Oracle Maximum
Availability Architecture

Oracle Database Appliance：Oracle Data Guardを使用したディザスタ・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 ホワイト・ペーパー | 2017 年 9 月

ORACLE

目次

はじめに 1

Oracle Active Data Guard を使用したデータ保護 1

Oracle Data Guard を使用する利点 2

構成のベスト・プラクティス 4

Oracle Database Appliance システム間での Oracle Data Guard の設定 6

- Oracle Data Guard の構成手順 6
- 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 7
- Oracle Database Appliance の小規模、中規模、大規模プラットフォームの構成 7

結論 7

付録 A：Oracle Database Appliance での 12c の設定例 8

付録 B：Oracle Database Appliance での 11gR2 の設定例 19

付録 C：Oracle RAC へのシングル・インスタンス・データベースの変換 32

付録 D：Oracle Data Guard を使用したデータベースのアップグレード 34

参考資料 36

はじめに

Oracle Database Appliances は構築と調整が完了していて、すぐに使えるようになっているクラスタ化されたおよびクラスタ化されていないデータベース・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、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につながることの多い構成や設定が防止されます。

Oracle Active Data Guardを使用したデータ保護

Oracle Database Appliance はそれ自体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、計画停止時間と計画外停止時間に対する保護と、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のデータ損失に対する保護を行うことができます。Oracle Maximum Availability Architecture（Oracle MAA）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、大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。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可用性とデータ保護を向上させるための MAA の不可欠な要素となっています。

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、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と停止時間の発生を防ぎます。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 12c のオフロード機能は強化され、読取り専用のレポートおよび任意の問合せ（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組み込まれました。Oracle Active Data Guard は可用性を高めるため、自動ブロック修復を行い、高可用性アップグレード（データベースのローリング・アップグレードの実装を容易にするための Oracle Database 12c の新しい自動化）を有効化します。

Oracle 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。スタンバイ・データベースが同じデータセンターに常駐しているか、同じ都市内の別のデータセンターに常駐している場合、Data Guard のロールのトランジションのために透過的にスタンバイにフェイルオーバーされるようアプリケーションを構成することができます。この Data Guard の構成は高い可用性、データ保護、一部の局所的な災害からの保護を備えています。スタンバイ・データベースが複数の地域をまたいで常駐している場合、この Data Guard の構成は通常、付加的なディザスタ・リカバリ保護を備えています。

Oracle 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 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。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・データベースを使用して、計画停電および計画外停電の間と停止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、予期しないシステム・コンポーネントまたはシステム全体の障害が含まれることがあります。

データベースのローリング・アップグレード – パッチ・バンドルが適用され、プライマリ Oracle Database Appliance の変更が行われている場合に、スタンバイ・データベースにより停止時間が最小限に抑えられます。パッチまたはその他のメンテナンスが最初にスタンバイ・データベースで適用されて検証され、その後本番のワークロードがプライマリ・システムからスタンバイ・システムに切り替えられます。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。

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・モードのままでプライマリ・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・データベースのブロックごとのコピーであるため、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。

» **ブロックの修復** – 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、ブロック破損シナリオからの自動復旧のために備えている能力です。プライマリ/スタンバイ構成では、破損ブロックを自動復旧し、この操作をエンドユーザーおよびデータベース管理者から完全に透過的に行うことができます。ブロック修復機能も Oracle Active Data Guard オプションの一部です。

» **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/書込み操作用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、http://www.oracle.com/technetwork/jp/database/features/availability/oracle-database-maa-best-practices-155386-ja.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。

» **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・インスタンスとスタンバイ・インスタンスを同様に構成** -- 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。再インストール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。

» **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・パスが不要な場合があります。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。また、リアルタイム適用を備えた Oracle Active Data Guard では自動ブロック破損の修復が可能です。

» **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・プラクティスが利用できる場合があります。

- Client Failover Best Practices for Data Guard 12c
- 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
- Best Practices for Synchronous Redo Transport - Data Guard and Active Data Guard
- Best Practices for Automatic Resolution of Outages to Resume Data Guard Zero Data Loss
- 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
- ブロック破損の防止、検出、および修復：Oracle Database 12c

上記のベスト・プラクティス、その他については、
http://www.oracle.com/technetwork/jp/database/features/availability/oracle-database-maa-best-practices-155386-ja.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。

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、このホワイト・ペーパーの付録 B に示します。

データベース・バージョン 12.1 および 12.2 - 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... 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 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、このホワイト・ペーパーの付録 A 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。

1. 現在、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、またはこのホワイト・ペーパーの付録に示した例に従って、11g および 12c 環境を構成することができます。
2. Oracle ACFS ストレージを使用している場合、スタンバイ・データベースをインストールする前に、スタンバイの Oracle Database Appliance システムのデータベース・ストレージを作成しておいてください。"oakcli create dbstorage"コマンドを root ユーザーで使用し、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、次のようにします。

   # oakcli create dbstorage -db stbydb

   このホワイト・ペーパー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。

3. 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。このホワイト・ペーパーで概要を示す 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 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、これらのシステムはシングル・ノード・システムであり、Oracle Real Application Clusters（Oracle RAC）を実行できるのは Oracle Database Appliance X6-2 HA ハードウェア・モデル上のみであるため、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 を最初に展開した瞬間から、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。このホワイト・ペーパーで概要を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、Database Configuration Assistance（DBCA）、RMAN、Oracle Data Guard などのツールを使用して自動で実行することができ、Oracle Database Appliance 上でスタンバイ・データベースを設定するのに使用できます。

付録A：Oracle Database Applianceでの12c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。

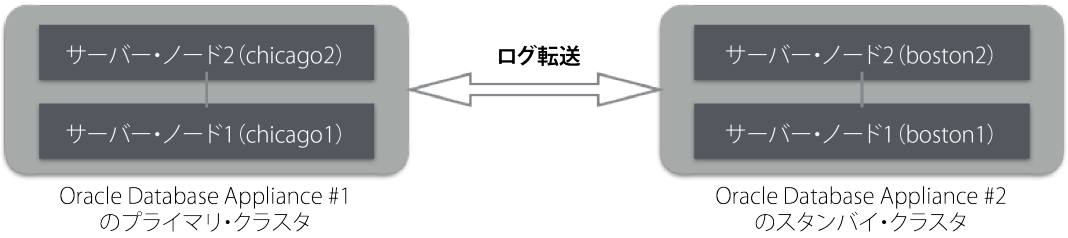

図1　Oracle Database ApplianceでのOracle RACの構成トポロジ

	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/12.1.0.2/grid		/u01/app/12.1.0.2/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	
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

プライマリ環境の構成

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/1024/1024/1024 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。

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 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 proddb
```

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 proddb
```

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. 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)
  )
)

on proddb1
chicago_local_listener =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = proddb1-vip.us.oracle.com)(PORT = 1521)))
on proddb2
chicago_local_listener =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = proddb2-vip.us.oracle.com)(PORT = 1521)))
```

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)
   )
)
```

on stbydb1

```
boston_local_listener =
   (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb1-vip.us.oracle.com)(PORT = 1521)))
```

on stbydb2

```
boston_local_listener =
   (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = stbydb2-vip.us.oracle.com)(PORT = 1521)))
```

7. 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;
```

8. 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**
*.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_unique_name=chicago *.remote_listener='proddb-scan:1521' *.local_listener='chicago_local_listener' *.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'	*.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_unique_name=boston *.remote_listener='stbydb-scan:1521' *.local_listener='boston_local_listener' *.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/chicago/adump *.fal_server='boston' *.remote_login_passwordfile='exclusive'	**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'
X5-2およびX6-2 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"	**X5-2およびX6-2 HAでのみ追加** *._cluster_flash_cache_slave_file=” *.db_flash_cache_file="/u02/app/oracle/oradata/flashdata/.ACFS/snaps/flashcache/boston/flash1"

9. 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

10. スタンバイでファイル・システム構造体を作成します。

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

11. 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: /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
```

12. 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/12.2.0.1/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ cp /u01/app/oracle/product/12.1.0.2/dbhome_1/dbs/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
```

13. 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、次のように指定します。control_files= '/u01/app/oracle/oradata/datastore/boston/BOSTON/controlfile/o1_mf_drw8zb81_.ctl'

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

14. 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/12.2.0.1/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;
```

15. 並列処理を有効化します。

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;
```

16. 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ネル間で分割し、各チャ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/1024/1024/1024 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、1 GB 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 1G;
RMAN> backup spfile;
```

17. スタンバイ側で管理リカバリを開始します。

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

18. 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/12.2.0.1/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/dat
astore' -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 RAC の例

```
[oracle@stbydb1]$ srvctl add database -db boston -oraclehome /u01/app/oracle/product/12.2.0.1/dbhome_1 -
dbtype RAC - dbname chicago -acfspath
'/u01/app/oracle/oradata/datastore,/u02/app/oracle/oradata/datastore,/u01/app/oracle/fast_recovery_area/da
tastore' -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 $ORACLE_HOME/dbs/initboston1.ora oracle@
stbydb2:/u01/app/oracle/product/12.2.0.1/dbhome_1/dbs/initboston2.ora
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston1 -node stbydb1
[oracle@stbydb1]$ srvctl add instance -database boston -instance boston2 -node stbydb2
```

19. 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/12.2.0.1/dbhome_1
[oracle@stbydb1]$ export ORACLE_SID=boston1
[oracle@stbydb1]$ export PATH=$ORACLE_HOME/bin:$PATH
[oracle@stbydb1]$ sqlplus / as sysdba
SQL> @PostCR.sql
```

20. 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;
```

21. 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
```

22. 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;
```

23. テストをフェイルオーバーします。

connect to standby before failover:

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

connect to former primary before failover:

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

付録B：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。

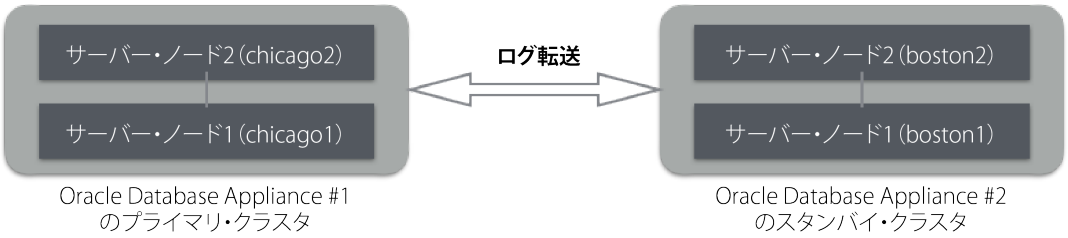

図2　Oracle Database ApplianceでのOracle RACの構成トポロジ

	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/12.1.0.2/grid		/u01/app/12.1.0.2/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1 - Oracle Databaseのネーミング規則の例

プライマリ環境の構成

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/1024/1024/1024 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。

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 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. 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='*';
```

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の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。ベア・メタル構成または仮想化プラットフォーム構成で Oracle Database Appliance を設定するには、http://docs.oracle.com/cd/E22693_01/doc.12/e55694.pdf で『Oracle Database Appliance Setup Poster』を参照してください。

10. 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)
  )
)
```

11. 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/12.1.0.2/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)))
```

12. 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
```

13. 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
```

14. パスワード・ファイルを作成します。

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リ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]$ orapwd file=/u01/app/oracle/product/11.2.0.4/dbhome_1/dbs/orapwboston
password=<primary sysdba passwd>
```

15. 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
```

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

    “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
    ```

17. 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
```

18. 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
```

スタンバイ・データベースのインスタンス化

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。

19. 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/onlinelog
[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/oradata/datastore/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
valid_for=(online_logfiles, primary_role) db_unique_name=chicago'
set fal_server='chicago'
NOFILENAMECHECK;
}
```

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
```

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。

20. 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。

21. 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
```

22. 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 C 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。

23. 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。Data Guard の REDO 転送のための専用ネットワーク・チャネルの設定に関する詳細は、MOS Note 1451810.1『Configuring Dedicated Disaster Recovery Network on Oracle Database Appliance』を参照してください。

24. 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#;
```

Data Guard Broker の構成の作成

この手順はオプションです。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。

25. 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)))
```

26. 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='*';
```

27. 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='*';
```

28. 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]$ dgmgrl
DGMGRL> connect sys/welcome1;
DGMGRL> CREATE CONFIGURATION 'ODADGConfig' AS PRIMARY DATABASE IS CHICAGO CONNECT IDENTIFIER is CHICAGO;
```

29. 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;
```

30. 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;
```

31. 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;
```

32. 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;
```

33. テストをフェイルオーバーします。

connect to standby before failover:

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

connect to former primary before failover:

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

付録C：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>stbydb</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。

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D：Oracle Data Guardを使用したデータベースのアップグレード

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

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（oakd など）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）。
3. スタンバイ ODA システムでサーバーおよび GI 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 システムのサーバーおよび GI 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・コンポーネントのみのアップグレード

サーバー・コンポーネント（OS、GI、一般的なファームウェア）とストレージ・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できます。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（oakd など）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（オプション。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。

参考資料

OTN 内の Oracle Database Appliance Web サイト

http://www.oracle.com/technetwork/server-storage/engineered-systems/database-appliance/index.html

Oracle Maximum Availability Architecture ベスト・プラクティス

http://www.oracle.com/technetwork/jp/database/features/availability/oracle-database-maa-best-practices-155386-ja.html

OTN 内の Oracle Real Application Clusters Web サイト

http://www.oracle.com/technetwork/jp/database/clustering/overview/index.html

OTN 内の Oracle Clusterware Web サイト

http://www.oracle.com/technetwork/jp/database/clusterware/overview/index.html

OTN 内の Oracle Data Guard Web サイト

https://www.oracle.com/technetwork/jp/database/options/active-data-guard/overview/index.html

Oracle Data Guard Concepts and Administration 12c Release 1

https://docs.oracle.com/database/122/SBYDB/title.htm

http://docs.oracle.com/database/121/SBYDB/toc.htm

My Oracle Support (MOS) Knowledge Content Notes

https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1075908.1

https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=2275154.1

OTN 内の Oracle Database High Availability Web サイト

http://www.oracle.com/technetwork/jp/database/features/availability/index.html

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

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

ORACLE

CONNECT WITH US

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

Integrated Cloud Applications & Platform Services

Copyright © 2017, 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。0615

Oracle Database Appliance：Oracle Data Guard を使用したディザスタ・リカバリ・ソリューションの実装　　日付：2017 年 8 月
著者：Ramasubramanian Athmanathan, Ravi Sharma, Krisztian Fekete, Sanjay Singh　　共著者：Oracle RAC Pack and MAA Team

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