



Oracle Database Appliance : Oracle Data Guardを使用した MAAディザスタ・リカバリ・ソリュー ションの実装



スタンバイ計算能力の利用時の本番システムの保護

Oracle Maximum Availability Architecture

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。

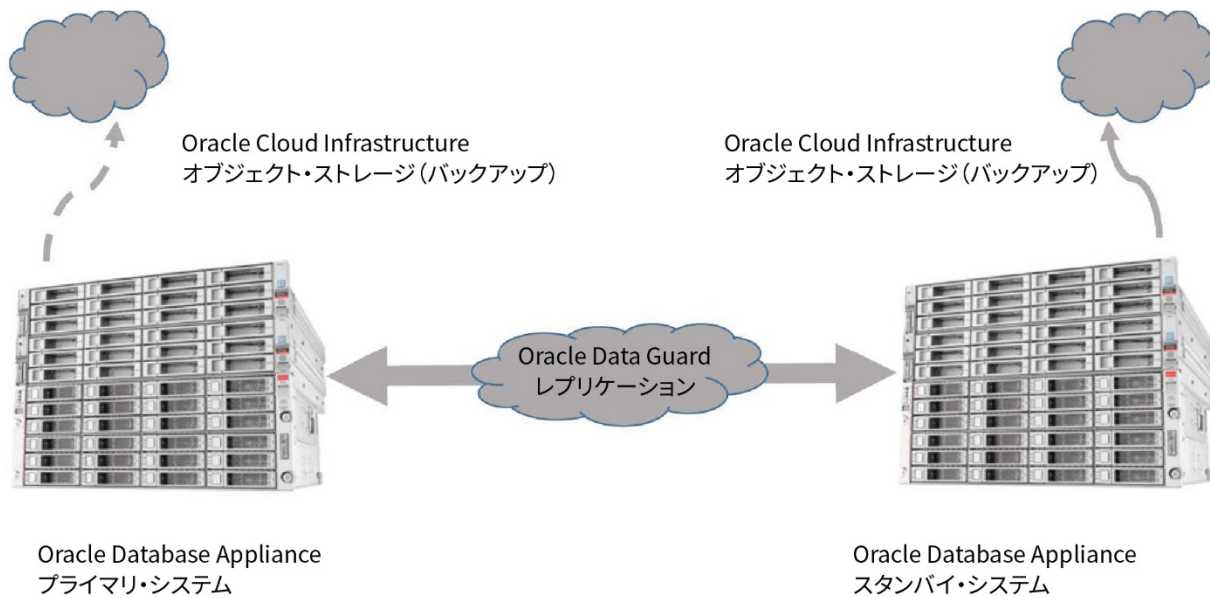
製品アーキテクチャの性質上、コードが大幅に不安定化するリスクなしに、本書に記載されているすべての機能を安全に含めることができない場合があります。

目次

免責事項.....	1
はじめに.....	3
Oracle Active Data Guardを使用したデータ保護.....	4
Oracle Data GuardとOracle Active Data Guardを使用する利点	4
構成のベスト・プラクティス	6
Oracle Database Applianceシステム間でのOracle Data Guardの設定.....	8
Oracle Data Guardの構成手順.....	8
Oracle Database Applianceベア・メタルおよび仮想化プラットフォームの構成	9
Oracle Database Applianceの小規模、中規模、大規模プラットフォームの構成	9
結論.....	9
付録A：DCSスタックを使用したOracle Database Applianceでの19cの設定例	10
付録B：OAKスタックを使用したOracle Database Applianceでの12.1.0.2の設定例	23
付録C：Oracle Database Applianceでの11gR2の設定例.....	33
付録D：Oracle RACへのシングル・インスタンス・データベースの変換.....	47
付録E：OAKを使用するOracle Database ApplianceでのOracle Data Guardを使用したデータベースのアップグレード	49
すべてのコンポーネントのアップグレード	49
データベースのみのアップグレード.....	50
付録F：DCSを使用するOracle Database ApplianceでのOracle Data Guardを使用したデータベースのアップグレード.....	51
すべてのコンポーネントのアップグレード	51
データベースのアップグレード	52
データベースへのパッチの適用.....	56
付録G：同じバージョンのRDBMSホームに一部の古いバージョンのPSUとバンドル・パッチが含まれる理由.....	58
参考資料.....	59
ドキュメント	59
技術概要.....	59
My Oracle Support (MOS) Knowledge Content Notes	59

はじめに

Oracle Database Appliancesは構築と調整が完了して、すぐに使えるようになっているクラスタ型および非クラスタ型のデータベース・システムで、サーバー、ストレージ、ネットワーク、ソフトウェアが最適な構成に組み込まれ、展開、操作、管理が容易に行えるようになっています。Oracle Database Applianceは小規模、中規模、大規模なデータベース実装に最適で完全なデータベース・プラットフォームであり、堅牢で実績のあるOracleのテクノロジー、たとえば、世界をリードするOracle Database、高い販売実績を誇るOracle Real Application Clusters (Oracle RAC) データベース・オプション、Oracle Clusterware、Oracle Automatic Storage Management (Oracle ASM) を内蔵しています。Oracle Database Applianceではハードウェアとソフトウェアを統合することにより、手作業で組み立てた非統合型のデータベース・ソリューションに固有の複雑さが解消され、数週間から数カ月かかっていた展開時間が数時間に短縮されます。同時に、管理が難しく、最適とは言えないデータベース環境の原因となることの多い構成や設定のエラーを防止できます。



Oracle Database ApplianceとOracle Data Guardを使用したOracle Maximum Availability Architecture (MAA)

Oracle Active Data Guardを使用したデータ保護

Oracle Database Applianceはそれ自身が可用性の高いシステムですが、スタンバイ・データベース環境により、プライマリ・データベース環境が使用不可になったり障害状態になったりした場合にデータを保護し、計画停止時間と計画外停止時間を短縮することができます。そのため、スタンバイ・データベースは常に、ミッション・クリティカルな本番システムの高可用性とデータ保護を向上させるための不可欠なMAAコンポーネントとなっています。[Oracle Maximum Availability Architecture \(Oracle MAA\) ゴールド層](#)のベスト・プラクティスを使用すると、スタンバイ・データベースをプライマリ・データベースと同期させることができるため、データベースのアップグレードなどの計画メンテナンス作業や、データ破損、データベース障害、クラスタの障害、停電、自然災害などの計画外停止によるデータベースの停止時間を最小限に抑えることができます。

最適なリカバリ計画を立てて実装するために考慮する必要のある2つの最重要指標が、リカバリ・ポイント目標（RPO）とリカバリ時間目標（RTO）です。Oracle Data Guardは、ミッション・クリティカルなOracle Databaseのシングル・ポイント障害を排除できるもっとも包括的なソリューションです。このソリューションは、MAAゴールド層を併用し本番データベースの同期された物理レプリカを遠隔地に保持することによって、もっとも単純かつ経済的な方法でデータの損失を防止し（RPOゼロ）、停止時間の発生を防止します（RTOゼロ）。何らかの理由で本番データベースが使用できなくなると、クライアント接続が、迅速に（また一部の構成では透過的に）、同期されているレプリカにフェイルオーバーされ、サービスがリストアされます。

Oracle Active Data Guardを使用して、プライマリ・データベースから読取り専用のフィジカル・スタンバイ・データベースに処理をオフロードすることによりパフォーマンスを向上させながら、プライマリ・データベースから受信した更新を適用することができます。Oracle Active Data Guardのオフロード機能には、書き込みや更新がほとんどない読取り専用のレポート（Oracle Database 19cのDMLリダイレクト経由）および任意の間合せ（グローバル一時表および一意のグローバル・シーケンスまたはセッション・シーケンスへのDMLを含む）、データの抽出、高速の増分バックアップ、REDO転送の圧縮、複数の遠隔地への効率的なサービス、プライマリ・データベースのパフォーマンスに影響を及ぼすことなくリモート・スタンバイ・データベースへのデータ損失ゼロ保護を拡張する機能が含まれています。

Oracle Active Data Guardは、可用性を高めるために自動ブロック修復を行い、高可用性アップグレードを有効化します（データベースのローリング・アップグレードの自動化を利用して、高可用性環境を維持しながら停止時間を回避します）。また、コミットされていない可能性のある処理中トランザクションに対するデータ保護を拡張した、アプリケーション・コンティニュイティ（AC）も搭載しています。

オラクルでは、Oracle Database Applianceのプライマリ・システム上で動作するミッション・クリティカルな本番システムのData Guardスタンバイ・システムをホスティングするために、独立した専用のOracle Database Applianceシステムを使用することを推奨しています。このMAAベスト・プラクティスは、ある程度の独立性が確保された近隣のデータセンター内にローカルの（同期レプリケーションによる）スタンバイ・データベースを配備し、リモート・スタンバイについては非同期レプリケーションによって定期的に維持するというものです。このようにすることで、大半の計画外停止のケースではRPOゼロを維持しながら、大規模停電などの地域全体に影響を及ぼすような災害からも保護できます。

Oracle Data GuardとOracle Active Data Guardを使用する利点

Oracle Data Guardは多数の利点を備えていて、導入したアーキテクチャの効率性と有効性を高めることができます。Oracle Data Guard単独でも優れた保護状態を確保できますが、MAAゴールド層ではActive Data Guardが必須となっています。それは、自動ブロック修復、アプリケーション・コンティニュイティ、DBMS_ROLLINGを使用しなければ、『[Oracle Maximum Availability Architecture \(Oracle MAA\) オンプレミス、Exadataベース、またはクラウドベースのOracle Databaseの計画停止時間と計画外停止時間の短縮構想](#)』のRTO/RPOを達成できないためです。Oracle Active Data Guardを使用すれば、スタンバイ・データベース環境が休んだままの状態、内部に隠れたままの状態になることはありません。積極的に多数の有益な目的を果たすことができます。このような付加的な使用により、労力と投資に対する全体的な収益率が大幅に増加します。

Oracle Database Applianceへの移行 - 既存のデータベースからOracle Database Applianceへの移行を計画している場合、Oracle Data Guardにより、使用中のデータベースからOracle Database Applianceへの移行を容易に行うことができます。Oracle Database Appliance上でフィジカル・スタンバイ・データベースを設定し、従来の環境から新しいOracle Database Appliance環境に処理をスイッチオーバーするだけです。ここには、特定のプラットフォーム間での移行も含まれます。たとえば、Windowsプラットフォーム上で動作中のデータベースをLinuxプラットフォーム上のOracle Database Applianceに移行するには、2つの環境間でOracle Data Guardを設定し、スイッチオーバーを行います。このアプローチによるプラットフォームの移行では、何らかの理由によりテスト後にスイッチバックを行いたい場合にも柔軟に対応できます。Oracle Data Guardを使用したプラットフォームの移行については、My Oracle Support（MOS）のNote [413484.1](#)『Data Guard Support for Heterogeneous Primary and Physical Standbys in Same Data Guard Configuration』を参照してください。

注：Oracle Data Guardでは一時ロジカル・スタンバイ・データベースを使用して、異なるバージョン間でのデータベースの移行も行えます。

ディザスタ・リカバリ - Oracle Data Guardのフィジカル・スタンバイ・データベースは災害保護のための最適なソリューションとして機能します。よく起こる災害の例は地域の停電ですが、災害のシナリオは、水道管や蒸気パイプの破裂から、火災、ハリケーン、破壊行為、地震、洪水、テロ行為まで、多岐にわたります。Oracle Data Guardのフィジカル・スタンバイ・データベースでは、本番データベースのブロックごとのコピーが維持されます。何らかの理由でプライマリ環境が利用できなくなった場合、スタンバイ環境をすばやく有効化して、アプリケーション用に継続的にデータベースを利用できる状態を維持することができます。

高可用性 - スタンバイ・データベースとOracle RACを使用して、計画停止および計画外停止の時間中に可用性を維持することもできます。このような事象には、構成の変更、ハードウェアの交換などのほか、データの破損、人為的エラーによる障害、システム・コンポーネントまたはシステム全体の予期しない障害が含まれることがあります。

Standby-First Patch - Active Data Guardでは、スタンバイ・データベースは、任意のハードウェア、オペレーティング・システム、グリッド・インフラストラクチャ、および正規のデータベース・ソフトウェアの更新を最初に適用することによって保護を強化できます。同様の変更をRACローリング方式でプライマリ・データベースに適用する前、またはData Guardのロール移行を実行する前には、保証の強化に数時間、数日、さらには数週間の検証が必要となる可能性があります。この保護強化により、不適切なパッチ適用による停止や、パッチ適用によるHAまたはパフォーマンスの低下を防止できます。データベースの停止時間は、プライマリとスタンバイの間でロールを変更するのに必要な短時間のみです。詳細は、My Oracle Support (MOS) のNote [1265700.1](#)『Oracle Patch Assurance - Data Guard Standby-First Patch Apply』を参照してください。

データベースのローリング・アップグレード - Active Data Guardと一時ロジカル・スタンバイにより、スタンバイ上でデータベースのメジャー・アップグレードなどの非ローリング・ソフトウェア変更を適用した後、スイッチオーバーすることによって、スタンバイ・データベースを使用して停止時間を最小限に抑えることができます。Data Guardのスイッチオーバーにより、停止時間を数秒に最小化できます。詳細は、12.1以上のデータベース・バージョンについては『Oracle Databaseローリング・アップグレード Oracle Data Guardフィジカル・スタンバイ・データベースの使用』(PDF) およびMAAの『Oracle Active Data GuardとDBMS_ROLLINGを使用した自動データベース・アップグレード』を参照してください。

ブロックの自動修復 - フィジカル・スタンバイ・データベースのそれ以外の利点の1つは、物理的なブロック破損を自動復旧する能力です。プライマリおよびスタンバイ構成では、破損ブロックを自動復旧し、この操作をアプリケーションおよびデータベース管理者から完全にシームレスに行うことができます。ブロック修復機能はOracle Active Data Guardオプションの一部です。

アプリケーション・コンティニューイティ (AC) - この機能は、Oracle Real Application Clusters (Oracle RAC)、Oracle RAC One Nodeオプション、Oracle Active Data Guardオプションで利用できる機能であり、リカバリ可能なシステム停止後に、実行中のデータベース・セッションをリカバリすることで、システム停止についてエンドユーザーやアプリケーションに気づかれないようにします。この機能では、停止後に影響を受けるデータベース・セッションの実行中の作業がリカバリされるため、停止がエンドユーザーやアプリケーションからマスクされます。アプリケーション・コンティニューイティはこのリカバリをアプリケーションの後方で実行するため、アプリケーションにとってこの停止は短時間の実行遅延のように見えます。アプリケーション・コンティニューイティによって、計画外停止および計画メンテナンスを実施する際、ユーザー・エクスペリエンスが向上します。最終的に、Oracleデータベースを利用するシステムやアプリケーションのフォルト・トレランスが強化されます。

ワークロードとアクティビティのオフロード - スタンバイ環境という名前ではありますが、アイドル状態である必要はありません。全体的な投資収益率を最大にするために積極的に利用することができます。フィジカル・スタンバイ・データベースを整備することにより、いくつかの主要なアクティビティをスタンバイ環境にオフロードすることができます。たとえば、次のことができます。

- » **読取り専用ワークロード** - Oracle Active Data Guardオプションを使用して、スタンバイ・モードのままプライマリ・データベースからREDOログの更新を受けながら、スタンバイ・データベースを読取り専用のクエリー・ワークロード用に開くことができます。多くの場合、スタンバイ・データベースに読取り専用ワークロードを割り当てると、本番のワークロードを大幅に低減することができるため、本番システムで利用可能な全体的な容量が増加します。
- » **バックアップ** - Oracle Data Guardのフィジカル・スタンバイ・データベースはプライマリ・データベースの物理コピーであるため、データベースのバックアップを完全にスタンバイ環境にオフロードし、これらのバックアップを、障害またはデータベース損失時のプライマリ・データベースのリストアおよびリカバリ用に透過的に使用することができます。Oracle Active Data Guardオプションのライセンスを取得すると、スタンバイ・データベースで高速な増分バックアップを行い、バックアップをスタンバイ・データベースにオフロードすることをさらに有意義なものにすることができます。
- » **スナップショット・スタンバイ** - スナップショット・スタンバイ・データベースは、プライマリ・データベース用の完全なデータ保護を備えた更新可能なスタンバイ・データベースです。プライマリから継続的にREDOデータを受信しますが、スタンバイ・データベースがテストのための読取り/書き込み操作に開かれている間は適用プロセスは停止します。テストが完了すると、単一のコマンドによってスタンバイ・データベースが元の状態に戻され、読取り/書き込みモードで開かれていた間に行われた変更が破棄され、蓄積されたREDOログが適用されて、現在のプライマリ・データベースの状態と同期化されます。

構成のベスト・プラクティス

このセクションでは、Oracle Database ApplianceでOracle Data Guardを設定するための重要なベスト・プラクティスについて説明します。一般的なOracle Data Guardのベスト・プラクティスの詳細なリストは、<https://www.oracle.com/database/technologies/high-availability/oracle-database-maa-best-practices.html>に掲載されたOracle Maximum Availability ArchitectureとOracle Data Guardのベスト・プラクティスを参照してください。これは、Oracle Database Appliance環境にも適用されます。

最新で最適なOracle Database Applianceバージョンを維持— データベース関連のメタデータの同期など、一部の機能は最新のOracle Database Applianceバージョンでのみ使用可能です。バックアップや他の一部の機能は、スタンバイ・データベースの最新のメタデータがないODAツール経由では動作しません。

Oracle Database Appliance 19.8リリースでは、Oracle Data GuardはOracle Database Applianceと統合されます。 odadiコマンドを使用して、Oracle Data Guardを別のOracle Database Applianceとともにすばやく設定して管理することができます。

プライマリ・データベースとスタンバイ・データベースの構成の一致— 整合性のあるサービス・レベルを維持し、プライマリ・データベースとスタンバイ・データベースを透過的に使用するには、プライマリ・システムとスタンバイ・システムのリソース、設定、構成をできるだけ一致させることが重要です。プライマリ・データベースとスタンバイ・データベースの構成が大きく異なっていると、ロールのトランジションが発生したときに最適なパフォーマンスが得られず、予期しない動作が生じる可能性があります。特に、次の推奨事項を考慮する必要があります。

- » **プライマリ・データベースとスタンバイ・データベースが個別のOracle Database Applianceで動作**— プライマリ・データベースとスタンバイ・データベースを、できれば地理的に離れた場所に個別に設置された専用のOracle Database Applianceユニット上で動作させることを推奨します。
- » **プライマリ・データベースとスタンバイ・データベースが同じ構成で動作**— Oracle Database Applianceでは3種類のデータベース構成、つまりOracle RACデータベース、Oracle RAC One、シングル・インスタンスEnterprise Editionデータベースがサポートされています。また、スタンバイ・データベースはプライマリ・データベースと同じ構成タイプでなければなりません。そのため、プライマリ・データベースがOracle RACデータベースとして構成されている場合、スタンバイ・データベースもOracle RACデータベースとして構成する必要があります。
- » **プライマリ・サイトとスタンバイ・サイトの対称性を維持**— プライマリ・データベースとスタンバイ・データベースのインスタンスは、メモリ、CPU、ネットワーク、ストレージなどのデータベース・パラメータ設定について、同様に構成する必要があります。これにより、データベースのロールを切り替えた場合に予期しない動作が発生するという状況を回避することができます。また、オペレーティング・システムの構成を2つの環境でミラー化する必要があります。
- » **プライマリ・データベースとスタンバイ・データベース両方でのフラッシュバック・データベースの構成**— フラッシュバック・データベース機能を利用すると、ロールのトランジションを高速で行い、トランジション後にデータベースのロールを再確立するのに必要な労力を軽減することができます。ベスト・プラクティスとしては、フラッシュバック・データベースをプライマリ・データベースとスタンバイ・データベースの両方で構成する必要があります。再インスタンス化のためだけにフラッシュバックが必要と考えられる場合、保存時間をデフォルトの24時間から2時間に短縮することをお勧めします。Oracle Database 19cのリリース時点では、すべてのリストア・ポイントはスタンバイ・データベースに自動的に伝播されます。
- » **スタンバイのトラフィック専用のネットワークの使用**— Oracle Database Applianceには複数の冗長ネットワーク・インタフェースが装備されています。必要に応じて、ユーザーとアプリケーションに関連するワークロードに対するパフォーマンスの影響を最小限に抑えるため、スタンバイのトラフィック用に専用のネットワーク・パスを構成することができます。Oracle Data Guardではプライマリ・データベースに対して行った変更のみをプライマリ・データベースからスタンバイ・データベースに転送する必要があるため、必要以上の要件がネットワークに課されることはありません。そのため、Oracle Data Guardを多数展開するために、プライマリとスタンバイ間でREDOログの転送用の専用のネットワーク・パスが不要場合があります。しかし、一部の高容量のアプリケーションまたは組織のベスト・プラクティスと基準では、REDOログの転送用に専用のネットワーク・パスが必要になることもあります。Oracle Database Applianceは、このために使用可能な各サーバー・ノードの追加のネットワーク・インタフェースを備えています。Oracle Database Applianceのディザスタ・リカバリ用の専用ネットワークの構成に関する詳細は、MOSのNote [1422563.1](#)を参照してください。
- » **Oracle Active Data Guardの利用**— REDO Applyはプライマリ環境とスタンバイ環境の間で引き続きアクティブのままとなるため、Oracle Active Data Guardにより、読取り専用のスタンバイで最新に近いデータが利用できます。そのため、読取り専用のワークロードをプライマリ環境からスタンバイ・データベースに分配またはオフロードし、スタンバイ・データベースでの投資収益率を高めることができます。Oracle Active Data Guardを使用すると、スタンバイ・データベースで増分バックアップを高速で実行することができます。高速な増分バックアップでは、バックアップ時間を時間単位から分単位に短縮できる可能性があります。スタンバイ・データベースを使用してローリング・アップグレードを実行し、停止時間をほぼゼロにすることもできます。また、リアルタイム適用を備えたOracle Active Data Guardでは双方向の自動ブロック破損の修復が可能で、ミッション・クリティカルなアプリケーションのためのデータ保護にもう1つのレイヤーが追加されます。
- » **Oracle Data Guard Brokerの使用**— Oracle Data Guard Brokerのインタフェースにより、使いやすさが向上し、Oracle Data Guard構成の管理と監視が一元化されます。これにより、管理全体が最小限に抑えられ、Data Guard構成固有のチェックアンドバランスが維持されます。詳細は、[Benefits of Oracle Data Guard Broker](#)を参照してください。
- » **ラスタウェアのロール・ベース・サービスのセットアップ**— 『高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティス』を参照してください

» **Oracle Database用のOracle Maximum Availability Architecture (Oracle MAA) のベスト・プラクティスのレビュー** - Oracle Data Guard環境の展開および使用、その他の要件によって、次のような多数の有益なOracle MAAのベスト・プラクティスを利用できる場合があります。

- o [Oracle Maximum Availability Architecture \(Oracle MAA\) オンプレミス、Exadataベース、またはクラウドベースのOracle Databaseの計画停止時間と計画外停止時間の短縮構想](#)
- o [高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティス Oracle Database 12c](#)
- o [Oracle Data GuardおよびOracle Active Data Guard 12cのREDO転送を構成するためのベスト・プラクティス](#)
- o [Best Practices for Asynchronous Redo Transport - Data Guard and Active Data Guard](#)
- o [同期REDO転送のベスト・プラクティス Oracle Data GuardとOracle Active Data Guard](#)
- o [Best Practices for Automatic Resolution of Outages to Resume Data Guard Zero Data Loss](#)
- o [ロール移行のベスト・プラクティス：Oracle Data GuardおよびOracle Active Data Guard](#)
- o [ブロック破損の防止、検出、および修復：Oracle Database 12c](#)
- o [高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティス](#)

上記のベスト・プラクティス、その他については、<https://www.oracle.com/database/technologies/high-availability/oracle-database-maa-best-practices.html> を参照してください。

Oracle Database Applianceシステム間でのOracle Data Guardの設定

Oracle Data Guardの構成手順

プライマリ・データベースのバージョンによって、Oracle Data Guardのフィジカル・スタンバイ・データベース環境を設定するための方法は異なります。

Oracle Database Appliance 19.8ベア・メタル・デプロイメント – データベースのリリース更新、バンドル・パッチ、PSUのバージョンにかかわらず、すべてのデータベース・バージョンについて、`odacli`コマンドを使用してOracle Data Guardを構成します。

「[Configuring Oracle Data Guard on Oracle Database Appliance](#)」を参照してください。

この19.8の新機能を使用してOracle Database Data Guardを構成する場合の前提条件：

- ▶ プライマリ・データベースとスタンバイ・データベースを別々のOracle Database Applianceハードウェア上で運用して、少なくとも2台のOracle Database Applianceマシンを確保することをお勧めします。
- ▶ プライマリ・データベースとスタンバイ・データベースを2台の同種のOracle Database Applianceシステム上で運用することをお勧めします。たとえば、プライマリ・データベースが高可用性システムの場合は、スタンバイ・データベースも同様の高可用性システム上で稼働させるようにします。
- ▶ プライマリ・システムとスタンバイ・システムを同じOracle Database Appliance構成にすることをお勧めします。データベースのロールを切り替えた場合に予期しない動作が発生しないようにするには、これらのデータベースのデータベース・シェイプ、バージョン、メモリ、ネットワーク、ストレージを同様に構成する必要があります（両方のデータベースにOracle ASM、Oracle ACFSのいずれかのストレージを構成する必要があります）。
- ▶ プライマリ・システムとスタンバイ・システムには同じOracle Database Applianceのリリースを使用し、その際にはOracle Database Applianceリリース19.8以上とする必要があります。
- ▶ オペレーティング・システムをカスタマイズした場合は、両方のマシンの環境が同一になるようにします。

この技術概要では、Oracle Data GuardをOracle Database Applianceのベア・メタルおよび仮想化プラットフォーム上に構成する方法について説明します。

同様に構成された2台のOracle Database Applianceベア・メタル（プライマリとスタンバイ）の両方で、Oracle Database Appliance 19.8以上を実行する場合に推奨されるOracle Data Guardの構成方法は、組み込みのOracle Database Applianceコマンドを使用することです。このコマンドにより、データベース・アップグレードやパッチ適用などの、Oracle Data Guard構成のライフサイクル全体を容易かつ効率的に管理できます。Oracle Database Appliance 19.8と統合されたOracle Data Guardについては、適用される制限事項がないか要件を確認してください。

19.8よりも前のバージョンのOracle Database Applianceベア・メタル、およびすべてのバージョンのOracle Database Appliance仮想化プラットフォーム

データベース・バージョン11.2 – データベース・バージョン11.2には標準のRMAN DUPLICATEメソッドをお勧めします。このメソッドは上位のバージョンでも有効ですが、以下のようなオプションも利用できます。詳細はMOS Note [1617946.1](#)の『Creating a Physical Standby Database using RMAN Duplicate (RAC or Non-RAC)』を参照してください。

注：Oracle Database Applianceプラットフォームを使用してOracle 11gデータベース用のプライマリ/スタンバイ構成を作成するための詳細な手順の例を、この技術概要の付録Cに示します。

データベース・バージョン12.1、12.2および18.x、19.x – データベース・バージョンが12.1.0.2以上の場合、RMAN 'restore... from service'メソッドも使用できます。'restore... from service'メソッドを使用してスタンバイ・データベースをインスタンス化する方法については、MOS Note [2283978.1](#)の『Creating a Physical Standby database using RMAN restore database from service』を参照してください。RMAN 'restore... from service'句を使用して、ネットワーク上でプライマリ・データベース・ファイルをスタンバイ・データベースにオンライン・リストアおよびリカバリすることができます。このメソッドでは、SECTION SIZE句を使用して複数のRMANチャンネル上でリストアをパラレル化することもできます。

注：Oracle Database Applianceプラットフォームを使用してOracle 19cおよび12cデータベース用のプライマリ/スタンバイ構成を作成するための詳細な手順の例を、この技術概要の付録AとBに示します。

上記の文書に従ってOracle Data Guard構成でプライマリ・データベース環境とスタンバイ・データベース環境を設定するときは、Oracle Database Applianceプラットフォームに固有の次のガイドラインに従ってください。

- » バグがあるため、スタンバイ・システムをインスタンス化しOracle Database Applianceをプラットフォームとして使用するためにOracle Enterprise Managerを使用することはできません。これはバグによるものです。しかし、上記の注、またはこの技術概要の付録に示した例に従って、11gおよび12c、18c、19c環境を構成することができます。
- » 古いスタック（OAK）でOracle ACFSストレージを使用している場合は、スタンバイ・データベースをインスタンス化する前に、スタンバイのOracle Database Applianceシステムのデータベース記憶域を作成しておいてください。“oakcli create dbstorage”コマンドをrootユーザーで使い、スタンバイ・データベース用のACFSストレージを作成してから、スタンバイ・データベースをインスタンス化してください。たとえば、

```
# oakcli create dbstorage -db stbydb
```

この技術概要の付録に記載した詳細な構成手順の例を参照してください。

- » 新しいスタック（DCS）でインスタンス専用データベースを事前に作成します。“oakcli create-database -io”コマンドをrootユーザーで使い、スタンバイ・データベース用のストレージ構造を作成してDCSに登録してから、スタンバイ・データベースをインスタンス化してください。たとえば、データベース記憶域がASM上にある場合：

```
# odacli create-database -m -u boston -n chicago -io -dh 2d7d2b0b-8489-4e28-a151-64f6e156a512
```

データベース記憶域がACFS上にある場合：

```
# odacli create-database -m -u boston -n chicago -r ACFS -io -dh 2d7d2b0b-8489-4e28-a151-64f6e156a512
```

この技術概要の付録に記載した詳細な構成手順の例を参照してください。

- » Oracle Data Guardは、データベース記憶域オプションにかかわらず、新しいスタック（DCS）と古いスタック（OAK）との間で構成できます。
- » Oracle Database Applianceベア・メタルおよびOracle Database Appliance仮想化プラットフォームの展開で、スタンバイ・データベースの展開手順を使用することができます。

Oracle Database Applianceベア・メタルおよび仮想化プラットフォームの構成

Oracle Database Applianceはベア・メタル（非仮想化）プラットフォームまたは仮想化プラットフォームとして構成することができます。この技術概要で示すOracle Data Guardフィジカル・スタンバイの設定プロセスは両方のOracle Database Appliance構成、つまりベア・メタルと仮想化プラットフォームで使用できます。Oracle Database Appliance仮想化プラットフォームでは、ODA_BASEドメイン内で構成手順が実行されます。また、Oracle Database Appliance仮想化プラットフォームで仮想LANを使用して、ディザスタ・リカバリ用に論理的に分離したネットワークを構成することができます。

Oracle Database Applianceの小規模、中規模、大規模プラットフォームの構成

Oracle Real Application Clusters（Oracle RAC）およびOracle Data GuardはOracle Maximum Availability Architecture（Oracle MAA）の基本的で不可欠なコンポーネントです。Oracle Database Appliance X6-2 S|M|L、X7-2 S|M、X8-2 S|Mハードウェア・モデル（小規模なシングル・ノード構成）間でOracle Data Guard構成を設定することもできますが、Oracle Real Application Clusters（Oracle RAC）を実行できるのはOracle Database Appliance HAハードウェア・モデル（X4-2、X5-2、X6-2 HA、X7-2 HA、X8-2 HA）上のみであるため、このような構成はOracle MAAのガイドラインに準拠しません。

結論

Oracle Data Guardを使用すると、Oracle Database Applianceの初期の展開時点から、効果的なディザスタ・リカバリ保護戦略をすばやく展開することができます。ディザスタ・リカバリ・ソリューションとして使用する以外にも、さまざまな目的でOracle Data Guardフィジカル・スタンバイ環境を使用できます。この技術概要で示しているフィジカル・スタンバイ構成と設定プロセスはすばやく簡単に使用できるもので、プライマリ・データベースの停止時間なしで完了させることができます。スタンバイの作成手順の大部分はOracle Appliance Manager、RMAN、Oracle Data Guard Brokerなどのツールを使用して自動で実行することができます。

付録A：DCSスタックを使用したOracle Database Applianceでの19cの設定例

環境の例

次のセクションで、Oracle Database Applianceを使用してOracle Data Guardを設定する次の例で使用する、プライマリ・データベース環境とスタンバイ・データベース環境のトポロジについて説明します。

注：Oracle Database Appliance 19.8 BMでは、`odacli`コマンドを使用してOracle Data Guardを構成および管理してください。前提条件と手順については、「[Configuring Oracle Data Guard on Oracle Database Appliance](#)」を参照してください。

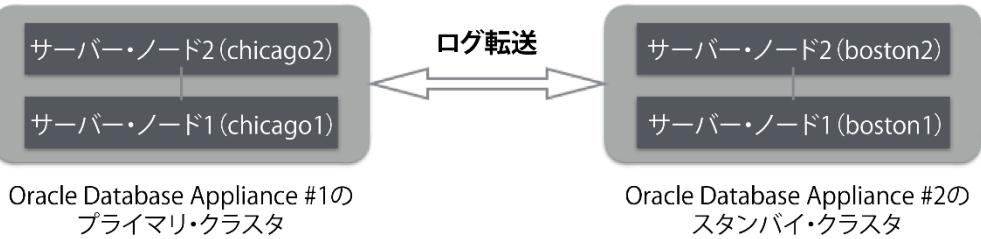


図1 Oracle Database ApplianceでのOracle RACの構成トポロジ

	プライマリOracle Database Appliance		スタンバイOracle Database Appliance	
アプライアンス名	アプライアンス #1		アプライアンス #2	
ホスト名	proddb1	proddb1	stbydb1	stbydb2
クラスタ名	PCLUSTER		SCLUSTER	
データベース名	chicago		chicago	
データベースの一意名	chicago		boston	
インスタンス名	chicago1	chicago2	boston1	boston2
スキャン名とIP	proddb-scan (10.1.27.2, 10.1.27.3)		stbydb-scan (10.1.27.4, 10.1.27.5)	
グリッド・インフラストラクチャ・ソフトウェアのインストール	/u01/app/19.0.0.0/grid		/u01/app/19.0.0.0/grid	
Oracle Databaseソフトウェアのインストール	/u01/app/oracle/product/19.0.0.0/db_home1		/u01/app/oracle/product/19.0.0.0/db_home1	
データベース記憶域	ASM		ASM	
ARCHIVELOGモード	対応		対応	
FORCE LOGGINGモード	対応		対応	

表1 - Oracle Databaseのプライマリ構成およびスタンバイ構成の例

オラクルのベスト・プラクティスでは、Oracle Data Guard Brokerを使用してOracle Data Guardを構成することを強くお勧めします。

1. スタンバイREDOログを作成します。

スタンバイREDOログ（SRL）はプライマリ・データベースからREDOデータをリアルタイムで受信し、転送ラグと適用ラグを最小限に抑えます。プライマリとスタンバイの設定前に、スタンバイREDOログをプライマリ・データベースでも作成し、Data Guardのロール移行後にすぐにREDOデータを受信できるようにしておくことをお勧めします。

プライマリ・データベースでスタンバイREDOログを作成します。スタンバイREDOログの各スレッドには、対応するオンラインREDOログのスレッドよりも少なくとも1つ多いREDOログ・グループが必要です。たとえば、次のようにします。

```
SQL> alter database add standby logfile thread 1 group 7 size 1G, group 8 size 1G, group 9 size 1G, group 10 size 1G;  
SQL> alter database add standby logfile thread 2 group 11 size 1G, group 12 size 1G, group 13 size 1G, group 14 size 1G;
```

オンラインREDOログの数とそのサイズを確認するには、次のクエリーを使用します。

```
SQL> select thread#, group#, bytes/1024/1024 SIZE_IN_GB, status from v$log;
```

スタンバイREDOログのサイズがREDOログのサイズと一致するようにしてください。Oracle Database Applianceプラットフォームのソリッド・ステート・ディスクに常駐するREDOディスク・グループ上に、スタンバイREDOログを作成する必要があります。ODA Small/Medium/LargeおよびX8-2 HAモデルの場合はREDOディスク・グループがないため、制御ファイル、オンライン・ログはRECOディスク・グループに格納されます。

スタンバイREDOログで各ログ・ファイルのサイズとログ・グループの数を検証するには、次のクエリーを使用します。

```
SQL> select group#, thread#, bytes/1024/1024 SIZE_IN_GB from v$standby_log;
```

2. プライマリ・データベースでアーカイブ・ログ・モードを有効化します。

アーカイブとは、アクティブなデータベースのREDOログが循環的に上書きされる前にアーカイブ・ファイルの形式でREDO情報を保存し、保護するプロセスのことです。Oracle Database Applianceで作成されたデータベースは、デフォルトでアーカイブがオンになっています。ただし、Oracle Database Applianceのデフォルト設定であるアーカイブ・ログ・モードでデータベースを動作させる必要は必ずしもありません。

プライマリ・データベースがARCHIVELOGモードで動作していることを確認します。

```
SQL> archive log list
```

プライマリ・データベースがARCHIVELOGモードで動作していない場合、次のようにARCHIVELOGモードを有効化します。Oracle Database Applianceで両方のインスタンスをシャットダウンします。

```
$ srvctl stop database -d chicago
```

1つのインスタンスを排他モードで起動マウントします。

```
SQL> startup mount exclusive;
```

アーカイブをオンにします。

```
SQL> alter database archivelog;
```

インスタンスをシャットダウンします。

```
SQL> shutdown immediate;
```

データベースを再起動します。

```
$ srvctl start database -d chicago
```

3. FORCE LOGGINGモードを有効化します。

強制ロギングを使用すると、NOLOGGING属性により実行されるデータベース操作を取得することができます。これにより、スタンバイ・データベースの整合性が保証されます。FORCE LOGGINGがプライマリ・データベースで有効化されているかどうか確認します。

```
SQL> select force_logging from v$database;
```

FORCE LOGGINGが有効化されていない場合、次のコマンドで有効化します。

```
SQL> alter database force logging;
```

4. フラッシュバック・データベース機能を構成します。

Oracleフラッシュバック・データベース機能は、不完全なデータベース・リカバリ実行に対する高速の代替手段となります。フラッシュバック・データベース機能の使用はオプションですが、フェイルオーバー後に古いプライマリ・データベースをすばやく元の状態に回復させるのに非常に有益な場合があります。そのため、スタンバイへのフェイルオーバーを行い、古いプライマリの修復が可能である場合、古いプライマリ・データベースをスタンバイ・データベースとして再構築する必要はなく、フラッシュバックし、Oracle Data Guardをそのポイントから再同期化させるだけで済みます。

プライマリ・データベースでフラッシュバック・データベースが有効化されているかどうか確認し、必要に応じて有効化します。

```
SQL> select flashback_on from v$database; SQL>  
alter database flashback on;
```

フラッシュバック・データベースを有効化するには、ファスト・リカバリ領域（RECOディスク・グループ）で追加の使用スペースが必要になります。フラッシュバック・ログによって使用されるスペースを制御するには、パラメータDB_FLASHBACK_RETENTION_TARGETを望ましい値に設定します。この値は分単位で指定します。たとえば、次のようにします。

```
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120 scope=both sid='*';
```

5. スタンバイ・ファイル管理を有効化します。

プライマリ・データベースでデータ・ファイルの追加または削除が行われる場合、スタンバイ・データベースでもそれに対応するアクションが自動的に実行されなければなりません。この操作を有効化するには、自動スタンバイ・ファイル管理を使用します。

```
SQL> alter system set STANDBY_FILE_MANAGEMENT=AUTO scope=both sid='*';
```

6. スタンバイのデータベース・ホームが存在しなかった場合は作成します。

```
[root@stbydb1]# odacli create-dbhome -v 19.8.0.0.200714
```

スタンバイのデータベース・ホームのバージョンはプライマリ・データベースのバージョンと同一である必要があります。

7. TNSエントリおよびリスナーを設定します。

データベース間でのREDO転送を有効化するには、Oracleネット・サービス名を構成する必要があります。プライマリ・データベースとスタンバイ・データベースの両方のTNSエイリアスが含まれるようにtnsnames.oraファイルを更新します。Oracle Database Applianceではtnsnames.oraファイルがOracleデータベース・ホームのnetwork/adminディレクトリに配置されることに注意してください。

```
$ vi $ORACLE_HOME/network/admin/tnsnames.ora Primary  
chicago =  
(DESCRIPTION =  
(ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))  
(CONNECT_DATA =  
(SERVER = DEDICATED)  
(SERVICE_NAME = chicago.us.oracle.com)  
)  
)
```

```

boston =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = boston.us.oracle.com)
    )
  )
)

```

Standby

```

chicago =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = chicago.us.oracle.com)
    )
  )
)

```

```

boston =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = boston.us.oracle.com)
    )
  )
)

```

8. REDO転送サービスを遅延モードで設定します。DG Brokerも構成する場合、この手順は不要です。

Oracle Data GuardのREDO転送メカニズムでは、Oracle Net接続を使用してデータベース間でREDOを送信します。REDO転送を有効化するには、LOG_ARCHIVE_DEST_nパラメータを設定します。たとえば、次の設定により、ログの送信が有効化され、LGWRベースの転送が非同期モードで使用されます。

```

SQL> alter system set log_archive_dest_2='SERVICE=boston LGWR ASYNC REGISTER VALID_FOR=(online_logfile,primary_role) REOPEN=60
DB_UNIQUE_NAME=boston' scope=both sid='*';
SQL> alter system set log_archive_dest_state_2='defer' scope=both sid='*';

```

REDOログ転送オプションの詳細は、『Oracle Data Guard概要および管理』を参照してください。

9. フェッチ・アーカイブ・ログ・サーバーを設定します。DG Brokerも構成する場合、この手順は不要です。

データベースがスタンバイ・ロールで、プライマリが欠落したログ・ファイルを送信できない場合、スタンバイ・データベースでFAL_SERVER設定を使用して欠落したログ・ファイルを取得することができます。FAL_SERVERパラメータではOracle Netサービス名を使用します。

```

SQL> alter system set FAL_SERVER=boston scope=both sid='*';

```

10. プライマリ・データベースのspfileからpfileを作成します。

```

[oracle@proddb1]$ export ORACLE_HOME=/u01/app/oracle/product/19.0.0.0/dbhome_1
[oracle@proddb1]$ export ORACLE_SID=chicago1
[oracle@proddb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@proddb1]$ sqlplus / as sysdba
SQL> create pfile='/tmp/chicago.pfile' from spfile;

```


11. プライマリ/スタンバイでパラメータを追加/変更します。以下に例を示します。

プライマリとスタンバイでのパラメータの変更	
プライマリ	スタンバイ
<pre> *.cluster_database=TRUE chicago2.instance_number=2 chicago1.instance_number=1 chicago2.thread=2 chicago1.thread=1 chicago2.undo_tablespace='UNDOTBS2' chicago1.undo_tablespace='UNDOTBS1' *.db_block_checking=FULL *.db_block_checksum=FULL *.db_lost_write_protect=TYPICAL *.db_unique_name=chicago *.listener_networks='((NAME=net1)(LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<primary node0's vip> (PORT=1521)))));'((NAME=net1)(LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(PR OTOCOL=TCP)(HOST=<primary node1's vip>)(PORT=1521)))));'((NAME=net1)(REMOTE_LISTENER=<primary's scan name>:1521))' *.LOG_FILE_NAME_CONVERT='+REDO/BOSTON/','+REDO/CHICAGO/' *.DB_FILE_NAME_CONVERT='+DATA/BOSTON/','+DATA/CHICAGO/' *.LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=chicago' 次のパラメータを追加 *.audit_file_dest='/u01/app/oracle/admin/chicago/adump' *.fal_server='boston' *.remote_login_passwordfile='exclusive' </pre>	<pre> *.cluster_database=TRUE boston2.instance_number=2 boston1.instance_number=1 boston2.thread=2 boston1.thread=1 boston2.undo_tablespace='UNDOTBS2' boston1.undo_tablespace='UNDOTBS1' *.db_block_checking=FULL *.db_block_checksum=FULL *.db_lost_write_protect=TYPICAL *.db_unique_name=boston *.listener_networks='((NAME=net1)(LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=<standby node0's vip> (PORT=1521)))));'((NAME=net1)(LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(P ROTOCOL=TCP)(HOST=<standby node1's vip>)(PORT=1521)))))';'((NAME=net1)(REMOTE_LISTENER=<standby's scan name>:1521))' *.LOG_FILE_NAME_CONVERT='+REDO/CHICAGO/','+REDO/BOSTON/' *.DB_FILE_NAME_CONVERT='+DATA/CHICAGO/','+DATA/BOSTON/' *.LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=boston' 次のパラメータを追加 *.audit_file_dest='/u01/app/oracle/admin/boston/adump' *.fal_server='chicago' *.remote_login_passwordfile='exclusive' </pre>

注：

データ保護パラメータは適宜設定する必要があります。 [Note 1302539.1](#) 『Data Guard構成における破損の検出、防止および自動修復に関するベスト・プラクティス』を参照してください。

Oracle Database Appliance Small/Medium/LargeおよびX8-2 HAモデルの場合はREDOディスク・グループがないため、制御ファイル、オンライン・ログはRECOディスク・グループに格納されます。

Oracle Database Appliance 19.6以上のベア・メタル・プラットフォームでは、`local_listener`と`remote_listener`の代わりに`listener_networks`パラメータがデータベースに使用されます。

12. すべてのスタンバイ・ホストで、bostonデータベース用の監査ディレクトリを作成します。

```
[oracle@stbydb1] mkdir -p /u01/app/oracle/admin/boston/adump
[oracle@stbydb2] mkdir -p /u01/app/oracle/admin/boston/adump
```

13. スタンバイでストレージ構造を作成し、スタンバイ・データベースをDCSに登録します。

```
[root@rwsodam001 ~]# odacli list-dbhomes
```

ID	Name	DB Version	Home Location	Status
755b4b5d-6211-4d94-81e8-cf611868fe39	OraDB19000_home1	19.8.0.0.200714	/u01/app/oracle/product/19.0.0.0/dbhome_1	Configured

```
[root@stbydb1]# odacli create-database -m -u boston -n chicago -io -dh 755b4b5d-6211-4d94-81e8-cf611868fe39
```

インスタンス専用データベースの準備ができるまで待機し、準備ができたら先に進みます。ステータスはodacli describe-jobsで確認します。

```
[root@unaodapoc1 ~]# odacli describe-job -i da26334e-c5ac-411b-b38d-2f6aef6ca090
```

Job details

```
ID: da26334e-c5ac-411b-b38d-2f6aef6ca090
Description: Database service creation with db name: chicago
Status: Success
Created: July 3, 2020 6:02:18 PM CEST
Message:
```

両方のノードでインスタンスがノーマウント・モードで動作している可能性があります。その場合はインスタンスを停止または強制終了します。

14. パスワードをコピーします。

パスワード・ファイルをプライマリ・データベースから最初のスタンバイ・ホストにコピーします。

```
[oracle@proddb1]$ srvctl config database -d chicago |grep Password Password
file:
+DATA/CHICAGO/PASSWORD/pwdchicago.277.1023633847

[grid@proddb1 ~]$ asmcmd
ASMCMDB> pwcop +DATA/CHICAGO/PASSWORD/pwdchicago.277.1023633847 /tmp/pwdboston copying
+DATA/CHICAGO/PASSWORD/pwdchicago.277.1023633847 -> /tmp/pwdboston
[grid@proddb1]$ scp /tmp/pwdboston oracle@stbydb1:/u01/app/oracle/product/19.0.0.0/dbhome_1/dbs/orapwboston

[grid@stbydb1 ~]$ asmcmd
ASMCMDB>
mkdir +DATA/BOSTON
ASMCMDB> mkdir +DATA/BOSTON/PASSWORDFILE
ASMCMDB> pwcop /u01/app/oracle/product/19.0.0.0/dbhome_1/dbs/orapwboston +DATA/BOSTON/PASSWORDFILE/pwdboston
```

15. 変更されたpfileを最初のスタンバイ・ホストにコピーし、スタンバイ・データベースをマウントします。スタンバイ制御ファイルが作成されたパスをメモします。

```
[oracle@proddb1]$ scp /tmp/chicago.pfile oracle@stbydb1.us.oracle.com:/tmp/boston.pfile
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/19.0.0.0/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ cp /u01/app/oracle/product/19.0.0.0/dbhome_1/dbs/orapwboston
/u01/app/oracle/product/19.0.0.0/dbhome_1/dbs/orapwboston1 [oracle@stbydb1]$ rman target /
RMAN> startup nomount pfile='/tmp/boston.pfile';
RMAN> restore standby controlfile from service chicago;
Starting restore at 05-JUL-20

using target database control file instead of recovery catalog allocated channel:ORA_DISK_1
channel ORA_DISK_1:SID=26 device type=DISK
channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: using network backup set from service Chicago
channel ORA_DISK_1: restoring control file
channel ORA_DISK_1: restore complete, elapsed time:00:00:01

output file name=+REDO/BOSTON/CONTROLFILE/current.1492.986894783 Finished restore at 05-JUL-20
```

16. 制御ファイル・パラメータを更新します。

pfile '/tmp/chicago.pfile'を編集し、前の出力からの新しいパスを示すようにcontrol_filesパラメータを置換します。たとえば、
control_files= '+REDO/BOSTON/CONTROLFILE/current.1492.986894783'

```
[oracle@stbydb1]$ vi /tmp/boston.pfile
```

17. スタンバイ・インスタンスを起動します。

変更後のpfileを使用してスタンバイ・インスタンスをノーマウント・モードで起動し、spfileを作成し、spfileを使用してインスタンスを再起動します。

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/19.0.0.0/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> startup nomount force pfile='/tmp/boston.pfile';

SQL> create spfile='+DATA/BOSTON/PARAMETERFILE/spfileboston' from pfile='/tmp/boston.pfile';
SQL> !echo "spfile='+DATA/BOSTON/PARAMETERFILE/spfileboston'" >
/u01/app/oracle/product/19.0.0.0/dbhome_1/dbs/initboston1.ora
SQL> startup mount force;
```

18. 並列処理を有効化してSECTION SIZE=64MBに設定します。

リストア中に並列処理を利用するには、以下を実行してサーバーのCPUの数を確認します。

```
[oracle@stbydb1]$ grep -c ^processor /proc/cpuinfo  
20
```

スタンバイで以下のRMANの構成変更を行います。

この例では、リカバリ・プロセス中に使用するためにRMAN用に構成済みの8つのチャンネルを使用します。

```
[oracle@stbydb1]$ rman target /  
RMAN> CONFIGURE DEFAULT DEVICE TYPE TO DISK;  
RMAN> CONFIGURE DEVICE TYPE DISK PARALLELISM 8;
```

19. プライマリ・データベース・サービスからスタンバイ・データベースをリストアします。

1つの大きなファイルを並行してバックアップするには、RMANのマルチセクション・バックアップ/リストア機能によりバックアップとリカバリの速度が向上します。RMANは内部で作業を複数のチャンネル間で分割し、各チャンネルがファイル内の1つのファイル・セクションに作用します。小さなセクション・サイズを指定してセクションの数が256を超えることになる場合、RMANにより、ちょうど256個のセクションとなる値にセクション・サイズが増加されます。

セクション・サイズ句はさまざまな要素、たとえば、ネットワーク帯域幅、チャンネル数、データ・ファイルのサイズ、アプリケーション・データベースのサイズなどによって決まります。

```
[oracle@stbydb1]$ sqlplus system/welcome1@chicago  
SQL> select TABLESPACE_NAME, bytes/1024/1024 SIZE_IN_GB from dba_data_files;
```

TABLESPACE_NAME	SIZE_IN_GB
SYSTEM	.68359375
SYSAUX	.5859375
UNDOTBS1	.297851563
UNDOTBS2	.1953125
USERS	.004882813

たとえば、スタンバイ・ホストで実行される次のコマンドでは、64 MBのバックアップ・セクション・サイズを指定します。

```
[oracle@stbydb1]$ rman target /  
RMAN> restore database from service chicago section size 64M;  
RMAN> switch database to copy;  
RMAN> recover database from service chicago;  
RMAN> backup spfile;
```

20. プライマリでログの送信を有効にします。この手順が必要なのは、Data Guard Brokerを構成しない場合のみです。

```
[oracle@proddb1]$ sqlplus / as sysdba  
SQL> alter system set log_archive_dest_state_2='enable' scope=both;
```

21. スタンバイでフラッシュバック・データベースを有効化し、必要に応じて保持期間を調節します。この手順が必要なのは、Data Guard Brokerを構成しない場合のみです。

```
SQL> alter database flashback on;  
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120;
```

22. スタンバイ側で管理リカバリを開始します。この手順が必要なのは、Data Guard Brokerを構成しない場合のみです。

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

23. Clusterwareを使用してスタンバイ・データベースを登録します。

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/19.0.0.0/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
```

シングル・インスタンスの例

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/19.0.0.0/dbhome_1 -dbtype SINGLE
-instance boston1 -node stbydb1 -dbname chicago -diskgroup 'DATA,REDO,RECO' -role physical_standby -spfile
'+DATA/BOSTON/PARAMETERFILE/spfileboston' -pwfile '+DATA/BOSTON/PASSWORDFILE/pwdboston'
```

Oracle RACの例

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/19.0.0.0/dbhome_1 -dbtype RAC -
dbname chicago -diskgroup 'DATA,RECO,REDO' -role physical_standby -spfile
'+DATA/BOSTON/PARAMETERFILE/spfileboston' -pwfile '+DATA/BOSTON/PASSWORDFILE/pwdboston'
[oracle@stbydb1]$ srvctl add instance -db boston -instance boston1 -node stbydb1
[oracle@stbydb1]$ srvctl add instance -db boston -instance boston2 -node stbydb2
[oracle@stbydb1]$ scp $ORACLE_HOME/dbs/initboston1.ora
oracle@stbydb2:/u01/app/oracle/product/19.0.0.0/dbhome_1/dbs/initboston2.ora
[oracle@stbydb1]$ srvctl start instance -db boston -instance boston1 -o mount
[oracle@stbydb1]$ srvctl start instance -db boston -instance boston2 -o mount
```

24. パラメータを設定し、Oracle Data Guard Broker構成を作成します。

環境に合わせて以下のスクリプトを変更し、PostCR.sqlとして保存します。

注：フェイルオーバーのロール・トランジション後に障害となったプライマリを再インスタンス化するにはフラッシュバック・データベースが必要です。オプションでプライマリとスタンバイの両方でフラッシュバックを有効化します。スタンバイ・データベースは、PostCRスクリプトを次のように使用して、フラッシュバックをオンにして開始することができます。

```
[oracle@stbydb1]$ cat PostCR.sql
connect / as sysdba
alter system set dg_broker_config_file1='+DATA/BOSTON/dr1.dat' scope=both;
alter system set dg_broker_config_file2='+DATA/BOSTON/dr2.dat' scope=both;
alter system set db_flashback_retention_target=120 scope=spfile;
alter database flashback on;
alter system set dg_broker_start=true;
shutdown immediate
startup mount
alter system register;
connect sys/welcome1@chicago as sysdba
alter system set dg_broker_config_file1='+DATA/CHICAGO/dr1.dat' scope=both;
alter system set dg_broker_config_file2='+DATA/CHICAGO/dr2.dat' scope=both;
alter system set dg_broker_start=TRUE;
host sleep 30
host dgmgrl sys/welcome1@chicago "CREATE CONFIGURATION dgconfig AS PRIMARY
DATABASE IS CHICAGO CONNECT IDENTIFIER IS CHICAGO";
host sleep 30
host dgmgrl sys/welcome1@chicago "ADD DATABASE BOSTON AS CONNECT IDENTIFIER IS BOSTON" ;
host dgmgrl sys/welcome1@chicago "ENABLE CONFIGURATION"
exit
```

スタンバイ・データベースでスクリプトPostCR.sqlを実行します。環境をスタンバイ・データベースに設定します。

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/19.0.0.0/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> @PostCR.sql
```

ORA-38788で‘ALTER DATABASE FLASHBACK ON’が失敗した場合、スタンバイを同期させ、次の手順を実行してからフラッシュバックを有効化してください。

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
SQL> alter database flashback on;
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT;
```

25. sqlplus/srvctlを使用して検証します。

```
[oracle@stbydb1]$ srvctl config database -d chicago
[oracle@stbydb1]$ srvctl config database -d boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> select FORCE_LOGGING, FLASHBACK_ON, OPEN_MODE, DATABASE_ROLE, SWITCHOVER_STATUS,
DATAGUARD_BROKER, PROTECTION_MODE from v$database;
SQL> select PROCESS,PID,DELAY_MINS from V$MANAGED_STANDBY;
```

26. dgブローカから検証します（dgmgrlを使用）。

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> show configuration verbose
DGMGRL>
show database verbose chicago DGMGRL> show
database verbose boston DGMGRL> validate
database chicago DGMGRL> validate database
boston
```

27. クラスタウェアのロール・ベース・サービスのセットアップ – 『高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティス』を参照してください

28. レジストリを同期します（18.7 BM新機能）。

```
# odacli list-databases
```

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	DbHomeID
e6450a56-5a7d-4dab-9ca9-25b004b66646	chicago	Rac	19.8.0.0.200714	false	Otp	Odb1	Asm	Configured	755b4b5d-6211-4d94-81e8-cf611868fe39

```
# odacli describe-database -i e6450a56-5a7d-4dab-9ca9-25b004b66646
```

Database details

ID: e6450a56-5a7d-4dab-9ca9-25b004b66646
Description: chicago DB
Name: chicago
DB Version: 19.8.0.0.200714
DB Type: Rac DB
Edition: EE DB
Role:
DB Target Node Name:
DBID:
Instance Only Database: true
CDB: false
PDB Name:
PDB Admin User Name:
SEHA Enabled: false
Class: Otp
Shape: Odb1
Storage: Asm
DB Redundancy: MIRROR
CharacterSet: AL32UTF8
National CharacterSet: AL16UTF16
Language: AMERICAN
Territory: AMERICA
Home ID: 755b4b5d-6211-4d94-81e8-cf611868fe39
Console Enabled: false
TDE Enabled: false
Level 0 Backup Day: Sunday AutoBackup
Enabled: true
Created: July 5, 2020 1:06:43 PM CEST
DB Domain Name: us.oracle.com

```
# odacli update-registry -n db -f
```

odacli describe-job -i 24c3a6fb-0538-4571-ae38-e6cf03a6e689

Job details

ID:24c3a6fb-0538-4571-ae38-e6cf03a6e689

Description:Discover Components : db

Status:Success

Created:July 7, 2020 9:40:41 PM CEST

Message:

Task Name	Start Time	End Time	Status
Rediscover DBHome	July 7, 2020 9:40:45 PM CEST	July 7, 2020 9:40:47 PM CEST	Success
Rediscover DB: boston	July 7, 2020 9:41:08 PM CEST	July 7, 2020 9:41:24 PM CEST	Success

odacli describe-database -i e6450a56-5a7d-4dab-9ca9-25b004b66646

Database details

ID: e6450a56-5a7d-4dab-9ca9-25b004b66646

Description: chicago DB

Name: chicago

DB Version:18.7.0.0.190716

DB Type:Rac

DB Role:Standby

DB Target Node Name:Rac

DB Edition:EE

DBID:19.8.0.0.200714

Instance Only Database: false

CDB: false

PDB Name:

PDB Admin User Name:

SEHA Enabled: false

Class:Oltp

Shape:Odb1

Storage:Asm

DB Redundancy:MIRROR

CharacterSet:AL32UTF8

National CharacterSet:AL16UTF16

Language:AMERICAN

Territory:AMERICA

Home ID:755b4b5d-6211-4d94-81e8-cf611868fe39

Console Enabled: false

TDE Enabled: false

Level 0 Backup Day:

AutoBackup Enabled: true

Created:July 5, 2020 1:06:43 PM CEST

DB Domain Name: us.oracle.com

29. テストをスイッチオーバーします。

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL>
switchover to boston
DGMGRL> connect sys/welcome1@chicago
DGMGRL> switchover to chicago;
```

30. テストをフェイルオーバーします。

フェイルオーバーの前にスタンバイに接続します：

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> failover to boston
DGMGRL> reinstate database chicago
```

フェイルオーバーの前に以前のプライマリに接続します：

```
DGMGRL> connect sys/welcome1@chicago
DGMGRL> failover to chicago;
DGMGRL> reinstate database boston
```

付録B：OAKスタックを使用したOracle Database Applianceでの12.1.0.2の設定例

環境の例

次のセクションで、Oracle Database Applianceを使用してOracle Data Guardを設定する次の例で使用する、プライマリ・データベース環境とスタンバイ・データベース環境のトポロジについて説明します。

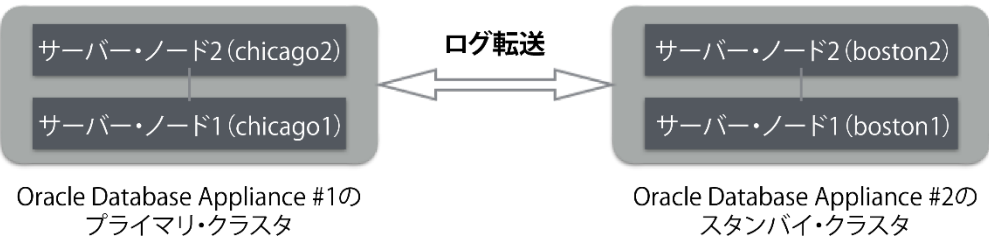


図1 Oracle Database ApplianceでのOracle RACの構成トポロジ

	プライマリOracle Database Appliance		スタンバイOracle Database Appliance	
アプライアンス名	アプライアンス #1		アプライアンス #2	
ホスト名	proddb1	proddb1	stbydb1	stbydb2
クラスタ名	PCLUSTER		SCLUSTER	
データベース名	chicago		chicago	
データベースの一意名	chicago		boston	
インスタンス名	chicago1	chicago2	boston1	boston2
スキャン名とIP	proddb-scan (10.1.27.2, 10.1.27.3)		stbydb-scan (10.1.27.4, 10.1.27.5)	
グリッド・インフラストラクチャ・ソフトウェアのインストール	/u01/app/18.0.0.0/grid		/u01/app/18.0.0.0/grid	
Oracle Databaseソフトウェアのインストール	/u01/app/oracle/product/12.1.0.2/db_home1		/u01/app/oracle/product/12.1.0.2/db_home1	
データベース記憶域	ACFS		ACFS	
ARCHIVELOGモード	対応		対応	
FORCE LOGGINGモード	対応		対応	

表2 - Oracle Databaseのプライマリ構成およびスタンバイ構成の例

プライマリ環境の構成

オラクルのベスト・プラクティスでは、Oracle Data Guard Brokerを使用してOracle Data Guardを構成することを強くお勧めします。

1. スタンバイREDOログを作成します。

スタンバイREDOログはプライマリ・データベースから受信したREDOデータをホスティングします。プライマリとスタンバイの設定前に、スタンバイREDOログをプライマリ・データベースでも作成し、スタンバイ・ロールへのスイッチオーバー後にすぐにREDOデータを受信できるようにしておくことをお勧めします。

プライマリ・データベースでスタンバイREDOログを作成します。スタンバイREDOログの各スレッドには、対応するオンラインREDOログのスレッドよりも少なくとも1つ多いREDOログ・グループが必要です。たとえば、次のようにします。

```
SQL> alter database add standby logfile thread 1 group 7 size 1G, group 8 size 1G, group 9 size 1G, group 10 size 1G;  
SQL> alter database add standby logfile thread 2 group 11 size 1G, group 12 size 1G, group 13 size 1G, group 14 size 1G;
```

オンラインREDOログの数とそのサイズを確認するには、次のクエリーを使用します。

```
SQL> select thread#, group#, bytes/1024/1024 SIZE_IN_GB, status from v$log;
```

スタンバイREDOログのサイズがREDOログのサイズと一致するようにしてください。Oracle Database Applianceプラットフォームのソリッド・ステート・ディスクに常駐するREDOディスク・グループ上に、スタンバイREDOログを作成する必要があります。ODA Small/Medium/LargeおよびX8-2 HAモデルの場合はREDOディスク・グループがないため、制御ファイル、オンライン・ログはRECOディスク・グループに格納されます。

スタンバイREDOログで各ログ・ファイルのサイズとログ・グループの数を検証するには、次のクエリーを使用します。

```
SQL> select group#, thread#, bytes/1024/1024 SIZE_IN_GB from v$standby_log;
```

2. プライマリ・データベースでアーカイブ・ログ・モードを有効化します。

アーカイブとは、アクティブなデータベースのREDOログが循環的に上書きされる前にアーカイブ・ファイルの形式でREDO情報を保存し、保護するプロセスのことです。Oracle Database Applianceで作成されたデータベースは、デフォルトでアーカイブがオンになっています。ただし、Oracle Database Applianceのデフォルト設定であるアーカイブ・ログ・モードでデータベースを動作させる必要は必ずしもありません。

プライマリ・データベースがARCHIVELOGモードで動作していることを確認します。

```
SQL> archive log list
```

プライマリ・データベースがARCHIVELOGモードで動作していない場合、次のようにARCHIVELOGモードを有効化します。Oracle Database Applianceで両方のインスタンスをシャットダウンします。

```
$ srvctl stop database -d chicago
```

1つのインスタンスを排他モードで起動マウントします。

```
SQL> startup mount exclusive;
```

アーカイブをオンにします。

```
SQL> alter database archivelog;
```

インスタンスをシャットダウンします。

```
SQL> shutdown immediate;
```

データベースを再起動します。

```
$ srvctl start database -d chicago
```

3. FORCE LOGGINGモードを有効化します。

強制ロギングを使用すると、NOLOGGING属性により実行されるデータベース操作を取得することができます。これにより、スタンバイ・データベースの整合性が保証されます。FORCE LOGGINGがプライマリ・データベースで有効化されているかどうか確認します。

```
SQL> select force_logging from v$database;
```

FORCE LOGGINGが有効化されていない場合、次のコマンドで有効化します。

```
SQL> alter database force logging;
```

4. フラッシュバック・データベース機能を構成します。

Oracleフラッシュバック・データベース機能は、不完全なデータベース・リカバリ実行に対する高速の代替手段となります。フラッシュバック・データベース機能の使用はオプションですが、フェイルオーバー後に古いプライマリ・データベースをすばやく元の状態に回復させるのに非常に有益な場合があります。そのため、スタンバイへのフェイルオーバーを行い、古いプライマリの修復が可能である場合、古いプライマリ・データベースをスタンバイ・データベースとして再構築する必要はなく、フラッシュバックし、Oracle Data Guardをそのポイントから再同期化させるだけで済みます。

プライマリ・データベースでフラッシュバック・データベースが有効化されているかどうか確認し、必要に応じて有効化します。

```
SQL> select flashback_on from v$database;
```

```
SQL> alter database flashback on;
```

フラッシュバック・データベースを有効化するには、ファスト・リカバリ領域（RECOディスク・グループ）で追加の使用スペースが必要になります。フラッシュバック・ログによって使用されるスペースを制御するには、パラメータDB_FLASHBACK_RETENTION_TARGETを望ましい値に設定します。この値は分単位で指定します。たとえば、次のようにします。

```
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120 scope=both sid='*';
```

5. スタンバイ・ファイル管理を有効化します。

プライマリ・データベースでデータ・ファイルの追加または削除が行われる場合、スタンバイ・データベースでもそれに対応するアクションが自動的に実行されなければなりません。この操作を有効化するには、自動スタンバイ・ファイル管理を使用します。

```
SQL> alter system set STANDBY_FILE_MANAGEMENT=AUTO scope=both sid='*';
```

6. スタンバイのデータベース・ホームが存在しなかった場合は作成します。

スタンバイのデータベース・ホームのバージョンはプライマリ・データベースのバージョンと同一である必要があります。

```
[oracle@stbydb1]# oakcli create dbhome -version 12.1.0.2.180417
```

7. TNSエントリおよびリスナーを設定します。

データベース間でのREDO転送を有効化するには、Oracleネット・サービス名を構成する必要があります。プライマリ・データベースとスタンバイ・データベースの両方のTNSエイリアスが含まれるようにtnsnames.oraファイルを更新します。Oracle Database Applianceではtnsnames.oraファイルがOracleデータベース・ホームのnetwork/adminディレクトリに配置されることに注意してください。

```
$ vi $ORACLE_HOME/network/admin/tnsnames.ora
```

Primary

```
chicago =
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = chicago)
  )
)
```



```

boston =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = boston)
    )
  )

```

Standby

```

chicago =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = chicago)
    )
  )

```

```

boston =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = boston)
    )
  )

```

8. REDO転送サービスを遅延モードで設定します。DG Brokerも構成する場合、この手順は不要です。

Oracle Data GuardのREDO転送メカニズムでは、Oracle Net接続を使用してデータベース間でREDOを送信します。REDO転送を有効化するには、LOG_ARCHIVE_DEST_nパラメータを設定します。たとえば、次の設定により、ログの送信が有効化され、LGWRベースの転送が非同期モードで使用されます。

```

SQL> alter system set log_archive_dest_2='SERVICE=boston LGWR ASYNC REGISTER VALID_FOR=(online_logfile,primary_role)
REOPEN=60 DB_UNIQUE_NAME=boston' scope=both sid='*';
SQL> alter system set log_archive_dest_state_2='defer' scope=both sid='*';

```

REDOログ転送オプションの詳細は、『Oracle Data Guard概要および管理』を参照してください。

9. フェッチ・アーカイブ・ログ・サーバーを設定します。DG Brokerも構成する場合、この手順は不要です。

データベースがスタンバイ・ロールで、プライマリが欠落したログ・ファイルを送信できない場合、スタンバイ・データベースでFAL_SERVER設定を使用して欠落したログ・ファイルを取得することができます。FAL_SERVERパラメータではOracle Netサービス名を使用します。

```

SQL> alter system set FAL_SERVER=boston scope=both sid='*';

```

10. プライマリ・データベースのspfileからpfileを作成します。

```

[oracle@proddb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@proddb1]$ export ORACLE_SID=chicago1
[oracle@proddb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@proddb1]$ sqlplus / as sysdba
SQL> create pfile='/tmp/chicago.pfile' from spfile;

```

11. プライマリ/スタンバイでパラメータを追加/変更します。以下に例を示します。

プライマリとスタンバイでのパラメータの変更	
プライマリ	スタンバイ
<pre> *.cluster_database=TRUE chicago2.instance_number=2 chicago1.instance_number=1 chicago2.thread=2 chicago1.thread=1 chicago2.undo_tablespace='UNDOTBS2' chicago1.undo_tablespace='UNDOTBS1' *.db_block_checking=FULL *.db_block_checksum=FULL *.db_lost_write_protect=TYPICAL *.db_unique_name=Chicago *.remote_listener='proddb-scan:1521' *.DB_FILE_NAME_CONVERT='boston','chicago','BOSTON','CHICAGO' *.LOG_FILE_NAME_CONVERT='boston','chicago','BOSTON','CHICAGO' *.LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=chicago' 次のパラメータを追加 *.audit_file_dest='/u01/app/oracle/admin/chicago/adump' *.fal_server='boston' *.remote_login_passwordfile='exclusive' HDDを構成したX5-2、X7-2 HA、X8-2 HAでのみ追加 *._cluster_flash_cache_slave_file="" *.db_flash_cache_file='/u02/app/oracle/oradata/flashdata/.ACFS/snaps /flashcache/chicago/flash1' </pre>	<pre> *.cluster_database=TRUE boston2.instance_number=2 boston1.instance_number=1 boston2.thread=2 boston1.thread=1 boston2.undo_tablespace='UNDOTBS2' boston1.undo_tablespace='UNDOTBS1' *.db_block_checking=FULL *.db_block_checksum=FULL *.db_lost_write_protect=TYPICAL *.db_unique_name=boston *.remote_listener='stbydb-scan:1521' *.DB_FILE_NAME_CONVERT='chicago','boston','CHICAGO','BOSTON' *.LOG_FILE_NAME_CONVERT='chicago','boston','CHICAGO','BOSTON' *.LOG_ARCHIVE_DEST_1='LOCATION=USE_DB_RECOVERY_FILE_DEST VALID_FOR=(ALL_LOGFILES,ALL_ROLES) DB_UNIQUE_NAME=boston' 次のパラメータを追加 *.audit_file_dest='/u01/app/oracle/admin/boston/adump' *.fal_server='chicago' *.remote_login_passwordfile='exclusive' HDDを構成したX5-2、X7-2 HA、X8-2 HAでのみ追加 *._cluster_flash_cache_slave_file="" *.db_flash_cache_file='/u02/app/oracle/oradata/flashdata/.ACFS/s naps/flashcache/boston/flash1' </pre>

12. すべてのスタンバイ・ホストで、bostonデータベース用の監査ディレクトリを作成します。

```

[oracle@stbydb1] mkdir -p /u01/app/oracle/admin/boston/adump
[oracle@stbydb2] mkdir -p /u01/app/oracle/admin/boston/adump

```

13. スタンバイでファイル・システム構造を作成します。この手順は古いスタック（OAK）にのみ適用されます。

```
[root@stbydb1]# oakcli create dbstorage -db boston -storage acfs
INFO:2017-08-12 06:16:44:Please check the logfile
/opt/oracle/oak/log/stbydb1/tools/12.1.2.10.0/createdbstorage_boston_69182.log' for more details
...
SUCCESS:All nodes in /opt/oracle/oak/onecmd/tmp/db_nodes are pingable and alive.
INFO:2017-08-14 04:47:45:Successfully setup the storage structure for the database 'boston'
INFO:2017-08-14 04:47:45:Set the following directory structure for the Database boston
INFO:2017-08-14 04:47:45:DATA: /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston
INFO:2017-08-14 04:47:45:REDO: /u01/app/oracle/oradata/datastore/boston
INFO:2017-08-14 04:47:45:RECO: /u01/app/oracle/fast_recovery_area/datastore/boston
SUCCESS:2017-08-14 04:47:45:Successfully setup the Storage for the Database : boston
```

14. パスワードをコピーします。

パスワード・ファイルをプライマリ・データベースから最初のスタンバイ・ホストにコピーします。

```
[oracle@proddb1]$ $ORACLE_HOME/bin/srvctl config database -d chicago |grep -i Password
Password file: /u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/orapwchicago
[oracle@proddb1]$ scp /u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/orapwchicago
oracle@stbydb1.us.oracle.com:/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston
```

15. 変更されたpfileを最初のスタンバイ・ホストにコピーし、スタンバイ・データベースをマウントします。

スタンバイ制御ファイルが作成されたパスをメモします。

```
[oracle@proddb1]$ scp /tmp/chicago.pfile oracle@stbydb1.us.oracle.com:/tmp/boston.pfile
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ cp /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston
/u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/orapwboston1
[oracle@stbydb1]$ rman target /
RMAN> startup nomount pfile='/tmp/boston.pfile';
RMAN> restore standby controlfile from service chicago;
Starting restore at 12-AUG-17
using target database control file instead of recovery catalog
allocated channel:ORA_DISK_1
channel ORA_DISK_1:SID=162 instance=boston1 device type=DISK
channel ORA_DISK_1: starting datafile backup set restore
channel ORA_DISK_1: using network backup set from service Chicago
channel ORA_DISK_1: restoring control file
channel ORA_DISK_1: restore complete, elapsed time:00:00:26
output file name=/u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile/o1_mf_drw8zb81_.ctl
Finished restore at 12-AUG-17
```

16. 制御ファイル・パラメータを更新します。

pfile '/tmp/chicago.pfile'を編集し、前の出力からの新しいパスを示すようにcontrol_filesパラメータを置換します。たとえば、
control_files= '/u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile/o1_mf_drw8zb81_.ctl'

```
[oracle@stbydb1]$ vi /tmp/boston.pfile
```

17. スタンバイ・インスタンスを起動します。

変更後のpfileを使用してスタンバイ・インスタンスをノーマウント・モードで起動し、spfileを作成し、spfileを使用してインスタンスを再起動します。

```
[oracle@stbydb1$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@stbydb1$ export ORACLE_SID=boston1
[oracle@stbydb1$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1$ mkdir -p /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> startup nomount force pfile='/tmp/boston.pfile';
SQL> create spfile='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora' from
pfile='/tmp/boston.pfile';
SQL> !echo "spfile='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora'" >
/u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/initboston1.ora
SQL> startup mount force;
```

18. 並列処理を有効化してSECTION SIZE=64MBに設定します。

リストア中に並列処理を利用するには、以下を実行してサーバーのCPUの数を確認します。

```
[oracle@stbydb1]$ grep -c ^processor /proc/cpuinfo
20
```

スタンバイで以下のRMANの構成変更を行います。

この例では、リカバリ・プロセス中に使用するためにRMAN用に構成済みの8つのチャンネルを使用します。

```
[oracle@stbydb1]$ rman target /
RMAN> CONFIGURE DEFAULT DEVICE TYPE TO DISK; RMAN>
CONFIGURE DEVICE TYPE DISK PARALLELISM 8;
```

19. プライマリ・データベース・サービスからスタンバイ・データベースをリストアします。

1つの大きなファイルを並行してバックアップするには、RMANのマルチセクション・バックアップ/リストア機能によりバックアップとリカバリの速度が向上します。RMANは内部で作業を複数のチャンネル間で分割し、各チャンネルがファイル内の1つのファイル・セクションに作用します。小さなセクション・サイズを指定してセクションの数が256を超えることになる場合、RMANにより、ちょうど256個のセクションとなる値にセクション・サイズが増加されます。

セクション・サイズ句はさまざまな要素、たとえば、ネットワーク帯域幅、チャンネル数、データ・ファイルのサイズ、アプリケーション・データベースのサイズなどによって決まります。

```
[oracle@stbydb1]$ sqlplus system/welcome1@chicago
SQL> select TABLESPACE_NAME, bytes/1024/1024 SIZE_IN_GB from dba_data_files;
```

TABLESPACE_NAME	SIZE_IN_GB
SYSTEM	.68359375
SYSAUX	.5859375
UNDOTBS1	.297851563
UNDOTBS2	.1953125
USERS	.004882813

たとえば、スタンバイ・ホストで実行される次のコマンドでは、64 MBのバックアップ・セクション・サイズを指定します。

```
[oracle@stbydb1]$ rman target /
RMAN> restore database from service chicago section size 64M;
RMAN> switch database to copy;
RMAN> recover database from service chicago;
RMAN> backup spfile;
```

20. プライマリでログの送信を有効にします。この手順が必要なのは、Data Guard Brokerを構成しない場合のみです。

```
[oracle@proddb1]$ sqlplus / as sysdba
SQL> alter system set log_archive_dest_state_2='enable' scope=both;
```

21. スタンバイでフラッシュバック・データベースを有効化し、必要に応じて保持期間を調節します。この手順が必要なのは、Data Guard Brokerを構成しない場合のみです。

```
SQL> alter database recover managed standby database cancel;
SQL> alter database flashback on;
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120;
SQL> alter database recover managed standby database disconnect;
```

22. スタンバイ側で管理リカバリを開始します。この手順が必要なのは、Data Guard Brokerを構成しない場合のみです。

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT FROM SESSION;
```

23. Clusterwareを使用してスタンバイ・データベースを登録します。

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
```

シングル・インスタンスの例

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/12.1.0.2/dbhome_1 -dbtype SINGLE
-instance boston1 -node stbydb1 -dbname chicago -acfspath
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore' -role
physical_standby -spfile '/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora' -pwfile
'/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston'
```

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/12.1.0.2/dbhome_1 -dbtype RAC -
dbname chicago -acfspace
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore' -role
physical_standby -spfile '/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora' -pwfile
'/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/orapwboston'
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston1 -node stbydb1
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston2 -node stbydb2
[oracle@stbydb1]$ scp $ORACLE_HOME/dbs/initboston1.ora
oracle@stbydb2:/u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/initboston2.ora
[oracle@stbydb1]$ srvctl start instance -database boston -instance boston1 -o mount
[oracle@stbydb1]$ srvctl start instance -database boston -instance boston2 -o mount
```

24. パラメータを設定し、Oracle Data Guard Broker構成を作成します。

環境に合わせて以下のスクリプトを変更し、PostCR.sqlとして保存します。

注：フェイルオーバーのロール・トランジション後に障害となったプライマリを再インスタンス化するにはフラッシュバック・データベースが必要です。オプションでプライマリとスタンバイの両方でフラッシュバックを有効化します。スタンバイ・データベースは、PostCRスクリプトを次のように使用して、フラッシュバックをオンにして開始することができます。

```
[oracle@stbydb1]$ cat PostCR.sql
connect / as sysdba
alter system set dg_broker_config_file1='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/dr1.dat'
scope=both;
alter system set dg_broker_config_file2='/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/dr2.dat'
scope=both;
alter system set db_flashback_retention_target=120 scope=spfile; alter
database flashback on;
alter system set dg_broker_start=true scope=spfile;
shutdown immediate
startup mount
alter system register;
connect sys/welcome1@chicago as sysdba alter
system set
dg_broker_config_file1='/u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/dr1.dat'
scope=both;
alter system set
dg_broker_config_file2='/u02/app/oracle/oradata/datastore/.ACFS/snaps/chicago/chicago/dr2.dat' scope=both;
alter system set dg_broker_start=TRUE;
host sleep 30
host dgmgrl sys/welcome1@chicago "CREATE CONFIGURATION dgconfig AS PRIMARY
DATABASE IS CHICAGO CONNECT IDENTIFIER IS CHICAGO";
host sleep 30
host dgmgrl sys/welcome1@chicago "ADD DATABASE BOSTON AS CONNECT IDENTIFIER IS BOSTON" ;
host dgmgrl sys/welcome1@chicago "ENABLE CONFIGURATION"
exit
```

スタンバイ・データベースでスクリプトPostCR.sqlを実行します。環境をスタンバイ・データベースに設定します。

```
[oracle@stbydb1]$ export ORACLE_HOME=/u01/app/oracle/product/12.1.0.2/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> @PostCR.sql
```


ORA-38788で'ALTER DATABASE FLASHBACK ON'が失敗した場合、スタンバイを同期させ、次の手順を実行してからフラッシュバックを有効化してください。

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE CANCEL;
SQL> alter database flashback on;
SQL> ALTER DATABASE RECOVER MANAGED STANDBY DATABASE DISCONNECT;
```

25. sqlplus/srvctlを使用して検証します。

```
[oracle@stbydb1]$ srvctl config database -d chicago
[oracle@stbydb1]$ srvctl config database -d boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> select FORCE_LOGGING, FLASHBACK_ON, OPEN_MODE, DATABASE_ROLE, SWITCHOVER_STATUS,
DATAGUARD_BROKER, PROTECTION_MODE from v$database;
SQL> select PROCESS,PID,DELAY_MINS from V$MANAGED_STANDBY;
```

26. dgブローカから検証します（dgmgriを使用）。

```
$ dgmgri
DGMGRL> connect sys/welcome1@boston
DGMGRL> show configuration verbose
DGMGRL> show database verbose chicago
DGMGRL> show database verbose boston
DGMGRL> validate database chicago
DGMGRL> validate database boston
```

検証コマンドの出力の"Current Log File Groups Configuration"セクションに"Warning: standby redo logs not configured for thread 0"と表示されることがあります。

MOS Note [20582405.8](#) - 『Bug 20582405 - dgmgri "validate database" shows warning "standby redo logs not configured for thread 0"』を参照してください。

27. クラスウェアのロール・ベース・サービスのセットアップ – [高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティスを参照してください](#)

28. スイッチオーバーをテストします。

```
$ dgmgri
DGMGRL> connect sys/welcome1@boston
DGMGRL> switchover to boston
DGMGRL> connect sys/welcome1@chicago
DGMGRL> switchover to chicago;
```

29. テストをフェイルオーバーします。

フェイルオーバーの前にスタンバイに接続します：

```
$ dgmgri
DGMGRL> connect sys/welcome1@boston
DGMGRL> failover to boston
DGMGRL> reinstate database chicago
```

フェイルオーバーの前に以前のプライマリに接続します：

```
DGMGRL> connect sys/welcome1@chicago
DGMGRL> failover to chicago;
DGMGRL> reinstate database boston
```

付録C：Oracle Database Applianceでの11gR2の設定例

環境の例

次のセクションで、Oracle Database Applianceを使用してOracle Data Guardを設定する次の例で使用する、プライマリ・データベース環境とスタンバイ・データベース環境のトポロジについて説明します。

注：Oracle Database Appliance 19.8 BMでは、`odacli`コマンドを使用してOracle Data Guardを構成および管理してください。前提条件と手順については、「[Configuring Oracle Data Guard on Oracle Database Appliance](#)」を参照してください。

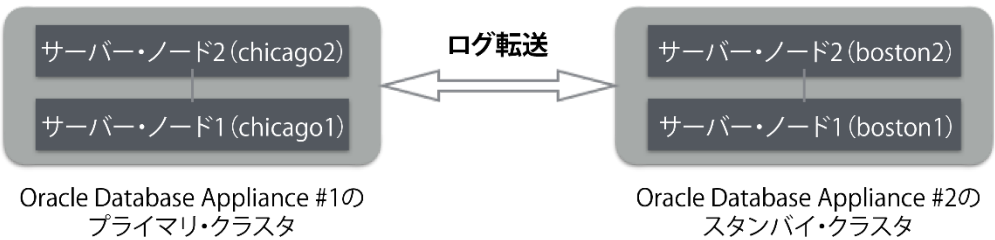


図2 Oracle Database ApplianceでのOracle RACの構成トポロジ

	プライマリOracle Database Appliance		スタンバイOracle Database Appliance	
アプライアンス名	アプライアンス #1		アプライアンス #2	
ホスト名	proddb1	proddb2	stbydb1	stbydb2
クラスタ名	PCLUSTER		SCLUSTER	
データベース名	chicago		chicago	
データベースの一意名	chicago		boston	
インスタンス名	chicago1	chicago2	boston1	boston2
スキャン名とIP	proddb-scan (10.1.27.2, 10.1.27.3)		stbydb-scan (10.1.27.4, 10.1.27.5)	
グリッド・インフラストラクチャ・ソフトウェアのインストール	/u01/app/19.0.0.0/grid		/u01/app/19.0.0.0/grid	
Oracle Databaseソフトウェアのインストール	/u01/app/oracle/product/11.2.0.4/db_home1		/u01/app/oracle/product/11.2.0.4/db_home1	
ARCHIVELOGモード	対応		対応	
FORCE LOGGINGモード	対応		対応	

表3 - Oracle Databaseのネーミング規則の例

プライマリ環境の構成

オラクルのベスト・プラクティスでは、Oracle Data Guard Brokerを使用してOracle Data Guardを構成することを強くお勧めします。

1. スタンバイREDOログを作成します。

スタンバイREDOログはプライマリ・データベースから受信したREDOデータをホスティングします。プライマリとスタンバイの設定前に、スタンバイREDOログをプライマリ・データベースでも作成し、スタンバイ・ロールへのスイッチオーバー後にすぐにREDOデータを受信できるようにしておくことをお勧めします。

プライマリ・データベースでスタンバイREDOログを作成します。スタンバイREDOログの各スレッドには、対応するオンラインREDOログのスレッドよりも少なくとも1つ多いREDOログ・グループが必要です。たとえば、次のようにします。

```
SQL> alter database add standby logfile thread 1 group 7 size 1G, group 8 size 1G, group 9 size 1G,  
group 10 size 1G;  
SQL> alter database add standby logfile thread 2 group 11 size 1G, group 12 size 1G, group 13 size 1G,  
group 14 size 1G;
```

オンラインREDOログの数とそのサイズを確認するには、次のクエリーを使用します。

```
SQL> select thread#, group#, bytes/1024/1024 SIZE_IN_GB, status from v$log;
```

スタンバイREDOログのサイズがREDOログのサイズと一致するようにしてください。ソリッド・ステート・ディスクに常駐するREDOディスク・グループ上に、スタンバイREDOログを作成する必要があります。Oracle Database Appliance Small/Medium/LargeおよびX8-2 HAモデルの場合はREDOディスク・グループがないため、制御ファイル、オンライン・ログはRECOディスク・グループに格納されます。

スタンバイREDOログで各ログ・ファイルのサイズとログ・グループの数を検証するには、次のクエリーを使用します。

```
SQL> select group#, thread#, bytes/1024/1024 SIZE_IN_GB from v$standby_log;
```

2. プライマリ・データベースでアーカイブ・ログ・モードを有効化します。

アーカイブとは、アクティブなデータベースのREDOログが循環的に上書きされる前にアーカイブ・ファイルの形式でREDO情報を保存し、保護するプロセスのことです。Oracle Database Applianceで作成されたデータベースは、デフォルトでアーカイブがオンになっています。ただし、データベースは必ずしもアーカイブ・ログ・モードで動作させる必要はありません。

プライマリ・データベースがARCHIVELOGモードで動作していることを確認します。

```
SQL> archive log list
```

プライマリ・データベースがARCHIVELOGモードで動作していない場合、次のようにARCHIVELOGモードを有効化します。

Oracle Database Applianceで両方のインスタンスをシャットダウンします。

```
$ srvctl stop database -d chicago
```

1つのインスタンスを排他モードで起動マウントします。

```
SQL> startup mount exclusive;
```

アーカイブをオンにします。

```
SQL> alter database archivelog;
```

インスタンスをシャットダウンします。

```
SQL> shutdown immediate;
```

データベースを再起動します。

```
$ srvctl start database -d chicago
```

3. FORCE LOGGINGモードを有効化します。

強制ログングを使用すると、NOLOGGING属性により実行されるデータベース操作を取得することができます。これにより、スタンバイ・データベースの整合性が保証されます。FORCE LOGGINGがプライマリ・データベースで有効化されているかどうか確認します。

```
SQL> select force_logging from v$database;
```

FORCE LOGGINGが有効化されていない場合、次のコマンドで有効化します。

```
SQL> alter database force logging;
```

4. フラッシュバック・データベース機能を構成します。

Oracleフラッシュバック・データベース機能は、不完全なデータベース・リカバリ実行に対する高速の代替手段となります。フラッシュバック・データベース機能の使用はオプションですが、フェイルオーバー後に古いプライマリ・データベースをすばやく元の状態に回復させるのに非常に有益な場合があります。そのため、スタンバイへのフェイルオーバーを行い、古いプライマリの修復が可能である場合、古いプライマリ・データベースをスタンバイ・データベースとして再構築する必要はなく、フラッシュバックし、Oracle Data Guardをそのポイントから再同期化させるだけで済みます。プライマリ・データベースでフラッシュバック・データベースが有効化されているかどうか確認し、必要に応じて有効化します。

```
SQL> select flashback_on from v$database;
```

```
SQL> alter database flashback on;
```

フラッシュバック・データベースを有効化するには、ファスト・リカバリ領域（RECOディスク・グループ）で追加の使用スペースが必要になります。フラッシュバック・ログによって使用されるスペースを制御するには、パラメータDB_FLASHBACK_RETENTION_TARGETを望ましい値に設定します。

この値は分単位で指定します。たとえば、次のようにします。

```
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120 scope=both sid='*';
```

5. スタンバイ・ファイル管理を有効化します。

プライマリ・データベースでデータ・ファイルの追加または削除が行われる場合、スタンバイ・データベースでもそれに対応するアクションが自動的に実行されなければなりません。

この操作を有効化するには、自動スタンバイ・ファイル管理を使用します。

```
SQL> alter system set STANDBY_FILE_MANAGEMENT=AUTO scope=both sid='*';
```

6. リモート特権ログインを有効化します。

プライマリ・データベースの各インスタンスがリモート・ログイン・パスワード・ファイルを使用して構成されていることを確認します。Oracle Database Applianceではデータベースがこの設定に基づいて展開されることに注意してください。初期化パラメータREMOTE_LOGIN_PASSWORDFILEをEXCLUSIVEに設定する必要があります。このパラメータが環境内でリセットされていて、次のように変更する必要がある場合、変更を反映させるにはデータベースを再起動する必要があります。

```
[oracle@proddb1]$ sqlplus / as sysdba
```

```
SQL> show parameter remote_login_passwordfile
```

```
SQL> alter system set remote_login_passwordfile='exclusive' scope=spfile sid='*';
```

7. TNSエントリを設定します。

データベース間でのREDO転送を有効化するには、Oracleネット・サービス名を構成する必要があります。プライマリ・データベースとスタンバイ・データベースの両方のTNSエイリアスが含まれるようにtnsnames.oraファイルを更新します。Oracle Database Applianceではtnsnames.oraファイルがOracleデータベース・ホームのnetwork/adminディレクトリに配置されることに注意してください。

```
chicago =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = chicago)
```

```
)
)

boston =
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = boston)
  )
)
```

8. REDO転送サービスを遅延モードで設定します。

Oracle Data GuardのREDO転送メカニズムでは、Oracle Net接続を使用してデータベース間でREDOを送信します。REDO転送を有効化するには、LOG_ARCHIVE_DEST_nパラメータを設定します。たとえば、次の設定により、ログの送信が有効化され、LGWRベースの転送が非同期モードで使用されます。

```
SQL> alter system set log_archive_dest_2='SERVICE=boston LGWR ASYNC REGISTER VALID_FOR=(online_logfile,primary_role)
REOPEN=60 DB_UNIQUE_NAME=boston' scope=both sid='*';
SQL> alter system set log_archive_dest_state_2='defer' scope=both sid='*';
```

REDOログ転送オプションの詳細は、『Oracle Data Guard概要および管理』を参照してください。

9. フェッチ・アーカイブ・ログ・サーバーを設定します。

データベースがスタンバイ・ロールで、プライマリが欠落したログ・ファイルを送信できない場合、スタンバイ・データベースでFAL_SERVER設定を使用して欠落したログ・ファイルを取得することができます。FAL_SERVERパラメータではOracle Netサービス名を使用します。

```
SQL> alter system set FAL_SERVER=boston scope=both sid='*';
```

スタンバイ環境の構成

このセクションでは、スタンバイ・データベースで実行する必要がある手順について説明します。スタンバイ環境でOracle Database Applianceシステムが設定されていることが前提となります。ペア・メタル構成または仮想化プラットフォーム構成でのOracle Database Applianceの設定方法については、Oracle Database Applianceの[Setup Poster](#)を参照してください。

10. データベース・ホームを作成します。

スタンバイのデータベース・ホームのバージョンはプライマリ・データベースのバージョンと同一である必要があります。

```
古いスタック (OAK)
[oracle@stbydb1]# oakcli create dbhome -version 11.2.0.4.200714
```

```
新スタック (DCS)
[oracle@stbydb1]# odacli create-dbhome -v 11.2.0.4.200714
```

11. TNSエントリを設定します。

データベース間でのREDO転送を有効化するには、Oracleネット・サービス名を構成する必要があります。プライマリ・データベースとスタンバイ・データベースの両方のTNSエイリアスが含まれるようにtnsnames.oraファイルを更新します。Oracle Database Applianceではtnsnames.oraファイルがOracleデータベース・ホームのnetwork/adminディレクトリに配置されることに注意してください。

```
chicago =
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = proddb-scan)(PORT = 1521))
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = chicago)
  )
)
```

)

```
boston =  
(DESCRIPTION =  
  (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb-scan)(PORT = 1521))  
  (CONNECT_DATA =  
    (SERVER = DEDICATED)  
    (SERVICE_NAME = boston)  
  )  
)
```

12. 静的リスナー構成を作成します。

インスタンス化の実行中に、Recovery Manager (RMAN) 接続用にグリッド・ユーザーとしてスタンバイ・データベースで静的リスナー・サービスを作成します。リスナーのホームがGrid Infrastructureのホームにあることに注意してください (/u01/app/18.0.0.0/grid/network/admin)。

```
SID_LIST_LISTENER =  
  (SID_LIST = (SID_DESC = (GLOBAL_DBNAME = boston)  
    (ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)(SID_NAME = boston)))
```

13. リスナーを再起動します。

リスナーの変更後は再起動する必要があります。

```
[grid@stbydb1]$ lsnrctl reload listener
```

14. 初期スタンバイ・パラメータ・ファイルを作成します。

スタンバイ・ホストのORACLE_HOME/dbsディレクトリで次のパラメータを使用してpfile (initboston1.ora) を作成します。sga_targetにはプライマリ・データベースと同じ値を設定することをお勧めします。以下に例を示します。

```
[oracle@stbydb1]$ vi /u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/initboston.ora  
db_name=chicago  
db_unique_name=boston  
sga_target=5G
```

15. パスワード・ファイルをプライマリ・データベースからコピーします。

RMAN複製プロセス中に、パスワード・ファイルの作成が必要なりモート認証を使用して、補助インスタンスにアクセスする必要があります。

```
[oracle@stbydb1]$ scp oracle@proddb1:/u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/orapwchicago1  
/u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/orapwboston
```

16. 監査ディレクトリを作成します。

両方のノードのスタンバイ側に監査ファイルの送信先ディレクトリを作成します。

```
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/admin/boston/adump  
[oracle@stbydb2]$ mkdir -p /u01/app/oracle/admin/boston/adump
```

17. 必要なACFSストレージ・ディレクトリをスタンバイのOracle Database Applianceシステムに作成します。

この手順は古いスタック (OAK) にも適用されます。

“oakcli create dbstorage”を‘root’ユーザー・コマンドとして使用し、スタンバイ用のACFSデータベース・ストレージを作成します。

```
[root@stbydb1]# oakcli create dbstorage -db boston
```

18. スタンバイでストレージ構造を作成し、スタンバイ・データベースをDCSに登録します。この手順は新しいスタック（DCS）にのみ適用されます。

スタンバイ用に作成したホームIDを調べます。

```
[root@stbydb1]# odacli list-dbhomes
```

ID	Name	DB Version	Home Location	Status

2d7d2b0b-8489-4e28-a151-64f6e156a512		OraDB11204_home1	11.2.0.4.200714	
/u01/app/oracle/product/11.2.0.4/dbhome_1		Configured		

インスタンス専用データベースを作成します。

```
[root@stbydb1]# odacli create-database -m -u boston -n chicago -r ACFS -io -dh 2d7d2b0b-8489-4e28-a151-64f6e156a512
```

インスタンス専用データベースの準備ができるまで待機し、準備ができたら先に進みます。ステータスはodacli describe-jobsで確認します。

```
[root@stbydb1]# odacli describe-job -i b8d02cbb-7f0d-48aa-bba5-9d11491560ea
```

Job details

```
-----
ID: b8d02cbb-7f0d-48aa-bba5-9d11491560ea
Description: Database service creation with db name: boston
Status: Success
Created: July 7, 2020 1:18:01 PM CEST
Message:
```

両方のノードでインスタンスがノーマウント・モードで動作している可能性があります。その場合はインスタンスを停止または強制終了します。

```
[root@stbydb1]# ps -elf|grep chic
```

```
0 S oracle 53721      1 0 80  0 - 1160761 SYSC_s 03:19 ?        00:00:00 ora_pmon_chicago
...
0 S oracle 54025      1 0 80  0 - 1160758 SYSC_s 03:19 ?        00:00:00 ora_gcr0_chicago
0 S root   59128 9659 0 80 0 - 25847 pipe_w 03:20 pts/2 00:00:00 grep chic
```

```
[root@scaoda7121 ~]# kill -9 53721
```

19. スタンバイ・インスタンスを起動します。

NOMOUNT状態の最初のスタンバイ・ホストでスタンバイ・データベース・インスタンスを起動し、インスタンス化の準備をします。

```
[oracle@stbydb1]$ export ORACLE_SID=boston
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> startup nomount
```

20. ネットワーク接続性を検証します。

この時点で、Oracle Netはスタンバイ環境からプライマリ環境とスタンバイ環境の両方のTNSエイリアスを解決できなければなりません。

```
[oracle@stbydb1]$ tnsping chicago
[oracle@stbydb1]$ tnsping boston
[oracle@stbydb1]$ sqlplus sys/<password>@//proddb1:1521/chicago as sysdba
```


このセクションでは、プライマリ環境とスタンバイ環境での設定の完了後に行われるスタンバイ・データベースのインスタンス化の概要を示します。

21. データベースを複製します。

Oracle Recovery Manager (Oracle RMAN) を使用して、DUPLICATE DATABASE コマンドによりスタンバイ・データベースを作成することができます。複製プロセスの一部として、パラメータ・ファイル、パスワード・ファイル、制御ファイル、データベースファイルがプライマリ環境からスタンバイ環境にコピーされます。

スタンバイの動作のためにパラメータ設定に対して行う必要のある適切な変更も、RMANのDUPLICATE DATABASE コマンドで指定する必要があります。RMANでプライマリ・パラメータ・ファイルがコピーされると、それに応じてDUPLICATE DATABASE コマンドで指定されたパラメータが変更されます。

パスワード・ファイルもコピーされると、スタンバイ・データベースのパスワードがプライマリ・データベースのパスワードと同じになります。

```
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile[oracle@stbydb1]$ mkdir -p
/u01/app/oracle/oradata/datastore/boston/BOSTON/onlineolog
[oracle@stbydb1]$ mkdir -p /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/BOSTON/datafile
[oracle@stbydb1]$ mkdir -p /u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/fast_recovery_area/datastore/boston/BOSTON/archivelog
[oracle@stbydb1]$ mkdir -p /u01/app/oracle/fast_recovery_area/datastore/boston/BOSTON/flashback

[oracle@stbydb1]$ export NLS_DATE_FORMAT="HH24:MI:SS"
[oracle@stbydb1]$ rman
connect target sys/welcome1@//proddb1:1521/chicago
connect auxiliary sys/welcome1@//stbydb1:1521/boston
run {
allocate channel tgt1 device type disk;
allocate auxiliary channel aux1 device type disk;
allocate channel tgt2 device type disk;
allocate auxiliary channel aux2 device type disk;
allocate channel tgt3 device type disk;
allocate auxiliary channel aux3 device type disk;
DUPLICATE TARGET DATABASE FOR STANDBY FROM ACTIVE DATABASE
SPFILE PARAMETER_VALUE_CONVERT='chicago','boston','CHICAGO','BOSTON'
SET DB_NAME="chicago"
SET DB_UNIQUE_NAME="boston"
SET CLUSTER_DATABASE='false'
SET REMOTE_LISTENER="stbydb-scan:1521"
SET LOCAL_LISTENER=""
SET STANDBY_FILE_MANAGEMENT='AUTO'
set audit_file_dest = '/u01/app/oracle/admin/boston/adump'
set db_create_file_dest = '/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston'
set diagnostic_dest="/u01/app/oracle"
set db_recovery_file_dest="/u01/app/oracle/fast_recovery_area/datastore"
set db_create_online_log_dest_1="/u01/app/oracle/oradata/datastore/boston"
set log_archive_dest_2 = 'service=chicago lgwr async
register )alid_for=(online_logfiles, primary_role) db_unique_name=chicago'
set fal_server='chicago'
NOFILENAMECHECK;
}
```

注：Oracle Database Appliance 19.6以上のベア・メタル・プラットフォームでは、*local_listener*と*remote_listener*の代わりに*listener_networks*パラメータがデータベースに使用されます。

22. db_file_name_convertパラメータとlog_file_name_convertパラメータを変更し、インスタンスを再起動します。

```
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> alter system set db_file_name_convert='chicago','boston','CHICAGO','BOSTON' scope=spfile;
SQL> alter system set log_file_name_convert='chicago','boston','CHICAGO','BOSTON' scope=spfile;
SQL> shutdown immediate
SQL> startup mount
```

23. 管理リカバリ・モードを開始します。

次のように、リアルタイム・モードでスタンバイ・データベースの管理リカバリを開始します。

```
SQL> alter database recover managed standby database disconnect;
```

リアルタイム適用では、REDO Applyサービスは現在のスタンバイREDOログがアーカイブされるのを待たずに、受信後すぐにスタンバイ・データベースにREDOログを適用することができます。その場合、フェイルオーバーまたはスイッチオーバーの発生時にはすでにスタンバイREDOログ・ファイルが適用されているため、フェイルオーバーとスイッチオーバーが高速化されます。

24. プライマリでREDO転送を有効化します。

```
SQL> alter system set log_archive_dest_state_2='enable' scope=both;
```

25. フラッシュバック・データベースを有効化します。

スタンバイ・データベースでフラッシュバック・データベースを有効化し、必要に応じて保存の調節を行います。たとえば、次のようにします。

```
SQL> alter database recover managed standby database cancel;
SQL> alter database flashback on;
SQL> alter system set DB_FLASHBACK_RETENTION_TARGET=120;
SQL> alter database recover managed standby database disconnect;
```

インスタンス化後の手順

スタンバイのインスタンス化の完了後に次の手順を実行します。

26. Oracle Clusterwareを使用してスタンバイ・データベースを登録します。

ORACLE_HOME環境変数が正しく設定されていることを確認してください。Oracle Clusterwareを使用して、スタンバイ・データベースをシングル・インスタンスとして登録し、クラスタの1つのノードから実行できるようにします。

```
[oracle@stbydb1]$ mv /u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/spfileboston.ora
/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora
[oracle@stbydb1]$ echo "spfile=/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora" >
/u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/initboston1.ora
```

Oracle Active Data Guard以外の構成の場合、次のようにします。

```
[oracle@stbydb1]$ srvctl add database -d boston -o /u01/app/oracle/product/11.2.0.4/dbhome_1 -p
/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora -r physical_standby
-s mount -c SINGLE -x stbydb1
-j '/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore'
```

Oracle Active Data Guard構成の場合、次のようにします。

```
[oracle@stbydb1]$ srvctl add database -d boston -o /u01/app/oracle/product/11.2.0.4/dbhome_1 -p
/u02/app/oracle/oradata/datastore/.ACFS/snaps/boston/boston/spfileboston.ora -r physical_standby
-s 'read only' -c SINGLE -x stbydb1
-j '/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/datastore'
```

データベース・リソースを起動します。

```
[oracle@stbydb1]$ sqlplus "/ as sysdba"
SQL> shutdown immediate
[oracle@stbydb1]$ srvctl start database -d boston
```

27. スタンバイ・データベースをOracle RACに変換します。

この手順はオプションです。この時点で、スタンバイ・データベースはシングル・インスタンス・データベースとして構成されています。プライマリ・データベースがOracle RACデータベースであった場合、スタンバイ・データベースもOracle RACスタンバイに変換することができます。[付録D](#)に、*rconfig*ツールを使用してシングル・インスタンス・データベースをOracle RACスタンバイ・データベースに変換する方法を示します。

28. 専用DRネットワークを設定します。

この手順はオプションです。REDO転送サービスは専用ネットワークを使用するように構成することができます。専用ネットワーク・チャネルは、特にパブリック・ネットワークで利用可能な帯域幅の大部分がアプリケーション・ネットワークのトラフィックによって使用される場合に、REDO送信のパフォーマンスを向上させるのに役立ちます。

29. 構成と設定を確認します。

スタンバイ・データベースで、内部データ・ディクショナリ・ビューを使用してスタンバイ・データベースの動作を確認することができます。

```
[oracle@stbydb1]$ srvctl config database -d boston
SQL> select database_role, switchover_status from v$database;
SQL> select thread#, sequence#, applied from v$archived_log order by sequence#;
```

30. クラスタウェアのロール・ベース・サービスのセットアップ – [高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティスを参照してください](#)

31. レジストリを同期します（18.7 BM新機能）。

odacli list-databases

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	DbHomeID
2d7d2b0b-8489-4e28-a151-64f6e156a512	chicago	Rac	11.2.0.4.200714	false	Oltp	Odb1	ACFS	Configured	
45e829ab-bf71-48ef-8f9a-0b0397007ba3									

odacli describe-database -i 2d7d2b0b-8489-4e28-a151-64f6e156a512

Database details

ID:2d7d2b0b-8489-4e28-a151-64f6e156a512
Description: chicago
DB Name: chicago
DB Version:11.2.0.4.200714
DB Type:Rac
DB Role:
DB Target Node Name:
DB Edition:EE
DBID:
Instance Only
Database: true
CDB: false
PDB Name:
PDB Admin User Name:
SEHA Enabled: false
Class:Otp
Shape:Odb1
Storage:ACFS
DB Redundancy:MIRROR
CharacterSet:AL32UTF8
National CharacterSet:AL16UTF16
Language:AMERICAN
Territory:AMERICA
Home ID:45e829ab-bf71-48ef-8f9a-0b0397007ba3
Console Enabled: false
TDE Enabled: false
Level 0 Backup Day:Sunday
AutoBackup Enabled: true
Created:November 5, 2019 6:02:19 PM CET
DB Domain Name: us.oracle.com
Associated Networks:

odacli update-registry -n db -f

odacli describe-job -i e367a5d4-526a-4d13-8ff2-81ab112a813a

Job details

ID: e367a5d4-526a-4d13-8ff2-81ab112a813a

Description: Discover Components : db

Status: Success

Created: July 7, 2020 10:32:37 PM CEST

Task Name	Start Time	End Time	Status
Rediscover DBHome	July 7, 2020 10:32:38 PM CEST	July 7, 2020 10:32:40 PM CEST	Success
Rediscover DB: boston	July 7, 2020 10:32:40 PM CEST	July 7, 2020 10:32:45 PM CEST	Success

odacli describe-database -i 2d7d2b0b-8489-4e28-a151-64f6e156a512

Database details

ID: 2d7d2b0b-8489-4e28-a151-64f6e156a512

Description: chicago

DB Name: chicago

DB Version: 11.2.0.4.200714

DB Type: Rac

DB Role: STANDBY

DB Target Node Name:

DB Edition: EE

DBID: 1654855519

Instance Only

Database: false

CDB: false

PDB Name:

PDB Admin User Name:

SEHA Enabled: false

Class: Oltp

Shape: Odb1

Storage: ACFS

DB Redundancy: MIRROR

CharacterSet: AL32UTF8

National CharacterSet: AL16UTF16

Language: AMERICAN

Territory: AMERICA

Home ID: 45e829ab-bf71-48ef-8f9a-0b0397007ba3

Console Enabled: false

TDE Enabled: false

Level 0 Backup Day:

AutoBackup Enabled: true

Created: July 7, 2020 10:32:40 PM CEST

この手順はオプションですが、強くお勧めします。Data Guard Brokerの構成を作成すると、Oracle Data Guard環境全体を1つのエントリとして容易に管理できるようになります。ローカルでもリモートでも使用可能な、管理、メンテナンス、監視の機能を備えています。

32. 静的な登録のためのリスナーを構成します。

プライマリ・データベースおよびスタンバイ・データベースのすべてのインスタンスの静的な登録のためのリスナーを構成します。Oracle Database Applianceで、リスナーはGrid Infrastructureのホームから実行されます。Oracle RACのプライマリ構成およびスタンバイ構成のための静的な登録の例を以下に示します。

ノードproddb1の場合：

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC = (GLOBAL_DBNAME = chicago_DGMGRL)
(ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
(SID_NAME = chicago1)))
```

ノードproddb2の場合：

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC = (GLOBAL_DBNAME = chicago_DGMGRL)
(ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
(SID_NAME = chicago2)))
```

ノードstbydb1の場合：

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(GLOBAL_DBNAME = boston_DGMGRL)
(ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
(SID_NAME = boston1)))
```

ノードstbydb2の場合：

```
SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC = (GLOBAL_DBNAME = boston_DGMGRL)
(ORACLE_HOME = /u01/app/oracle/product/11.2.0.4/dbhome_1)
(SID_NAME = boston2)))
```

33. Broker設定ファイルを構成します。

プライマリ・データベースとスタンバイ・データベースの両方でBroker設定ファイルの場所を構成します。

```
[oracle@proddb1]$ mkdir -p /u02/app/oracle/oradata/datastore/chicago/broker
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE1='/u02/app/oracle/oradata/datastore/chicago/broker/dr1.dat'
SCOPE=BOTH SID='*';
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE2='/u02/app/oracle/oradata/datastore/chicago/broker/dr2.dat'
SCOPE=BOTH SID='*';
[oracle@stbydb1]$ mkdir -p /u02/app/oracle/oradata/datastore/boston/broker
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE1='/u02/app/oracle/oradata/datastore/boston/broker/dr1.dat'
SCOPE=BOTH SID='*';
SQL> ALTER SYSTEM SET DG_BROKER_CONFIG_FILE2='/u02/app/oracle/oradata/datastore/boston/broker/dr2.dat'
SCOPE=BOTH SID='*';
```

34. Oracle Data Guard Brokerを有効化します。

プライマリ・データベースとスタンバイ・データベースの両方でOracle Data Guard Brokerを有効化します。ノードproddb1の場合：

```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE SCOPE=BOTH SID='*';
```

ノードstbydb1の場合：

```
SQL> ALTER SYSTEM SET DG_BROKER_START=TRUE SCOPE=BOTH SID='*';
```

35. Broker構成を作成します。

プライマリ・データベースのDB_UNIQUE_NAMEとそれに対応するTNSエイリアスを使用して、プライマリのBroker構成を作成します。

```
[oracle@proddb1]$ dgmgri
DGMGRL> connect sys/welcome1
DGMGRL> CREATE CONFIGURATION 'ODADGConfig' AS PRIMARY DATABASE IS CHICAGO CONNECT IDENTIFIER is
CHICAGO;
```

36. Oracle Data Guard Broker構成にスタンバイ・データベースを追加します。

スタンバイ・データベースのDB_UNIQUE_NAMEを使用して、構成にスタンバイ・データベースを追加します。

```
DGMGRL> ADD DATABASE BOSTON AS CONNECT IDENTIFIER IS BOSTON;
```

37. 構成を有効化します。

Oracle Data Guard Broker構成を次のように有効化します。

```
DGMGRL> ENABLE CONFIGURATION;
```

38. 構成を確認します。

次のコマンドを実行して、確立された構成を確認します。

```
DGMGRL> show configuration;
DGMGRL> show database verbose chicago;
DGMGRL> show instance verbose chicago1;
DGMGRL> show instance verbose chicago2;
DGMGRL> show database verbose boston;
DGMGRL> show instance verbose boston1;
DGMGRL> show instance verbose boston2;
```

“show instance verbose”の出力の‘StaticConnectIdentifier’パラメータが以下のようにになっていることを確認します。

```
StaticConnectIdentifier      =  
'(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=proddb1)(PORT=1521))(CONNECT_DATA=(SERVICE_NAME=chicago_ DGMGRL.us.oracle.com)(INSTANCE_NAME=chicago1)(SERVER=DEDICATED)))'
```

以下のようにになっている場合：

```
StaticConnectIdentifier      =  
'(DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=DUMMY))(CONNECT_DATA=(SERVICE_NAME=chicago_DGMGRL.us.o racle.com)(INSTANCE_NAME=chicago1)(SERVER=DEDICATED)))'
```

その場合は、次のコマンドを使用してこのパラメータを更新します。

```
DGMGRL> edit instance chicago1 set property  
StaticConnectIdentifier='(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=proddb1)(PORT=1521))(CONNECT_DATA=(SE RVICE_NAME=chicago_DGMGRL.us.oracle.com)(INSTANCE_NAME=chicago1)(SERVER=DEDICATED)))';
```

```
DGMGRL> edit instance chicago2 set property  
StaticConnectIdentifier='(DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=proddb2)(PORT=1521))(CONNECT_DATA=(SE RVICE_NAME=chicago_DGMGRL.us.oracle.com)(INSTANCE_NAME=chicago2)(SERVER=DEDICATED)))'
```

この問題は、Oracle Database Appliance 19.6以上のベア・メタル・バージョンにデータベースが作成された場合に発生することがあります。

39. テストをスイッチオーバーします。

```
$ dgmgrl  
DGMGRL> connect sys/welcome1@boston  
DGMGRL> switchover to boston  
DGMGRL> connect sys/welcome1@chicago  
Connected as SYSDBA.  
DGMGRL> switchover to chicago;
```

40. テストをフェイルオーバーします。

フェイルオーバーの前にスタンバイに接続します：

```
$ dgmgrl  
DGMGRL> connect sys/welcome1@boston  
DGMGRL> failover to boston  
DGMGRL> reinstate database chicago
```

フェイルオーバーの前に以前のプライマリに接続します：

```
DGMGRL> connect sys/welcome1@chicago  
DGMGRL> failover to chicago;  
DGMGRL> reinstate database boston
```


付録D：Oracle RACへのシングル・インスタンス・データベースの変換

rconfigコマンドライン・ユーティリティを使用して、シングル・インスタンス・データベースをOracle RACデータベースまたはOracle RAC One Nodeに変換することができます。

この機能を利用するには、次の手順を実行します。

1. XML構成ファイルを作成します。

convert.xmlとして保存するXML構成ファイルの例を以下に示します。このファイルは、システムの要件に合わせて変更することができます。サンプルXMLファイルは\$ORACLE_HOME/assistants/rconfig/sampleXMLディレクトリにあります。

注：変換が正しく行われるように、テスト変換の実行用に変換オプションConvert verify="ONLY"を最初に設定してください。

```
<?xml version="1.0" encoding="UTF-8"?>
<n:RConfig xmlns:n="http://www.oracle.com/rconfig"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.oracle.com/rconfig rconfig.xsd">
  <n:ConvertToRAC>
    <n:Convert verify="YES">
      <n:SourceDBHome>/u01/app/oracle/product/11.2.0.4/dbhome_1</n:SourceDBHome>
      <n:TargetDBHome>/u01/app/oracle/product/11.2.0.4/dbhome_1</n:TargetDBHome>
      <n:SourceDBInfo SID="boston">
        <n:Credentials>
          <n:User>sys</n:User>
          <n:Password>welcome1</n:Password>
          <n:Role>sysdba</n:Role>
        </n:Credentials>
      </n:SourceDBInfo>
      <n:NodeList>
        <n:Node name="stbydb1"/>
        <n:Node name="stbydb2"/>
      </n:NodeList>
      <n:InstancePrefix>boston</n:InstancePrefix>
      <n:SharedStorage type="CFS">
        <n:TargetDatabaseArea></n:TargetDatabaseArea>
        <n:TargetFlashRecoveryArea></n:TargetFlashRecoveryArea>
      </n:SharedStorage>
    </n:Convert>
  </n:ConvertToRAC>
</n:RConfig>
```

2. reconfigツールを実行します。

変更が完了したら、ファイルを保存します。スタンバイ・データベースで次のコマンドを実行します。 *convert.xml*は上記で構成したXML入力ファイルの名前です。

```
[oracle@stbydb1]$ rconfig convert.xml
```

3. Oracle Cluster Ready Servicesのリソースを更新します。

変換したデータベースのためにOracle Cluster Ready Services (Oracle CRS) リソースを更新する必要があります。

```
[oracle@stbydb1]$ srvctl modify database -d boston -r physical_standby -s mount
```

スタンバイ・データベースを再起動します。

```
[oracle@stbydb1]$ srvctl stop database -d boston
```

```
[oracle@stbydb1]$ srvctl start database -d boston
```

4. 構成を検証します。

5.

スタンバイ・データベースの構成を検証します。

```
[oracle@stbydb1]$ srvctl config database -d boston
```

付録E：OAKを使用するOracle Database ApplianceでのOracle Data Guardを使用したデータベースのアップグレード

すべてのコンポーネントのアップグレード

ODA環境のアップグレードは、サーバー、ストレージ、およびデータベース・コンポーネントのアップグレードで構成されます。スタンバイ・システムがすでに展開されているODA環境をアップグレードする場合、スタンバイ・システムを利用してアップグレード・アクティビティを完了させるために必要な停止時間を短縮することができます。このセクションの目的は、プライマリ/スタンバイ設定でのアップグレード・プロセスについて大まかな概要を示すことです。

1. システムが正常に動作しているか確認します（事前チェック、ハードウェアおよびシステム・プロセスの検証、`orachk`を使用したシステム構成の確認など）。
2. OS、GI、Oracleのホーム、データベースのバックアップを取ります（プライマリ環境）。
ベア・メタル・**デプロイメント**については、以下を参照してください。MOS Note [2466177.1](#) - ODA (Oracle Database Appliance):ODABR a System Backup/Restore Utility
3. スタンバイODAシステムでSERVERコンポーネントをアップグレードします。
4. プライマリ・データベースのロールとアプリケーション接続をスタンバイODAシステムにスイッチオーバーします。
5. 元のプライマリODAシステムのSERVERコンポーネントをアップグレードします。
6. DBホームが存在しない場合、または既存のホームを使用したくない場合は、"`oakcli create dbhome`"コマンドを使用して望ましいバージョンで現在のスタンバイ上に新しいDBホームを作成します。
7. DBホームが存在しない場合、または既存のホームを使用したくない場合は、望ましいバージョンで現在のプライマリ上に新しいDBホームを作成します。
8. プライマリからスタンバイへのREDO送信を保留にします。
9. 現在のスタンバイ・データベースで"`oakcli upgrade database`"コマンドを実行します（バイナリのみがアップグレード/スイッチされ、カタログ・スクリプトは実行されません）。
10. アプリケーション・トラフィックを停止します。
11. "`oakcli upgrade database`"コマンドを使用して現在のプライマリ・データベースをアップグレードします。
12. アプリケーション・トラフィックを開始します。
13. プライマリ・データベースからスタンバイ・データベースへのREDO送信を有効化します。
14. プライマリ側とスタンバイ側でDBの動作を確認します。
15. オプションで、プライマリ環境とスタンバイ環境の間でロールをスイッチバックします。

上記のプロセスを使用することにより、アップグレード中の停止時間が最小限に抑えられ、システムの可用性への影響がデータベース・コンポーネントのアップグレード中のみに限定されます。

データベースのみのアップグレード

SERVERコンポーネント（OS、GI、一般的なファームウェア）とSTORAGEコンポーネント（共有ディスクのファームウェア）をアップグレードする場合、スイッチオーバー/スイッチバックを利用すると停止時間が短縮されます。しかし、データベース・コンポーネントのみをアップグレードし、ゼロ停止時間ソリューション（アクティブ/アクティブGoldenGateソリューションなど）を使用しない場合、アプリケーションで多少の停止時間が発生します。スタンバイ構成が存在する場合にデータベース・アップグレードを実行するための一般的なプロセスは、次のようになります。

1. システムが正常に動作しているか確認します（事前チェック、ハードウェアおよびシステム・プロセスの検証、`orachk`を使用したシステム構成の確認など）。
2. データベースとOracleホームのバックアップを取ります。
3. DBホームが存在しない場合、または既存のホームを使用したくない場合は、`"oakcli create dbhome"`コマンドを使用して望ましいバージョンでスタンバイ上に新しいDBホームを作成します。
4. DBホームが存在しない場合、または既存のホームを使用したくない場合は、望ましいバージョンでプライマリ上に新しいDBホームを作成します。
5. アプリケーションを停止します。
6. スタンバイDBをプライマリと同期させ、プライマリで生成された最新のSCNがスタンバイに適用されているか確認します。
7. プライマリ・データベースからスタンバイ・データベースへのREDO送信を保留にします（オブション。REDOフローを停止）。
8. スタンバイ・データベースで`"oakcli upgrade database"`コマンドを実行します（バイナリのみがアップグレード/スイッチされ、カタログ・スクリプトは実行されません）。
9. `"oakcli upgrade database"`コマンドを使用してプライマリ・データベースをアップグレードします。
10. アプリケーションを起動します。
11. プライマリ・データベースからスタンバイ・データベースへのREDO送信を有効化します。
12. プライマリ側とスタンバイ側でDBの動作を確認します。

必要な合計停止時間はDBのアップグレード時間です。データベースのアップグレード中は、スイッチオーバーとスイッチバックは不要です。

付録F：DCSを使用するOracle Database ApplianceでのOracle Data Guardを使用したデータベースのアップグレード

注：Oracle Database Appliance 19.8のベア・メタル・デプロイメントでは、ドキュメントの説明に従い、ODAのツールを全面的に使用するようになっています。このツールは、Oracle Data Guardをodacliコマンドで構成した場合に、データベースのパッチ適用やアップグレードを含むOracle Data Guard環境のライフサイクル全体の管理にも対応するからです。「[Configuring Oracle Data Guard on Oracle Database Appliance](#)」を参照してください。odacliコマンドで構成されたOracle Data Guard環境については、以下の手順は適用されません。

すべてのコンポーネントのアップグレード

ODA環境のアップグレードは、DCS、SERVER、STORAGE、およびDATABASEコンポーネントのアップグレードで構成されます。スタンバイ・システムがすでに展開されているODA環境をアップグレードする場合、スタンバイ・システムを利用してアップグレード・アクティビティを完了させるために必要な停止時間を短縮することができます。このセクションの目的は、プライマリ/スタンバイ設定でのアップグレード・プロセスについて大まかな概要を示すことです。

1. システムが正常に動作しているか確認します（事前チェック、ハードウェアおよびシステム・プロセスの検証、orachkを使用したシステム構成の確認など）。
2. IS、GI、Oracleのホーム、データベースのバックアップを取ります（プライマリ環境）。

以下を参照してください。MOS Note [2466177.1](#) - ODA (Oracle Database Appliance):ODABR a System Backup/Restore Utility

3. スタンバイODAシステムでDCSおよびSERVERコンポーネントをアップグレードします。
4. プライマリ・データベースのロールとアプリケーション接続をスタンバイODAにスイッチオーバーします。
5. 現在のスタンバイ（以前のプライマリ）ODAでDCSおよびSERVERコンポーネントをアップグレードします。

6. 要件に応じてデータベースに[パッチを適用](#)またはデータベースを[アップグレード](#)します。

Oracle Database Appliance 19.8のベア・メタル・デプロイメントでは、ドキュメントの説明に従い、ODAのツールを全面的に使用するようになっています。このツールは、Oracle Data Guardをodacliコマンドで構成した場合に、データベースのパッチ適用やアップグレードを含むOracle Data Guard環境のライフサイクル全体の管理にも対応するからです。「[Configuring Oracle Data Guard on Oracle Database Appliance](#)」を参照してください。

上記のプロセスを使用することにより、アップグレード中の停止時間が最小限に抑えられ、システムの可用性への影響がデータベース・コンポーネントのアップグレード中のみに限定されます。

データベースのアップグレード

DCS、SERVER（OS、GI、一般的なファームウェア）、およびSTORAGE（共有ディスクのファームウェア）コンポーネントをアップグレードする場合、スイッチオーバー/スイッチバックを利用すると停止時間が短縮されます。しかし、DATABASEコンポーネントのみをアップグレードし、ゼロ停止時間ソリューション（アクティブ/アクティブGoldenGateソリューションなど）を使用しない場合、アプリケーションで多少の停止時間が発生します。スタンバイ構成が存在する場合にデータベース・アップグレードを実行するための一般的なプロセスは、次のようになります。

1. システムが正常に動作しているか確認します（事前チェック、ハードウェアおよびシステム・プロセスの検証、`orachk`を使用したシステム構成の確認など）。
2. データベースとOracleホームのバックアップを取ります。
3. スタンバイ・データベースを停止します。

```
[oracle@stbydb1]$ srvctl stop database -d boston
```

4. プライマリでのデータベースのアップグレード先のバージョンのスタンバイで既存のデータベース・ホームを使用、または新しいデータベース・ホームを作成します。

```
[oracle@stbydb1]# odacli create-dbhome -v 18.7.0.0.190716
```

5. プライマリでログの送信を停止します。

```
[oracle@proddb1] dgmgrl
connect sys/welcome1@chicago
DGMGRL> SHOW DATABASE 'boston' 'LogShipping';
LogShipping = 'ON'
DGMGRL> edit database 'boston' SET PROPERTY 'LogShipping'='OFF';
Property "LogShipping" updated
DGMGRL> SHOW DATABASE 'boston' 'LogShipping';
LogShipping = 'OFF'
```

6. データベースのアップグレード先のバージョンのプライマリで既存のデータベース・ホームを使用、または新しいデータベース・ホームを作成します。

```
# odacli create-dbhome -v 18.7.0.0.190716
```

7. アプリケーションを停止します。
8. “odacli upgrade database”コマンドを使用してプライマリ・データベースをアップグレードします。

```
[root@proddb1]# odacli list-databases
```

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	DbHomeID
e97cc2f3-bdd8-4775-b959-d5f79a6c59fc	chicago	Rac	12.2.0.1.190716	false	Oltp	Odb1	Asm	Configured	
88ce2c7-fa3d-4f93-802a-bfa50d180758									

```
[root@proddb1]# odacli list-dbhomes
```

ID	Name	DB Version	Home Location	Status
863c8cbe-1c5f-450e-866c-15c384580ad3	OraDB18000_home1	18.7.0.0.190716	/u01/app/oracle/product/18.0.0.0/dbhome_1	Configured
288ce2c7-fa3d-4f93-802a-bfa50d180758	OraDB12201_home1	12.2.0.1.190716	/u01/app/oracle/product/12.2.0.1/dbhome_1	Configured

```
[root@proddb1]# odacli upgrade-database -i 713b68d3-8c43-4d10-973e-90a3fa88a84a --destDbHomeId 863c8cbe-1c5f-450e-866c-15c384580ad3 --sourceDbHomeId 288ce2c7-fa3d-4f93-802a-bfa50d180758
[root@proddb1]# odacli list-databases
```

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	DbHomeID
713b68d3-8c43-4d10-973e-90a3fa88a84a	chicago	Rac	18.7.0.0.190716	false	Oltp	Odb1	Asm	Configured	863c8cbe-1c5f-450e-866c-15c384580ad3

- アプリケーションを起動します。
- スタンバイのtnsnames.oraファイルを古いOracleホームから全ノード上の新しいOracleホームにコピーします。
- パスワード・ファイルをプライマリからスタンバイにコピーします。

```
[oracle@proddb1]$ srvctl config database -d chicago |grep Password
Password file: +DATA/CHICAGO/PASSWORD/pwdchicago.277.1023633847
[grid@proddb1 ~]$ asmcmd
ASMCMD> pwcopyp +DATA/CHICAGO/PASSWORD/pwdchicago.277.1023633847 /tmp/pwdboston
copying +DATA/CHICAGO/PASSWORD/pwdchicago.277.1023633847 -> /tmp/pwdboston
[oracle@proddb1]$ scp /tmp/pwdboston oracle@stbydb1:/u01/app/oracle/product/18.0.0.0/dbhome_2/dbs/orapwboston
[grid@stbydb1 ~]$ asmcmd
ASMCMD> pwcopyp /u01/app/oracle/product/18.0.0.0/dbhome_2/dbs/orapwboston
+DATA/BOSTON/PASSWORDFILE/pwdboston
copying /u01/app/oracle/product/18.0.0.0/dbhome_2/dbs/orapwboston -> +DATA/BOSTON/PASSWORDFILE/pwdboston
```

- 12.2データベースをスタンバイのClusterwareから削除します。

```
[oracle@stbydb1]# srvctl remove database -db boston
Remove the database boston? (y/[n]) y
```

- データベースをスタンバイのClusterwareに戻します。Oracleホームは新しいバージョンのホームを参照する必要があります。

シングル・インスタンスの例

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/18.0.0.0/dbhome_1 -dbtype
SINGLE -instance boston1 -node stbydb1 -dbname chicago -diskgroup 'DATA,REDO,RECO' -role physical_standby -spfile
'+DATA/BOSTON/PARAMETERFILE/spfileboston' -pwfile '+DATA/BOSTON/PASSWORDFILE/pwdboston'
```

Oracle RACの例

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/18.0.0.0/dbhome_1 -dbtype RAC -
dbname chicago -diskgroup 'DATA,RECO,REDO' -role physical_standby -spfile
'+DATA/BOSTON/PARAMETERFILE/spfileboston' -pwfile '+DATA/BOSTON/PASSWORDFILE/pwdboston'
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston1 -node stbydb1
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston2 -node stbydb2
[oracle@stbydb1]$ srvctl start instance -db boston -instance boston1 -o mount
[oracle@stbydb1]$ srvctl start instance -db boston -instance boston2 -o mount
```

14. ログの送信を有効にしてData Guard構成を検証します。

```
[oracle@stbydb1]$ dgmgrl
DGMGRL> connect sys/welcome1@chicago
DGMGRL> edit database 'boston' SET PROPERTY 'LogShipping'='ON';
Property "LogShipping" updated
DGMGRL> SHOW DATABASE 'boston' 'LogShipping';
LogShipping = 'ON'
DGMGRL> show configuration verbose
DGMGRL> show database verbose chicago
DGMGRL> show database verbose boston
DGMGRL> validate database chicago
DGMGRL> validate database boston
```

15. スイッチオーバーとフェイルオーバーをテストします。

スイッチオーバーをテストします。

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> switchover to boston
DGMGRL> connect sys/welcome1@chicago
DGMGRL> switchover to chicago;
```

フェイルオーバーをテストします。

フェイルオーバーの前にスタンバイに接続します：

```
$ dgmgrl
DGMGRL> connect sys/welcome1@boston
DGMGRL> failover to boston
DGMGRL> reinstate database chicago
```

フェイルオーバーの前に以前のプライマリに接続します：

```
DGMGRL> connect sys/welcome1@chicago
DGMGRL> failover to chicago;
DGMGRL> reinstate database boston
```

ヘルス・チェック

```
DGMGRL> show database verbose Chicago
DGMGRL> show database verbose boston
DGMGRL> validate database Chicago
DGMGRL> validate database boston
```


16. スタンバイのレジストリを同期します（18.7 BM新機能）。

DBのバージョンを検証します。

```
[root@ stbydb1~]# odacli list-databases
```

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	DbHomeID
e6450a56-5a7d-4dab-9ca9-25b004b66646		chicago	Rac	12.2.0.1.190716	false	Oltp	Odb1	Asm	Configured

レジストリ・エントリを同期します。

```
[root@ stbydb1~]# odacli update-registry -n db -f
```

```
[root@ stbydb1~]# odacli describe-job -i 25ec2987-4c93-4d25-97db-bad2f6f602f6
```

Job details

```

ID:25ec2987-4c93-4d25-97db-bad2f6f602f6
Description:Discover Components : db
Status:Success
Created:November 6, 2019 11:00:50 PM CET
Message:
Task Name          Start Time          End Time          Status
-----
Rediscover DBHome   November 6, 2019 11:00:54 PM CET   November 6, 2019 11:00:56 PM CET   Success
Rediscover DB: boston November 6, 2019 11:00:56 PM CET   November 6, 2019 11:01:02 PM CET   Success
  
```

レジストリが変更されたことを確認します。

```
[root@ stbydb1~]# odacli list-databases
```

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	bHomeID
e6450a56-5a7d-4dab-9ca9-25b004b66646		chicago	Rac	18.7.0.0.190716	false	Oltp	Odb1	Asm	Configured

必要な合計停止時間はDBのアップグレード時間です。データベースのアップグレードでは、スイッチオーバーとスイッチバックは不要です。

データベースへのパッチの適用

Oracle Database Appliance上でのデータベースへのパッチ適用は完全なオンライン操作です。スタンバイ構成が存在する場合にデータベースへのパッチ適用を実行するためのプロセスは、次のようになります。

1. システムが正常に動作しているか確認します（事前チェック、ハードウェアおよびシステム・プロセスの検証、`orachk`を使用したシステム構成の確認など）。
2. データベースのバックアップを取ります。
3. プライマリでログの送信を停止します。

```
$ dgmgrl
DGMGRL> connect sys/welcome1@chicago
DGMGRL> edit database 'CHICAGO' SET STATE="LOG-TRANSPORT-OFF";
DGMGRL> SHOW DATABASE 'boston' 'LogShipping';
LogShipping = 'ON'
DGMGRL> edit database 'boston' SET PROPERTY 'LogShipping'='OFF';
Property "LogShipping" updated
DGMGRL> SHOW DATABASE 'boston' 'LogShipping';
LogShipping = 'OFF'
```

4. スタンバイ・データベースを停止して“読み取り専用”モードで再起動します。

```
[oracle@stbydb1]$ srvctl stop database -d boston
[oracle@stbydb1]$ srvctl start database -db boston -o "read only"
```

5. 先にスタンバイ・データベースにパッチを適用します。

データベースのOracleホームを特定します。

```
[root@ocboda10 ~]# odacli list-databases
```

ID	DB Name	DB Type	DB Version	CDB	Class	Shape	Storage	Status	DbHomeID
667a0eec-910c-404b-9820-aedcddf668d7 e4e9fcbd-63d4-4c56-bb0c-b239a4e749f3	chicago	Rac	18.5.0.0.190115	false	Olt	Odb1	Asm	Configured	

```
[oracle@stbydb1]# odacli list-dbhomes
```

ID	Name	DB Version	Home Location	Status
e4e9fcbd-63d4-4c56-bb0c-b239a4e749f3	OraDB18000_home3	18.5.0.0.190115	/u01/app/oracle/product/18.0.0.0/dbhome_3	Configured

Oracleホームで事前チェックを実行します。

```
[oracle@stbydb1]# odacli update-dbhome -p -i 6d05e3f1-e948-4482-bcba-c560d9c8e5e5 -v 18.7.0.0
```

```
[oracle@stbydb1]# odacli describe-job -i b4ee24d9-2b82-4c80-b789-ced90013e4b3
```

Job details

ID: b4ee24d9-2b82-4c80-b789-ced90013e4b3

Description:DB Home Prechecks

Status:Success

Created:November 7, 2019 6:26:51 PM CET

パッチを適用します。

```
[oracle@stbydb1]# odacli update-dbhome -i 6d05e3f1-e948-4482-bcba-c560d9c8e5e5 -v 18.7.0.0
```

```
[oracle@stbydb1]# odacli describe-job -i "e3556125-7ce6-4560-9f22-3fdd9738f955"
```

Job details

ID: e3556125-7ce6-4560-9f22-3fdd9738f955

Description:DB Home Patching:Home Id is e4e9fcbd-63d4-4c56-bb0c-b239a4e749f3

Status:Success

Created:November 7, 2019 7:09:52 PM CET

結果を確認します。

```
[oracle@stbydb1]# odacli list-dbhomes
```

<i>ID</i>	<i>Name</i>	<i>DB Version</i>	<i>Home Location</i>	<i>Status</i>
-----------	-------------	-------------------	----------------------	---------------

<i>e4e9fcbd-63d4-4c56-bb0c-b239a4e749f3</i>	<i>OraDB18000_home3</i>	<i>18.7.0.0.190716</i>		
<i>/u01/app/oracle/product/18.0.0.0/dbhome_3</i>	<i>Configured</i>			

```
[oracle@stbydb1]# odacli list-databases
```

<i>ID</i>	<i>DB Name</i>	<i>DB Type</i>	<i>DB Version</i>	<i>CDB</i>	<i>Class</i>	<i>Shape</i>	<i>Storage</i>	<i>Status</i>	<i>DbHomeID</i>
-----------	----------------	----------------	-------------------	------------	--------------	--------------	----------------	---------------	-----------------

<i>667a0eec-910c-404b-9820-aedcddf668d7</i>	<i>chicago</i>	<i>Rac</i>	<i>18.7.0.0.190716</i>	<i>false</i>	<i>Olt</i>	<i>tp</i>	<i>Odb1</i>	<i>Asm</i>	<i>Configured</i>
<i>e4e9fcbd-63d4-4c56-bb0c-b239a4e749f3</i>									

6. プライマリ・データベースにパッチを適用します。手順は手順5と同じです。

7. プライマリでログの送信を開始し、Data Guard構成を確認します。

```
DGMGRL> connect sys/welcome1@chicago
```

```
DGMGRL> edit database 'boston' SET PROPERTY 'LogShipping'='ON';
```

Property "LogShipping" updated

```
DGMGRL> SHOW DATABASE 'boston' 'LogShipping';
```

LogShipping = 'ON'

```
DGMGRL> show configuration verbose
```

```
DGMGRL> show database verbose chicago
```

```
DGMGRL> show database verbose boston
```

```
DGMGRL> validate database chicago
```

```
DGMGRL> validate database boston
```

付録G：同じバージョンのRDBMSホームに一部の古いバージョンのPSUとバンドル・パッチが含まれる理由

12.1データベースでは、いくつかの古いリリースの古いスタック（OAK）とは異なるパッチが新しいスタック（DCS）に含まれていました。

- » S/M/Lモデルは、常に新しいスタック（DCS/odacli）で実行されてきました。バンドル・パッチは、これらのモデルの12.1データベースに適用されます。
- » HAモデルでは、PSUは古いスタック（OAK/oakcli）を含むいくつかの古いOracle Database Applianceリリースで12.1 RDBMSホームに適用されました。
- » バンドル・パッチには、PSUよりも多くの修正が含まれます。具体的にはPSUのスーパーセットです。
- » バンドル・パッチは、12.2.1.2 Oracle Database Applianceのリリース以来、すべてのスタックを含むすべてのモデルの12.1 RDBMSに適用されてきました。

上記の相違により、RDBMSホームの一方にPSUが含まれ、他方にBPが含まれる場合、Oracle Database Appliance間でData Guardが構成されるのを阻止します。プライマリ・データベースとスタンバイ・データベースとでパッチ・レベルを同一にする必要があることに注目してください。MOS Note [785347.1](#) 『Mixed Oracle Version support with Data Guard Redo Transport Services』を参照してください。

この問題は、データベース・ホームに3年以上パッチが適用されていない構成に影響します。

解決策は、データベースにパッチを適用して12.1 RDBMSバージョンのBPを含む最新バージョンにすることです。

参考資料

ドキュメント

[Oracle Database Appliance Webサイト](#)

[Configuring Oracle Data Guard on Oracle Database Appliance](#)

[Oracle Maximum Availability Architectureベスト・プラクティス](#)

[Oracle Databaseの高可用性Webサイト](#)

[Oracle Real Application Clusters Webサイト](#)

[Oracle Clusterware Webサイト](#)

[Oracle Data Guard Webサイト](#)

Oracle Data Guard概要および管理 [19c](#)、[18c](#)、[12.2](#)、[12.1](#)、[11.2](#)

技術概要

[Oracle Maximum Availability Architecture \(Oracle MAA\) オンプレミス、Exadataベース、またはクラウドベースのOracle Databaseの計画停止時間と計画外停止時間の短縮構想](#)

[Oracle Data GuardおよびOracle Active Data Guard 12cのREDO転送を構成するためのベスト・プラクティス](#)

[Best Practices for Asynchronous Redo Transport - Data Guard and Active Data Guard](#)

[同期REDO転送のベスト・プラクティス Oracle Data GuardとOracle Active Data Guard](#)

[Best Practices for Automatic Resolution of Outages to Resume Data Guard Zero Data Loss](#)

[ブロック破損の防止、検出、および修復：Oracle Database 12c](#)

[ロール移行のベスト・プラクティス：Oracle Data GuardおよびOracle Active Data Guard](#)

[高可用性Oracle Databaseでのクライアント・フェイルオーバーのベスト・プラクティス](#)

[Client Failover Best Practices for Data Guard 12c](#)

[Data Guardを使用したデータベース・ローリング・アップグレード](#)

[Oracle Active Data GuardとDBMS_ROLLINGを使用した自動データベース・アップグレード](#)

My Oracle Support (MOS) Knowledge Content Notes

[Note 2466177.1 - ODA \(Oracle Database Appliance\):ODABR a System Backup/Restore Utility](#)

[Note 1265700.1 - Oracle Patch Assurance - Data Guard Standby-First Patch Apply](#)

[Note 1617946.1 - Creating a Physical Standby Database using RMAN Duplicate \(RAC or Non-RAC\)](#)

[Note 2283978.1 - Creating a Physical Standby database using RMAN restore database from service](#)

[Note 785347.1 - Mixed Oracle Version support with Data Guard Redo Transport Services](#)

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

Copyright © 2020, Oracle and/or its affiliates. All rights reserved. 本文書は情報提供のみを目的として提供されており、ここに記載されている内容は予告なく変更されることがあります。本文書は、その内容に誤りがないことを保証するものではなく、また、口頭による明示的保証や法律による黙示的保証を含め、商品性ないし特定目的適合性に関する黙示的保証および条件などのいかなる保証および条件も提供するものではありません。オラクルは本文書に関するいかなる法的責任も明確に否認し、本文書によって直接的または間接的に確立される契約義務はないものとします。本文書はオラクルの書面による許可を前もって得ることなく、いかなる目的のためにも、電子または印刷を含むいかなる形式や手段によっても再作成または送信することはできません。

OracleおよびJavaはOracleおよびその子会社、関連会社の登録商標です。その他の名称はそれぞれの会社の商標です。

IntelおよびIntel XeonはIntel Corporationの商標または登録商標です。すべてのSPARC商標はライセンスに基づいて使用されるSPARC International, Inc.の商標または登録商標です。AMD、Opteron、AMDロゴおよびAMD Opteronロゴは、Advanced Micro Devicesの商標または登録商標です。UNIXは、The Open Groupの登録商標です。0120

Oracle Database Appliance：Oracle Data Guardを使用したMAAディザスタ・リカバリ・ソリューションの実装 2020年8月

著者：Krisztian Fekete, Ramasubramanian Athmanathan, Ravi Sharma, Sanjay Singh

共著者：Oracle RAC Pack and MAA Team

