ORACLE
NOSQL DATABASE

Oracle NoSQL Database

高速、信頼性、予測可能

Oracleホワイト・ペーパー | 2022年1月

目次

*Oracle France WeLoveStartups*での取組みに促されて、当社では*Oracle REST Data Services*を使用して*Oracle NoSQL Database Enterprise Edition*を利用することで、ビッグ・データ機能が大幅に向上しました。オラクルだけが、完全に機能するビッグ・データ・インフラストラクチャを数時間で実現してくれたのです。

BUSIT SAS
CEO
WASSEL GUERBAA氏、CEO

はじめに

Oracle NoSQL Databaseを使用すると、顧客との関係を築き、絶えず変化するビジネス要件に合わせてビジネス価値を創出する、革新的なアプリケーションを作成できます。組織は、データの非常に高速なストレージときわめて短い待機時間での取得を必要とする新しい機会に迅速に対応できます。Oracle NoSQL Databaseは、スケーラブルな分散型NoSQLデータベースであり、構成可能なストレージ・ノード・セット全体で、信頼性、柔軟性、可用性に優れたデータ管理を実現できるように設計されています。Oracle NoSQL Databaseは、以下のように多くの分野で柔軟性を持つように設計されています。

- 開発者は、普及している多数のプログラミング言語を使用して革新的なアプリケーションを作成でき、多くの方法でデータをモデリングできます。
- 管理者は、業界標準のサーバーを使用して、垂直/水平方向に拡張することにより、さまざまなワークロードを計画できます。

Oracle NoSQL Databaseの概要

NoSQLデータベースは、エンタープライズ・アプリケーション・アーキテクチャにおける最近の進化を象徴しており、過去20年の進化を継続しています。1990年代には、垂直に統合されたアプリケーションがクライアント/サーバー・アーキテクチャに移行し、最近は、クライアント/サーバー・アーキテクチャが3層のWebアプリケーション・アーキテクチャに移行しました。並行して、Webスケールのサービスの需要により、きわめて短い待機時間によるリアルタイムのアクセスと、オフラインのMapReduce処理が加わりました。データ・アーキテクトは、増分スケーラビリティと大規模な分散の代わりに、トランザクション一貫性を回避し始めました。NoSQLムーブメントは、この第二のエコシステムから現れました。

NoSQLは多くの場合、それが何でないかという表現で特徴づけられます。尋ねる相手によって異なりますが、NoSQLは、SQLベースのリレーショナル・データベース管理システムに限らないものか、または単にSQLベースのリレーショナル・データベース管理システム（RDBMS）ではないものかのいずれかです。これらの定義は、NoSQLは何でないかを説明していますが、NoSQLが何であるかについてはほとんど説明していません。過去40年間にデータ管理を導いてきた基盤を考えてみましょう。RDBMSシステムや大規模なデータ管理は、原子性、一貫性、独立性、および永続性という、トランザクションのACIDプロパティを特徴としています。一方、NoSQLは、次のようにBASEという頭字語を特徴とする場合があります。

Basically **A**vailable（基本的には常に利用可能）：レプリケーションを使用して、データが使用できなくなる可能性を減らし、シャーディングを使用（多数のさまざまなストレージ・サーバー間でデータをパーティション化）して、残された障害を分割します。可用性のためのシャーディングとレプリケーションの出現により、シェアード・ナッシング・アーキテクチャと水平スケーリングのメリットが生じます。結果として、システムは、たとえデータのサブセットが短期間使用できなくなっても、非常にスケーラブルで常に利用可能になります。

Soft state（一貫性がない状態を許容）：ACIDシステムがデータ整合性を厳格な要件としている一方で、NoSQLデータベースは、データに一貫性がない状態を許容し、そのような非一貫性にまつわる設計をアプリケーション開発者に委任しています。

Eventually consistent（結果的には一貫性が取れている）：アプリケーションは瞬間的な非一貫性に対応する必要がありますが、NoSQLシステムは、将来のどこかの時点でデータに一貫性があることを保証します。ACIDシステムがトランザクション・コミット時に一貫性を強制するのに対して、NoSQLは不確定な将来のある時点での一貫性を保証します。

NoSQLは、Amazon、Google、LinkedIn、Yahooなどの企業が、待機時間の厳しい制約の下で、今までに類を見ないデータ量や操作量の処理に取り組む中で生まれました。予測可能な待機時間を維持しながらリアルタイムまたはほぼリアルタイムで何百万ものリクエストに応えることが、豊富なユーザー・エクスペリエンス、高度なターゲット広告掲載システム、そしてオフライン分析の元になる大量のマシン生成データの収集を実現するための技術面での重要な推進力となりました。従来のリレーショナル・データベースでは役割を果たせなかったため、企業は分散ハッシュ表（DHT）や、慣例にとらわれないリレーショナル・データベース・システムまたはOracle Berkeley DBなどの組み込みのキー/値ストアに関する10年にわたる研究を基に、可用性の高い分散キー/値ストアを開発しました。

過去数年間に多くのNoSQLデータベースが進化しましたが、真に柔軟と言えるトランザクション・モデルを提供できたものはほんのわずかです。通常は、NoSQLデータベースは、パフォーマンス、スケーラビリティ、プログラミングの容易さのいずれか、またはこれらすべてを実現するために、データの一貫した分離されたビューを開発者に提供する機能を犠牲にしなければなりません。一方Oracle NoSQL Databaseは、パフォーマンスとスケーラビリティにほぼ影響を与えることなくACID/BASEのハイブリッド型モデルを提供すると同時に、プログラミングを容易にして開発者を満足させます。アプリケーション・アーキテクトやアプリケーション開発者は、トランザクションの制約を緩和または強化するのに適切な時期を判断できます。データベースにアクセスするたびに、アプリケーション固有のニーズに基づいて、一貫性（読取りの場合）および永続性（書込みの場合）の適切なレベルを選択できます。

組織は、NoSQLテクノロジーを、エンタープライズ・アプリケーション・アーキテクチャのテクノロジー ディメンションの1つとして採用してきました。そして、Oracle NoSQL Databaseは、あらゆるエンタープライズ・アプリケーション・アーキテクチャへのシームレスな統合に必要なNoSQLソリューションの望ましい機能すべてを提供します。図1は、標準的な取得-体系化-分析のデータ・サイクルを示したもので、Oracle NoSQL Databaseがこのようなエコシステムに適合する様子を表しています。オラクルが提供するモジュールによって、Oracle NoSQL Databaseは、Hadoop MapReduce、Hive、Spark、またはOracle Databaseとの統合が可能です。Oracle DatabaseをOracle NoSQL Databaseのデータへのアクセスに利用することによって、ビジネス向けの統合環境が拡張され、既存のBIツール、Oracle Enterprise R、またはOracle Advanced Analyticsを使用して、Oracle NoSQL Databaseに格納したデータにアクセスできます。

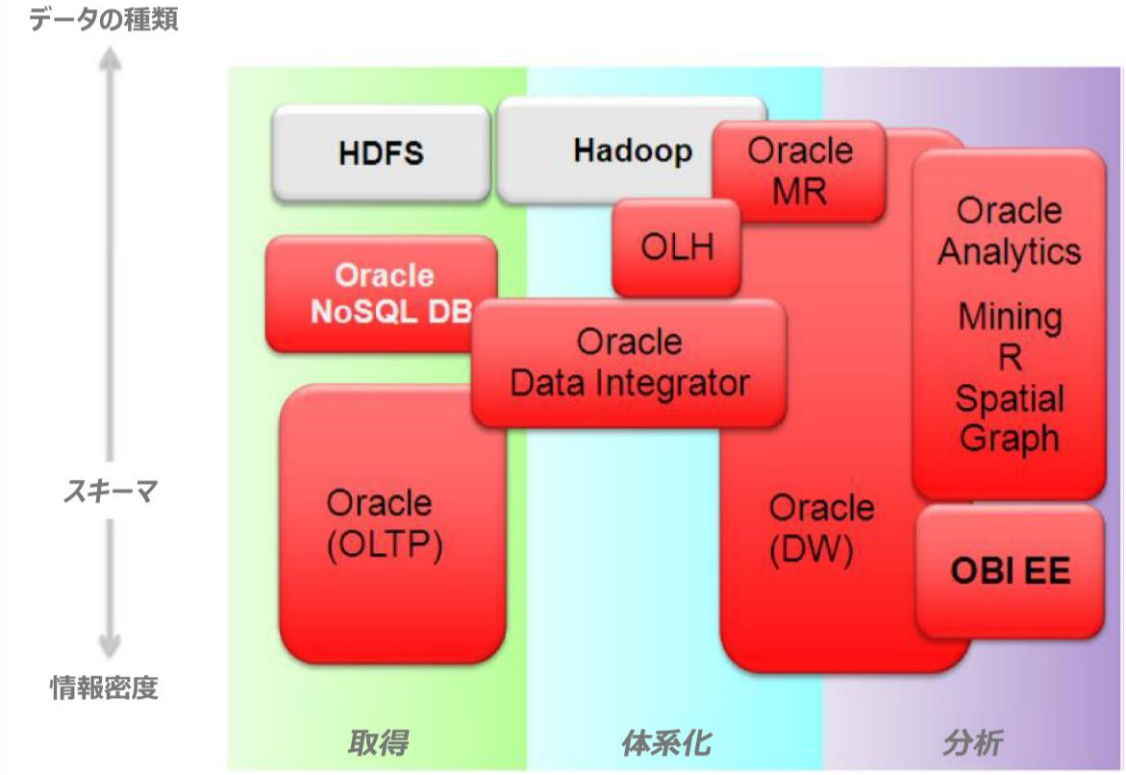

図1：取得-体系化-分析

“シングル・ポイント障害なし”のアーキテクチャを備えたOracle NoSQL Databaseが適切なソリューションとなるのは、データ・アクセスが本質的に"シンプル"で、アプリケーションの要求が従来のデータ管理ソリューションのボリュームや待機時間の能力を超えている場合です。たとえば、大容量のWebサイトからのクリックストリーム・データ、高スループットのイベント処理、モノのインターネット（IoT）のセンサー・データなどはすべて、極めて大量のシンプルな鍵付きデータを生成するアプリケーション・ドメインを表します。オンラインの小売り行動の監視、顧客プロファイルへのアクセス、適切な顧客広告の取得、およびリアルタイムのコミュニケーションの保存と転送は、究極の短い待機時間でのアクセスを要求するドメインの例です。リアルタイムのセンサー情報集計やスケーラブルな認証などの高度に分散化されたアプリケーションも、Oracle NoSQL Databaseに適したドメインを表します。

技術概要

Oracle NoSQL Databaseは、Oracle Berkeley Database Java Edition High Availabilityストレージ・エンジンを利用して、待機時間の影響を受けやすい大容量のアプリケーションやWebサービスのための分散化された高可用性キー/値および表ストレージを提供します。Oracle Berkeley DBは基礎になるストレージ管理機能を提供する一方で、Oracle NoSQL Databaseの上位のレイヤーは、オンラインでの柔軟なスケールアウト/スケールイン、複数のデータ・モデル（キー/値、表、JSON（JavaScript Object Notation）ドキュメント）のサポート、SQL Query、全文検索、認証、認可、マルチ・データセンターのディザスタ・リカバリなどの重要な機能を提供します。

アーキテクチャのデータ・モデル

基本的に、Oracle NoSQL Databaseは、ユーザー定義キーを不明確なデータ項目に関連付けるキー/値マップを実装します。このマップはキー/データ・ペアのバージョン番号を記録しますが、ストアに保持されるのは1つの最新バージョンのみとなっています。Oracle NoSQL Databaseはリーダーベースのレプリケーションを使用するため、アプリケーション側で非互換バージョンの調整を心配する必要はありません。このレプリケーションでは、Paxosマスター・ノードは特定のキーの最新値を常に保持していますが、マスター以外のレプリカはやや古いバージョンを保持していることがあります。アプリケーションは、バージョン番号を使用してread-modify-write操作の一貫性を確保できます。

データ・モデリングのオプション

真のマルチモデル・データ・ストアとして、Oracle NoSQL Databaseは、以下のようにさまざまなデータ・モデリングのオプションを提供しています。

- キー/値ペア – このモデリング・オプションを使用して、開発者は文字列キーと不明瞭なバイト配列を値として指定します。Oracle NoSQL Databaseは、バイト配列を解析しません。バイト配列は、アプリケーションによってシリアライズおよびデシリアライズされます。
- 表 – このモデリング・オプションを使用して、開発者はSQLデータ定義言語を使用して表定義を指定します。これは、リレーショナル・データベースによく似ています。Oracle NoSQL DatabaseまたはOracle NoSQL Database APIの問合せのためのコマンドラインSQLは、表データのアクセスに使用できます。
- JSONドキュメント – このモデリング・オプションを使用して、開発者はJSONオブジェクトをOracle NoSQL Databaseに格納し、SQLを使用してそれらのオブジェクトにアクセスできます。Oracle NoSQL Databaseは、JSONオブジェクトによるスライシングのために特別に設計された特殊な演算子を含む、豊富なSQL言語を提供します。JSONパス式によって指定される2次索引も、このSQL言語経由でサポートされます。

開発者は、RAWキー/値ペアまたは表の使用を選択して、アプリケーションのデータをモデリングできます。さまざまな方法でデータのモデリングを選択する場合は、いくつかの要素を考慮する必要があります。

- 2次索引または全文索引は必要ですか。アプリケーションに2次索引または全文索引によるメリットがある場合は、RAWキー/値ペアではなく表を使用してデータをモデリングする必要があります。Oracle NoSQL Databaseは、表定義を使用することで2次索引の作成を可能にしています。
- ファイングレイン認可は必要ですか。ファイングレイン認可を使用してデータを保護する必要がある場合は、RAWキー/値ペアではなく表を使用してデータをモデリングする必要があります。
- SQL問合せ – SQLまたはAPIコールを使用してデータにアクセスする計画ですか。SQLの使用を計画している場合は、RAWキー/値ペアではなく表を使用してデータをモデリングする必要があります。
- JSONドキュメント – アプリケーションは、JSONをプライマリ・データ形式として使用するように設計されますか。このJSONを永続的に格納して問合せをする必要がある場合は、表モデルおよび関連するJSONデータタイプを使用する必要があります。

シャーディング・モデル

Oracle NoSQL Databaseは、シャードに対するキーのハッシュ値を計算して、データベースにストレージを提供するストレージ・ノードのコレクションに対して分散を行います。ただし、アプリケーションはサブキー（子表とも呼ばれる）機能を利用してデータの局所性を実現できます。主キーとはシャード・キーと非シャード・キーを結合する可能性のあるもので、どちらもアプリケーションによって指定されます。シャード・キーを共有するすべてのレコードは、データの局所性を実現するために同じ場所に配置されます。同じ場所に配置されたシャード・キーのコレクション内で、シャード部分と非シャード部分で構成された完全な主キーは、高速で索引付けされた参照を提供します。たとえば、ユーザーの電子メールを格納するアプリケーションは、ユーザーIDをシャード・キーとして使用して、受信ボックス、削除済み、下書きなど、ユーザーが所有するさまざまな電子メール・フォルダにいくつかの子表を持つ場合があります。ログイン時にユーザー・インタフェースをレンダリングすると、ユーザーの受信ボックス・メッセージ（またはそのユーザーの他のフォルダ）の最初のページが単一のAPIコールを使用してアプリケーションから読み取られます。言い換えると、単一のネットワーク呼出しをOracle NoSQL Databaseに対して行うことで、特定のユーザーのすべてのメッセージを取得できます。

一貫性モデルと永続性モデル

多くのNoSQLデータベースが最終的な整合性を提供している一方で、Oracle NoSQL Databaseは、さまざまな異なる一貫性ポリシーを提供しています。広範な一貫性ポリシーの一端では、アプリケーションが絶対的な一貫性を規定できます。これによって、すべての読取りが、指定されたキーに対応する最新の書込み値を返すことが保証されます。広範な一貫性ポリシーのもう一端では、一貫性のないデータを許容するアプリケーションの能力で、一貫性の弱さを規定できます。これによって、データベースは、完全な最新データとはいえない場合でも効率的に値を返すことが可能になります。対極にあるこれら2つのポリシーの間で、アプリケーションは、最新バージョンに関してレコードの古さに制約を設けて時間に基づいた一貫性を規定すること、またはread-modify-write操作の原子性と読取りにおいて遡ることが許されるバージョンの指定を両方ともサポートするバージョンに基づいた一貫性を規定することができます。図2は、柔軟な一貫性ポリシーの範囲によって、アプリケーションの待機時間およびスケーラビリティ要件を満たしながら、データの保証を提供するビジネス・ソリューションを開発者が簡単に作成できる様子を示したものです。

図2：CAPモデル

Oracle NoSQL Databaseは、幅広い永続性ポリシーも提供しています。これらのポリシーは、クラッシュの発生後に何がシステム動作を保証するかを指定します。広範なポリシーの一端では、アプリケーションは、すべてのコピーにおいてレコードが永続的なストレージに書き込まれるまで書込み要求をブロックするように、要求することができます。これは、明らかにパフォーマンスと可用性に影響しますが、アプリケーションがデータの書込みに成功した場合は、そのデータは保持され、複数の障害の同時発生によってすべてのコピーが一時的に使用できなくなっても、データのリカバリが可能です。広範なポリシーのもう一端では、アプリケーションは、システムが既存の書込みの記録を完了するとすぐに、データが永続的なストレージになかったとしても、書込み操作の再開を要求できます。このようなポリシーでは、最適な書込みパフォーマンスが得られますが、永続性は保証されません。データベースがレコードをディスクに書き込む時期と、レコードのコピーで永続性が必要な部分（なし、すべて、過半数）を指定することによって、アプリケーションは広範な永続性ポリシーを適用し、待機時間と永続性との重要なトレードオフを実行できます。

プログラミング・モデル

前に述べたように、開発者には、Oracle NoSQL Databaseにおけるモデリング構成体について豊富な選択肢があります。これらのデータ・モデリング構成は、プログラミング・モデルに直接つながります。Oracle NoSQL Databaseを使用すると、開発者はデータのモデリングとNatural APIの使用が可能になります。たとえば、JSONをデータ・モデルとして使用する必要がある場合、Oracle NoSQL Database JSON APIはアプリケーションの開発を簡素化するように開発されています。同様に、アプリケーションのモデリングに表を使用することにより、Oracle NoSQLのDatabase Table APIを自然に使用できます。このホワイト・ペーパーでは、これら2つの固有モデルの例を挙げます。

表モデル

Oracle NoSQL Databaseが公開しているデータ・モデルの1つが、表モデルです。このモデルはリレーショナル・データベースに慣れている開発者にはより馴染みがある一方で、Oracle NoSQL Databaseの表モデルは、配列、マップ、組込みレコードなど、リレーショナルの世界にはない独自のデータタイプを提供します。Oracle NoSQL Databaseでの表の作成は、リレーショナルSQLと同様にSQLデータ定義言語（DDL）を通じて行います。次に示すのは、以下の列を使用してUserと呼ばれる表を作成するDDL文の例です。

» Id – これは、主キーおよびシャード・キーになります（指定しない場合、デフォルトではシャード・キーは主キーです）。

» firstNameおよびlastName – ユーザーの姓名を格納する文字列。

» 住所レコードの配列 – このユーザーの複数の住所を格納します。各住所レコードには、以下の属性が含まれています。

» Street、City、State – この住所の都道府県、市区町村、番地。

» zip – この住所の整数の郵便番号。

» addrType – “work”または“home”を含むことができる列挙リスト。

Oracle NoSQL Databaseでは、この表を作成するためのDDLは次のように指定できます。

```
Statement result res = KVStore.executeSync(“CREATE TABLE user(id INTEGER,
        firstName STRING, lastName String,
        addresses ARRAY(RECORD (street STRING, city String, state STRING,
                                zip INTEGER, addrType ENUM(home, work),
                                primary key(id))))”);
```

Oracle NoSQL Databaseはアプリケーションに簡単に組み込むことができます。基本の作成、読取り、更新、削除（CRUD）操作のためのAPI、およびSQLアクセスは、単一のjarファイルにパッケージ化されます。アプリケーションは、スタンドアロンのOracle NoSQL Databaseサーバー・プロセスにアクセスする1つまたは複数のクライアント・プロセスからAPIまたはSQL問合せを使用して、初期の開発およびテストのために複数システムの構成を設定する必要性を緩和できます。

表データへのSQL経由のアクセス

シンプルなキー/値アクセスのパターンに適合しない問合せ、またはセカンダリ・キーによるシンプルなレンジ・スキャンには、Oracle NoSQL Databaseはデータへのネイティブな SQL アクセスを提供します。Oracle NoSQL Databaseはかなり豊富な SQL 言語を提供しますが、このSQLは、Oracle RDBMSが提供するより強力で包括的なSQLのシンプルなサブセットと見なされます。このレベルのSQLアクセスを必要とするアプリケーションでは、Oracle NoSQL DatabaseがOracle RDBMSと統合することで、RDBMSはOracle NoSQL Database表から直接データへの問い合わせを実行できます。

Oracle RDBMSとOracle NoSQL DatabaseのSQLの言語の違いで注目すべき点は、配列やマップなどの非リレーショナル・データに対するSQL問合せ機能の概念です。Oracle NoSQL DatabaseのSQL言語は、配列やマップを介したスライスのための強力な構成体を提供すると同時に、これらの構造にフィルタリング式を提供します。以下の例は、この機能の小さなサブセットを示したものです。

1. New York Cityのすべてのユーザーの姓名を返します。

```
select
      firstName,
      lastName
from
       user
where
       addresses.phones[$element.city = “New York”]
```

2. 郵便番号94107に住所があるすべてのユーザーの姓名を返し、その結果をlastName属性によって順序付けます。

```
select
      firstName,
      lastName
from
       user
where
       addresses.phones[$element.zip= 94107] and
       addresses.phones[$element.attrType = “home”]
order by lastName
```

3. New Haven、Newark、およびNew York Cityのすべてのユーザーの最初のページ（1ページにつき25の結果）を表示します。

```
select
    *
from
    user
where

    addresses.phones[$element.city = “New York”] or
    addresses.phones[$element.city = “New Haven”] or
    addresses.phones[$element.city = “Newark”]

order by lastName limit 25 offset 1
```

JSONドキュメントの使用

JSONは、Webアプリケーション、サービスAPIコールからの結果の返し、およびアプリケーション間のデータ交換に便利なデータ形式として、過去数年間で劇的に成長してきました。自己記述的なデータタイプであるJSON形式のデータは、スキーマを事前に指定することなく非定型のコンテンツを保存しようとするアプリケーションに対して、究極の柔軟性を提供します。固定されたスキーマはいったん保存される一方で、すべてのJSONドキュメントにはスキーマのコピーが含まれます。

Oracle NoSQL Databaseは、固定されたスキーマ表の独自の組合せと、JSONドキュメント向けの非定型のスキーマレス・サポートを提供します。開発者は、アプリケーションのどの部分でスキーマレスJSONの柔軟性を利用すべきか、そしてアプリケーションのどの部分で固定されたスキーマと引換えにストレージとコンピューティングを最適化すべきかを選択できます。

前の例の固定されたスキーマのバージョンは、以下のようにスキーマレスJSONとして表されます。

```
Statement result res = KVStore.executeSync(“CREATE TABLE user(id INTEGER,
                                            userData JSON primary key(id))”);
```

以下を使用してJSONデータを挿入できます。

```
Row r = Table.createRow().put(“id”, 123456).put("userData",
                                                FieldValueFactory.createValueFromJson(
    “{
        “firstName” : “John”,
        “lastName” : “Doe”
          "addresses": [
          {“street” : “127 Spring St”, “city” : “New York”, “state” : “NY”,
            “zip” : 10012, ”addrType” : “work”},
          {“street” : “625 Ridgewood Rd”, “city” : “Paramus”, “state” : “NJ”,
            “zip” : 07675, ”addrType” : “home”}
      ]
    }”);
    ));
```

JSONデータは、表で実行されるのと同じSQL言語を使用して問い合わせることもできます。さらに、非定型のJSONドキュメントでは、まとまりのない形式や未知の形式を含むドキュメントの処理に役立つ、いくつかの新しい演算子と"CASE"式が導入されています。

1. New York cityのすべてのユーザーを検索します。この場合、ドキュメントに住所の配列が含まれる場合は、条件を配列に適用します。それ以外の場合は、条件を直接メイン・ドキュメントの“city”という属性に適用します。

```
select
      firstName,
      lastName
from
       User u
where
       (case when u.addresses instanceof ARRAY then
         u.addresses[$element.city = “New York”] else
         u.city = “New York” end)
```

2. JSONドキュメントにcity属性が含まれる場合に、問合せ条件のドキュメントでNew York Cityのすべてのユーザーを検索します。

```
select *
from User u
where
        (case when exists(u.address.city) then u.address.city = “New York”
```

Oracle NoSQL Databaseはアプリケーションに簡単に組み込むことができます。基本の作成、読取り、更新、削除（CRUD）操作のためのAPI、およびSQLアクセスは、単一のjarファイルにパッケージ化されます。アプリケーションは、スタンドアロンのOracle NoSQL Databaseサーバー・プロセスにアクセスする1つまたは複数のクライアント・プロセスからAPIまたはSQL問合せを使用して、初期の開発およびテストのために複数システムの構成を設定する必要性を緩和できます。

障害回復力

Oracle NoSQL Databaseでは、障害分離コンテナとして可用性ゾーンの概念を公開しています。可用性ゾーンは、サーバーのラック、ラック・スイッチの上部、建物のフロア、またはデータセンター全体という形を取ることができます。ゾーンは、プライマリまたはセカンダリとしての特徴も持ち、プライマリ・ゾーンはマスターの選択および割当ての承認に使用できます。Oracle NoSQL Databaseはさらに、任意の数のプライマリ・ゾーンおよびセカンダリ・ゾーンの構成が可能です。一度構成されると、データ・ストアの各シャードのレプリカがこれらのゾーン間でレイアウトされ、1つのゾーンの障害がこの障害を制限（または格納）するようになるため、割当ては常にデータ・ストアの各シャードによって維持されます。

Oracle NoSQL Databaseは、可用性の高い別個の管理サービスをデプロイします。Oracle NoSQL Databaseでの“シングル・ポイント障害なし”の哲学に沿って、実行中のインストールの操作は管理サービスの可用性には依存しません。そのため、データベースと管理サービスは、構成の変更中でも使用できます。

アーキテクチャ

ここでは、Oracle NoSQL Databaseのアーキテクチャを説明します。システムの論理コンポーネントによる書込み操作の実行を把握してから、これらのコンポーネントが実際のハードウェアおよびソフトウェア操作にどのようにマッピングされているかを説明します。“Katana”キーおよび“{“a”: “foo”}”の値のJSONオブジェクトを含むレコードを作成します。図3は、putIfAbsent(“Katana”, “{“a”: “foo”}メソッドの起動を表したものです。

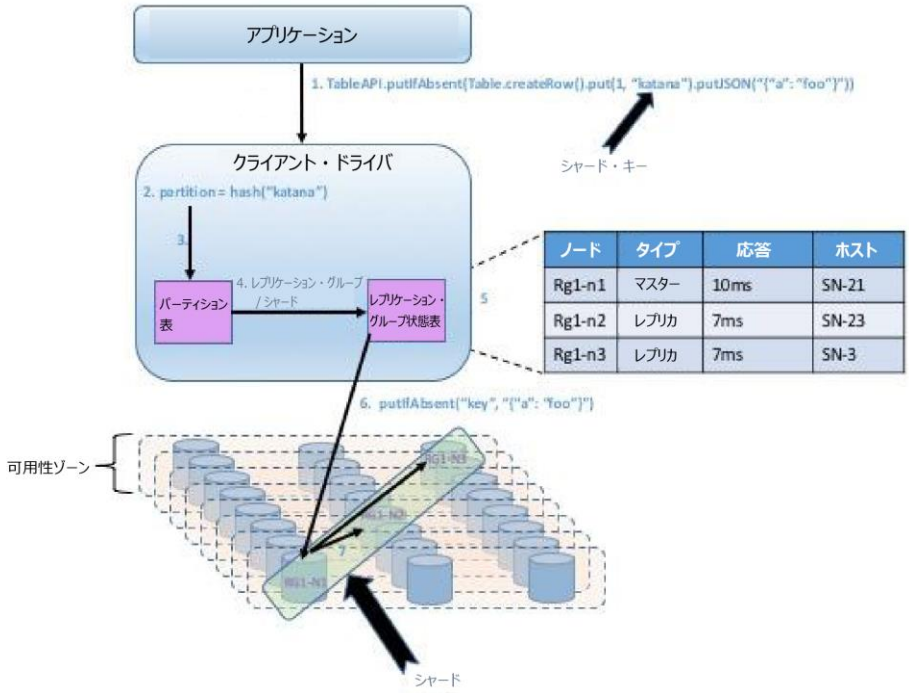

図3：アーキテクチャ - Oracle NoSQL Database – 書込みリクエストのフロー

アプリケーションは、クライアント・ドライバにputIfAbsentメソッドを発行します（手順1）。クライアント・ドライバは、“Katana”キーのハッシュ値を計算して、多数のパーティションのうちの1つを選択します（手順2）。各パーティションが特定のレプリケーション・グループに割り当てられます（シャード）。ドライバはパーティション表を照会して（手順3）、パーティション番号をシャードにマッピングします。

シャードはいくつかの（構成可能な）レプリケーション・ノードで構成され、各レプリケーション・ノードは個別の可用性ゾーンに常駐しています（障害の封じ込めのため）。シャード内のレプリケーション・ノードの数によって、システムにレジリエンスがある障害の数が決まります。1シャードあたり3つのノードがあるシステムは、読取りリクエストを継続しながら2つの障害に耐えることができます。書込み時の障害に耐える機能は、書込みAPIコールでリクエストされた永続性ポリシーに基づきます。アプリケーションが大多数の参加者に書込みの確認を求めない場合、システムは最大2つの書込み障害に耐えることもできます。たとえアプリケーションが大半のサイトに書込み操作の確認を要求する永続性ポリシーを必要とする場合でも、5つのノードを備えたグループは、最大4つの読取り障害と最大2つの書込み障害に耐えることができます。

シャードがあれば、クライアント・ドライバは次にレプリケーション・グループ状態表（RGST）を照会します（手順4）。シャードごとにRGSTは、シャード内のレプリケーション・ノードのグループで構成される各レプリケーション・ノードに関する情報を含みます（手順5）。レプリケーション・グループのさまざまなノード上のマスターのIDやロードなどのRGSTの情報に基づいて、クライアント・ドライバはリクエストを送信するノードを選択し、適切なノードにリクエストを転送します（手順6）。この場合、書込み操作を発行しているため、リクエストはマスター・ノードへ進む必要があります。

続いて、レプリケーション・ノードが操作を適用します。putIfAbsentの例でキーが存在している場合、操作は影響を与えることなく、指定されたエントリはすでにストアに存在するというエラーを返します。キーが存在しない場合は、レプリケーション・ノードはキー/値ペアをストアに追加してから、新しいキー/値ペアをレプリケーション・グループの他のノードに伝播します（手順7）。

実装

Oracle NoSQL Databaseのインストールは、クライアント・ドライバと、ストレージ・ノードのコレクションという2つの主要部分で構成されます。図3で示したように、クライアント・ドライバがパーティション・マップとRGSTを実装する一方、ストレージ・ノードはシャードで構成されるレプリケーション・ノードを実装します。このセクションでは、これらの各コンポーネントについて詳しく見ていきます。

ストレージ・ノード

ストレージ・ノード（SN）とは、一般的に、ローカルの永続的なストレージ（ディスクまたはSSD）が接続された物理マシンのことで、1つ以上のコアを持つCPUとメモリを搭載し、IPアドレスが割り当てられています。システムは、ストレージ・ノードの数が多くなるほど集計スループットが向上し、ストレージ容量も増加します。そして、レプリケーション・グループのレプリケーションの割合が大きくなるほど、レプリケーションの割合が小さいインストールよりもリクエスト待機時間が短くなります。さらに、SNが多いほど、システム全体の可用性が高まります。

ストレージ・ノード・エージェント（SNA）は、各ストレージ・ノード上で実行され、ノードの動作を監視します。SNAは、管理サービスから構成を受け取り、監視情報を管理サービスにレポートを提供します。SNAは、業務系データをストレージ・ノードから継続的に収集し、要求を受けた際に管理サービスにそのデータを提供します。

1つのストレージ・ノードは、1つまたは複数のレプリケーション・ノードに対応します。各レプリケーション・ノードは、単一のレプリケーション・グループに属しています。単一のレプリケーション・グループにあるノードはすべて、同じデータに対応します。各グループには、すべてのデータ変更操作（作成、更新、削除）を処理する動的に選出されたマスター・ノードが含まれます。他のノードは読取り専用レプリカですが、マスター・ノードに障害が発生した場合は、マスターの役割を果たします。一般的なインストールでは、レプリケーション・グループのレプリケーション係数として3が使用されます。これによって、2つの障害が同時に発生しても引き続き読取り操作に対応できるため、システムの動作を確実に継続できます。信頼性の増減を必要とするアプリケーションでは、このパラメータを適宜調整できます。

図4は、10個のレプリケーション・グループ（0～9）（シャードとも呼ばれる）によるインストールを表しています。各レプリケーション・グループには、2つのデータセンターにわたって3のレプリケーション係数（1つのマスターと2つのレプリカ）が含まれます。2つのデータセンターのうち、大きいほうに2つのレプリケーション・ノードを置き、小さいほうのデータセンターに残りのレプリケーション・ノードを置くことに注意してください。この種の仕組みは、プライマリ・データのアクセスに大規模なデータセンターを使用するアプリケーションに適しています。小規模なデータセンターは、プライマリ・データセンターに致命的な障害が発生した場合に備えて維持されます。

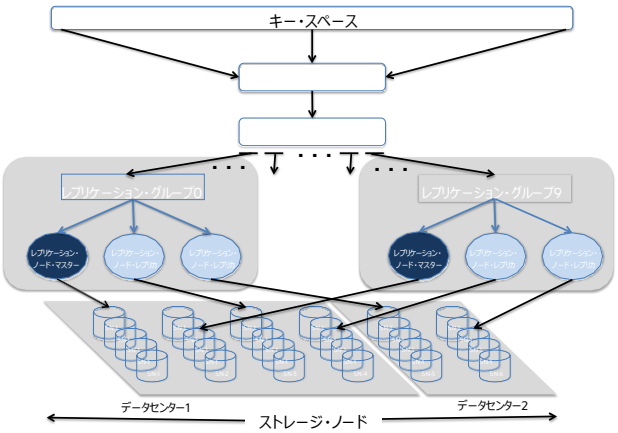

図4：アーキテクチャ

図4：10個のシャードおよびRF=3を使用したアーキテクチャ

10個のレプリケーション・グループは、2つのデータセンターにわたって30個のストレージ・ノードに格納されます。

レプリケーション・ノードは、Oracle NoSQL Database APIをクライアントからのRMIコール経由でサポートし、ログ構造化ストレージ・システムからデータを直接取得し、ログ構造化ストレージ・システムにデータを直接書き込みます。これにより、短い待機時間の読取りパフォーマンスを提供する索引構造を維持しながら、高パフォーマンスの書込みが実現します。

Oracle NoSQL Databaseは、レプリケーションを使用することによって、障害発生時のデータ可用性を確保しています。単一マスター・アーキテクチャでは、書込みをマスター・ノードに適用してから、レプリカを伝播します。マスター・ノードで障害が発生した場合、レプリケーション・グループ内のノードは、自動的に信頼性の高い選択を実行し（Paxosプロトコルを使用）、残っているノードの1つをマスター・ノードに指定します。新しいマスター・ノードは、書込みについて責任を負うことになります。

クライアント・ドライバ

クライアント・ドライバは、APIをアプリケーションにエクスポートするJava jarファイルです。クライアント・ドライバは、レプリケーション・グループ状態表（RGST）を使用するトポロジに対応しています。そのトポロジにより、キーがパーティションに、そしてパーティションからレプリケーション・グループに効率的にマッピングされます。各レプリケーション・グループには、グループで各レプリケーション・ノードをホストしているストレージ・ノードのホスト名、レプリケーション・ノードに関連するサービス名、および各ストレージ・ノードが常駐するデータセンターが含まれます。クライアントは、主に2つの目的でRGSTを使用します。つまり、レプリケーション・グループのマスター・ノードを特定し、書込みリクエストをマスターに送信できるようにするため、そして、読取りのためにレプリケーション・グループのすべてのノード間でロード・バランシングを行うためです。RGSTは重要な共有データ構造をしているため、各クライアントとレプリケーション・ノードは、そのコピーを保持することでシングル・ポイント障害を回避します。クライアントおよびレプリケーション・ノードは、RGSTを使用して書込みリクエストをマスターに、読取りリクエストをレプリケーション・グループの適切なメンバーに直接接続（リダイレクト）するRequestDispatcherを実行します。

トポロジはクライアントまたはレプリケーション・ノードの初期化中にロードされ、トポロジに変更がある場合は管理者が更新できます。RGSTは動的であるため、継続的な保守が必要です。各レプリケーション・ノードは、Replication Node State Updateスレッドと呼ばれるスレッドを実行します。これは、RGSTの継続的な保守を担っています。UpdateスレッドおよびRequestDispatcherは、折を見てリモート・レプリケーション・ノードに関する情報を収集します。これには、レプリケーション・グループのノードの現在の状態、ノードの最新状態の程度を示す指標、ノードとの相互作用が最後に成功した時刻、ノードの後続の平均応答時間、未処理のリクエスト・キューの現在の長さなどが含まれます。さらに、Updateスレッドは、ネットワーク接続を維持し、切断されたネットワーク接続を再確立します。この保守は、RequestDispatcherのリクエスト/レスポンス・サイクルの外部で実行され、切断された接続が待機時間に与える影響を最小限に抑えます。

セキュリティ

Oracle NoSQL Databaseはセキュアに構成できます。セキュアな構成では、NoSQLクライアント、ユーティリティ、NoSQLサーバー・コンポーネント間のネットワーク通信はSSL/TLSを使用して暗号化され、すべてのプロセスは接続するコンポーネントに対して認証される必要があります。

セキュリティの2つのレベルを認識する必要があります。これらはネットワーク・セキュリティで、ネットワーク・レベルでの保護の外部レイヤーと、ユーザー認証/認可を提供します。ネットワーク・セキュリティは通常はインストール・プロセス中にファイル・システム・レベルで構成されますが、ユーザー認証/認可はNoSQLユーティリティを通じて管理されます。

Oracle NoSQL Databaseは、ストアを保護し、パスワードの複雑さを設定するために以下の機能を提供します。

- セキュリティ構成ユーティリティ。新規または既存のOracle NoSQL Databaseインストールにセキュリティを追加する構成を実現します。
- 認証方式。Oracle NoSQL Databaseは、ユーザーおよびシステムにパスワード認証を提供します。Enterprise EditionバージョンのOracle NoSQL Databaseでは、Kerberos認証もサポートしています。

» 暗号化。データをネットワーク上で暗号化することで、そのデータへの認証されていないアクセスを阻止します。

» 外部パスワード・ストレージ。Oracle NoSQL Databaseは、操作可能な2つのタイプの外部パスワード・ストレージ方式を提供します（タイプの1つはCEデプロイメント向け）。これらのタイプはクリア・テキストで、Oracle Wallet（Oracle NoSQL Database Enterprise Editionで使用可能）を使用しています。

» セキュリティ・ポリシー。Oracle NoSQL Databaseは、セキュアな環境を保証するために行動を設定できます。

» ロールベースの認可。Oracle NoSQL Databaseは、事前定義されたシステム・ロール、権限、およびユーザー定義のロールをユーザーに提供します。ロールの付与により、必要な権限をユーザーに設定できます。

統合

Oracle NoSQL Database（Enterprise Editionバージョン）は、提供された状態のままで、Oracle Database、Oracle GoldenGateなどのさまざまなOracleテクノロジーや、Hadoop、Hive、Sparkなどのオープンソースのテクノロジーと統合できます。

Oracle Databaseの統合

Oracle NoSQL DatabaseからOracle Databaseの外部表にデータを読み取るには、1つまたは複数のロケーション・ファイルを用いて外部表を定義します。ロケーション・ファイルには、実際にはデータは含まれていませんが、Oracle DatabaseおよびOracle NoSQL Databaseへの接続、クエリ制約、および書式設定に関連する構成情報が含まれます。

外部表の定義で複数のロケーション・ファイルを指定することで、データ読取り中の並列度を設定できます。外部表を定義したら、Publishユーティリティを実行して、外部表に関する情報およびOracle NoSQL Databaseのデータにアクセスする方法を"パブリッシュ"します。この情報は、XMLドキュメントとしてそれぞれのロケーション・ファイルに書き込まれます。

パブリッシュが完了したら、SQL SELECTが外部表に対して実行されます。これにより、<KVHOME>/exttab/bin/nosql_streamスクリプトが起動し、次にPreprocクラスが起動します。プリプロセッサは、以下を実行します。

» ロケーション・ファイルからの構成情報およびNoSQL Databaseのアクセス情報の読取り

» NoSQL Databaseからのデータの読取り

» ユーザー定義のフォーマッタまたはデフォルトの形式を使用して読み取ったデータのフォーマット

» フォーマットしたデータのstdoutへの書込み

Hadoop、Hive、Big Data SQLの統合

HadoopからOracle NoSQL Databaseへの統合は、Hadoop MapReduceジョブをオラクル NoSQL Database表に格納されるデータに対して実行することでサポートされます。カスタムのInputFormatクラスにより、MapReduce入力データの読取りおよび書込みの方式を定義します。ジョブの論理"分割"のリストを取得する方法を定義するInputSplitのカスタム実装や、MapReduceおよびSerDeに使用されるキー/値ペアを検索して返す作業を実行するRecordReadクラスのカスタム実装もあります。

Oracle Big Data SQLを使用すると、ユーザーはSQL言語を採用して、異なるデータベースの多数の異なる場所に保存されているデータを管理して操作できます。Oracle Big Data SQLは、Oracle NoSQL Databaseのデータを強化された外部表としてOracle Databaseに提示することでこれを実現します。そのためには、これらのソースのデータへアクセスするための外部セマンティックをOracle Databaseの内部構造へマッピングします。

Oracle Big Data SQLは、HadoopおよびNoSQLソースのデータを簡単に統合できるだけでなく、基礎となるストレージ・メカニズムを利用して可能な限り最適なパフォーマンスを提供します。Big Data SQLの条件プッシュダウン・テクノロジーを使用すると、Oracle Databaseで発行される問合せの条件をリモート・システムで実行し、特定のファイル形式にプッシュできます。

Hiveとの統合は、HiveQL（HQL）と呼ばれるSQLに似た言語を使用してOracle NoSQL Databaseに保存されるデータにHive問合せを実行することで実行できます。カスタムのHive TableStorageHandlerは、HiveおよびBig Data SQLによって要求され、Oracle NoSQL Databaseストアの表に対して問合せを実行する際に使用されるメカニズムを定義します。さらに、カスタムのObjectInspectorクラスは、対応するOracle NoSQL Databaseのデータ形式にデータタイプを変換するために使用されます。

問合せパフォーマンスを向上させるため、HiveおよびBig Data SQLは、条件プッシュダウンの形式をサポートします。そこでは、クライアント側のフロントエンドが、問合せのWHERE句を列情報および対応する比較操作に分解します。これにより、結果のコンポーネントが処理のためにデータベース・サーバー側にプッシュされます。Big Data SQLは、Hiveインタフェースを使用して、KVStore表データに対してBig Data SQL問合せを実行する際の条件プッシュダウンをサポートします。条件プッシュダウン機能をサポートするために、HiveStoragePredicateHandlerのカスタム実装では、条件の分解が表スキャンおよび索引スキャンにプッシュダウンされる基準をサポートして定義します。

パフォーマンス

これまで、Oracle NoSQL Databaseのさまざまな構成を試し、Yahoo! Cloud Serving Benchmark（YCSB）のいくつかのパフォーマンス結果を提示し、システムのノード数によってシステムを拡張する方法を示してきました。あらゆるパフォーマンスの測定結果と同様に、結果は多数の要素によって異なる可能性があります。

一貫性のあるYCSBロードをストレージ・ノードごとにさまざまなサイズの構成に適用しました。各ストレージ・ノードは、6コア/ソケットと24 GBのメモリを備えたIntel Xeon Processor X5670デュアル・ソケット・マシンで構成されました。各マシンは単一のローカル・ディスクを備え、RedHat 2.6.18-164.11.1.el5.crt1を実行しました。ディスク・サイズは各ノード上で300 GBのスケール制限のあるリソースであり、構成全体に影響するため、各ノードが1億のレコードを保持し、平均のキー・サイズは13バイト、データ・サイズは1,108バイトとなるように構成しました。

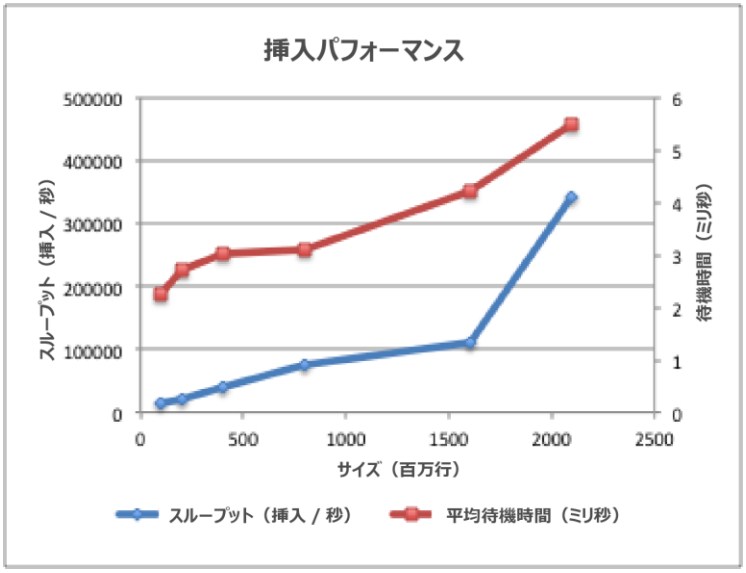

図5：YCSBの結果 – データベース・サイズの増加に伴うパフォーマンス

図5は、Oracle NoSQL DatabaseのRAW挿入パフォーマンスを示したものです。対象となっている構成は、1億のレコードを保存する3つのノードを備えた単一のレプリケーション・グループ・システムから、21億のレコードを保存する96のノード上に32のレプリケーション・グループを備えたシステムまでです（YCSBベンチマークは最大21億のレコードに制限されています）。このグラフでは、1秒あたりの操作のスループット（青線と左軸）と1ミリ秒単位の応答時間（赤線と右軸）を示しています。システムのスループットは、データベース・サイズとレプリケーション・グループの数が増加するにつれてほぼ線形にスケーリングし、応答時間の増加はわずかです。

結論

オラクルのNoSQL Databaseは、エンタープライズ品質のストレージおよびパフォーマンスを、広く分散された可用性の高いNoSQL環境に提供します。商用に実績のある、書込みに最適化されたストレージ・システムにより、卓越したパフォーマンスと堅牢性および信頼性が実現します。また、“シングル・ポイント障害なし”の設計により、システムを継続して実行でき、さまざまな障害が発生した場合もデータを使用し続けることができます。

Oracle Corporation, World Headquarters

500 Oracle Parkway

Redwood Shores, CA 94065, USA

海外からのお問い合わせ窓口

電話：+1.650.506.7000

ファクシミリ：+1.650.506.7200

CONNECTWITHUS

blogs.oracle.com/oracle

facebook.com/oracle

twitter.com/oracle

oracle.com

Integrated Cloud Applications & Platform Services

Oracle NoSQL Database
2022年1月 V1