Oracle Exadata Database Machine上のOracle PeopleSoft

Oracle Maximum Availability Architecture

ホワイト・ペーパー

2012年2月

Oracle Maximum Availability Architecture

高可用性に関するオラクルのベスト・プラクティス

ORACLE

概要

Oracle Exadata Database MachineにOracle PeopleSoftを展開すると、Oracle Maximum Availability Architecture（Oracle MAA）とパフォーマンス機能の両方の大きなメリットがあります。このホワイト・ペーパーでは、Exadata Database MachineへのOracle PeopleSoftの展開方法について、MAAのベスト・プラクティスを含めて説明します。また、Oracle PeopleSoft HR Payrollのバッチ・プロセスのパフォーマンスを中心に事例を紹介します。事例では、PeopleSoft（北米）のHR Payrollベンチマーク・キットを使用してワークロードを生成しました。

Oracle MAA [1]は、オラクルの高可用性テクノロジーを導入するためのベスト・プラクティス構想です。Oracle PeopleSoftをExadata Database Machineに展開して高可用性と耐災害性を最大化するベスト・プラクティスは、Oracle MAAチームによって文書化され検証されています。

はじめに

Oracle PeopleSoftのデータベースは、他のプラットフォームと同様にOracle Exadata Database Machine上で実行されます。ただし他のプラットフォームとは違い、Exadata Database Machineには企業にとって革新的な一連のテクノロジーが搭載されています。次の一連のテクノロジーによって、現在ITが直面している多くの課題に対応できます。

- Oracle Real Application Clusters（Oracle RAC）とインテリジェントなExadataストレージ・グリッドがInfiniBandネットワークによって統合されているので、全機能搭載型の事前検証済みクラスタリング・ソリューションを短期間（日数）で実現できます。
- Exadata Smart Flash Cacheは、データベース・サーバーからのI/Oリクエストのタイプを判断し、フラッシュ・キャッシュの利用効率を最適化します。たとえば、Oracle Recovery Manager（Oracle RMAN）バックアップでは、フラッシュ・キャッシュにデータを送らないため、OLTPアプリケーションへの影響を低減します。
- Exadata Smart Flash Cache内に使用頻度の高いデータ・ブロックをキャッシュすることにより、ランダムI/Oのボトルネックを解消します。このため、OLTPアプリケーションの読取りIO性能（IOPS）[1]を、20倍に増やすことができます。
- Exadataストレージ・セルとOracle RACの両方でInfiniBandの高帯域幅ネットワーク・ファブリック（40Gb/秒）が使用されるため、I/OおよびOracle RAC間のインターコネクトのレイテンシがそれぞれ短縮されます。
- OLTPと分析の両方のワークロードをより多くのメモリ、高性能なCPUとIO性能を保有した1台のExadataに統合することによりフロア設置スペースを節約でき、コストを削減できます。
- Exadata MAAには、自動的に構成展開できる検証済みのベスト・プラクティスがあります。これらはMAAのベスト・プラクティス『Oracle Exadata Database Machineを使用したOracle Maximum Availability Architectureの実現』で説明されています。

[1] 1秒あたりの入力/出力操作（IOPS）

このホワイト・ペーパーは、読者がOracle PeopleSoftの管理タスク（プロセス・スケジューラの構成など）に精通しており、Oracle PeopleSoft HR Payrollの機能に関する知識があることを前提としています。このホワイト・ペーパーで説明する一部の推奨事項を実行するには、Oracle Real Application Clusters（Oracle RAC）、Oracle Automatic Storage Manager（Oracle ASM）の基礎知識と、SQLの実用的な知識が必要です。このホワイト・ペーパーでは、ExadataでのPeopleSoft HR Payrollの実行を中心に、Exadataをさまざまな側面から説明します。このため、Exadataの主要コンポーネント（コンピュート・ノード、ストレージ・セル、InfiniBandネットワーク、Smart Flash Cacheなど）を理解しておくことは非常に有益です。

ベスト・プラクティスのサマリー

この項では、PeopleSoft HRをOracle Exadata Database Machineに実装するためのベスト・プラクティスのサマリーを説明します。

Exadata Database Machine

次の推奨事項には、より詳細な情報を提供するため、My Oracle Support IDや他のMAAホワイト・ペーパーへのリンクが含まれます。

- Exadataヘルス・チェックの実行については、My Oracle Support ID 1070954.1に従ってください。
- Exadata Database MachineおよびExadata Storage Server 11g Release 2のサポート対象バージョンで推奨されるソフトウェアについては、My Oracle Support ID 888828.1を確認してください。
- Exadata Database Machineの監視とAutomatic Service Request（ASR）については、My Oracle Support ID 1110675.1を確認してそれに従ってください。
- Exadataのテストとパッチ適用のベスト・プラクティスについては、My Oracle Support ID 1262380.1を参照してください。
- MAAのホワイト・ペーパー『Oracle Exadata Database Machineを使用したOracle Maximum Availability Architectureの実現』の推奨事項を確認してそれに従ってください。上記のホワイト・ペーパーには、このホワイト・ペーパーに含まれないその他のMAAトピックが含まれます。

PeopleSoft

次の推奨事項は、このホワイト・ペーパーで後述する内容の詳細な説明のリンクです。

- 各データベース・マシンのコンピュート・ノードでLinux HugePagesを構成します。詳細は、My Oracle Support ID 744769.1を参照してください。HugePagesを構成すると、サイズの大きいデータベースSGAがスワッピングの影響を受けなくなります。
- RAC対応データベースを新しく作成するか、既存のPeopleSoftデータベースを移行します。このホワイト・ペーパーの、"Exadata Database MachineへのPeopleSoftデータベースのインストールまたは移行"の項の手順に従ってください。
- MAAのホワイト・ペーパー『Deploying an Oracle PeopleSoft Maximum Availability Architecture』のベスト・プラクティスを使用して、PeopleSoft環境を実装します。
- このホワイト・ペーパーの項"表と索引のパーティション化"の説明に従って、特定のPeopleSoftアプリケーションに適した表と索引のパーティション化を使用します。
- このホワイト・ペーパーの項"複数のpayroll実行制御によるワークロードの分散"の説明に従って、PeopleSoftプロセス・スケジューラ内の複数の実行制御を定義します。
- このホワイト・ペーパーの項"Oracle RACインスタンス間でのロードバランシングの使用"の説明に従って、クライアントTNSのロードバランシングを構成してOracle RACを利用します。
- このホワイト・ペーパーの項"PeopleSoftプロセス・スケジューラの配置"の説明に従って、十分なCPUリソースがあり、データベース・マシンとのネットワーク接続のレイテンシが短い個別のサーバーにプロセス・スケジューラを配置します。

Exadata Database Machine上のOracle PeopleSoft MAA

図1に、高可用性、スケールアップ/スケールアウト、および災害復旧用のサイトを示します。

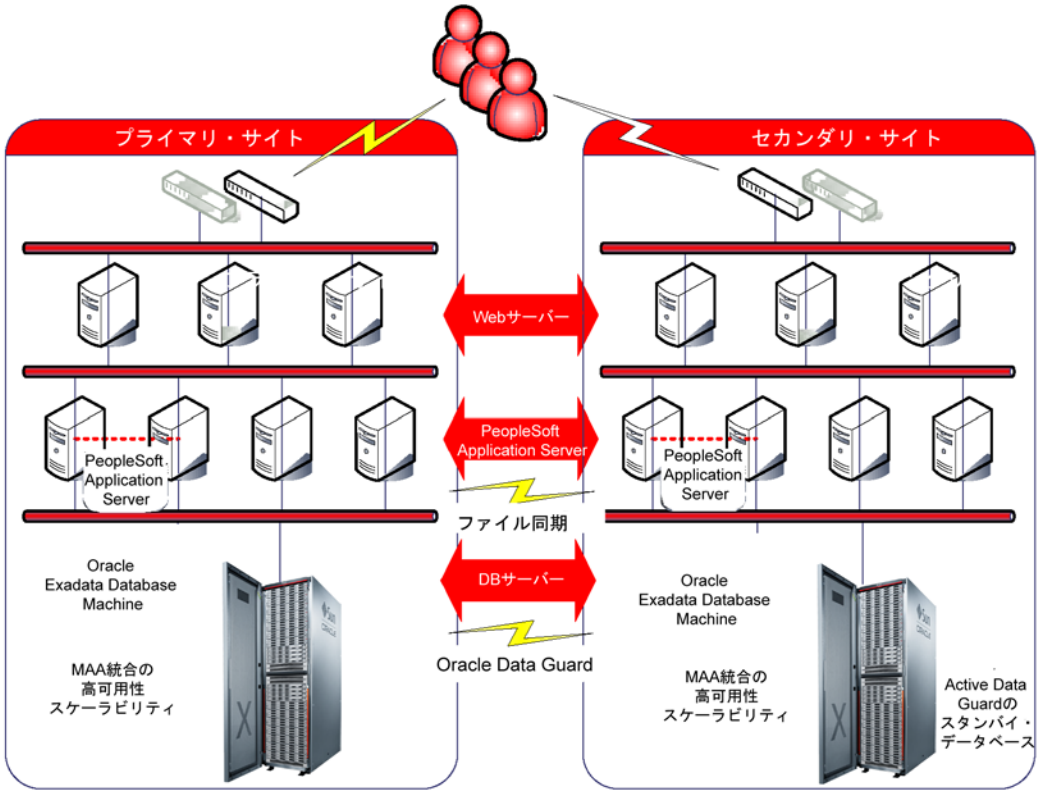

図1：Oracle PeopleSoftの高可用性構成とディザスタ・リカバリ・サイト

ホワイト・ペーパー『Deploying an Oracle PeopleSoft Maximum Availability Architecture』に記載されているMAAのベスト・プラクティスはすべて、Exadata Database Machineで実行されるPeopleSoftアプリケーションに適用できます。これらのMAAのベスト・プラクティスの一部は、データベース・マシンに既に実装済みです。具体的には、Oracle Grid Infrastructure、Oracle RAC、およびOracle ASMの利用です。Oracle Data Guardを実装すると、ディザスタ・リカバリのセカンダリ・サイトでデータを保護できます。またData Guardをローカルで実装することもでき、データベースやExadata Database Machine全体で障害が発生した場合、迅速にフェイルオーバーできます。Oracle Active Data Guardには、アプリケーションのダウンタイムなしでブロック破損を自動的に修復する自動ブロック修復機能があります。これは、アプリケーションに対して透過的です。

またリリース8.51以降、Oracle PeopleSoft PeopleToolsのサポートには次の機能が含まれます。

- Oracle Active Data Guardにより、Active Data Guardデータベース・インスタンスにレポート処理がオフロードされます。Exadata Database MachineでOracle Active Data Guardデータベースを利用すると、その問合せやレポートでセル・オフロード、セル・ストレージ索引、Smart Scan、およびその他の問合せの最適化によるあらゆるメリットを享受しながら、災害復旧用データベースとしても使用できます。
- Fast Application Notification（FAN）（リリース8.5.0.9以降）と透過的アプリケーション・フェイルオーバー（TAF）により、プライマリ・データベース・サービスが失われた場合のクライアント・フェイルオーバーが高速化されます。Oracle PeopleSoft MAAの詳細は、MAAのホワイト・ペーパー『Deploying an Oracle PeopleSoft Maximum Availability Architecture』を参照してください。

PeopleSoftをExadata Database Machineで使用開始するには

PeopleSoftデータベースをExadata Database Machineで使用開始するには、Exadata Database Machineにデータベースを新規インストールするか、既存のPeopleSoftデータベースをExadata Database Machineに移行します。

最低限、次のソフトウェア・バージョンを使用する必要があります。

- Oracle Database 11g Release 2（11.2.0.1）以降
- PeopleToolsバージョン8.49以降

  Oracle Database 11g Release 2（11.2）では、PeopleToolsバージョン8.49以降がサポートされます。PeopleSoftアプリケーションのバージョンは、実装されているPeopleToolsのバージョンに依存します。実際にはPeopleTools 8.51以上の実装を推奨します。FANやクライアント・フェイルオーバーなどのMAA機能のほか、Active Data Guardもサポートしているためです。サポートされているPeopleToolsの最小バージョンを確認するには、PeopleSoftアプリケーションのインストールに関するPeopleBookを参照してください。

新しいデータベースのインストール

Exadata Database MachineでOracle PeopleSoftのインストールを実行するには、PeopleToolsソフトウェア・パッケージで提供される標準ツール（DataMoverやApplication Designerなど）を使用します。Oracleデータベースを構築し、PeopleSoftスキーマをインストールするためのツールと方法は、他のすべてのOracleプラットフォームと同じです。

Exadataでは、まずDBCAを使用してRAC対応の空のデータベースを作成し、次にPeopleToolsツール・セットを使用してPeopleToolsスキーマとアプリケーション・スキーマのインストールを完了することを推奨します。通常、PeopleTools設定スクリプトの表領域作成のSQL文内ではファイル・システムのパス名が使用されています。これらのスクリプトの編集して、ファイルシステムの代わりにOracle ASMを使用するようにスクリプトを変更する必要があります。

注：PeopleTools、Application Server、およびプロセス・スケジューラをExadata Database Machineに直接インストールしないでください。詳細は、このホワイト・ペーパーの後の項"PeopleSoftプロセス・スケジューラの配置"を参照してください。

設定とインストールの詳細は、PeopleToolsのPeopleBook [5]のドキュメントを参照してください。

既存のデータベースの移行

既存のPeopleSoftデータベースを移行する場合、移行するデータベースのソース・プラットフォームによっていくつかの方法があります。これらの方法については、MAAのプレゼンテーション『Best Practices for Migrating to Exadata Database Machine』を参照してください。本番システムで移行を実施する前に問題なくデータベース移行できることを確認しておくことが重要です。

既存のPeopleSoftアプリケーションでリリース8.49より前のバージョンのPeopleToolsが使用されている場合は、データベースも古いリリース・バージョン上にあります。その場合、次のオプションがあります。

- ソース・システム上のPeopleToolsをリリース8.49以降にアップグレードします。そして『Best Practices for Migrating to Exadata Database Machine』内で説明されている移行方法の1つを使用して、Exadataに移行します。たとえば、トランスポータブル表領域の移行方法を使用できます。
- ソース・システムでPeopleToolsをリリース8.49以降にアップグレードします。次に、Oracle Data Pumpを使用してExadata Database Machine上の新しいデータベースにデータを移行します。

事例：Exadata Database Machine上のPeopleSoft HR Payroll

この項では、PeopleSoft HR payrollのバッチ処理の事例について、Exadata Database Machine上のpayrollのPaySheet、PayCalc、およびPayConfirmフェーズの最適化を中心に説明します。このホワイト・ペーパーで説明しない、主要なpayrollプロセス・フェーズもあります。この事例は、実在するほとんどのPeopleSoft環境に適用できるベスト・プラクティスを作り出すための、継続的なMAAプロジェクトです。

HR payrollは、従業員のさまざまな福利厚生の変更、ビジネス・ガバナンス、法的要件、年間の支払いサイクルごとに、支払いの再計算が必要になる反復的な処理です。国によっては、法律、税金、および支払い計算の要件が非常に複雑な場合があります。このため、支払い計算フェーズを最終的に実行する前に、数回のやり直しが必要な場合があります。

payrollは反復的な処理であるため、各支払いサイクルを完了するには、かなり時間がかかります。このホワイト・ペーパーでは、Exadata Database Machineでpayrollを実行する際に、スケーラビリティの実現と全体の処理時間を短縮するための、特定のアプローチについて説明します。

テスト環境の構成

Exadata Database Machineのテスト環境の構成は次のとおりです。

サーバー - クォーターラックのOracle Exadata X2-2 Database Machine

- コンピュート・ノード（SunFire X4170）×2
  - 2ソケット、4コア（対称型マルチスレッディング（SMT）対応）、コンピュート・ノードあたり16基の論理CPU
  - コンピュート・ノードあたり72GBの物理RAM
- ストレージ・セル（X4275）×3
  - 2ソケット、4コア（SMT対応）、ストレージ・セルあたり16基の論理CPU
  - ストレージ・セルあたり24GBの物理RAM
  - ストレージ・セルあたり12台のSATAディスク
  - ストレージ・セルあたり384GBのSmart Flash Cache

ソフトウェア

- Oracle Database 11g Release 2（11.2.0.2）
- Oracle PeopleSoft PeopleTools 8.49

  32ビットのOracleクライアントが必要（11.2.0.2）
- Oracle PeopleSoft HCM 9.0

テスト環境の構成

- 両方のOracle RACインスタンスで、データベース・パラメータBUFFER_CACHE_SIZEを32GBに設定
- 両方のOracle RACインスタンスで、データベース・パラメータSGA_TARGETを37GBに設定
- データベース・マシンの両方のコンピュート・ノードで、HugePagesを42GBに設定

ワークロード・プロファイル

この事例で使用するワークロードは、PeopleSoft（北米）のHR Payrollベンチマーク・キットをベースにしていますが、この事例はベンチマークではありません。この事例は、最適なパフォーマンスとスケーラビリティを実現するベスト・プラクティスを引き出すための、MAAの演習環境として使用します。

payrollのバッチ・ワークロード部分だけを、オンライン・ユーザーなしで実行しました。次のバッチ・プロセスを実行しました。

- PaySheet：従業員のpayrollデータ・ワークシートを生成するCOBOLプロセス
- PayCalc：PaySheetを参照して、これらの従業員の賃金を計算するCOBOLプロセス
- PayConfirm：Payroll Calculationで生成される情報を取得し、計算額に基づき従業員の差引合計額を更新するCOBOLプロセス

これらのバッチ・プロセスは、年間支払いサイクルの最終月（12月）にpayrollを初めて実行したかのように実行されました。実際にpayrollを初めて実行すると、他の人事処理を除いて一番処理時間がかかります。この事例では、このpayrollの実行でPayCalcプロセスを反復的に実行していません。PeopleSoft（北米）のPayrollキットでは、1つの会社内で128個の支払いグループを定義した500,000人の従業員が含まれており、11ヶ月の支払い履歴がデータに存在しています。

このテスト環境では、HR payrollをSINGLE CHECK = "NO"の設定で構成したため、PayCalcバッチ・プロセスで、複数の実行制御をパラレルで実行しました。

すべてのフェーズのpayrollワークロードで次の共通特性が設定されています。

- スイープ・スタイル処理。この処理では、データの再読取りがほとんどなしで、プロセスのすべての従業員がスイープされます。
- 1行毎に対して単一の行処理（フェッチ、挿入、更新）
- 索引スキャンによる高速な小規模I/O
- 索引を利用した処理 - 全表スキャンなし
- payrollのCOBOLプロセスとデータベース・サーバー間の高いネットワーク間の往復通信
- COBOLプロセスによる、中程度のCPU使用率でのビジネス・ロジックの実行

次の項では、Exadata Database Machineに実装できるベスト・プラクティスの推奨事項を説明します。またExadataによって、Oracle PeopleSoftのパフォーマンスがより向上します。

ExadataでのHR Payrollのスケーリング

この事例は、Exadataで実行されるPeopleSoft（北米）のHR Payrollを中心に説明します。他のPeopleSoftアプリケーション・スイート（Financials、CRM、EPMなど）は考慮しません。以下の事例の推奨事項から次の点を理解できます。

- PeopleSoft HR PayrollのOracle RACでのスケーラビリティ
- PeopleSoft HR PayrollではOracle RACインスタンス・アフィニティが不要である
- データのパーティション化とワークロードの再分散による、Exadataでの優れたスケーラビリティ

Exadata Database Machineの拡張性で重要な点は、データベース・マシン上に事前構成されているOracle RACを利用することです。Oracle RACインスタンス間で、表や索引ブロックのインスタンス間競合を最小限にすることが重要です。大規模な単一の表と索引では、複数のOracle RACノードで実行される多くのプロセスで、表の索引リーフ・ブロックやデータ・ブロックが競合し、クラスタ全体の競合の可能性が高くなります。

このため、ほとんどのERPバッチ処理は複数のOracle RACインスタンス間では実行されません。Oracle RACデータベースで実行される多くのアプリケーションでは、バッチ・ワークロードを特定のOracle RACインスタンスに割り当てることで、クラスタ競合を最小限に抑えています。このために、いくつかの固有のカスタマイズが必要です。

この事例の目標の1つは、特定のOracle RACインスタンスにワークロード・アフィニティを必要要件にすることなく、HR payrollをスケーリングすることでした。バッチ処理は、一貫したパフォーマンスで複数のOracle RACインスタンス間で実行できる必要があります。この目標を達成するために3つの主要なチューニングを実行しました。

- 表と索引のパーティション化
- 複数のpayroll実行制御によるワークロードの再分散
- Oracle RACインスタンス間でのロードバランシングの構成

これらのトピックは、次の各項でベスト・プラクティスとして説明します。

表と索引のパーティション化

パーティション化された表と索引を利用して、特定のワークロード・プロセスを（パーティション・キーを使用して）特定のパーティションに対して動作させることで、payrollバッチ・プロセスの優れたスケーラビリティ性能が実現されます。payrollは支払いグループごとに処理され、各payrollバッチ・プロセスが1つまたは複数の支払いグループに割り当てられます。このpayrollテストには1つの会社と128個の支払いグループしか含まれないため、PAYGROUP列をパーティション・キーとして支払いグループごとにレンジパーティション化することは理にかなっています。各支払いグループはそれ自体のパーティションに配置されるため、各表のパーティション数は128個になりました。この事例では、すべてのpayroll表ではなく、インストール・スクリプトによって提供される表だけをパーティション化しました。

アプリケーションの実行中に表や索引をパーティション化することはできますが、ローカルのパーティション索引として、索引をリビルドすることになります。このため、表と索引をパーティション化する間は、アプリケーションをオフラインにすることを推奨します。

パーティションの構築手順の概要は次のとおりです。

1 パーティション化していない現在の表と同じ定義（同じ列）のシャドウ表を1つのパーティションで作成します

2 現在の表を新しくパーティション化した表と置き換えます

3 データを最終形のパーティションに分割します

4 ローカルのパーティション索引を作成します

5 表のグローバル索引をリビルドします（存在する場合）

以下の例は、当社のMAA事例に固有のものです。ただしパーティション化の構造定義はPAYGROUP列の値の範囲の知識が必要です。もし企業が複数の会社で構成されている場合は、最適なパーティション方法を決定するには、まずCOMPANY、次にPAYGROUPによるパーティション化が必要かもしれません。ただ各パーティションに、データが均一に分散するわけではありません。payroll表をパーティション化する方法と最適な表設計はこの記事の範囲外であり、PeopleSoft HR Payroll機能の専門家による分析が必要です。

たとえば、PS_JOB表をパーティション化する場合を考えてみます。表が作成されて、既にデータが投入済みであり、パーティション化が必要であるとします。この表を128個のパーティションにパーティション化する（支払いグループあたり1つのパーティション）には、次の手順を実行します。

手順1： PS_JOB_PART表のシャドウ・バージョンの表をパーティション１つで作成する

この表の列は、PS_JOB表と同じである必要があります。この表の作成で使用するDDLの抜粋は次のとおりです。

```
CREATE TABLE PS_JOB_PART (
EMPLID VARCHAR2(11) NOT NULL,
EMPL_RCD SMALLINT NOT NULL,
EFFDT DATE NOT NULL,
EFFSEQ SMALLINT NOT NULL,
PER_ORG VARCHAR2(3) NOT NULL,
…
AUTO_END_FLG VARCHAR2(1) NOT NULL,
LASTUPDDTTM DATE,
LASTUPDOPRID VARCHAR2(30) NOT NULL
)
PARTITION BY RANGE (PAYGROUP)
(
PARTITION JOB_P128 VALUES LESS THAN (MAXVALUE) TABLESPACE PSTABLE)
/
```

手順2：EXCHANGE PARTITION句を使用して、元のPS_JOB表と交換する

```
alter table PS_JOB_PART
exchange partition JOB_P128
with table PS_JOB
/
```

手順2では、既存のPS_JOB表のセグメントとシャドウ表の1つのパーティションを変換します。このパーティションは、手順3で残りのすべてのパーティションに分割されます。

手順3：パーティションを支払いグループに分割する

次のパーティション分割SQL文のパーティション・キーの値は、特定の支払いグループ（102、103、430など）です。各支払いグループは、それ自体のパーティション内に重複なしで配置する必要があります。このため、支払いグループの境界にレンジ・キーを指定することが重要です。既存のいずれかの支払いグループがマージまたは分割されると、その新しい境界でパーティションがマージまたは分割されます。新しい支払いグループの追加は、（可能な場合は）最後のパーティション（MAXVALUE）を分割することで実行できます。既存の支払いグループ間の値を使用して新しい支払いグループを追加する場合は、既存のパーティションを新しいパーティションにマージして再分割する必要があります。N+1個のパーティションが必要です。Nは支払いグループの数です。

```
ALTER TABLE PS_JOB_PART split partition JOB_P128 at (102) into ( partition
JOB_P00 tablespace pstable, partition JOB_P128 ) parallel 8;

ALTER TABLE PS_JOB_PART split partition JOB_P128 at (103) into ( partition
JOB_P01 tablespace pstable, partition JOB_P128 ) parallel 8;

…

ALTER TABLE PS_JOB_PART split partition JOB_P128 at (430) into ( partition
JOB_P126 tablespace pstable, partition JOB_P128 ) parallel 8;

ALTER TABLE PS_JOB_PART split partition JOB_P128 at (431) into ( partition
JOB
```

SQL文のPARALLEL句の使用に注目してください。この句によって、パーティション分割操作がスケーリングされ、はるかに短時間で完了できます。パラレル処理では、Exadata Database Machineで広いI/O帯域幅とInfiniBandネットワークが使用されるため、スケーラビリティに優れています。

手順4：ローカルのパーティション索引の構築

ローカル・パーティションを使用して、PS_JOB表に索引を構築します。次に例を示します。

```
CREATE INDEX PS0JOB ON PS_JOB (PER_ORG,
EMPLID,
EMPL_RCD,
EFFDT DESC,
EFFSEQ DESC)
LOCAL (
PARTITION JOB_P00 TABLESPACE PSINDEX,
PARTITION JOB_P01 TABLESPACE PSINDEX,
PARTITION JOB_P02 TABLESPACE PSINDEX,
…
PARTITION JOB_P127 TABLESPACE PSINDEX,
PARTITION JOB_P128 TABLESPACE PSINDEX
)
PCTFREE 10 PARALLEL NOLOGGING
/
ALTER INDEX PS0JOB NOPARALLEL LOGGING
/
```

手順5：グローバル索引の構築

注：必要な索引がローカルのパーティション索引として適切でない場合にのみ、グローバル索引を構築します。

PS_JOBでは、PS_JOBの一意索引がグローバル索引です。次に例を示します。

```
CREATE UNIQUE INDEX PS_JOB ON PS_JOB (EMPLID,
  EMPL_RCD,
  EFFDT DESC,
  EFFSEQ DESC)
TABLESPACE PSINDEX
PCTFREE 10
PARALLEL NOLOGGING
/
ALTER INDEX PS_JOB NOPARALLEL LOGGING
/
```

上記の手順を実行して表をパーティション化した場合、Exadata Database Machineでは（数時間ではなく）数10分しかかかりません。当社のテストでは、9個のpayroll表のパーティション化（すべての索引ビルドを含む）を300GBのデータベースで、開始から終了まで40分かかりませんでした。これら9個の表には、payrollデータの大部分が含まれていました。

複数のpayroll実行制御によるワークロードの分散

データのパーティション化後、適切なスケーラビリティ性能を実現するには、ワークロードをできるだけ均等に分散することが重要です。簡単なラウンドロビン法でpayrollプロセスを支払いグループに割り当てた場合、プロセスによっては、すぐに終了したり、完了に時間がかかったりする場合があります。

パーティション化する手法によって、支払いグループのデータが特定のパーティションに分割されます。payrollデータはすべての支払いグループ（パーティション）に均一に分散されるわけではないため、スキームを考案してワークロードを均等に分散する必要があります。この操作により、スケーリングとスループットが向上します。

この事例では、次の範囲の4タイプの支払いグループがあります。

- 101～132には、この範囲内の10,948個のジョブがあります。
- 201～232には、この範囲内の6,256個のジョブがあります。
- 301～332には、この範囲内の3,128個のジョブがあります。
- 401～432には、この範囲内の3,128個のジョブがあります。

一番上の例では、32個の支払いグループにわたる100シリーズの支払いグループ・タイプに10,948個のジョブが含まれます。この範囲には、実質的には200、300、400の支払いグループ・シリーズより多くのジョブが含まれ、均等に分散されていません。PeopleSoft内で実行指示を定義し、複数のpayroll作業ストリーム間の作業を分散できます。ワークロードを均等に分散するために64個の実行指示を定義しました。次に、これらを2つのグループに分けました。

- 実行指示ストリーム1～32は、100シリーズ（101～132）のすべての支払いグループに割り当てました。
- 実行指示ストリーム33～64は、残りの200、300、および400シリーズの支払いグループに割り当てました。

最初のグループでは実行指示1に支払いグループ101を、2番目のグループでは実行指示ストリーム33を3つの支払いグループ（201、301、および401）に割り当てました。1つの実行指示が、他の実行指示と支払いグループを共有することはありません。定義した実行指示は、上記のパーティション化スキームと一致します。このスキームによる主要な利点は次の2つです。

- 各payrollプロセスが、一連のパーティションで排他的に動作する
- 各payrollプロセスを、アフィニティなしで所定のOracle RACインスタンスで実行できる

これらの実行制御は、payrollのすべてのフェーズ（PaySheet、PayCalc、およびPayConfirm）で定義する必要があります。

PayCalcと実行制御

PayCalcプロセスでは、同じ組織内の複数のジョブで実行される全従業員の給与計算範囲を単一のチェック処理（シングルチェック）で実行できます。これは、次のようにpayrollのパフォーマンスに影響します。

- PayCalcでシングルチェックを実行すると、それ以降の処理は順次実行されます。実行指示は、PayCalcでは1つずつ順番に実行されます。
- PayCalcで個々のチェックを計算する場合は、パラレルで実行できます。PayCalcでは、すべての実行指示を同時に開始できます。これは高いスケーラビリティを実現し、実行経過時間を大幅に減らすのに適した方法です。

PayCalcで、複数の実行制御によるパラレル処理を実行できるようにするには、Payrollを設定する際に、HR Payrollで"`SingleCheck`"を"`NO`"に設定して実行する必要があります。

SINGLE_CHECK機能の詳細は、PeopleSoft Enterprise Payroll（北米）のPeopleBook（http://www.oracle.com/pls/psft/homepage）を参照してください。

PeopleSoft HRMSバージョンを実装済みの場合は、特定のPeopleBook： "Handling Employees with Multiple Jobs in the same Organization"を見つける必要があります。

Oracle RACインスタンス間でのロードバランシングの使用

Oracle RACインスタンス間でスケーリングするには、TNSエイリアス接続文字列で`LOAD_BALANCE = YES`と指定する必要があります。Exadata Database Machineでは、TNSエイリアス接続文字列のHOSTパラメータでSingle Client Access Name（SCAN）[2]を指定する必要があります。

SCANを使用するには、PeopleToolsでOracle Database 11g Release 2（11.2）以降のデータベース・クライアント・ソフトウェアのインストールを使用する必要があります。SCANを使用すると、Exadata上のGrid Infrastructureによって、接続記述子で指定したサービスをサポートする、すべてのOracle RACインスタンス間で接続が分散されます。

次に、TNS接続文字列の例を示します。

```
PSFT =
   (DESCRIPTION =
   (SDU=32767)
   (ADDRESS = (PROTOCOL = TCP)(HOST = sclcz-scan2.us.oracle.com)(PORT = 1521))
   (LOAD_BALANCE = yes)
  (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = PSFT)
   )
```

[2] SCANの詳細は、ホワイト・ペーパー『Oracle Real Application Clusters 11g Release 2 Overview of SCAN』を参照してください。

)

パーティション化、ワークロード分散、およびロードバランシングの結果

オラクル社によるOracle Exadata Database Machineの2ノードのOracle RACクラスタによる、上位待機イベントの割合の分布を以下の表に示します。

待機イベントの分散			
待機イベントの割合	PAYSHEET	PAYCALC	PAYCONFIRM
DB CPUの割合	50.97	44.44	68.54
I/Oの割合	36.88	47.48	24.02
RACクラスタの割合	13.15	8.54	8.30

待機時間はほぼDB CPUかI/Oで発生するため、Oracle RACインスタンス間でワークロードをスケーリングし、より生産性の高い作業を実行できます。payrollプロセスの3つのフェーズすべてで、クラスタ待機イベントが最小になります。このためには、どの実行指示間でも支払いグループが重複しないようにします。このOracle RACインスタンス間のロードバランシングによって、各payrollプロセスをどのOracle RACインスタンスで実行するかに関係なくスケーリングが可能です。

Exadata Smart Flash Cache

Exadata Database Machine X2-2およびX2-8には、各ストレージ・セルに384GBのSmart Flash Cacheを搭載しています。フラッシュ・キャッシュの総量は、Exadata Database Machineラック内のストレージ・セルの数によって異なります。この事例で使用されているクォーターラックには、3つのストレージ・セルに1.01TBのフラッシュ・キャッシュが含まれます。

Exadata Smart Flash CacheはI/Oリクエストを監視し、フラッシュ・キャッシュの利用を高度に最適化します。たとえば、使用頻度の高いデータ・ブロックをキャッシングすることにより、読取りIO性能を向上し、ランダムI/Oのボトルネックを解消できます。I/Oのタイプをインテリジェントに管理してフラッシュ・キャッシュが枯渇することを防ぎます。たとえば、Oracle RMANでバックアップの実行中にフラッシュ・キャッシュ全体を使用して、OLTPアプリケーションに影響を与えるようなことがありません。

通常、SANおよびNASのストレージ・アレイのI/O待機時間（それぞれSCSI、ファイバまたはGigEを前提とする）は、平均で6～8ミリ秒です。I/Oパフォーマンスは、2つの要素によって決まります。IOPS（1秒あたりのI/O操作）とI/O帯域幅です。テストの結果、PeopleSoft payrollがストレージ・セルのExadata Smart Flash Cacheの恩恵を受けていることが分かりました。PeopleSoftでは、Exadata Storageセル上のSmart Flash Cacheの広いI/O帯域幅と高性能なIOPSが利用できます。

フラッシュ・キャッシュを明示的に使用するのに、特別なチューニングやアプリケーション変更は不要です。

フラッシュ・キャッシュの結果

フラッシュ・キャッシュを使用したpayrollテストの結果、I/O待機時間が30%のヒット率で、平均で7ミリ秒から3ミリ秒間の短縮が見受けられました。これにより、全体的なスループットが向上し、処理実行時間も15%短縮されました。HR payrollはスイープ・スタイルのワークロードであるため、表の再読取りはほとんどなく、ほとんどのデータ処理は1回です。

アプリケーション・パフォーマンスはさまざまですが、Exadata Smart Flash Cacheにより、全体のパフォーマンスが向上していることが分かります。

PeopleSoftプロセス・スケジューラの配置

テストの結果、PayCalcとPayConfirmの同時32個のプロセスによって、8コア・サーバーのCPUリソースの25～40%が消費されることが分かりました。これは、payrollを実行するほとんどのビジネス・ロジックがCOBOLアプリケーション・プロセスに含まれるためです。

データベース・マシンは、データベースのパフォーマンスとスケーラビリティ性能が最高になるようチューニングされています。データベース・マシンのコンピュート・ノードにアプリケーション・コンポーネントをインストールすると、CPUリソースの競合につながり、高いスケーラビリティを達成しようとするとスループットが低下します。データベース・サーバー・インスタンスとPeopleSoft payroll COBOLプロセス間で両方をスケーリングする場合、CPUリソースのバランスを取ることが重要になります。

すべてのアプリケーション・コンポーネント（アプリケーション・サーバーやプロセス・スケジューラ）を、Exadata以外の個別の中間層サーバーに配置することを推奨します。次に、パフォーマンスに影響する可能性があるネットワークレイテンシを解消する方法を説明します。

前述のとおり、payrollプロセスのほとんどのビジネス・ロジックは、アプリケーション層のCOBOLプロセスにあります。この設計によって、データベースとやり取りされるネットワーク通信のラウンドドリップ量が増えます。

次のような場合、パフォーマンス上の問題が発生している可能性があります。

- データベースのアイドル待機イベント時間が非常に長くなる場合（AWRレポート内で数千秒）。これは、クライアントへのSQL*Netメッセージと、クライアントからのSQL*Netメッセージのの待機イベントの"合計待機時間（秒）"を合算したものです。

  この場合、バッチ・プロセスによって、CPUリソースの競合、ネットワーク帯域幅とレイテンシ制約、またはその両方が原因でパフォーマンス・ボトルネックが発生している可能性があります。

- "SQL*Net round trip"と"SQL*Net message from / to client"の待機イベントで記録されているラウンド・トリップあたりの待機時間が「0.数ミリ秒（0.3ミリ秒以上）」で返される場合。

この事例は、次の構成でテストしました。

- 平均0.264ミリ秒の待機時間で、GigE経由で接続されている個別の中間層に、プロセス・スケジューラを配置
- 平均0.121ミリ秒の待機時間で、GigE経由で接続されている個別の中間層に、プロセス・スケジューラを配置

プロセス・スケジューラ専用サーバー上の実行結果

プロセス・スケジューラを個別のサーバーに配置する場合は、COBOLプロセスのワークロードをサポートするのに十分なCPUリソースが必要です。また、これらのバッチ・プロセスのネットワーク・リンクが、スループットとパフォーマンスで重要な役割を果たします。前述のとおり、これらのCOBOLプロセスでは非常に頻繁に通信が行われます。したがってネットワークの待機時間が長くなると、処理の経過時間が長くなります。専用サーバー上のMAAテストは、2つのネットワーク・トポロジ（ExadataサーバーからのGigEの2ホップ（0.264ミリ秒のネットワーク待機時間）と、Exadataサーバーからの GigEのシングルホップ（0.121ミリ秒のネットワーク待機時間））で構成しました。

図2と図3の円グラフは、ネットワーク待機時間がpayroll処理に与える影響を示しています。この円グラフでは、2つのネットワーク構成間で、クライアントでかかった時間の割合の違いを比較しています。

図2の円グラフは、2ホップのネットワークを示しています。ここでは待機時間が0.264ミリ秒で、クライアントでの合計経過時間のうちの43%です。

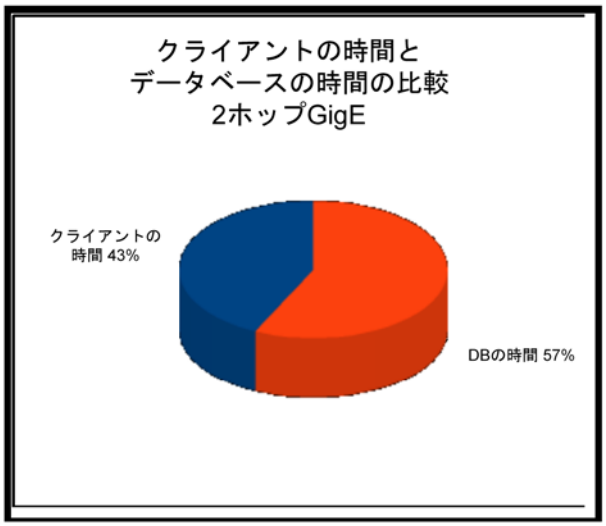

図2：Exadata Database Machineコンピュート・ノードのCPU統計

図3の円グラフは、シングルホップ・ネットワークで構成した場合（待機時間が0.121ミリ秒）、クライアントの時間が50%超も大幅に減少することを示しています。クライアントには全体の時間の27%しか使用されておらず、残りの73%はデータベースに使用されています。

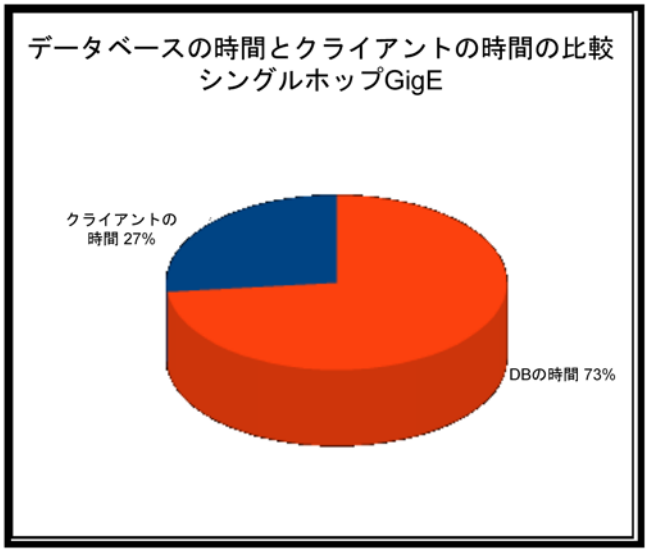

図3：Exadata Database Machineコンピュート・ノードのCPU統計

図4では、これを棒グラフで示しています。この棒グラフは、各ネットワーク・トポロジのクライアント側の時間（秒単位）を比較したものです。両方で、同じワークロードを使用しました。同じ32個の同時payrollストリームで、500,000人の従業員を処理しています。両方のテストで、"DB時間"は同じです。高レイテンシのネットワークでは、ネットワークの待機時間が長くなります。棒グラフによると、0.264ミリ秒のネットワークでは、クライアント側でかかった時間は約77,000秒でした（32ストリームすべての集計）。待機時間が0.121ミリ秒のネットワークでは、この時間が約37,000秒となり、ネットワーク待機時間が50%短縮されたことになります。

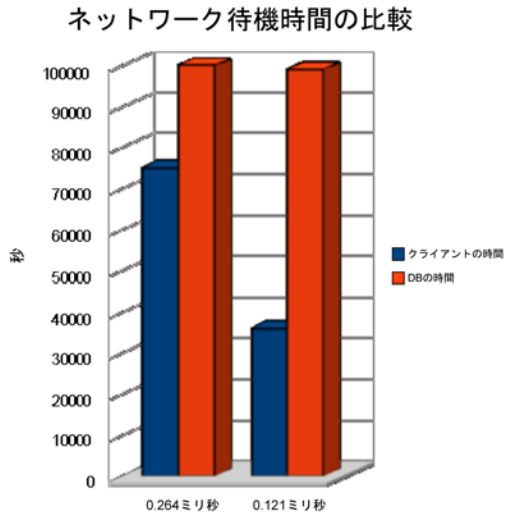

図4：Exadata Database Machineコンピュート・ノードのCPU統計

これらのテストに基づき、プロセス・スケジューラを待機時間の短いネットワークで専用のサーバーに配置することを推奨します。これにより、payroll処理のスケールアップとスケールアウトが可能です。

PeopleSoftでは、複数のプロセス・スケジューラを各スケジューラ専用のサーバー上に柔軟に構成できます。この場合、各プロセス・スケジューラはスケジューラ自身のキューを使用して構成するのがベスト・プラクティスです。高可用性を実現するには、"マスター"のプロセス・スケジューラを構成し、いずれかのプロセス・スケジューラで障害が発生した場合にマスター・スケジューラがそのワークロードを実行できるようにします。

結論

Oracle Exadata Database Machine上でのERP OLTPアプリケーションの実行を検討しているお客様は、アプリケーションが高可用性と高パフォーマンスというメリットを享受出来ることを理解されたと思います。事例で説明したとおり、アプリケーションの変更は最小限で済みます。その変更は、移行したアプリケーション固有のものです。

高いスケーラビリティを実現するには、アプリケーション層のコンポーネントをExadata Database Machineとは別にしておくことを推奨します。アプリケーション層サーバー用に、COBOLプロセス用の十分なCPU容量を確保し、広い帯域幅と待機時間の短いネットワーク（GigE、10GigE、InfiniBandなど）を使用して、Database Machineと接続します。この構成によって、"通信が頻繁な"バッチ・アプリケーション・プロセスで高いスケーラビリティとより簡単なパフォーマンス管理が可能となります。

CBAなどのお客様は、重要なPeopleSoft Financialsやその他のアプリケーションをExadataで実行することで、大きなメリットを得ています。お客様によるハードウェアとソフトウェアのインストールと設定が不要になったため、Exadata Database Machineへの移行にかかる時間を短縮できます。プロジェクトチームは、インストールや移行タスクの後に、テストや検証を行う必要がなくなりました。本番稼働すると、オンライン・ユーザーは安定したパフォーマンスとバッチ・プロセスによって、実行経過時間を大幅に短縮できることが分かります。

参考資料

1 Oracle Maximum Availability ArchitectureのWebサイト

Maximum Availability Architecture Best Practices

2 Oracle Database高可用性概要（部品番号B14210）

http://otn.oracle.com/pls/db111/db111.to_toc?partno=b28281

3 Oracle Database高可用性ベスト・プラクティス（部品番号B25159）

http://otn.oracle.com/pls/db111/db111.to_toc?partno=b28282

4 Oracle Exadata Bundle Patch Support Note:

My Oracle Support ID 888828.1 - Database Machine and Exadata Storage Server 11g Release 2 (11.2) Supported Versions

5 Oracle PeopleSoft PeopleBook

http://www.oracle.com/pls/psft/homepage

ORACLE

Oracle Exadata Database Machine上の

Oracle PeopleSoft
2011年2月

著者：Darryl Presley
共著者：Lawrence To、Lyn Pratt、
Richard Exley、Ray Dutcher、Marc Varennes、
Michelle Lam、Viv Schupmann

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

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

www.oracle.com

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

0109

Hardware and Software, Engineered to Work Together