで、データベースとシステムの可用性が向上します。アイドル・セッションの終了を許容できないアプリケーションも存在します。そのような状況では、MAX_IDLE_BLOCKER_TIMEを使用する方が適している可能性があります。

MAX_IDLE_BLOCKER_TIME設定（Oracle 10gで導入）は、他のセッションをブロッキングしているアイドル状態のデータベース・セッションを自動的に終了できるものであり、この設定によって、他のセッションもブロッキングしているアイドル・セッションを終了させることで、データベースとシステムの可用性が向上します。

CPU_COUNTは、インスタンスが使用可能な（ハイパースレッディング・システムの）スレッド数を指定するためのインスタンス・レベルの設定です。Oracle Real Application Clustersの各データベースには、クラスタのすべてのノードにわたって、1つ以上のインスタンスが含まれています。このセクションで説明した他のパラメータも（DB_PERFORMANCE_PROFILEを除き）インスタンス・レベルの設定であることに注意してください。

上記のメモリおよびO/Sプロセスの制御機能は、システム可用性の問題につながりうるシステム・リソースの割当て超過を防ぐために、非統合環境でも使用すべき機能です。PGAメモリの（PL/SQLアプリケーションなどでの）割当て超過はメモリ不足につながります。接続ストームの原因となるアプリケーションはO/Sプロセスを過度に割り当てることによって、O/Sレベルでのリソース不足を引き起こします。

プラガブル・データベースの場合、これらの設定はCDBレベルの“ロックダウン・プロファイル”で構成できます。この設定によって、個別のPDBデータベースの管理者による設定変更を予防できます。この機能は、複数のデータベース管理者がいる大規模な統合環境で重要になります。

データベース内のリソース管理 - コンシューマ・グループ

本書ではおもに複数のデータベース全体にわたるリソースのプロビジョニングと管理を取り扱っていますが、Oracle Database Resource Managerには1つのデータベース内部でリソースを管理する機能もあります。リソース・マネージャのコンシューマ・グループを使用すれば、各種ルールおよびポリシーを使用して、ユーザー、セッションまたはそれらのコンシューマ・グループへの接続に対してシステム・リソースを割り当てて、それらのリソースの使用状況を管理することができます。データベース内のリソース管理について詳しくは、MOS Note 1484302.1（Master Note:Overview of Oracle Resource Manager and DBMS_RESOURCE_MANAGER、Doc ID 1484302.1）を参照してください。

復帰しない問合せ

リソース・マネージャのコンシューマ・グループには、“復帰しない”問合せを実行するセッションなど、許可された量を超えるリソースを消費するセッションを管理する機能があります。この機能は、データウェアハウスや分析システムなどの非定型問合せを実行できるあらゆる環境で重要です。リソース・マネージャでは、過剰なリソースを消費するセッションまたは問合せに対して以下の操作を実行できます。

- グループの切替え
- SQLの取消し
- セッション終了
- ログ記録のみ

リソース・マネージャ・プラン・ディレクティブを使用して、セッションを別の（通常は優先順位の低い）コンシューマ・グループに切り替える、特定のSQL文を取り消す、ユーザー・セッションを終了（キャンセル）する、または単に管理者が後で分析するためにそのイベントのログを記録するという操作が可能です。復帰しない問合せへの対処は重要なトピックですが、本書では説明を割愛します。詳しくは、MOS Note 1484302.1（Master Note:Overview of Oracle Resource Manager and DBMS_RESOURCE_MANAGER、Doc ID 1484302.1）を参照してください。

リソース制御機能の使用

以下の表に、システム・リソース、Oracle Database Resource Manager（DBRM）とExadataのI/Oリソース管理機能におけるそれらのリソースの制御機能、および非Exadataシステムでの設定についてまとめます。

表6：システム・リソースの制御（12c Release 2以降）

リソース	DB（スタンドアロン）	CDB	PDB
CPU	**インスタンス・ケージング手法** CPU_COUNT（最小2を指定） RESOURCE_MANAGER_PLAN = ’DEFAULT_PLAN’	RESOURCE_MANAGER_PLAN = ‘DEFAULT_CDB_PLAN’	**パラメータによる制御：** CPU_COUNT **CDBリソース・プランによる制御：** SHARE = PDBあたりのCPUリソースの割当て率 UTILIZATION_LIMIT = PDBのリソース制限値
16コア有効	SGA_TARGET PGA_AGGREGATE_TARGET PGA_AGGREGATE_LIMIT（デフォルトはターゲットの2倍）	**CDBレベル：** コンテナ全体に適用 SGA_TARGET PGA_AGGREGATE_TARGET	**パラメータによる制御：** SGA_TARGET PGA_AGGREGATE_TARGET PGA_AGGREGATE_LIMIT（デフォルトはターゲットの2倍）

		PGA_AGGREGATE_LIMIT（デフォルトはターゲットの2倍）	**CDBリソース・プランによる制御：** SHARE = PDBあたりのCPUリソースの割当て率 UTILIZATION_LIMIT = PDBのリソース制限値
プロセスとセッション	**データベース設定：** SESSIONS MAX_IDLE_TIME MAX_IDLE_BLOCKER_TIME PARALLEL_MAX_SERVERS MAX_IDLE_BLOCKER_TIME PARALLEL_TARGET_SERVERS **SQL Netによる制御：** CONNECTION_RATE_listener_name RATE_LIMIT	**CDB設定** SESSIONS MAX_IDLE_TIME MAX_IDLE_BLOCKER_TIME PARALLEL_MAX_SERVERS MAX_IDLE_BLOCKER_TIME PARALLEL_TARGET_SERVERS **SQL Netによる制御：** CONNECTION_RATE_listener_name RATE_LIMIT	**パラメータによる制御：** SESSIONS MAX_IDLE_TIME MAX_IDLE_BLOCKER_TIME PARALLEL_MAX_SERVERS MAX_IDLE_BLOCKER_TIME PARALLEL_TARGET_SERVERS **CDBリソース・プランによる制御：** SHARE = PDBあたりのCPUリソースの割当て率 UTILIZATION_LIMIT = PDBのリソース制限値
I/Oリソース管理	**Exadata** IORM Objective = 'auto' IORM Share – システム上の他のデータベースとの相対的なI/Oリソース割当て率を指定します。IORM Limit – データベースが使用するI/Oに上限（%）を課します。 FlashCache Min – 指定したデータベースのFlash Cache最小量です。 FlashCache Limit – システム内の他のアクティビティに応じて、このサイズまで使用可能です。FlashCache Size – データベース用に、この領域を予約します。 非**Exadataシステム** 12cR2以降 – PDBのみ（Exadata上でない） » MAX_IOPS » MAX_MBPS		
RPMディスク	**Exadata** ネットワーク・リソース管理は自動 – 設定不要**すべてのプラットフォーム（Exadataを含む）** ワークロードに必要となる最小数のインスタンス上でRACデータベースを有効化		

PDBコンピューティング・リソースを制御するパラメータ（CPU_COUNT）を使用することで、CDBリソース・プラン内で割当て率や制限を使用する場合と比較して、それらのリソースをより高い精度で制御できます。

標準化されたDBリソース・シェイプ

標準化されたリソース・シェイプを使用することで、リソース管理が大幅に簡潔になります。標準化されたシェイプでは、リソース間の関係に対して共通の定義を使用します。システム内のすべてのリソースを、そのシステムで実行中の各データベースに使用されるCPU数の単純な関数として割り当てることができます。たとえば、あるデータベースにCPUの10 %が割り当てられる場合、メモリの10 %、I/Oの10 %、およびExadataシステムの使用可能なFlash Cacheの10 %も割り当てられることになります。

シェイプ

オラクルは、ワークロード・タイプ別の全般的なリソース割当て方法で構成される、2種類のリソース・シェイプを作成することを推奨します。それぞれのシェイプに対して全体的に同じ比率が適用されますが、異なるリソース要件に対応するために多数のバリエーションが生じることになります。推奨されるリソース・シェイプの種類は以下のとおりです。

- データウェアハウス（DW）

- オンライン・トランザクション処理（OLTP）

通常、データウェアハウス・ワークロードでは大きめのPGA（プロセス・グローバル領域）と小さめのSGA（システム・グローバル領域）が必要です。一方、OLTPワークロードでは大きめのSGAと小さめのPGAが必要です。また、データウェアハウス・ワークロードでは、同時ユーザー・セッション数は少なく、パラレル問合せ処理の使用頻度が高くなる傾向にあります。OLTPワークロードは通常、主にパラレルでない操作による“混合型”ワークロードとなり、パラレル問合せ処理の割合は低くなります。

DWシェイプとOLTPシェイプの全体的なサイズは、CPU、メモリ、プロセス、I/Oの割当てという意味においてはほぼ同一になりますが、メモリとプロセスの配分は、以下の表に示すとおり異なります。

倍数

上述の標準化されたシェイプでは、すべてのシステム・リソース間の共通比率が確立されます。データベースにリソースを割り当てるために、倍数を使用してこれらのシェイプの規模を調整できます。このように標準化されたシェイプと倍数を使用すれば、システム・リソースの割当てが簡潔になり、かつ管理者は各アプリケーションに適正量のリソースをプロビジョニングできるようになります。

データベース・リソース・シェイプの例

このセクションで説明した標準化されたデータベース・リソース・シェイプは、後のビンパッキング・プロセスで、“ビン”（サーバー・リソース シェイプとコンテナ・データベース）のサイズや、それらのビンへのデータベースの配置方法を決定するために使用できます。各データベースに必要となるCPU_COUNTを使用して、各データベースに割り当てられる他のシステム・リソースの比率を決定してください。注意点として、これらはデータベースのリソース・シェイプであり、仮想マシンまたはベア・メタルのシェイプとは異なります。データベース・リソース・シェイプは、下層の仮想マシンまたはベア・メタルのシェイプの上に階層化されます。1つの仮想マシンまたはベア・メタルのExadataサーバーに複数のデータベースが搭載される場合もあります。

表7と表8に、以下のExadata構成を使用するOLTPおよびDWのワークロードに対する、標準化されたデータベース・リソース・シェイプを示します。

- Exadata X8-2フル・ラック
- 8つのデータベース・ノード
- ノードあたり48 CPUコア（ノードあたり96ハイパースレッド）
- サーバーあたり2 VM、VMあたり24 CPUコア（VMあたり48ハイパースレッド）
- 14のExadataストレージ・セル（セルあたり25.6 TBのFlash Cache）

これらのデータベース・リソース・シェイプはスタンドアロン・データベースのほか、コンテナ・データベース・レベルにも適用されます。これらのデータベース・リソース・シェイプと同じものをプラガブル・データベースにも使用できますが、PDBの柔軟性を高めより高い精度のリソース共有を行うために、SHARE設定とLIMIT設定を使用することもできます。詳しくは、PDBリソース・シェイプに関するセクションを参照してください。高可用性を確保するためには、各データベースまたはコンテナ・データベースを、Oracle Real Application Clustersを使用する少なくとも2つのデータベース・ノード上で実行するように構成する必要があります。注意点として、メモリ設定とプロセス設定はノードごとの設定である一方、IORMはノード数にかかわらずデータベースに適用されます。

表7：OLTP DBリソース・シェイプ – Exadata X8-2

				メモリ				プロセス		IORM		
シェイプ	CPU_COUNT	ノード	合計	DBメモリ	SGA	PGA ターゲット	PGA制限	セッション	PQプロセス	共有値	制限（%）	Flash Cache制限
OLTP Small 2V	4	VM 2	8	30 GB	15 GB	7.5 GB	15 GB	128	4	8	5 %	3.7 TB
OLTP Medium 2V	8	VM 2	16	60 GB	30 GB	22 GB	30 GB	256	8	16	5 %	7.4 TB
OLTP Large 2V	16	VM 2	32	120 GB	60 GB	30 GB	60 GB	512	16	32	5 %	14.9 TB
OLTP 2X Large 2V	32	VM 2	64	240 GB	120 GB	60 GB	120 GB	1024	32	64	8 %	29.8 TB
OLTP 3X Large 2V	48	VM 2	96	360 GB	180 GB	90 GB	180 GB	1536	48	96	12 %	44.8 TB
OLTP 3X Large 4V	48	VM 4	192	360 GB	180 GB	90 GB	180 GB	1536	48	192	25 %	89.6 TB
OLTP 6X Large 2V	96	VM 2	192	720 GB	360 GB	180 GB	360 GB	3072	96	192	25 %	89.6 TB
OLTP 6X Large 2B	96	BM 2	192	1.5 TB	768 GB	384 GB	768 GB	3072	96	192	25 %	89.6 TB
OLTP 12X Large 4B	96	BM 4	384	1.5 TB	768 GB	384 GB	768 GB	3072	96	384	50 %	179.2 TB
OLTP 24X Large 8B	96	BM 8	768	1.5 TB	768 GB	384 GB	768 GB	3072	96	768	100 %	358.4 TB

前のセクションで説明したとおり、データウェアハウス（DW）ワークロードでは集計処理に合わせて最適化するために、通常OLTPシステムよりもSGAが小さく、PGAにより多くのメモリが割り当てられます。また、DWワークロードでは通常ユーザー数は少なく、並列処理の利用頻度が高くなります。そのため、DWシステムではOLTPシステムよりもセッション制限を低くして、PQプロセス数を増やすことになります。これらのシェイプの例すべてで、高可用性の確保とスケーリングのために、Oracle RACを使用してRACクラスタあたり2つ以上のインスタンス（またはノード）を構成しています。

表8：DW DBリソース・シェイプ – Exadata X8-2

				メモリ				プロセス		PDB、IORM		
シェイプ	CPU_COUNT	ノード	合計	DBメモリ	SGA	PGA ターゲット	PGA制限	セッション	PQプロセス	共有値	制限（%）	Flash Cache制限
DW Small 2V	4	VM 2	8	30 GB	10 GB	10 GB	20 GB	32	16	8	5 %	3.7 TB
DW Medium 2V	8	VM 2	16	60 GB	20 GB	20 GB	40 GB	64	32	16	5 %	7.4 TB
DW Large 2V	16	VM 2	32	120 GB	40 GB	40 GB	80 GB	128	64	32	5 %	14.9 TB
DW 2X Large 2V	32	VM 2	64	240 GB	80 GB	80 GB	160 GB	256	128	64	8 %	29.8 TB
DW 3X Large 2V	48	VM 2	96	360 GB	120 GB	120 GB	240 GB	384	192	96	12 %	44.8 TB
DW 3X Large 4V	48	VM 4	192	360 GB	120 GB	120 GB	240 GB	384	192	96	25 %	89.6 TB
DW 6X Large 2V	96	VM 2	192	720 GB	240 GB	240 GB	480 GB	768	384	192	25 %	89.6 TB
DW 6X Large 2B	96	BM 2	192	1.5 TB	360 GB	588 GB	1176 GB	768	384	192	25 %	89.6 TB
DW 12X Large 4B	96	BM 4	384	1.5 TB	360 GB	588 GB	1176 GB	768	384	384	50 %	179.2 TB
DW 24X Large 8B	96	BM 8	768	1.5 TB	360 GB	588 GB	1176 GB	768	384	768	100 %	358.4 TB

注意点として、このスキームで“3X Large 2”または“3X Large 4”よりも大きいデータベース・リソース・シェイプでは、ベア・メタルを使用する必要があります（ノード構成をBM 2、BM 4、BM 8として示しています）。これは、Exadataに仮想マシンあたりメモリ容量として720 GBの制限があるからです。システムのスラッシングの発生を回避するために、UTILIZATION_LIMITで極端に低い値（5 %未満）を使用しないでください。上記の例では、使用率の下限を5 %とし、それよりも高い値については共有値の2倍に相当するようにしています。

プラガブル・データベース間のリソース共有

Oracle Multitenantを使用すれば、コンテナ・データベース内のリソースを、そのコンテナ内にあるプラガブル・データベース間で共有できます。CDB内のコンピューティング・リソースの共有は、前のセクションで説明したCPU_COUNTによる厳密な制御（インスタンス・ケージングとも呼ばれる）ではなく、各PDBのリソース・プランの以下の設定によって制御します。

- 割当て率
- 使用率制限
- パラレル実行サーバー最大数
- パラレル実行サーバー・ターゲット

あるデータベースのコンピューティング・リソースの割当て率（SHARE）は、そのコンテナ内の他のデータベースとの相対値です。たとえば、コンテナ内に10個のプラガブル・データベースがあり、各データベースの割当て率の値が10の場合、各PDBにはそのコンテナのCPUリソースの10 %以上が割り当てられます。割当て率は使用可能なリソースの最小の割当てを示しますが、コンテナ内のプラガブル・データベースでは、他のデータベースがアクティブではなくそれらのリソースを消費していない場合に、その最小の割当てを超えるCPUリソースを使用できます。

データベースの使用率制限（UTILIZATION_LIMIT）は、PDBで使用可能なコンピューティング・リソース量に上限を設定するものです。使用率制限のおもな目的は、データベースで使用可能なリソース量のばらつきを抑えることです。ばらつきがあると、エンドユーザーのパフォーマンスにもばらつきが生じる可能性があります。使用率制限によってアイドル状態のシステム・リソースが使用されなくなります。その場合、通常は全体的なシステム使用率が減少し、ユーザーがアイドル状態のシステム・リソースを活用できなくなりますが、エンドユーザー・エクスペリエンスの一貫性は高まります。

パラレル実行サーバー最大数（PARALLEL_MAX_SERVERS）設定は、プラガブル・データベースが利用可能なパラレル実行サーバー・プロセス数に制限を課すものです。パラレル実行サーバー・ターゲット（PARALLEL_SERVERS_TARGET）には、需要が少ないときにパラレル実行サーバー数を縮小する際の目標数を指定します。

プラガブル・データベースのリソース・シェイプ

前述のとおり、プラガブル・データベースではスタンドアロン・データベースとコンテナ・データベースについて上記の割当て方法を使用できるほか、以下の表に詳しく示すとおり、PDBパフォーマンス・プロファイルを使用できます。いずれかの方法を使用して、プラガブル・データベースはコンテナ・データベースに割り当てられたリソースの一部分を割り当てられることになります。CPU、メモリ、プロセスのシステム・リソースを固定値として割り当てる場合、ピーク時のリソースを各データベースに割り当てる必要があるため、全体的なシステム・リソースの使用率は低下します。共有値と制限による手法を使用する場合は、リソースがPDB間で共有され、さらに各プラガブル・データベースのリソース消費量に制限が置かれることになります。

表9：PDBパフォーマンス・プロファイル – Exadata X8-2

		コンピューティング、メモリ、プロセスの共有値			IORM共有値		
プロファイル名	ノード	共有値	使用率制限	パラレル実行サーバー制限	共有値	制限（%）	Flash Cache制限
PDB XX Small 2	VM 2	1	5 %	4 %	1	5 %	9 TB
PDB Extra Small 2	VM 2	2	8 %	8 %	2	5 %	1.8 TB
PDB Small 2	VM 2	4	16 %	16 %	4	5 %	3.7 TB
PDB Medium 2	VM 2	8	32 %	33 %	8	5 %	7.4 TB
PDB Large 2	VM 2	16	64 %	66 %	16	5 %	14.9 TB
PDB 2X Large 2	VM 2	32	100 %	100 %	32	8 %	29.8 TB
PDB 3X Large 2	VM 2	48	100 %	100 %	48	16 %	44.8 TB
PDB 3X Large 4	VM 4	48	100 %	100 %	96	32 %	89.6 TB
PDB 6X Large 2	BM 2	96	100 %	100 %	96	32 %	89.6 TB
PDB 12X Large 4	BM 4	96	100 %	100 %	192	64 %	179.2 TB
PDB 24X Large 8	BM 8	96	100 %	100 %	384	100.00 %	358.4 TB

表9の例では、各PDBのリソース・シェイプには、PDBパフォーマンス・プロファイルと呼ばれる方法で、リソースの“共有値”に加え、その共有値の2倍に相当する使用率制限も指定されています。たとえば、“PDB XX Small”パフォーマンス・プロファイルでは、共有値“1”は、構成された仮想マシン上の48コアの約2 %に相当します。システム・リソースのスラッシングを予防するため、制限の値は5 %以上に設定する必要があります。このPDBに対する“共有値と制限”による手法はOLTPワークロードとDWワークロードで同一になります。適切なリソース割当てがコンテナ・データベース・レベルで指定されているためです。PDBには単に、コンテナに割り当てられたリソースについての共有値（および制限）が適用されます。

ここでは、PDBに対するIORM共有値と各種制限が、スタンドアロン・データベースの設定と同じものになっています。Exadata I/O Resource Managerはコンピューティング層から独立しているためです。

データベース統合の実装

この章では、本書で説明した手法によりデータベース統合を実装するためのプロセスについて説明します。データベース統合は、Exadata、Exadata Cloud@Customer、またはExadata Cloud Serviceに統合する場合と同じプロセスを踏みます。データベース統合の実装は、各データベースのリソース容量ニーズを決定し、複数のデータベースをグループにまとめて管理を簡潔にするための標準化されたプロセスを踏む必要があります。このプロセスでは次の手順を実行します。

1. 統合対象のデータベースのインベントリ
2. リソース使用率および増加率に関するメトリックの収集
3. リソース要件の新プラットフォームへのマッピング
4. データベース独立性の要件と手法の決定
5. データベース統合手段の選択
6. データベースのHA層へのグループ化
7. データベースのリソース・シェイプへのビンパッキング
8. データベースのリソース・プランの作成

統合対象のデータベースのインベントリ

データベース統合ではまず、統合対象のすべてのデータベースのインベントリを実行します。統合の候補となるデータベースについて、以下の情報を収集する必要があります。

- データベース名
- データベースのバージョン
- 物理的な位置、地域、またはデータセンター
- サーバー名
- サーバー・プラットフォームのO/S
- サーバーO/Sのバージョン
- 関連アプリケーション
- HA層（ブロンズ、シルバー、ゴールド、プラチナ）
- 環境（開発、テスト、QA、本番、DRなど）

この候補データベースのリストは今後、同じ物理サーバーまたは仮想サーバー、もしくはその両方に統合されるデータベースを決定するために、この統合計画プロセス全体を通して使用されます。

リソース使用率および増加率に関するメトリックの収集

リソース使用率メトリックは、統合候補リスト内の各データベースについて必要になります。各データベースのオラクルの自動ワークロード・リポジトリ（AWR）から以下のメトリックを収集できます。

- CPU使用率
- プロセス数とセッション数
- メモリ
- ストレージ領域
- I/O

既存システム上で収集するメトリックは、容量の監視と計画の目的で収集するメトリックと同じものです。データベース統合環境への移行を計画するためのこれらのメトリックの収集方法について詳しくは、本書の容量使用率の測定に関するセクションを参照してください。現行システムのリソース使用率分析によって、現在の状態や、ある程度の過去のリソース使用率が明らかになりますが、増加率はもっと複雑です。

あらゆる組織で、データ量と処理ワークロードは増加していきます。この増加によってストレージ領域、コンピューティング能力、システム・メモリ、I/O速度への要求がより厳しくなります。将来の計画のために過去の増加率が使用されるのはよくあることですが、過去の増加率には、今後予測されるビジネス状況の変化、アプリケーション機能の変更、その他容量に影響する外部要因は当然反映されていません。そのため、容量計画プロセスでは多くの場合、過去のシステム統計値にとらわれずに検討する必要があります。

計画期間（Planning Horizon）とは、容量計画が実際に適用される期間のことです。ほとんどの組織では3〜5年ごとに容量計画を実施し、その計画期間に向けてハードウェアを購入したり必要なサービスを獲得したりします。たとえば、容量計画において、5年間に年平均成長率10％、つまりその5年全体で61％の成長が見込まれているとします。ビジネス環境、企業買収、アプリケーションの機能変更、規制の変更、その他の要因があるため、前年比の増加率は常に同一にはなりません。それでも、前年までの過去の増加率を使用して将来の増加率を予測するという手法はよくとられます。

リソース制限間近のシステムでは、リソース使用率を正確に把握することはできません。需要が供給よりも多いときは常に、正確な需要レベルは不明であり、リソースの供給を上回っている可能性があります。この現象について認識し、容量計画の際に考慮するようにしてください。

リソース使用率の新プラットフォーム特性へのマッピング

リソース使用率を既存システム上で収集しましたが、統合プロセスでは通常、その使用率は個別のシステムに適用されることになります。新しいコンピューティング・プラットフォームでは一般的に旧システムよりもプロセッサ速度、メモリ容量、I/O速度、I/Oスループットが向上し、容量とパフォーマンスが追加されます。システム・ベンダーはそれぞれ独自の方法論によって、特定のアプリケーションやデータベース一式に関するシステム容量要件を決定しています。オラクルはM-Valuesというシステム・パフォーマンス・メトリックを使用して、複数世代のサーバーにおけるパフォーマンスの差異を追跡しています。

新システムへの統合によって、旧システムとは異なるアーキテクチャになる場合もあります。従来のサーバーとストレージの環境からExadata Database Machineに移行するような場合です。プラットフォームのアーキテクチャ上の差異は多くの場合、データベースのシステム容量要件に影響を及ぼします。たとえば、Exadataでは、データベース処理がExadataストレージ層にオフロードされるため、必要となるコンピューティング能力は従来のシステムと比較して低くなります。これはExadataデータベース・サーバー最適化と呼ばれるもので、ワークロードの内容や、そのワークロードのストレージへのオフロードの程度によって変わります。本書で示したベンチマークでは、統合密度が2倍に上がりました（つまり、コンピューティング・リソースの消費量が50 %低下しました）。

リソース要件を新プラットフォームにマッピングする際に必ず考慮しなければならない現象として、**ペントアップ・デマンド**（繰越需要）というものがあります。この現象は特に、システムの使用がそのシステムのパフォーマンスにより制限されることの多いデータウェアハウスや分析システムで顕著に見られます。システムのパフォーマンス向上によってそのシステムの使用量が増加し、必要なリソースが予期しないほど増加するのです。特に、データウェアハウスや分析用のデータベースをExadataプラットフォームに移行する場合にそうなります。

独立性の要件と手法の決定

すべてのIT組織は、データベース同士の独立性をある程度確保することが求められます。一般的な独立性の要件は以下のとおりです。

- 環境（開発、テスト、QA、本番、DRなど）
- データ・ガバナンス（PII、PCI、HIPAAなど）
- アップグレード、パッチ適用

通常、非本番環境（開発、テスト、QAなど）は本番環境と分離する必要があり、Payment Card Industry（PCI）要件などの特殊なガバナンス規制の影響を受けるデータは、それらの規制の影響を受けないデータから分離する必要があります。Exadataシステム上のOracleデータベースでは以下の2つの分離手法を利用できます。

- 物理的な分離
- 仮想マシンによる分離
- コンテナ・データベースによる分離

物理的に分離されたマシンでは、環境間で完全な独立性を確保できますが、仮想マシンを使用すればその同じ利点の多くを同じ物理マシン内で実現できます。非本番データベース（開発、テスト、QAなど）は、物理的に分離されたマシンを使用して、本番データベース（本番、DR）から分離する必要があります。データ・ガバナンスを目的とした分離も、物理的な分離と仮想マシンによる分離のいずれかによって達成できます。

1つの物理サーバー、仮想マシン、または物理/仮想クラスタ内で複数のコンテナ・データベースを使用することで、データベースの独立性のレベルがさらに向上します。複数のコンテナ・データベースを同じO/S上に配置し、異なるOracle DBMSパッチやバージョンのレベルを使用することもできます。

データベース統合手段の選択

各HA層のリファレンス・アーキテクチャでは、すべての統合手段とデプロイメント・モデルに対応しています。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場合の詳細なベスト・プラクティスは、利用する統合手段によって異なります。

サーバーあたり複数のデータベースの使用（Oracle RACとOracle Resource Managerを併用）は、Oracle Database 12c以降のリリースにおけるMAAのベスト・プラクティスです。Oracle RACによって可用性とスケーラビリティを確保しながら、Resource Managerによってリソースの独立性を確保します。

Oracle MultitenantはOracle Database 12cR1以降のリリースに適用できます。このマルチテナント・アーキテクチャによって、最大限の統合密度が実現され、多数のデータベースの管理性も向上します。

物理環境、仮想環境でのMAAベスト・プラクティスの差異はありません。スキーマ統合は大部分がOracle Multitenantに置き換わりましたが、今でも複数のアプリケーションやマイクロサービスを1つのデータベースに統合する場合には選択肢の1つになります。この統合によるデータベースが、上記のいずれかの手法によって他のデータベースと統合されることになります。

スタンドアロン・データベースをOracle Multitenantに移行する場合のプロセスは簡単で、各データベースをPDBに変換して、そのデータベースを定められたHA層の適切なCDBに移行するだけで完了します。

データベースのHA層へのグループ化

すべてのデータベースを、本書で説明したいずれかのHA層に分類する必要があります。分類の基準には、統合の候補となっている各データベースに関する、データ消失に対するビジネス側の許容度（リカバリ・ポイント目標、RPO）、停止時間（リカバリ時間目標、RTO）が含まれます。データベース間に存在する可能性のあるすべての依存関係について十分に把握し、ビジネス目標を達成できるように、それらのデータベースを適切にグループ化してください。

MAAベスト・プラクティスとしては、前述のHA層の標準セットに従って、同じようなRTO（リカバリ時間目標）とRPO（リカバリ・ポイント目標）を持つデータベースを統合するようにします。注意点として、データベース間に依存関係がある場合は、HA要件がもっとも厳しいデータベースに該当するHA層にそれぞれのデータベースを割り当てるようにします。

RTOとRPOの要件に従って標準HA層にデータベースをグループ化するプロセスによって、以下の3つの目標が達成されます。限られたHA層上で標準化することによって、複雑さを緩和し、規模の経済を達成します。

HAインフラストラクチャやプロセスへの過度な投資や不要な複雑さを避けて、効率的に統合します。たとえば、ブロンズ層のデータベース（RTOとRPOをバックアップのリストアによって達成できる）をシルバー層やゴールド層のデータベースと統合するのは非効率です。シルバー層やゴールド層では、RTOとRPOの要件から追加のHAインフラストラクチャが必要になるからです。

DBaaS用サービス・カタログのHAおよびデータ保護の部分を決定します。このサービス・カタログでは、IT組織が開発者、アーキテクト、エンドユーザーなどのユーザー・コミュニティ対して提供するサービスと、データベース・サービスの提供と管理の方法に関する詳細を説明します。

データベースのリソース・シェイプへのビンパッキング

独立性の要件、統合手段、HA層へのグループ化（前述のとおり）を完了したら、次にビンパッキングと呼ばれるプロセスで、データベースをリソース・シェイプ内にまとめます。このビンパッキング・プロセスでは、どのデータベースを一緒にリソース・シェイプ（このコンテキストにおける“ビン”）と呼ばれる形に統合するかを決定します。このシェイプは以下のいずれかの構成になります。

- 物理サーバー
- 仮想マシン
- 物理サーバー・クラスタ
- 仮想マシン・クラスタ
- コンテナ・データベース

ビンパッキング・プロセスでは、各リソース・シェイプ（“ビン”）のサイズ要件と構成について決定します。これには、仮想マシンのサイズ、クラスタに統合する物理マシンまたは仮想マシンの個数などがあります。ビンパッキング・プロセスは、必ず最大から最小の順序で実行してください。このサイズは、ストレージ容量ではなく、各データベースに割り当てられたコンピューティング・リソースによって測定します。リソース・シェイプに必要となるストレージ領域の容量は、ビンパッキング・プロセスの終了後、関連するデータベースのサイズに基づいて算出します。その後、リソース・シェイプに割り当てられたデータベースの合計サイズに従って、必要なストレージ容量を各リソース・シェイプ（物理サーバー、仮想マシン、または物理/仮想クラスタ）に割り当てます。

コンテナ・データベース（CDB）内のプラガブル・データベース（PDB）として統合するデータベースはすべて、同じHA層（ブロンズ、シルバー、ゴールド、プラチナ）に配属してください。

基盤のハードウェアに容量制限があり、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・クラスタに統合できない場合があります。パフォーマンス要件を評価することによって、データベース統合の候補としてのデータベースの適合性を判断し、統合対象の現行マネージド・サーバーを統合先のサーバーにマッチングする必要があります。

同時に複数のOracle Databaseリリース/パッチセットをサポートしなければならない場合もよくあります。たとえば、1つ目のCDBがOracle Database 12c Release 1（12.1.0.2）で、2つ目のCDBがOracle Database 18.1.0.0となるような構成です。このような構成により、他の12.1 PDBをアップグレードできない場合に、ある個別のPDBをOracle 12.1.0.2 CDBから'切断'しOracle 18.1.0.0 CDBに'接続'して、アップグレードを実施することができます。たとえば、ベンダー供給のアプリケーションでデータベースの新リリースのサポートが遅れている場合に、このような手段をとることができます。

通常、データベースのブロンズ層は、統合に適したもっとも多くのデータベースが格納されるため、Oracle Multitenantを使用する候補になる可能性がもっとも高い層です。オラクルは、どのような統合作業でも、アプリケーションの重大性が低く投資回収率が高いブロンズ層のデータベースから始めることを推奨します。

コンテナ・データベースも"ビン"の1つであり、複数のデータベースがコンテナにまとめられます。コンテナ・データベースのリソースは、そのCDBに含めるすべてのプラガブル・データベースを格納できるサイズにします。CDBはバージョン/パッチ適用の観点で独立性を確保しますが、同時に、CDB管理下のPDBによって使用されるリソースを制約する役割も果たします。

データベース用のリソース・マネージャ計画

Oracle Database Resource Manager（DBRM）とExadataのI/O Resource Manager（IORM）は、対象のデータベースとアプリケーションに対して、ビジネス・ニーズにあった適正量のシステム・リソースを割り当てる役割を果たします。

- 個別のデータベースのリソース・プラン
- データベース・グループ用のリソース・プロファイル

個別のリソース・マネージャのプランは、より注意して積極的に管理する必要のある大規模でクリティカルなデータベースに対して作成します。リソース・マネージャのプロファイルは、リソース要件が比較的均一となる小規模で重要度の低いデータベースに対して使用します。Oracleリソース・マネージャ（DBRMとIORM）の使用について詳しくは、本書のリソース・マネージャに関するセクションを参照してください。

統合計画プロセスの概要を表10に示します。

表10：統合計画

統合のためのおもなステップ	推奨事項	利点
1.統合対象のデータベースのインベントリ	統合対象のデータベースのリストには、位置、プラットフォーム、アプリケーション、HA層、統合の決定をサポートするその他の特性を記録する。	統合作業の全体的なスコープを決定する。
2.リソース使用率および増加率の収集	現行システムのリソース使用率（CPU、メモリ、I/Oなど）が、統合後のデータベース・プラットフォームのリソース計画の基礎になる。システムがビジネス側のリソース・ニーズを満たすように、増加率も考慮する必要がある。リソース使用率によってリソース・マネージャの構成も定まる。	ステップ7のビンパッキング実行のために必要。各データベースまたはデータベース・グループに対してリソース管理プランを定義する。
3.リソース要件の新プラットフォームへのマッピング	システム・ベンダーに相談して、旧世代と新世代のハードウェア間のパフォーマンスについて相対的な差異を明らかにする。	新コンピューティング・プラットフォームでは通常、旧システムよりもパフォーマンスと容量が向上する。Exadataへの移行などのシステム・アーキテクチャの変更も、システムの容量要件に影響する可能性がある。
4.データベース独立性の要件と手法の決定	異なる物理マシン上または異なる仮想マシン内に配置することで、独立性を確保できる。開発およびテスト用のデータベースは通常、本番データベースから分離する。特殊なガバナンス規則（PII、PCI、HIPAAなど）の影響を受けるデータベースは通常、他のデータベースから分離する。	独立性のビジネス要件を満たしながら、多数の物理サーバーや仮想マシンを継続的に管理することによる複雑さを最小限に抑えることができる。
5.データベース統合手段の選択	Oracle Multitenantのコンテナ・データベース（CDB）によって最大限の統合密度を実現できる。また、Oracle Multitenantを使用せずに、複数のデータベースをリソース・シェイプ（単一のサーバー、単一のVM、物理サーバーまたは仮想サーバーから成るクラスタ）に統合することも可能。 Exadata上の統合では、いずれかのアプローチと併用することでより高い統合密度を達成できる。Multitenantのデータベースとスタンドアロン・データベースを組み合わせて使用することも可能。	統合密度が高いほど、システム・インフラストラクチャのコストのほか、システムやデータベースの保守にかかる運用コストを削減できる。
6.データベースのHA層への グループ化	MAA HA層（プラチナ、ゴールド、シルバー、ブロンズ）に従う。CDB内の すべてのデータベースを同じ層に配属する必要がある。	標準化により運用コストとリスクを削減。
7.データベースのリソース・シェイプへのビンパッキング	サイズを概算したリソース・シェイプ（"ビン"）を作成する。典型的なリソース・シェイプでは、ハーフ・ラックから最大1つのフル・ラック（CloudおよびCloud@Customer）または2つのフル・マルチラック・システム（オンプレミスのみ）を使用する。 それぞれのリソース・シェイプを1つのHA層に割り当てる。各層のHAアーキテクチャは異なるため、リソース・シェイプあたりのHA層は1つのみとする。 最大から最小の順序で、データベースを"ビン"にパッキングする。	ビンパッキングによってリソース・シェイプ構成を決定。

8.データベース用のリソース・マネージャ計画	最大またはもっともクリティカルなデータベースのそれぞれに対して、個別のリソース・マネージャのプランを作成する。小規模で重要度の低いデータベースに対しては、リソース・マネージャのプロファイルを使用する。	ビジネス・ニーズに従って、適正量のシステム・リソースを各データベースに割り当てるようにする。リソース管理は、障害の発生したアプリケーションやデータベースによってシステムの安定性が損なわれないようにする役割も果たす。

結論

データベース統合は、現代の企業の目的を達成するための有用なアプローチであり、以下の5つのおもなビジネス目標のバランスをとるように設計されています。

- **コスト**の削減
- 管理の**簡素化**
- **セキュリティ**の強化
- データベースとアプリケーションの**可用性**確保
- ビジネス・アプリケーションに必要なレベルの**パフォーマンス**実現

本書では、Oracle Database、仮想マシン、Oracle Multitenant、Exadata（オンプレミスにデプロイされたか、Cloud ServiceやCloud@Customerモデルにデプロイされたかは問わない）を使用してこれらの目的を達成するための戦略について説明しました。これらすべてのデプロイメント・モデルで、Exadataは他のプラットフォームよりも高い統合密度を達成できるため、プラットフォーム全体のコストが削減されます。管理の簡潔性は、データベースの統合に加えて、Exadataプラットフォームの使用により達成できます。Exadataプラットフォームは、サーバー、ストレージ、ネットワーク、オペレーティング・システム、仮想マシン、クラスタウェア、論理ボリューム・マネージャ、Oracleデータベース・ソフトウェアの統合ソリューションとして提供されるためです。

Oracle Multitenantは、（Exadataによる高密度に加えて）さらに高いレベルの統合密度を実現することによって、コストをさらに削減し、かつ大規模統合環境に対して簡潔な“一元管理”機能をもたらします。

本書では、オラクルのMaximum Availability Architecture（MAA）についても説明しました。この説明には、データベース統合のコンテキストにおけるMAAの適用方法も含まれていました。データベースとアプリケーションの可用性を必要とされるレベルで提供することが統合環境のおもな目標であり、本書ではそれらのビジネス・ニーズへの対応方法について説明しました。

複数のシステムや管理プラクティスが異なる多彩なプラットフォームがもたらす複雑さが軽減されるため、データベース統合によってセキュリティが強化されます。また、Exadataプラットフォーム上のデプロイメントではセキュリティがさらに強化されます。“デフォルトで高セキュリティ”の構成で提供され、事前にスキャン済みのシステム・スタックが搭載され、Advanced Intrusion Detection Environment（AIDE）の実装によって攻撃から保護されるためです。

最後に、どのような統合作業でも、ビジネス・アプリケーションが必要とする適正レベルのパフォーマンスを提供する必要があります。パフォーマンスを損なわずに統合できるようにシステムのアーキテクチャを設計する必要があります。統合し、さらに必要なパフォーマンスを提供するために、Exadataがこれまで実証してきたパフォーマンス能力を活用できます。

参考資料

My Oracle Supportのドキュメント

DocID	ドキュメントのタイトル
1338988.1	Scripts and Tips for Monitoring CPU Resource Manager
1337265.1	Tool for Gathering I/O Resource Manager Metrics: metric_iorm.pl
888828.1	Exadata Database Machine and Exadata Storage Server Supported Versions
361468.1	HugePages on Oracle Linux 64-bit
401749.1	Oracle Linux:Shell Script to Calculate Values Recommended Linux HugePages / HugeTLB Configuration
1484302.1	Master Note:Overview of Oracle Resource Manager and DBMS_RESOURCE_MANAGER
1585184.1	Using PROCESSOR_GROUP_NAME to bind a database instance to CPUs or NUMA nodes on Linux
1338988.1	Scripts and Tips for Monitoring CPU Resource Manager
1070954.1	Oracle Exadata Database Machine exachk or HealthCheck

ホワイト・ペーパーとプレゼンテーション**Exadata Security Guide**

https://docs.oracle.com/en/engineered-systems/exadata-database-machine/dbmsq/security-guide-exadata-database-machine.pdf

MAAベスト・プラクティス

https://www.oracle.com/technetwork/database/availability/maa-overview-onpremise-2019-5391126.pdf

仮想マシンとExadata

https://www.oracle.com/technetwork/jp/database/availability/exadata-ovm-2795225-ja.pdf

『継続的可用性』ホワイト・ペーパー

https://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/adb-continuousavailability-5169724.pdf

オラクルの情報を発信しています

+1.800.ORACLE1までご連絡いただくか、oracle.comをご覧ください。

北米以外の地域では、oracle.com/contactで最寄りの営業所をご確認いただけます。

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

データベース統合の
ためのベスト・プラ
クティス 2020年7月

著者：Christian A. Craft
共著者：Glen Hawkins、Lawrence To、Tina Rose、Dan Norris