

Oracle Advanced Queuing: 概要と チュートリアル

オラクル・ホワイト・ペーパー
2005年6月

Oracle Advanced Queuing: 概要とチュートリアル

概要	3
メッセージ・キューイング	3
STREAMS ADVANCED QUEUING	4
データベースの統合	4
強力な機能	4
サービスの品質	4
メッセージの順序付け	5
通知	5
変換	5
伝播	5
結論	6
STREAMS ADVANCED QUEUING – ショート・チュートリアル	7
管理インタフェース – Streams Advanced Queuing の設定	7
ペイロード・タイプの作成	7
キューの作成/起動	7
マルチ・コンシューマ・キュー用サブスクリバの追加	7
運用インタフェース – AQ の使用	8
メッセージのエンキュー	8
メッセージのデキュー	8
キューの停止/削除	9
簡単なインタフェース	9

Oracle Advanced Queuing: 概要とチュートリアル

概要

企業に対する技術投資は、今日まで長期にわたり期待されてきました。特にこの数年、多種多様な技術がみられます。技術の中には、情報にとって有効なものとしていないもの、また、役に立たないものもあります。これらのシステム間で連携を築くことは難しい問題です。すべてのシステムで、使う言語も技術的なつかみ所も違います。

これらの様々なシステムをどのように取り扱えばよいのでしょうか。多様な機能すべてをどのように管理できるのでしょうか。また、どのようにしてこれらの機能を一体化したシステムに統合できるのでしょうか。答えは、メッセージ・キューイングにあります。

メッセージ・キューイング

メッセージ・キューイングは、キューに様々なタスクやアプリケーションに関する情報を保管するインフラストラクチャです。キューは、アプリケーションが「メール」をメッセージ形式で検索できるように、「郵便ポスト」の役割を果たします。このため、1つのアプリケーションが特定のタスクを実行するために、特定のアプリケーションへのアクセスが必要な場合、メッセージをキューに保管し、必要に応じてそれらのメッセージを検索できます。受信側アプリケーションでは、必要な処理について情報が得られます。また、受信側アプリケーションは非同期に要求を処理できるため、要求側は他のアプリケーションを待つ間、その作業を停止する必要がありません。この「切り離し」処理は、メッセージ・キューイングの最も重要な機能です。

ここで一番大きな問題となるのは、各部門システムのベスト・パーツへの同時接続です。メッセージ・キューイングは、このようなすべてのシステムに対しブリッジおよび統合の役割を果たします。メッセージ・キューイングは、アプリケーションが絶えず再実装しなければならない情報を確実に転送します。重要な交換情報を持つアプリケーションは、この情報を失うわけにはいきません。メッセージ・キューイング・システムを使用して、一度だけメッセージ配信を保証する必要があります。レガシー・システムと新規システムは共存可能です。メッセージ・キューイングにより、企業は、様々なインフラストラクチャのベスト・パーツを機能的なエコシステムに統合できます。

様々なシステムの統合以外にも、メッセージ・キューイングは、よりスケーラブルな IT インフラストラクチャを構築できます。アプリケーションは、簡単な論理で重要な結果データをキューに保存できます。また、データを蓄積するキュー

へリソースを追加するだけで、ボトルネックが軽減できます。これは、アプリケーション・ロジックを介してではなく構造的に処理できるためであり、管理作業が軽減でき、インフラストラクチャは様々な業務ニーズに対応できます。

STREAMS ADVANCED QUEUING

データベースの統合

Streams Advanced Queuing (SAQ) (以前の Advanced Queuing) は、Oracle Database に搭載済みの強力なアプリケーション・メッセージ・キューイング・インフラストラクチャです。SAQ には、データベースの重要な特性すべてが網羅されています。また、可用性も高く、Oracle でも定評の特性です。SAQ は、シングル・インスタンスから大規模なマルチノード RAC インストールまで拡張可能なインフラストラクチャで、そのセキュリティは、実証済みのデータ保護基準と同様のレベルが保証されています。SAQ には、データベースからの監査レベルも期待できます。

さらに、SAQ はデータベース表を実装の一部として使用します。このデータベース表は、データおよび業務を管理する表と同じタイプです。表の構築により、SAQ は、他のデータと同じようにキューに簡単に問合せできます。高性能表の管理における Oracle の豊富な経験により、迅速にキューにアクセスできます。

最も重要な点は、SAQ が Oracle Database に搭載され統合されていることです。Oracle Database は、すでに内部操作に対して SAQ を使用しているため、優秀なテスト結果と高度な性能が期待できます。つまり、データベースにいかなるパーツも統合する必要がありません。一方、IBM の Websphere MQ、SonicMQ または TIBCO など他の製品では、パーツの統合が必要です。さらに、Streams Messaging の永続レイヤーは、データベースです。メッセージの安全性を確保することとは、無駄な管理作業を行わずにデータベースの安全性を確保することです。

強力な機能

サービスの品質

永続的なメッセージ・キューイングは、信頼性のあるインフラストラクチャのバックボーンです。アプリケーションは、メッセージ・キューイング・インフラストラクチャによって保証された完全一括配信メカニズムにより、情報の整合性を維持する必要があります。あるアプリケーションが失敗すると、別のアプリケーションがこれに対応するか通常どおりの業務を継続するためにオンラインになります。この永続的なサービスは、SAQ ではデフォルトで提供されています。

アプリケーションによっては、このような強力な保証は必要ない場合があります。たとえば、携帯電話のテキスト・メッセージは配信される必要はありますが、その保証は必要ありません。テキスト・メッセージ・ユーザーは、むしろ配信待ち時間が短くなることを望んでいます。これが、テキスト・メッセージングの最も重要な特長です。このようなアプリケーションに対して、SAQ は、待ち時間が最も短いメッセージ管理インフラストラクチャに対して、メモリー内メッセージング機能を提供します。

メッセージの順序付け

通常、アプリケーションは、受信順にメッセージとデータを処理しなければなりません。たとえば、銀行では、金融取引の順序を厳密に管理し、金融の整合性を維持する必要があります。このコミット時間順は、SAQ のデフォルト順に基づいています。

処理によっては、別の順序付けメカニズムが必要な場合があります。たとえば、顧客への最も慎重な対応が必要な航空会社では、乗客の優先順序があります。運送会社には、荷物の急送サービスがあります。このようなアプリケーションでは、依頼された日時に関係なく優先順位の高い順に処理する必要があります。SAQ は、強力な順序付けスキーマにも対応しています。

通知

アプリケーションは、新しいメッセージを絶えず確認する必要がある場合にのみ、非同期となり分離されます。SAQ により、アプリケーションはキューの新しいメッセージをポーリングしたり、新しいメッセージが到着するまでキューで待機できます。さらに、SAQ によりアプリケーションは、新しいメッセージを常時検索する必要がありません。SAQ は、必要なメッセージの受信時にアプリケーションに通知できるため、重要なメッセージが到着するまでアプリケーションは本来の業務に専念でき、メッセージを気にすることなく業務をこなせます。

変換

すべての人が同じ言語を話さないのと同様に、アプリケーションも同じ言語で書かれているわけではありません。アプリケーション間の連携には相互理解が必要です。SAQ には、送受信両方のメッセージを翻訳し変換する機能があります。各アプリケーションは、仲介の翻訳サービスを行う SAQ Rules Engine の変換ユーティリティを使用して、1つの言語を理解し伝達することができます。それにより、アプリケーションの同時処理が可能になります。

伝播

アプリケーションは、単一データベース、単一サーバーに限らず接続できます。SAQ には、キューのある場所に関係なくキューからキューへ伝播するメッセージ伝播機能があります。伝播機能により、メッセージは必要に応じてルーティングが作成されるアプリケーションに対して、幅広く収集・配信を行い集約 (fan-in) と分岐 (fan-out) を実行します。さらに SAQ には、データベース・リンクまたは各キュー・レベルで伝播をスケジューリングするオプションがあります。この伝播処理は、データベース・リンクまたは HTTP 転送アダプタを介して実行されます。SAQ を使用すると、アプリケーションは局所的かつ地理的に分散できます。

結論

企業には、様々な業種から様々な要件が寄せられます。これらの要件に応じて別のソリューションが発生すると、インフラストラクチャの関連付けが困難になります。Oracle Database に搭載された Streams Advanced Queuing は、企業を管理する様々なテクノロジーやアプリケーションを標準化し活用するプラットフォームを提供します。異なるサービスの品質、自動変換および伝播処理などの強力な機能により、最良の状態で行うツールを企業に提供できます。データベースに統合された Streams Advanced Queuing は、ますます成長し、スケーラブルで高可用性なビジネスのために、管理作業を軽減し、生産性を向上することを意味します。

STREAMS ADVANCED QUEUING – ショート・チュートリアル

ここでは、最も柔軟な API を使用して、Streams Advanced Queuing を迅速に設定、使用する方法のチュートリアルを簡単に説明します。このチュートリアルでは、典型的なアプリケーション・ニーズについて説明し、Streams Advanced Queuing の簡単な使用例を示します。

管理インタフェース – Streams Advanced Queuing の設定

ペイロード・タイプの作成

最初の手順では、必要なメッセージ情報を決定します。
この確実な情報でペイロード・タイプを簡単に作成できます。

```
--create the type to go into the queue
create or replace type fun_silly_type as object (fun
    varchar2(1000), silly number(8))
/
```

キューの作成/起動

```
--create the multi-consumer queue table, queue, and
    start it
begin
DBMS_AQADM.create_queue_table(queue_table=>'fun_silly_qt
    ab', queue_payload_type=>'sys.anydata',
    multiple_consumers=>TRUE);
DBMS_AQADM.create_queue(queue_name=>'fun_silly_q',
    queue_table=>'fun_silly_qtab');
DBMS_AQADM.start_queue(queue_name=>'fun_silly_q');
end;
/
```

これでキューが使用できます。

マルチ・コンシューマ・キュー用サブスクリバの追加

```
--add subscriber to the queue
declare
sub sys.aq$_agent;
begin
sub := sys.aq$_agent('sub1', NULL, NULL);
DBMS_AQADM.add_subscriber(
    queue_name=>'fun_silly_q',
    subscriber=>sub,
    delivery_mode=>DBMS_AQADM.PERSISTENT_OR_BUFFERED);
end;
/
```

運用インタフェース - AQ の使用

メッセージのエンキュー

```
-- Enqueue
declare
e_opt DBMS_AQ.enqueue_options_t;
m_prop DBMS_AQ.message_properties_t;
m_handle raw(16);
message anydata;

begin
-- --Buffered Message Flags
-- e_opt.delivery_mode := DBMS_AQ.buffered;
-- e_opt.visibility := DBMS_AQ.immediate;

--create the message payload
message :=
    ANYDATA.ConvertObject(fun_silly_type('blahblahblah
    blah', 172));

--enqueue
DBMS_AQ.enqueue(
    queue_name=>'fun_silly_q',
    enqueue_options=>e_opt,
    message_properties=>m_prop,
    payload=>message,
    msgid=>m_handle);
commit;
end;
/
```

メッセージのデキュー

```
declare
d_opt DBMS_AQ.dequeue_options_t;
m_prop DBMS_AQ.message_properties_t;
m_handle RAW(16);
message ANYDATA;
obj fun_silly_type;
ret pls_integer;

begin
-- --Buffered Message Flags
-- d_opt.delivery_mode := DBMS_AQ.buffered;
-- d_opt.visibility := DBMS_AQ.immediate;

d_opt.navigation := DBMS_AQ.first_message;
d_opt.consumer_name := 'sub1';
d_opt.wait := DBMS_AQ.no_wait;
```

```

DBMS_AQ.dequeue(
    queue_name=>'fun_silly_q',
    dequeue_options=>d_opt,
    message_properties=>m_prop,
    payload=>message,
    msgid=>m_handle);
ret := message.getobject(obj=>obj );
-- Output Object Information
DBMS_OUTPUT.put_line('Stuff: '||obj.fun|| ' and '
    ||obj.silly);
commit;
end;
/

```

キューの停止/削除

```

begin
DBMS_AQADM.stop_queue('fun_silly_q');
DBMS_AQADM.drop_queue('fun_silly_q');
DBMS_AQADM.drop_queue_table('fun_silly_qtab');
end;
/

```

簡単なインタフェース

チュートリアルでは、最も柔軟な構成用 API コールを使用していますが、制限付きで次の簡単な API コールを使用できます。詳細は、『Streams Advanced Queuing アプリケーション開発者ガイド』を参照してください。

目的	標準インタフェース	簡単なインタフェース
運用	<i>DBMS_AQ</i>	<i>DBMS_STREAMS_MESSAGING</i>
エンキュー	enqueue	enqueue
デキュー	dequeue	dequeue
管理	<i>DBMS_AQADM</i>	<i>DBMS_STREAMS_ADM</i>
キューの作成/起動	create_queue_table create_queue start_queue	set_up_queue
キューの停止/削除	stop_queue drop_queue drop_queue_table	remove_queue



Oracle Advanced Queuing: 概要とチュートリアル

2005年6月

著書: Toliver Jue

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

海外からのお問合せ窓口:

電話: +1.650.506.7000

ファックス: +1.650.506.7200

www.oracle.com

Copyright © 2005, Oracle. All rights reserved.

この文書はあくまで参考資料であり、掲載されている情報は予告なしに変更されることがあります。

オラクル社は、本ドキュメントの無謬性を保証しません。また、本ドキュメントは、法律で明示的または暗黙的に記載されているかどうかに関係なく、商品性または特定の目的に対する適合性に関する暗黙の保証や条件を含む一切の保証または条件に制約されません。オラクル社は、本書の内容に関していかなる保証もいたしません。また、本書により、契約上の直接のおよび間接的義務も発生しません。本書は、事前の書面による承諾を得ることなく、電子的または物理的に、いかなる形式や方法によっても再生または伝送することはできません。

Oracle、JD Edwards、PeopleSoftは米国 Oracle Corporation および関連会社の登録商標です。他の製品名は、それぞれの所有者の商標です。