

ORACLE®

ORACLE®

Oracle Database 12c Release 1

Global Data Services

日本オラクル株式会社
井上 克己

ORACLE®
DATABASE 12^c



Plug into the **Cloud**.

以下の事項は、弊社の一般的な製品の方向性に関する概要を説明するものです。また、情報提供を唯一の目的とするものであり、いかなる契約にも組み込むことはできません。以下の事項は、マテリアルやコード、機能を提供することをコミットメント(確約)するものではないため、購買決定を行う際の判断材料になさらないで下さい。オラクル製品に関して記載されている機能の開発、リリースおよび時期については、弊社の裁量により決定されます。

OracleとJavaは、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。文中の社名、商品名等は各社の商標または登録商標である場合があります。

アジェンダ

1. 複製データを活用している環境での現状
2. Global Data Services と RAC
3. 想定されるユース・ケース
4. グローバル・サービス
5. ロード・バランシング

GDS実装の背景

※ GDS ... Global Data Services

コンソリデーション vs ディストリビューション



マルチテナ
ント・アーキ
テクチャー

コンソリデーションの動機

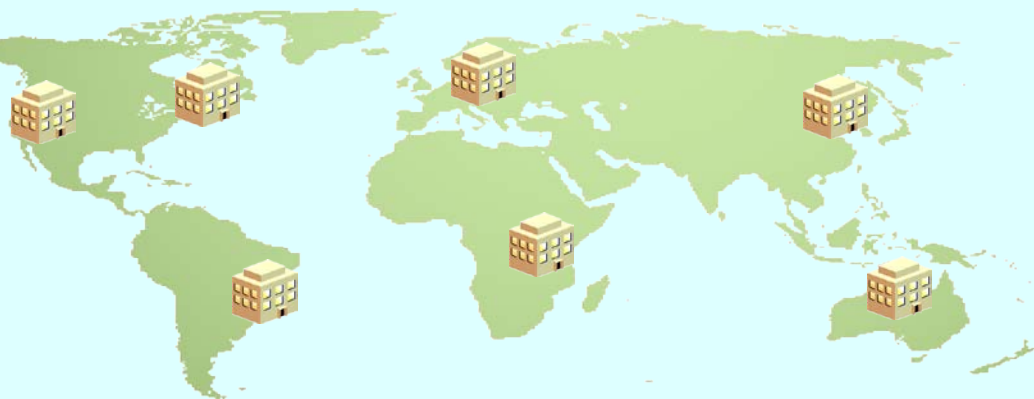
- 管理の容易さ
- コスト削減



ディストリビューションの動機

- DR ディザスタリカバリー
- ロードバランシング/ パフォーマンス

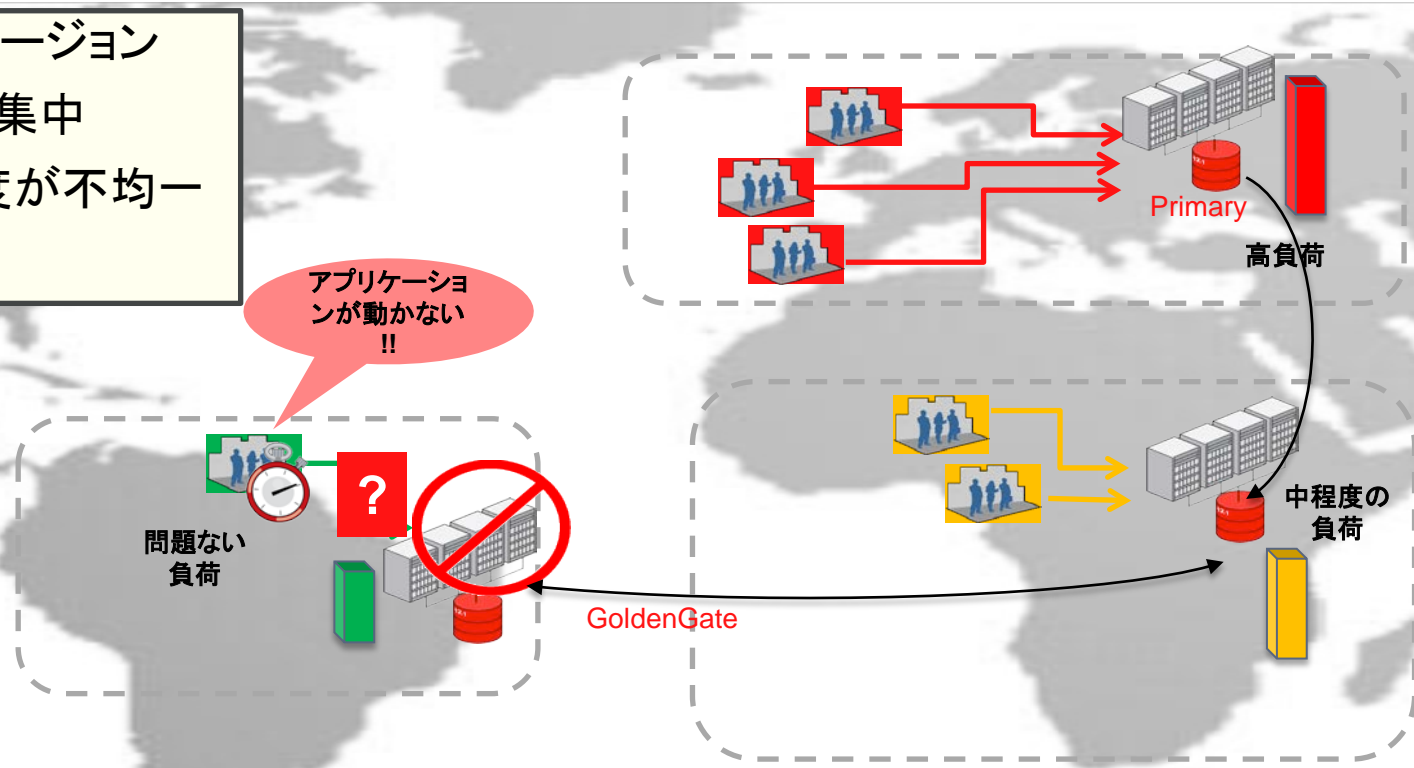
GDS



ORACLE

複製データを活用している環境での問題

- サイロ化されたリージョン
- 負荷がーか所に集中
- リソースの使用度が不均一
- 耐障害性がない

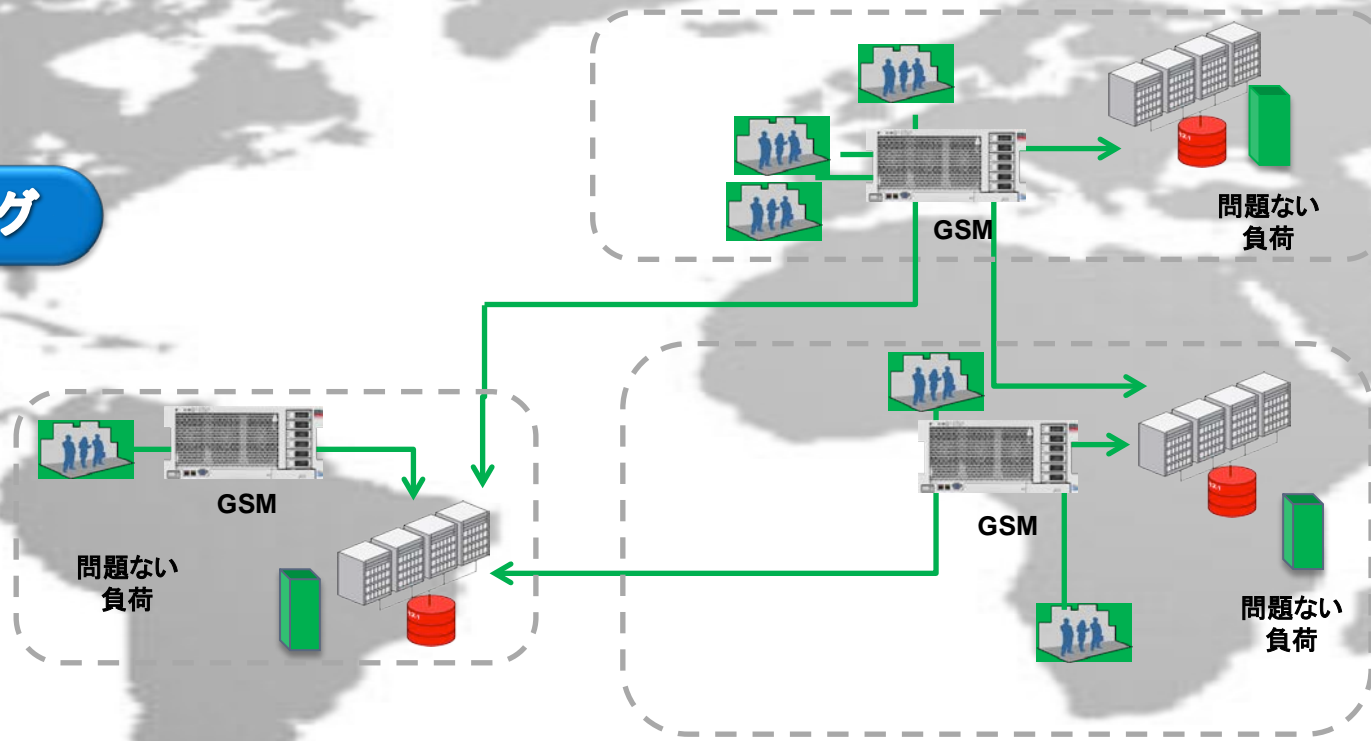


アジェンダ

1. 複製データを活用している環境での現状
2. Global Data Services と RAC
3. 想定されるユース・ケース
4. グローバル・サービス
5. ロード・バランシング

複製データを活用している環境 – GDS 導入後

ロードバランシング



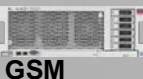
ORACLE

複製データを活用している環境 – GDS 導入後

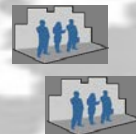
- 負荷を均等に分散
- リソース利用率の最適化
- 管理が楽に
- アプリケーションの耐障害性



耐障害性



SALES_REPORTING_SRVC



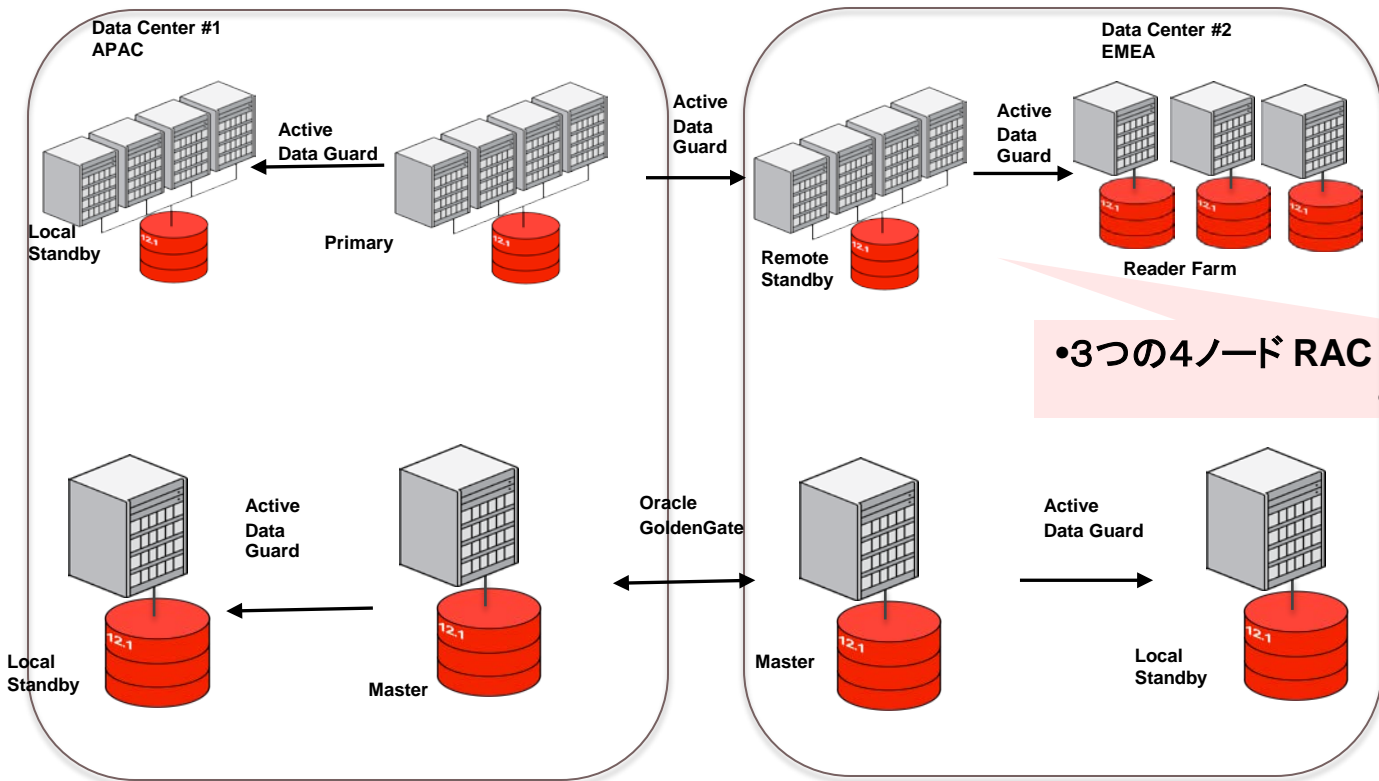
GSM



GSM



GDS 導入前後の物理的なイメージ — 導入前

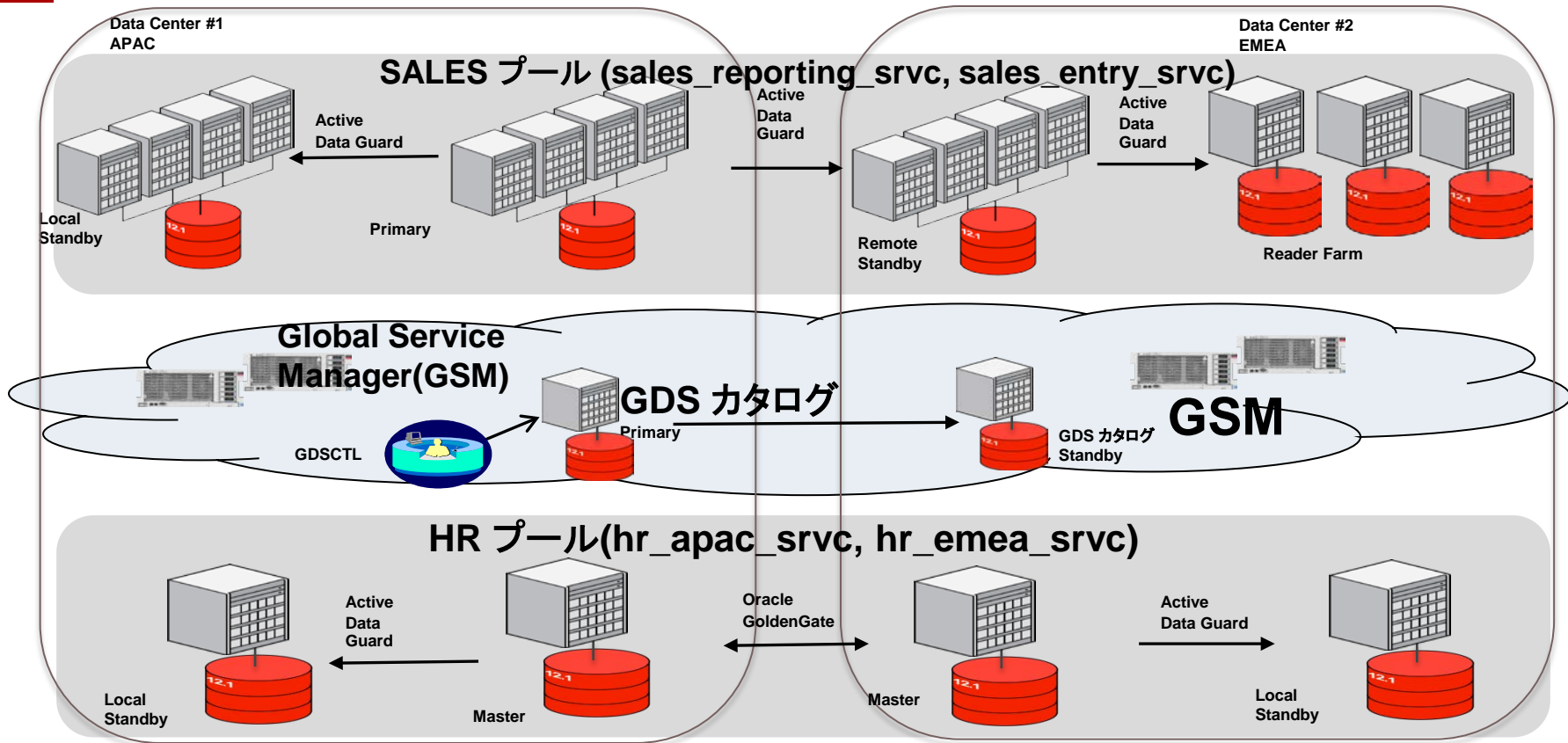


- データセンター 2カ所
- 2組の独立した複製グループ

• 3つの4ノード RAC と 3 シングル・インスタンスDB
• 混在可能

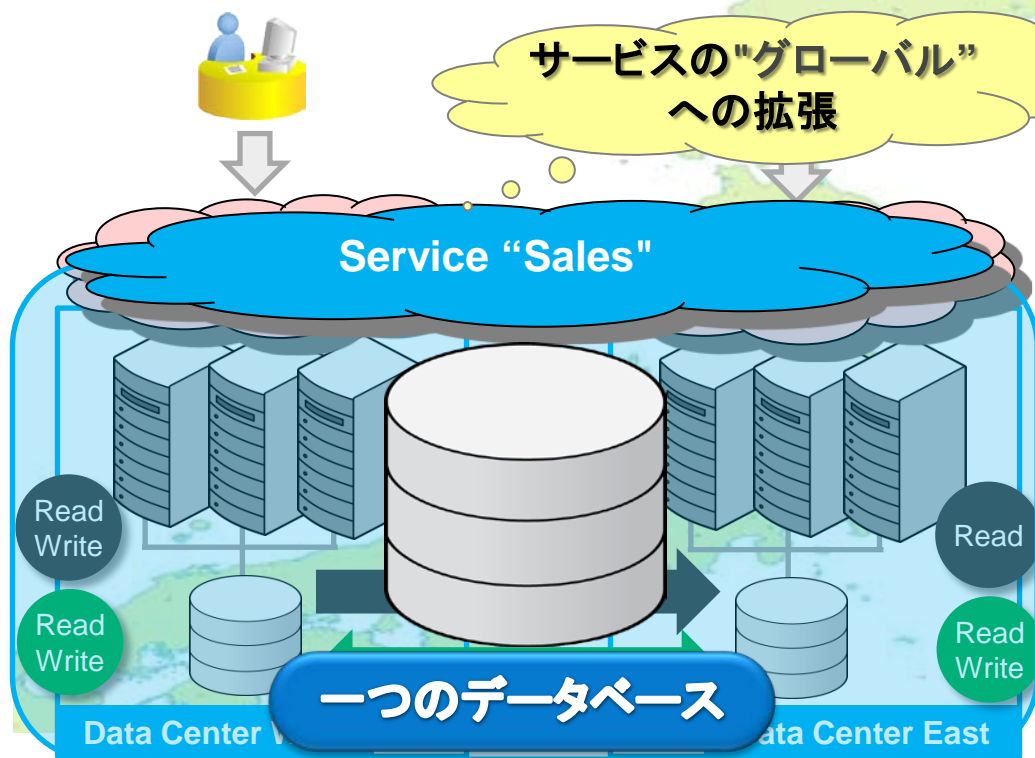
- レプリケーション手段は問わない
例) Streams, Advanced Repl, スクリプト, ストレージ・コピー

GDS 導入前後の物理的なイメージ - 導入後



Global Data Services – RAC クラスタ

独立したDB群を1つのRACクラスタのように使うことが可能に



GDSの特徴

- 負荷分散
DCを跨いだロードバランシングを実現
- 障害時フェイルオーバー
自動でのフェイルオーバーが可能
- 集中管理
単一(or少数)のサービスで一括管理

アジェンダ

1. 複製データを活用している環境での現状
2. Global Data Services と RAC
3. 想定されるユース・ケース
4. グローバル・サービス
5. ロード・バランシング

業務、アプリケーション の適性

全てのサーバーで最新データにアクセスできるとは限らないので制約がある

読み取りのみの処理
と書き込みありの処理
の分離

例)レポーティング系

データが最新でなくて
も良いか

例)
30秒前
前日
先月末

コネクション・プール
使用

ORACLE製プール

ユース・ケース – データベース側

大分類

読み取り専用ファーム
での負荷分散

(リーダー・ファーム、
Reader Farm)
Active Data Guard
GoldenGate

フェイル・オーバー
スイッチ・オーバー

Data Guard
Active Data Guard
コールド・スタンバイ

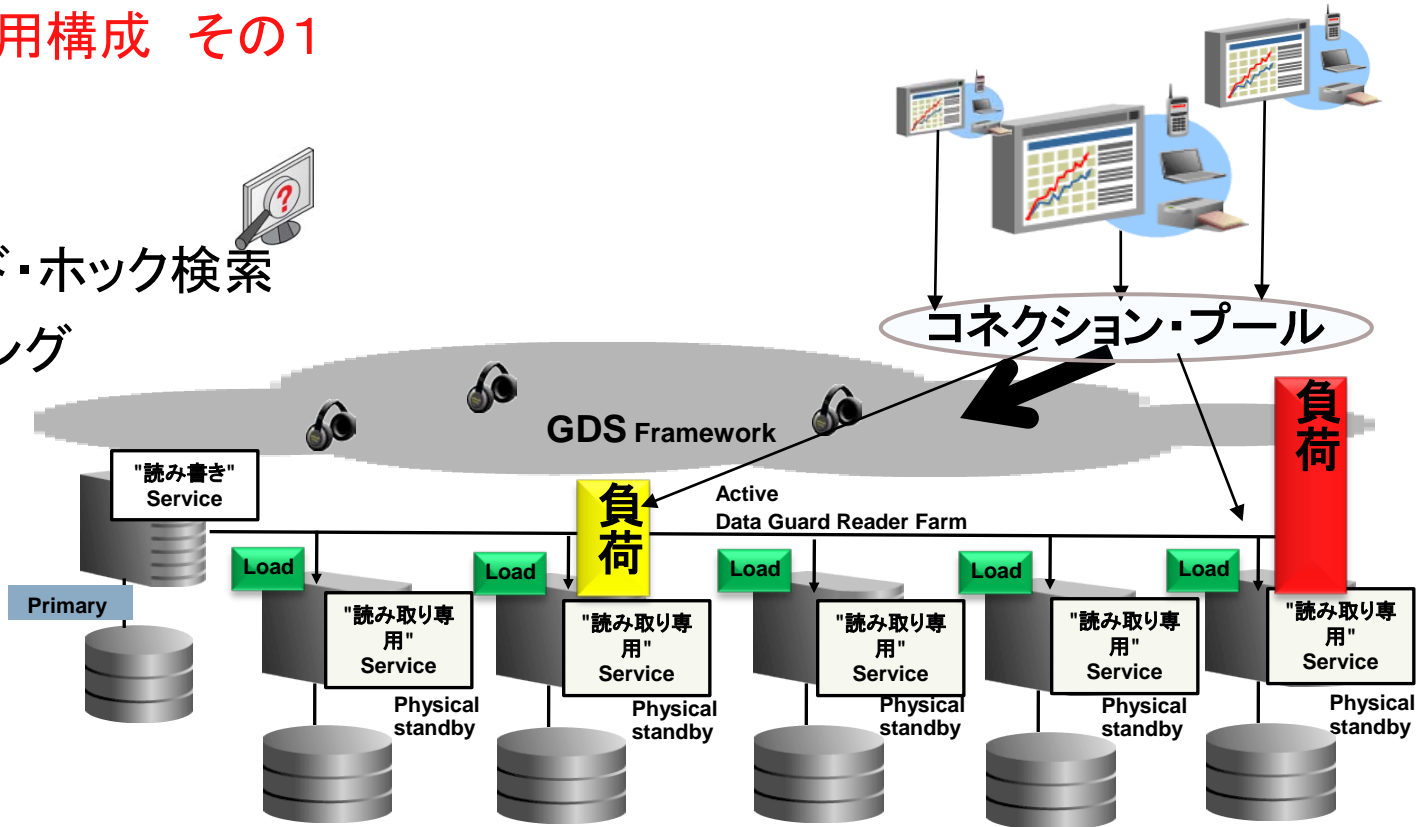
マルチ・マスター環境
での負荷分散

GoldenGate
Active-Active 構成

リーダー・ファームの使用用途と負荷の偏り

想定される使用構成 その1

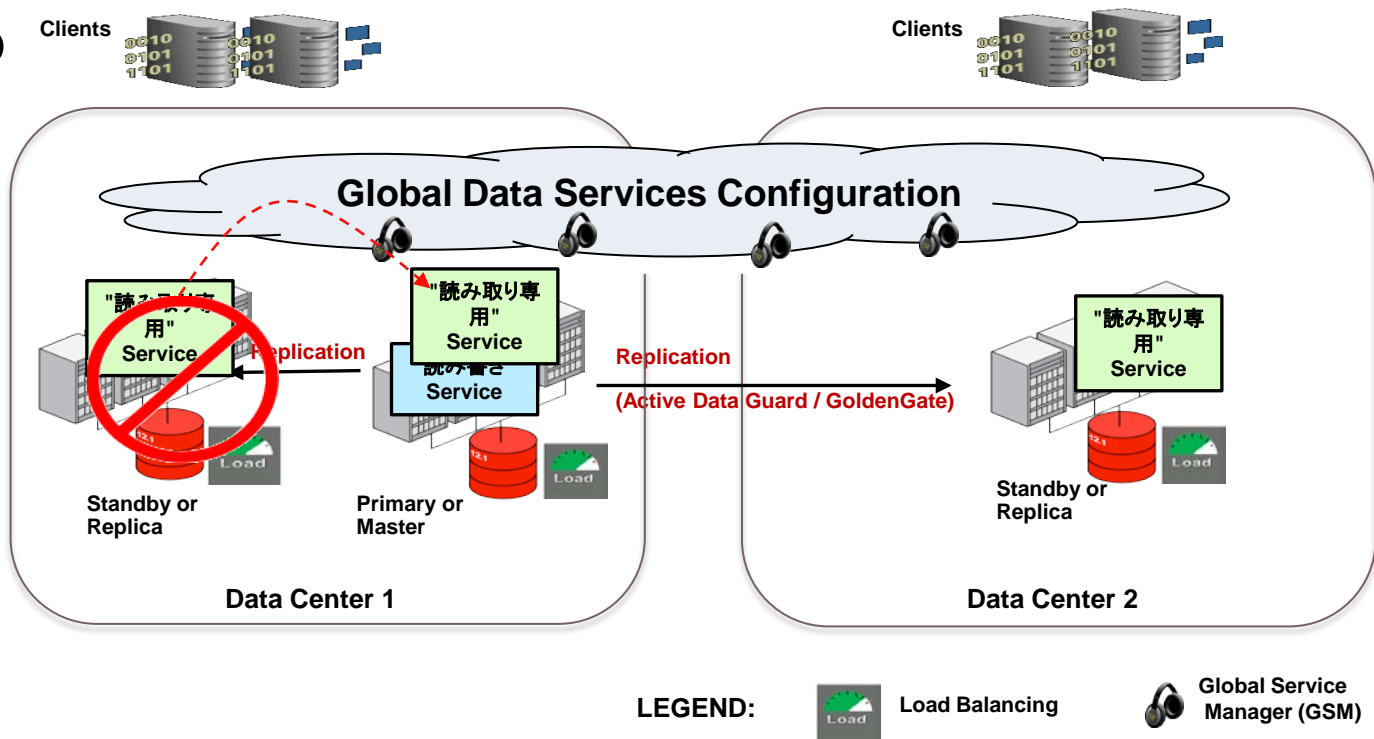
- バックアップ
- レポート作成
- 制限のないアド・ホック検索
- データ・マイニング



フェイル・オーバー — Data Guard, GoldenGate

想定される使用構成 その2

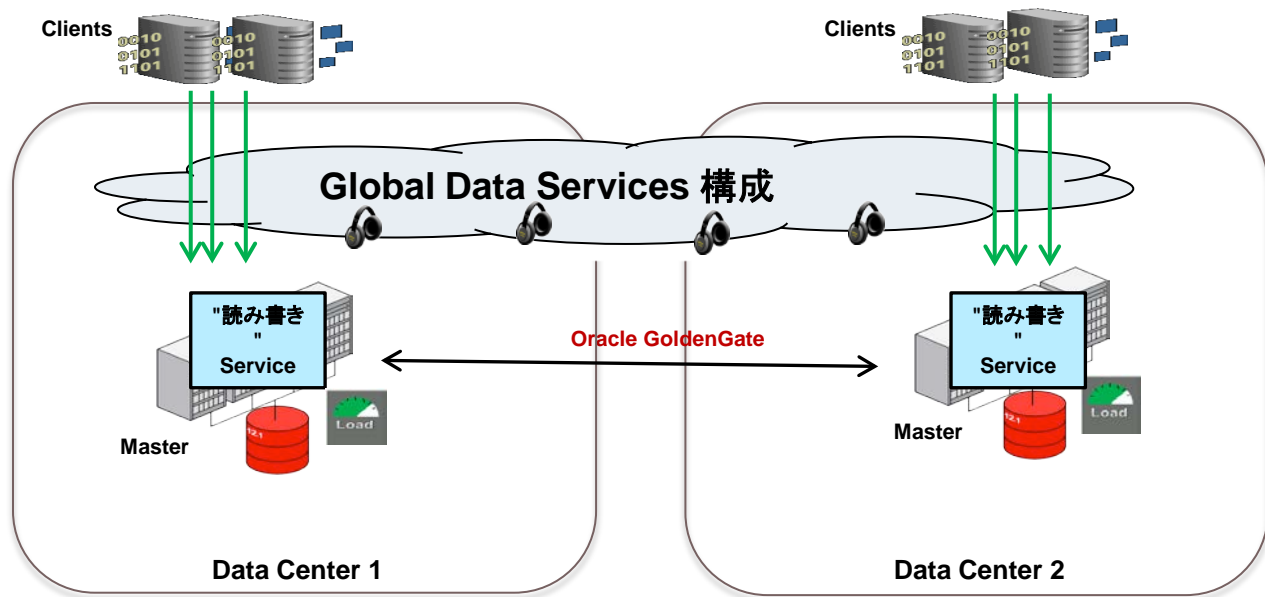
- “DBサービス”のフェイル・オーバー
- カーディナリティーを維持する
- DBが“OPEN”状態であることが前提です



GoldenGate Active-Active でのロード・バランシング

想定される使用構成 その3

- データ更新を含むアプリケーションのロードバランシング
- データ競合(コンフリクト)の解決は GoldenGate または アプリケーションで対応されることが想定されています。



LEGEND:



Load Balancing



Global Service Manager (GSM)

ORACLE

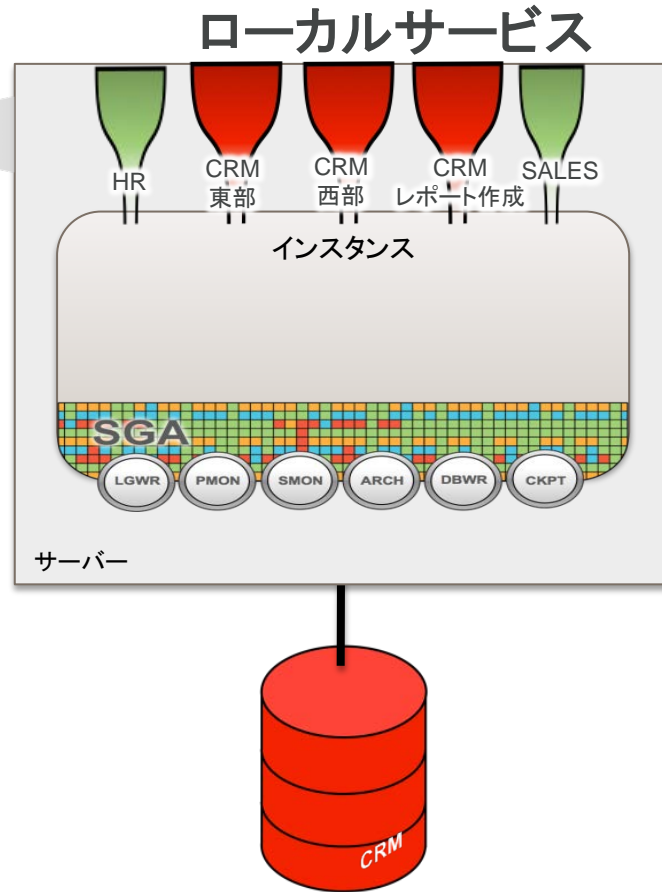
アジェンダ

1. 複製データを活用している環境での現状
2. Global Data Services と RAC
3. 想定されるユース・ケース
4. グローバル・サービス
5. ロード・バランシング

データベース “サービス”

- ローカルな”サービス”は以前のバージョンからあった仕組みで主にワークロード管理のために使われていました。
- シングルインスタンスDBでも作成可能
- クラスタリソースとして作成可能
- Global Data Services でのグローバルサービスと対比させるため旧来のサービスが “ローカル” サービスと呼ばれることがあります。

サービス



グローバル・サービス固有の 3 属性

ローカル・サービスには存在しない新しい属性

カーディナリティー

- サービスを提供する DBの数
- リージョン毎に設定可能

遅延(ラグ、LAG)

- Active Data Guard 固有
- REDO適用遅延

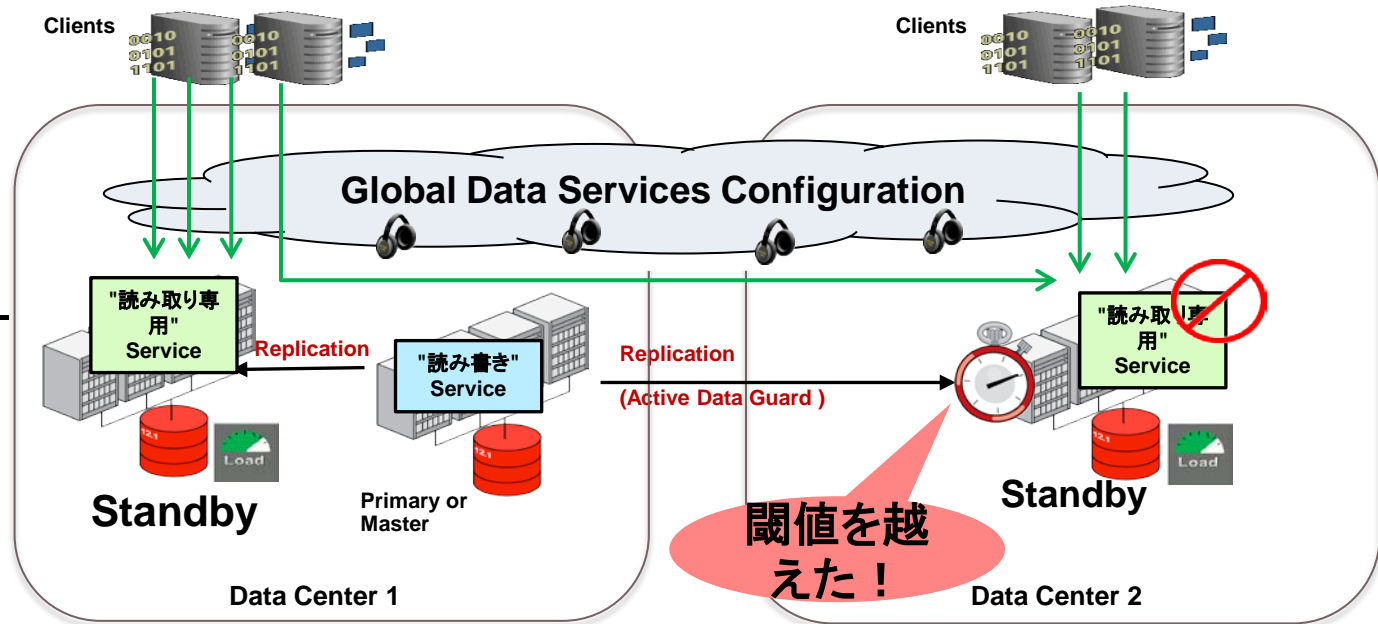
リージョン・アフィニティ

遠隔データセンターのDBへの接続を許可するかどうか

ラグ属性を持った”グローバル”サービスの動作

Active Data Guard スタンバイでのREDO適用遅延

- 1種のSLA (Service Level Agreement)
- 右図はプライマリー1に対し、ローカルとリモートのDCにそれぞれスタンバイが1個



LEGEND:



Load Balancing

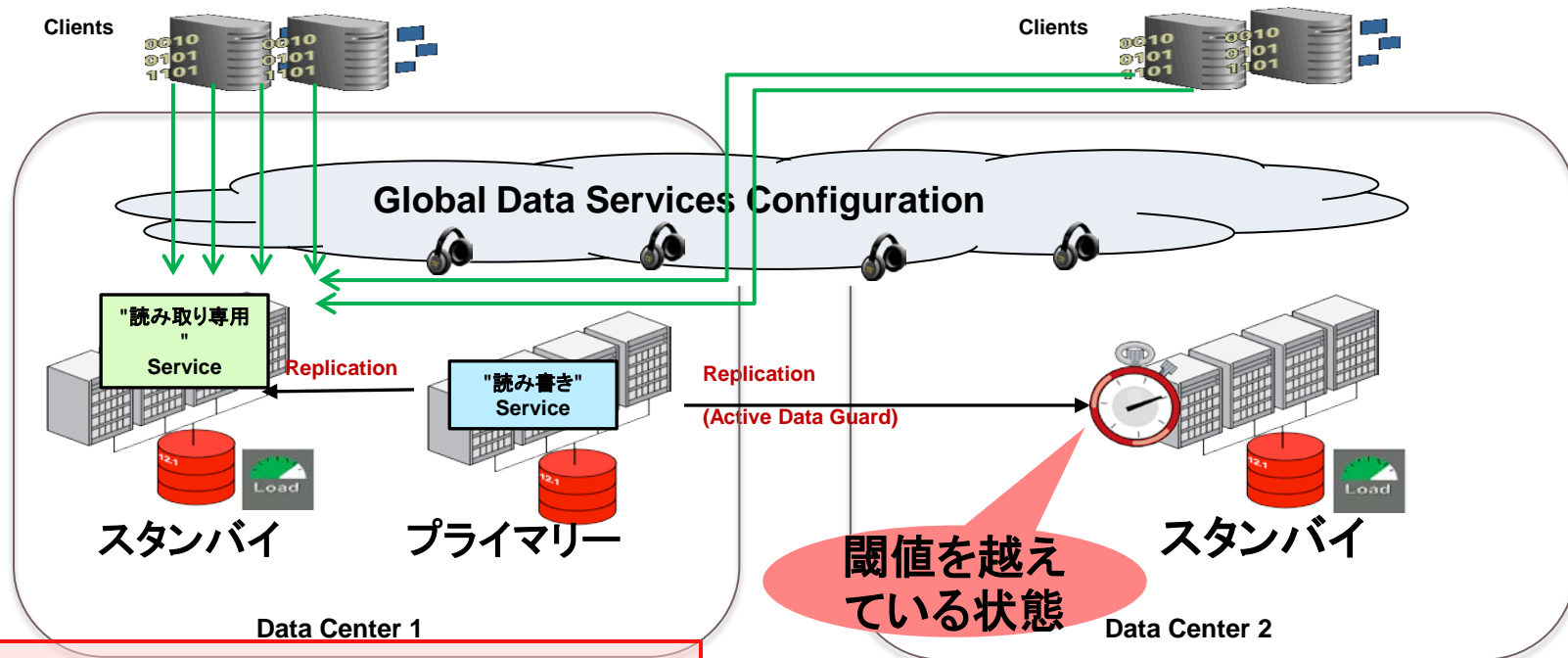


Global Service Manager (GSM)

ORACLE

ラグ属性を持った”グローバル”サービスの動作

Active Data Guard スタンバイでのREDO適用遅延



適用が追いついて再度閾値を下回ればサービスは再起動されます

LEGEND:



Load Balancing



Global Service Manager (GSM)

ORACLE

“リージョン・アフィニティ” サービス属性

“グローバル”サービス<->クライアント、および、サービス<->DB 間

全リージョン

- デフォルト
- ロード・バランシングの際はネットワーク・レイテンシーが計算に入る

ローカル・リージョン

- クライアントは自分自身が属しているリージョンで稼働しているサービスにのみ接続可能
- サービス・カーディナリティーはリージョン毎に保たれる

Local with Inter-Regional Fail-Over

ローカルに該当サービスが一個も起動してない場合のみ他リージョンを探す

アジェンダ

1. 複製データを活用している環境での現状
2. Global Data Services と RAC
3. 想定されるユース・ケース
4. グローバル・サービス
5. ロード・バランシング

サービスのロード・バランシング属性

```
GDSCTL>config service -service sales_read_only
```

名前: sales_read_only

ネットワーク名: sales_read_only.hr.oradbcloud

プール: hr

起動済: はい

すべて優先: はい

ローカリティ: ANYWHERE

リージョン・フェイルオーバー: いいえ

ロール: PHYSICAL_STANDBY

プライマリ・フェイルオーバー: はい

ラグ: ANY

ランタイム・バランス SERVICE_TIME

接続バランス LONG

通知: はい

TAFポリシー: NONE

ポリシー: AUTOMATIC

DTP: いいえ

フェイルオーバー・メソッド: NONE

フェイルオーバー・タイプ: NONE

フェイルオーバー再試行:

フェイルオーバー遅延:

エディション:

PDB:

コミット結果:

保存タイムアウト:

再生開始タイムアウト:

セッション状態一貫性: DYNAMIC

SQL翻訳プロファイル:

SERVICE_TIME または THROUGHPUT または NONE

データベース

データベース

優先

ステータス

--

dbvm3

はい

有効

dbvm4

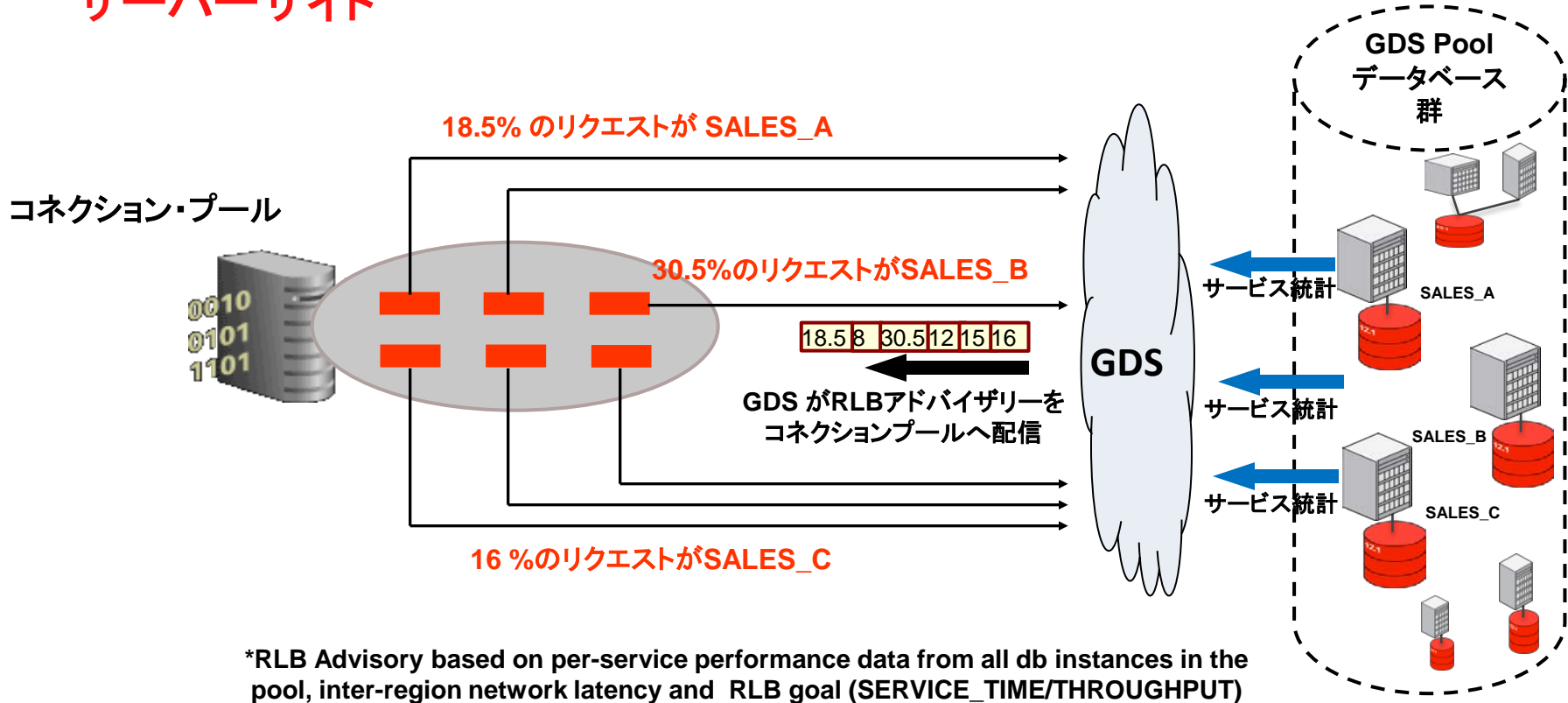
はい

有効

LONG
または
SHORT

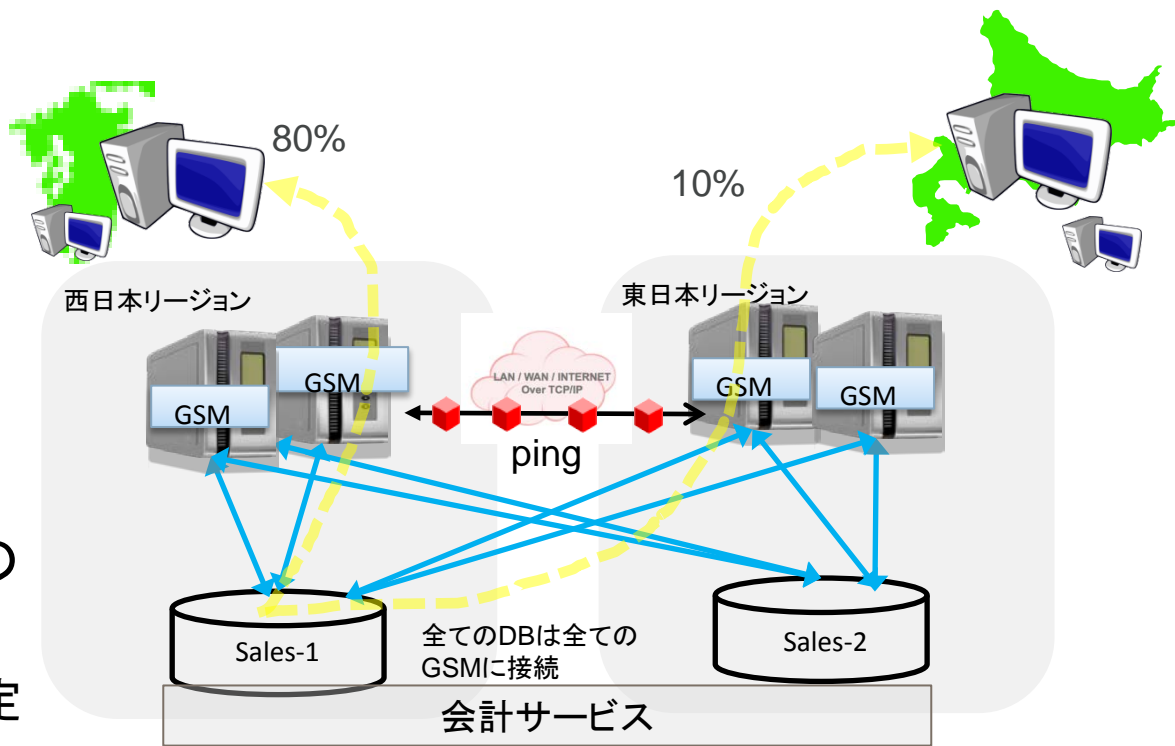
グローバル・ランタイム・ロード・バランシング (RLB)

サーバーサイド



DB12c RAC でのロードバランシングとの比較

- ロードバランシングの設定や計算方法は 11g,12c RAC と 12c の GDS は似ています。
- 異なる点: GSMは各DBから取ってくる負荷情報に REGION 間のネットワーク的な距離を加味した値をクライアントにガイドします
- RLB_GOAL が THROUGHPUT の時はDBの負荷情報のみでガイド
 - (DB時間) ≫ network と仮定



ORACLE®

Oracle Database 12c Release 1

Application Continuity

日本オラクル株式会社
井上 克己

ORACLE®
DATABASE 12^c



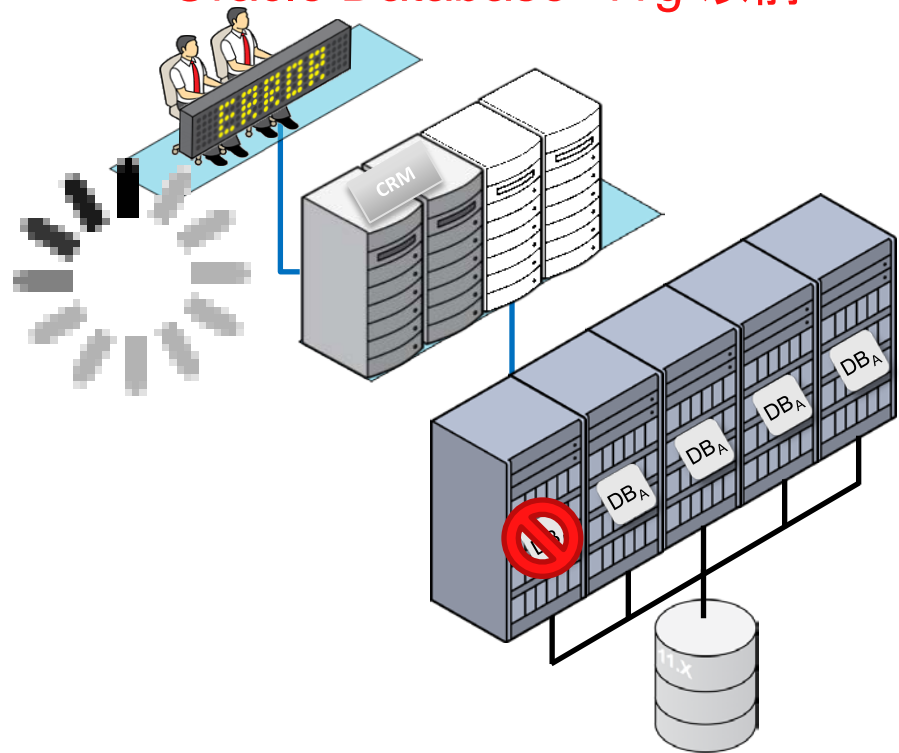
Plug into the **Cloud**.

アジェンダ

1. **アプリケーションのフェイルオーバー: 現状**
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

想定外(計画外)の停止、クラッシュ

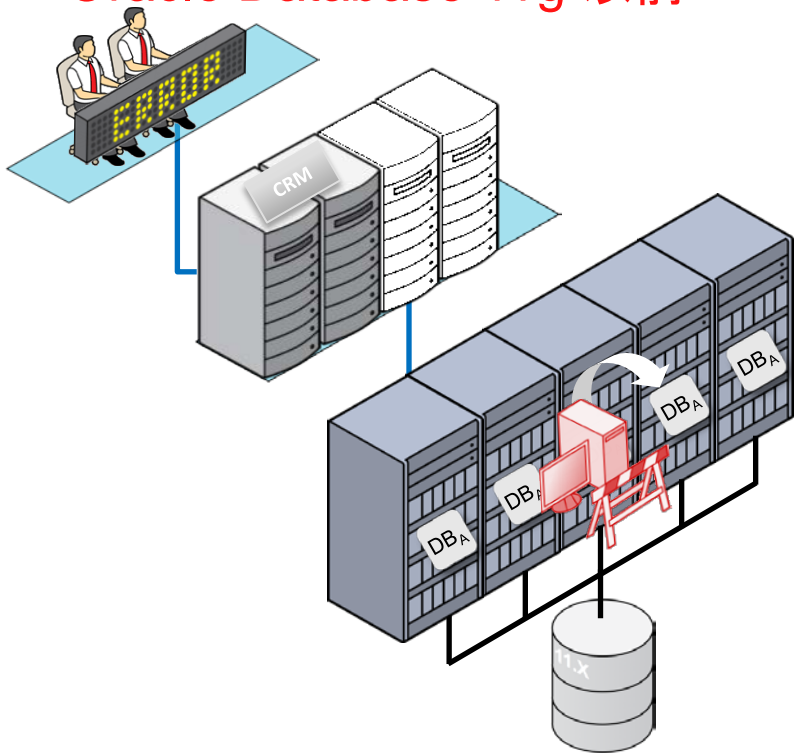
Oracle Database 11g 以前



- データベースの異常停止により実行中だったトランザクションのどこまでをコミットされたのか、どこから再入力する必要があるのか判断できないケースがありました。
- 2重コミット
- 不明トランザクションをクリーンアップするための中間層の初期化
- DB異常に対応するためのアプリケーションコードの複雑化

計画停止時にトランザクションが失われるケース

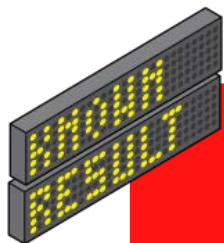
Oracle Database 11g 以前



- 計画停止の方が非計画停止より頻度が多い傾向があります
- FAN と Oracle Connection Pool(Weblogic,UCP,OCI,..) の組み合わせによりアイドル(使われていない)接続のクリーンアップは 10gR2 から可能
- FAN と Oracle Connection Pools でも長時間接続を離さないアプリには対応できていなかった

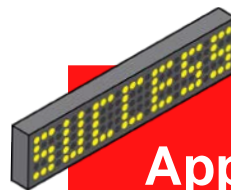
アプリケーション開発で実装が難しい点を解決

Oracle Database 12c 新機能



Transaction Guard

最後のトランザクション結果を確実に取り出すためのプロトコルとAPI



Application Continuity

想定外の停止、計画停止時に実行中だったセッションを安全に再実行する

アジェンダ

1. アプリケーションのフェイルオーバー: 現状
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

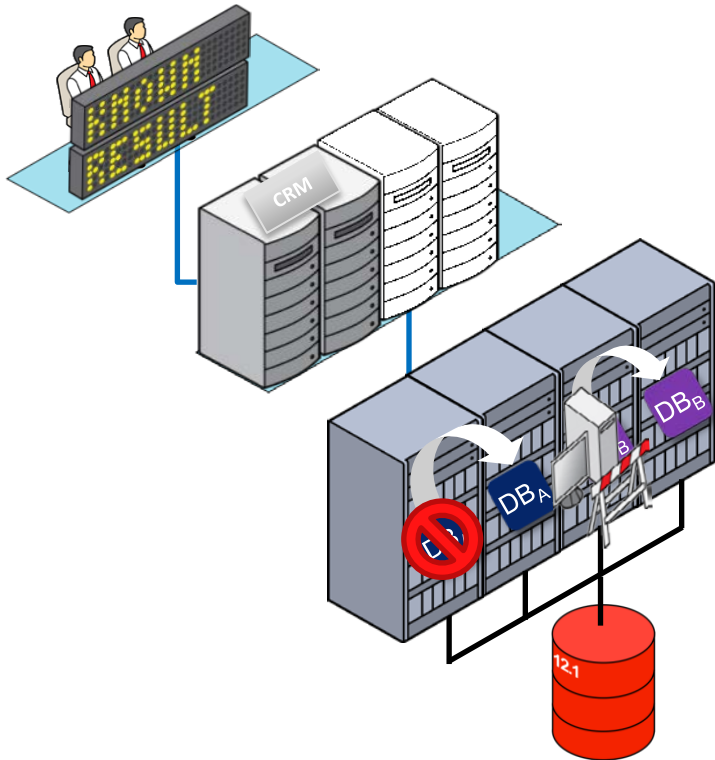
Transaction Guard



Plug into the **Cloud.**

Transaction Guard

コミット結果を保存。後から取り出し可能



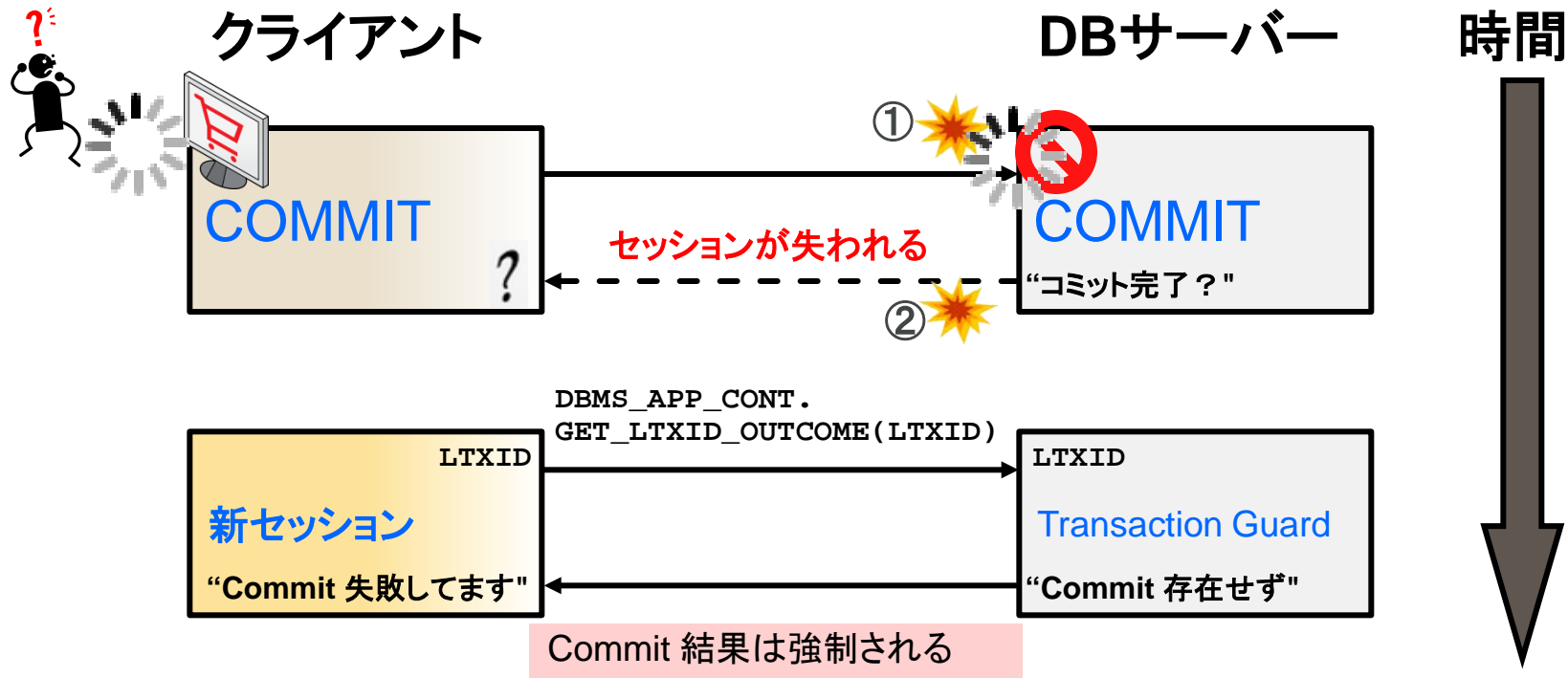
- トランザクションのコミット結果を取り出す API
- このAPIを使うことにより2重発注などのよくある問題を未然に防ぐことができる
- Transaction Guard APIによりアプリケーションは自分で再実行するのか、または、エンドユーザーに再入力を促すかなどの判断が可能に
- “Application Continuity for Java” 機能でも使われる
- JDBC-Thin, C/C++ (OCI/OCCL), ODP.Net クライアント用 API

アジェンダ

1. アプリケーションのフェイルオーバー: 現状
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

API使用 および 動作

コミット発行後に返事がなく実際には 失敗していた場合



アジェンダ

1. アプリケーションのフェイルオーバー: 現状
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

Application Continuity for Java



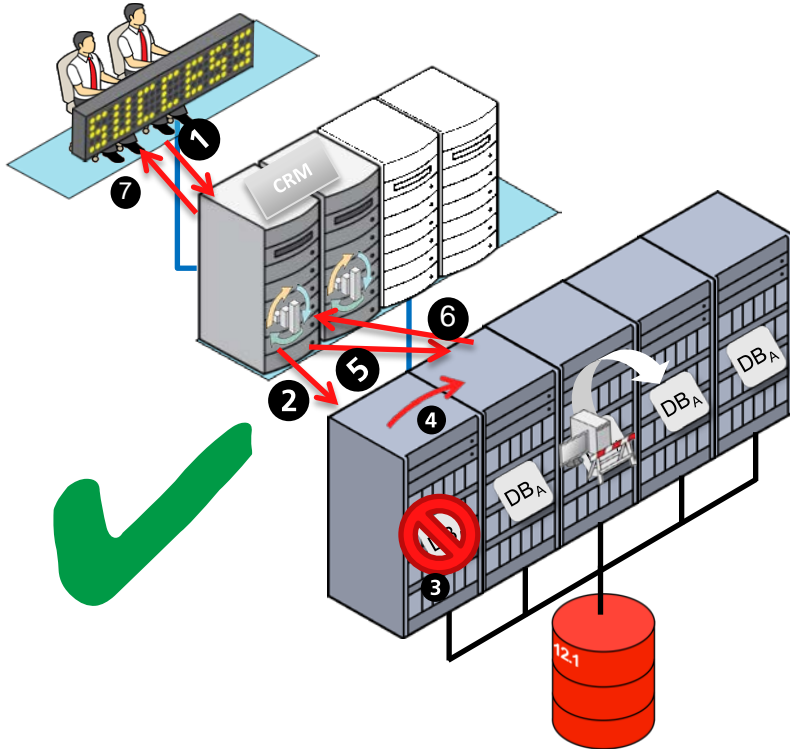
Plug into the **Cloud.**

アジェンダ

1. アプリケーションのフェイルオーバー: 現状
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

Application Continuity for Java

計画外停止、および、計画停止をエンドユーザーに気づかせない



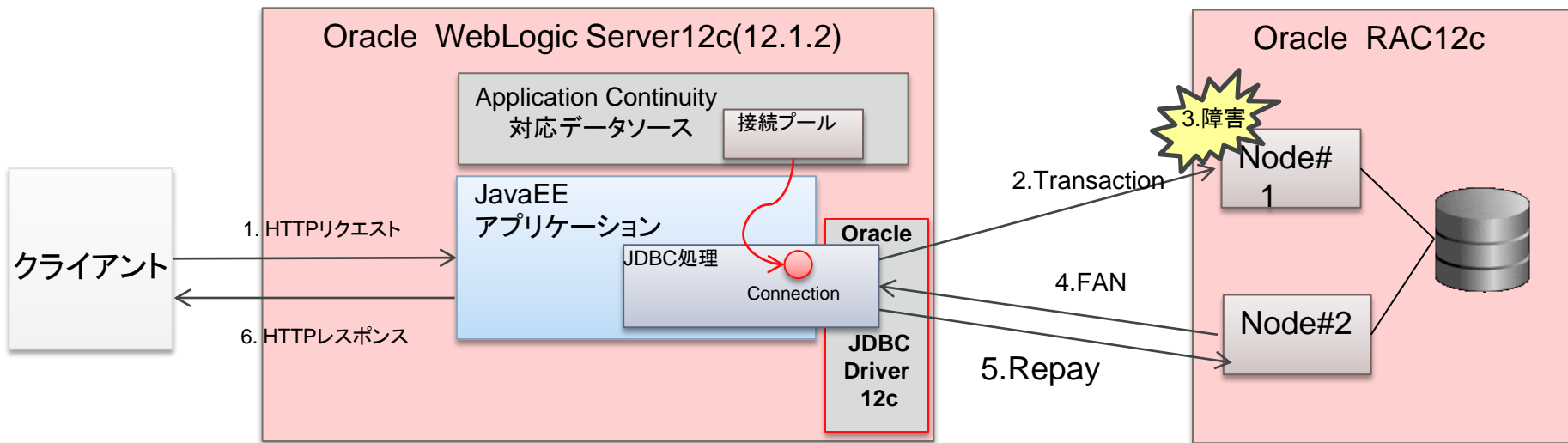
- Recoverable(回復可能)エラー発生時にDML(トランザクション)を再実行し再度データ投入
- 多くのハードウェア、ネットワーク、OS、ストレージのエラーや停止をエンドユーザーに気づかせずにトランザクション実行

アジェンダ

1. アプリケーションのフェイルオーバー: 現状
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

WebLogicデータソースのApplication Continuity対応

- WebLogic12c(12.1.2)では Application Continuityに対応したデータソースを構成できます。
- これにより、WebLogicで動作するアプリケーションがDB障害発生に透過的にJDBC処理をフェイルオーバー(Replay)させることが可能になります。



データソースの作成

新規接続ファクトリクラス

- 「ドライバ・クラス名」が「oracle.jdbc.replay.OracleDataSourceImpl」になっていることを確認
- DB12c に含まれるJDBCドライバーに含まれる
 - ojdbc6.jar
 - ojdbc7.jar
- UCP(Universal Connection Pool)でもこのクラスを使い Application Continuity 機能を利用することができます

メッセージ

✔ 接続テストが成功しました。

新しいJDBCデータソースの作成

構成のテスト | 戻る | 次 | 終了 | 取消

データベース接続のテスト

データベースの可用性、および指定した接続プロパティをテストします。

接続プールでのデータベース接続の作成に使用するJDBCドライバ・クラスの完全パッケージ名を指定してください。
(このドライバ・クラスは、デプロイ先のいずれかのサーバーのクラスパスに含まれる必要があります。)

ドライバ・クラス名: **oracle.jdbc.replay.OracleDataSourceImpl**

接続先データベースのURLを指定してください。使用するJDBCドライバによって、URLの書式が異なります。

URL: jdbc:oracle:thin:@//

使用例①: アプリからのINSERT文発行

- Application Continuity データソースを利用し、INSERT文を発行するテスト用のJavaEEアプリケーションをWebLogicにデプロイします。
- INSERT文の対象となる表は、行が入っていない状態にしておきます。
- アプリケーションから(敢えて時間のかかる)INSERT文を発行します。

Firefox

デプロイメントのサマリー - base_... x 接続中...

SimpleDS (UPDATE, INSERT ,DELETE)

DataSource: jdbc/db12cAC

SQL:

```
insert into actest values (1,sleep(20),sysdate)
```

Submit

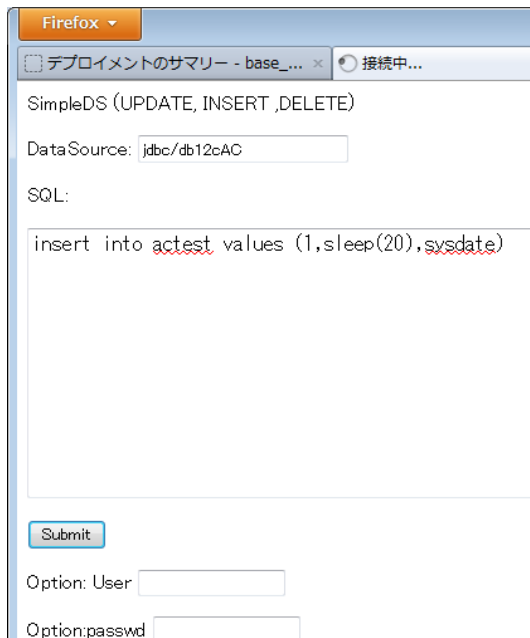
Option: User

Option:passwd

使用例②: アプリ側は実行中の状態であることを確認

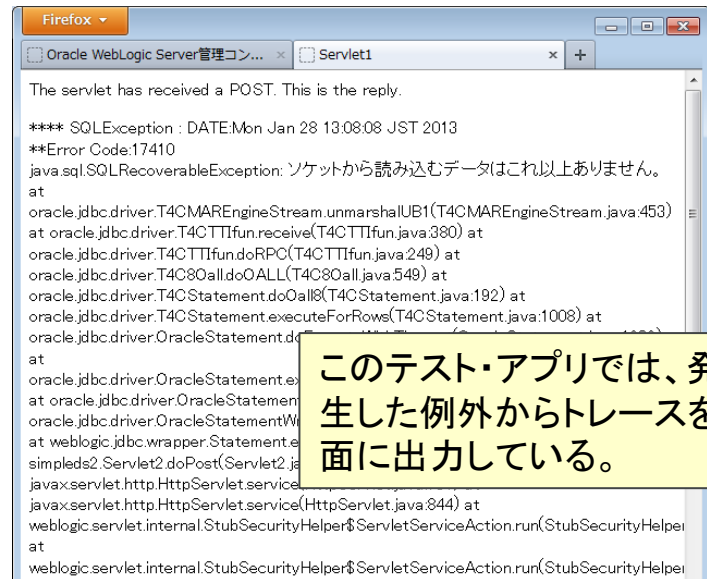
- データベースを強制停止(shutdown abort) しても、アプリはまだ実行状態であることを確認します。

アプリはエラーにならず、実行中の状態



The screenshot shows a web browser window with a single tab titled "デプロイメントのサマリー - base_...". The page content includes a form for testing database operations. At the top, it says "SimpleDS (UPDATE, INSERT,DELETE)". Below that, there is a text input field for "DataSource:" containing "jdbc/db12cAC". Underneath is a "SQL:" label followed by a larger text area containing the SQL query: "insert into actest values (1,sleep(20),sysdate)". At the bottom of the form, there is a "Submit" button and two more input fields labeled "Option: User" and "Option:passwd".

【参考】Application Continuity対応データソースを使用していない場合、同じ処理でも下図のようにエラーになる。



The screenshot shows a web browser window with two tabs: "Oracle WebLogic Server管理コン..." and "Servlet1". The main content area displays a stack trace for a database error. The text reads: "The servlet has received a POST. This is the reply." followed by "**** SQLException : DATE:Mon Jan 28 13:08:08 JST 2013" and "**Error Code:17410". The stack trace includes the following lines: "java.sql.SQLException: ソケットから読み込むデータはこれ以上ありません.", "at oracle.jdbc.driver.T4CMAREngineStream.unmarshalUB1(T4CMAREngineStream.java:453)", "at oracle.jdbc.driver.T4CTTIfun.receive(T4CTTIfun.java:380)", "at oracle.jdbc.driver.T4CTTIfun.doRPC(T4CTTIfun.java:249)", "at oracle.jdbc.driver.T4C8Oall.doOALL(T4C8Oall.java:549)", "at oracle.jdbc.driver.T4CStatement.doOall8(T4CStatement.java:192)", "at oracle.jdbc.driver.T4CStatement.executeUpdateForRows(T4CStatement.java:1008)", "at oracle.jdbc.driver.OracleStatement.d...", "at oracle.jdbc.driver.OracleStatement.ex...", "at oracle.jdbc.driver.OracleStatement...", "at oracle.jdbc.driver.OracleStatementW...", "at weblogic.jdbc.wrapper.Statement.e...", "at simplesds2.Servlet2.doPost(Servlet2.ja...", "at javax.servlet.http.HttpServlet.servic...", "at javax.servlet.http.HttpServlet.servic...", "at weblogic.servlet.internal.StubSecurityHelper\$ServletServiceAction.run(StubSecurityHelper...", "at weblogic.servlet.internal.StubSecurityHelper\$ServletServiceAction.run(StubSecurityHelper...".

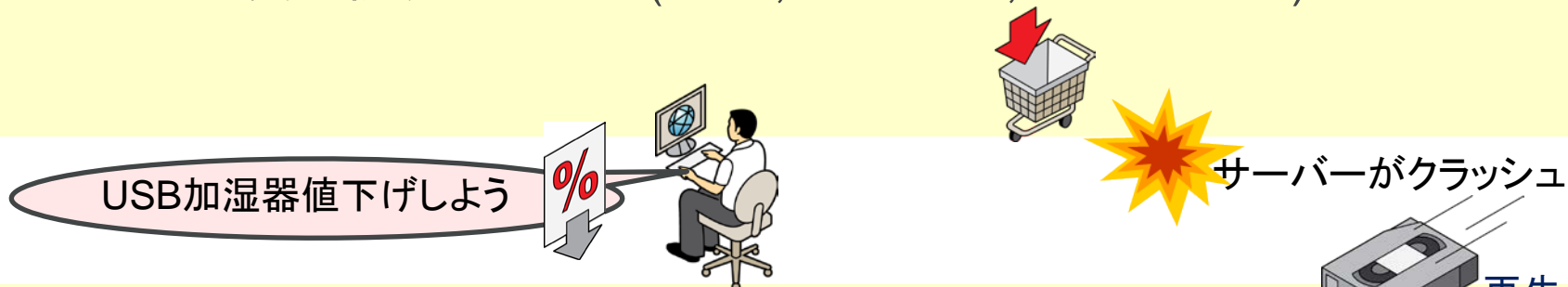
このテスト・アプリでは、発生した例外からトレースを画面に出力している。

アジェンダ

1. アプリケーションのフェイルオーバー: 現状
2. Transaction Guard
 - i. 動作
3. Application Continuity for Java
 - i. 動作概要
 - ii. Weblogic Server 12.1.2 での使用
 - iii. 動作詳細

再生時に結果が変わる場合、再生は中断される

```
UPDATE 買い物カゴ SET OWNER='U12345' WHERE ...  
SELECT メーカー、値段 FROM 商品 WHERE .. 値段 < 3000  
INSERT INTO 買い物候補 values(2480,'BrandCo','USB加湿器')
```



```
UPDATE 買い物カゴ SET OWNER='U12345' WHERE ...  
SELECT メーカー、値段 FROM 商品 WHERE .. 値段 < 3000
```

```
INSERT INTO 買い物候補 values(2480,'BrandCo','USB加湿器')
```



SQLException: ORA-41412: results changed during replay; failover cannot continue

ORACLE

初回実行時と同じ値を使うよう設定が可能なMutable

Mutable: 呼び出す毎に違う値を返す関数

シーケンス

mySeq.NEXTVAL

タイムスタンプ

SYSDATE
SYSTEMTIMESTAMP

ユニークID生成

SYS_GUID

*再生時に Parallel Query で実行される場合同じ値は使われません

補足: Database Replay(RAT,Real Application Testing) にある機能と似た機能

Hardware and Software

ORACLE®

Engineered to Work Together

ORACLE®