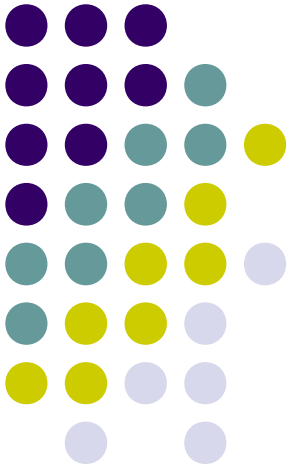
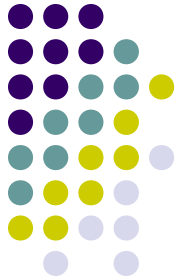
マルチ・スレッド・アンローダー新機能

version 4.1



強化ポイント

- 対応プラットフォームの拡大
  - x64ベースのWindowsへ対応
- 高信頼性強化
  - TAFイベント対応
  - 領域あふれ時のフェイルオーバー
- 高性能化
  - 並列化可能なパイプ処理
  - 名前付きパイプ
  - ヒープ構成表の並列アンロード
  - 集合化クエリ
- 機能追加
  - メインフレーム向けデータ移行
  - 並列処理データの単一ファイル出力

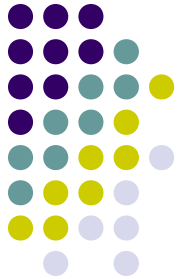


対応プラットフォームの拡大

x64ベースのWindowsへ対応

- 旧版ではx64ベースのWindowsでMTUを使うには32-bitのOracle Clientを導入する必要がありました。
- 新しい版ではx64ベースのWindowsでネイティブに動作する実行形式ファイルを追加しました。
- 従来通りx86ベースWindowsに対応する実行形式ファイルも提供されます。
- x86用、x64用の間で有意な性能差はありません。

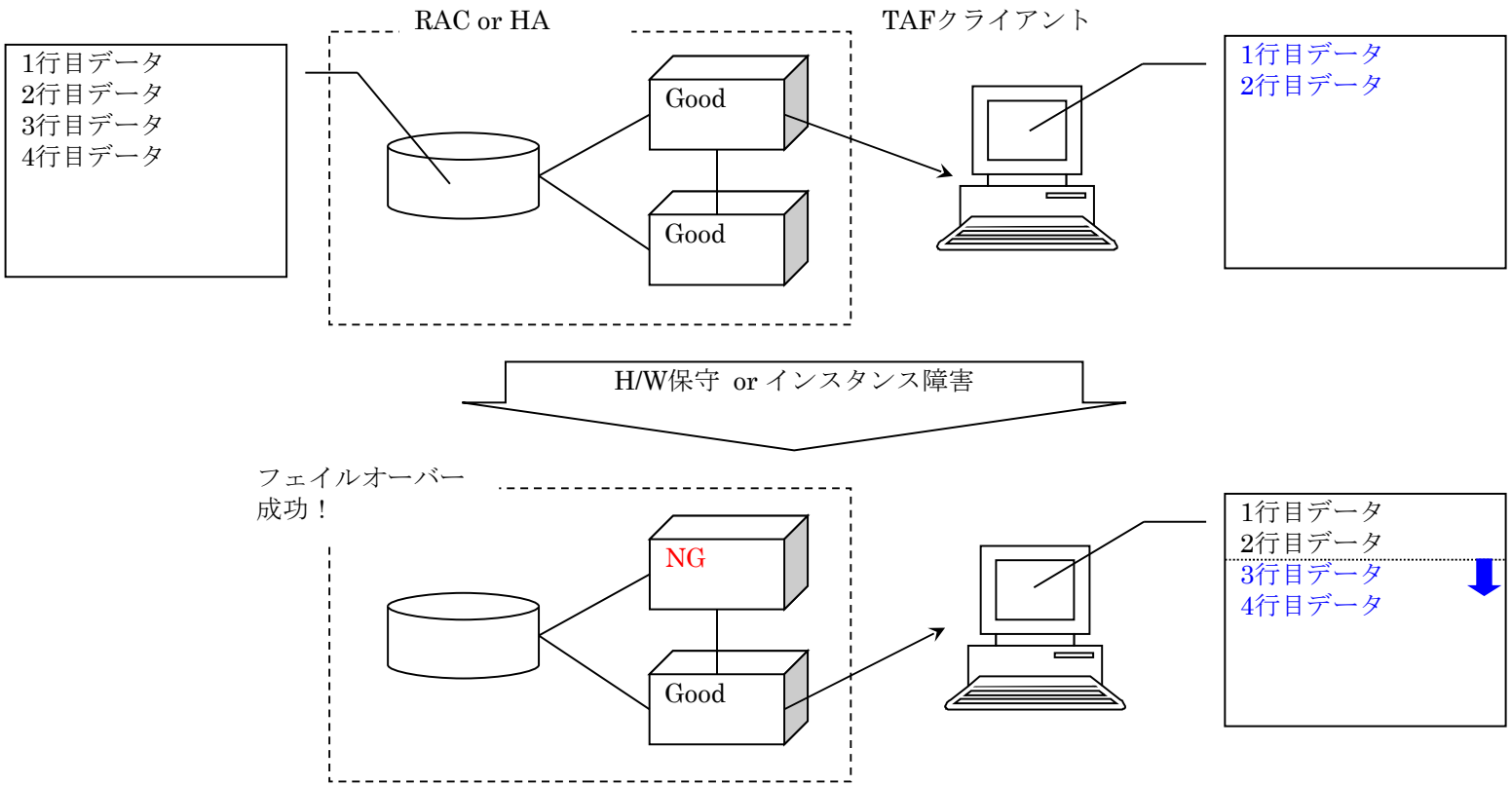
製品を動作させる条件として、Microsoft 社が提供しているMicrosoft Visual C++ 2010 SP1再頒布可能パッケージ (x86) またはMicrosoft Visual C++ 2010 SP1再頒布可能パッケージ (x64) を事前にインストールしていただく必要がございます。詳しくはMicrosoft 社のホームページでご確認ください。



高信頼性強化

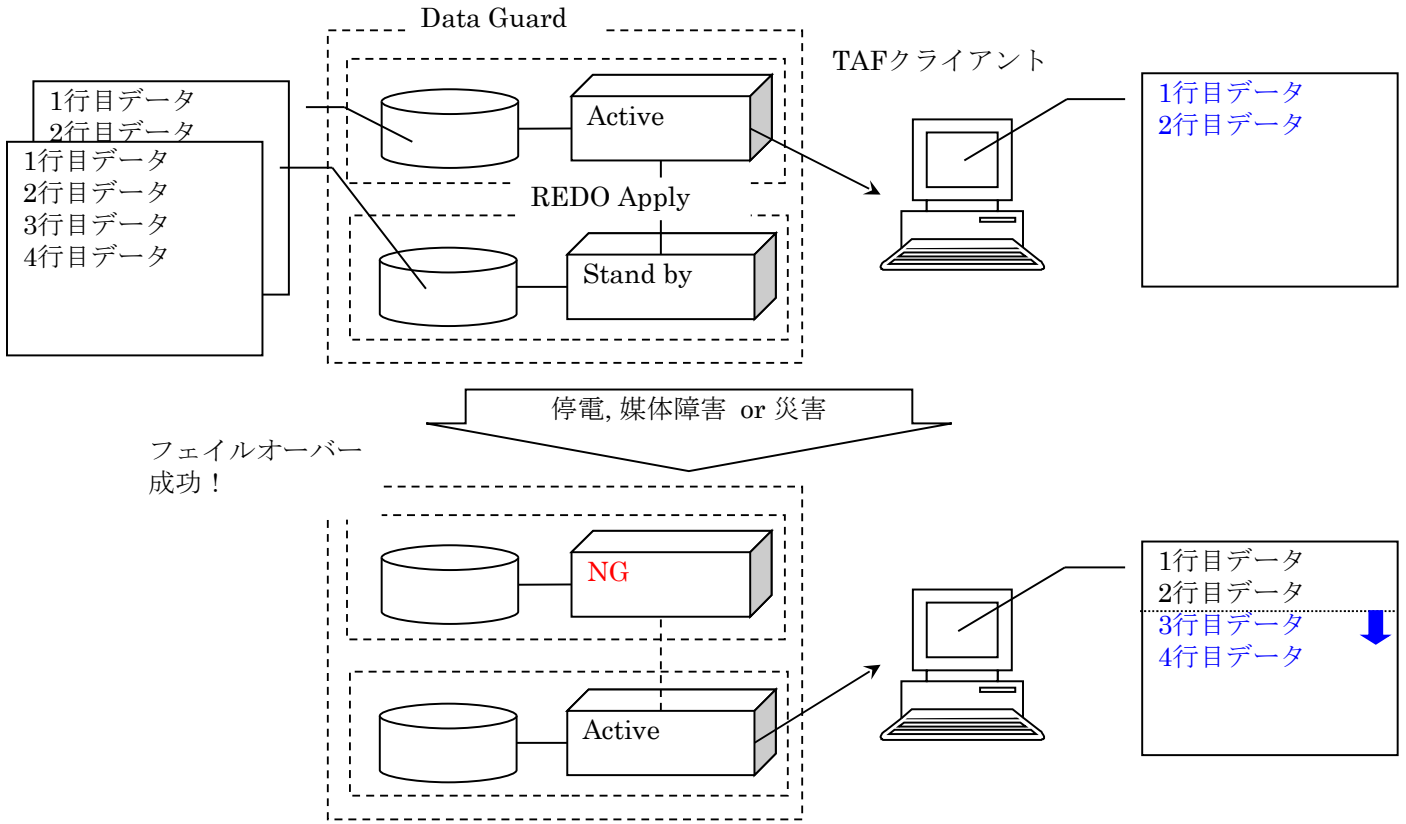
TAFイベント対応

- Real Application Clusters またはHA構成

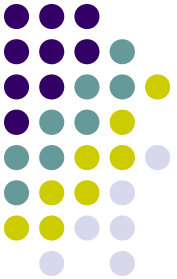


TAFイベント対応

- Data Guard構成



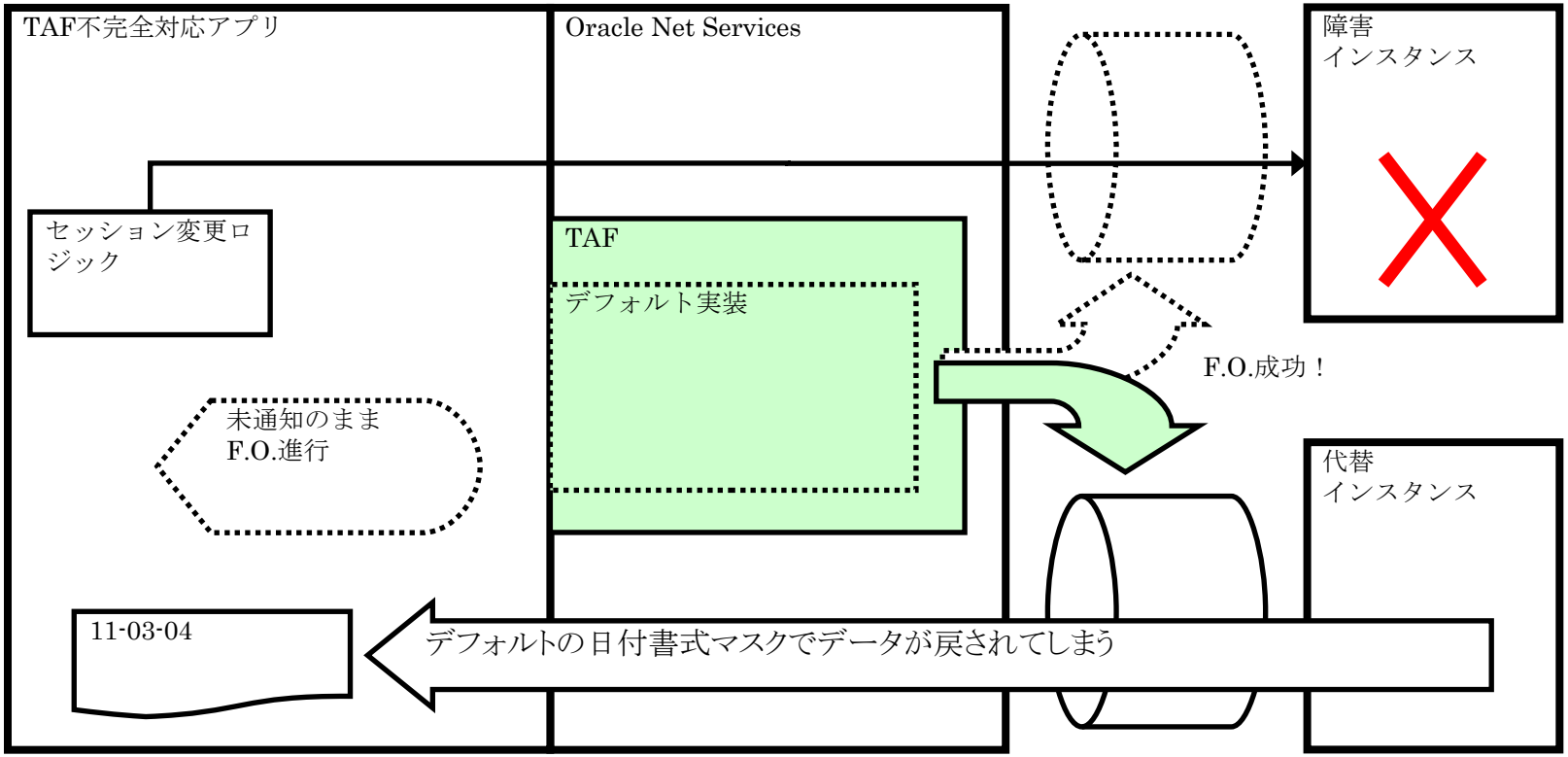
TAFイベント対応



- TAFはクライアント・ソフトウエア向けの高信頼性システムを構築する為のオプションです。Oracle Database Net Servicesによって機能が提供されます。TAFが構成されていると、データ読取中にインスタンス障害が発生しても、その影響がクライアントへ及ぶのを回避することが出来ます。
- TAFが有効な接続でフェイルオーバーに成功すると、まだ読み取られていなかった箇所から読み取りが再開されます。フェイルオーバーが無かった場合の読み取り結果と同一のデータセットが戻される点が“透過的“とされる理由です。
- TAFクライアントの接続先として典型的にはRACが選ばれますが、その他にもOracle Fail Safe(OFS)などのHA、Data Guard、アドバンスト・レプリケーションなど、耐障害性を高めたOracle Databaseを選ぶことが出来ます。

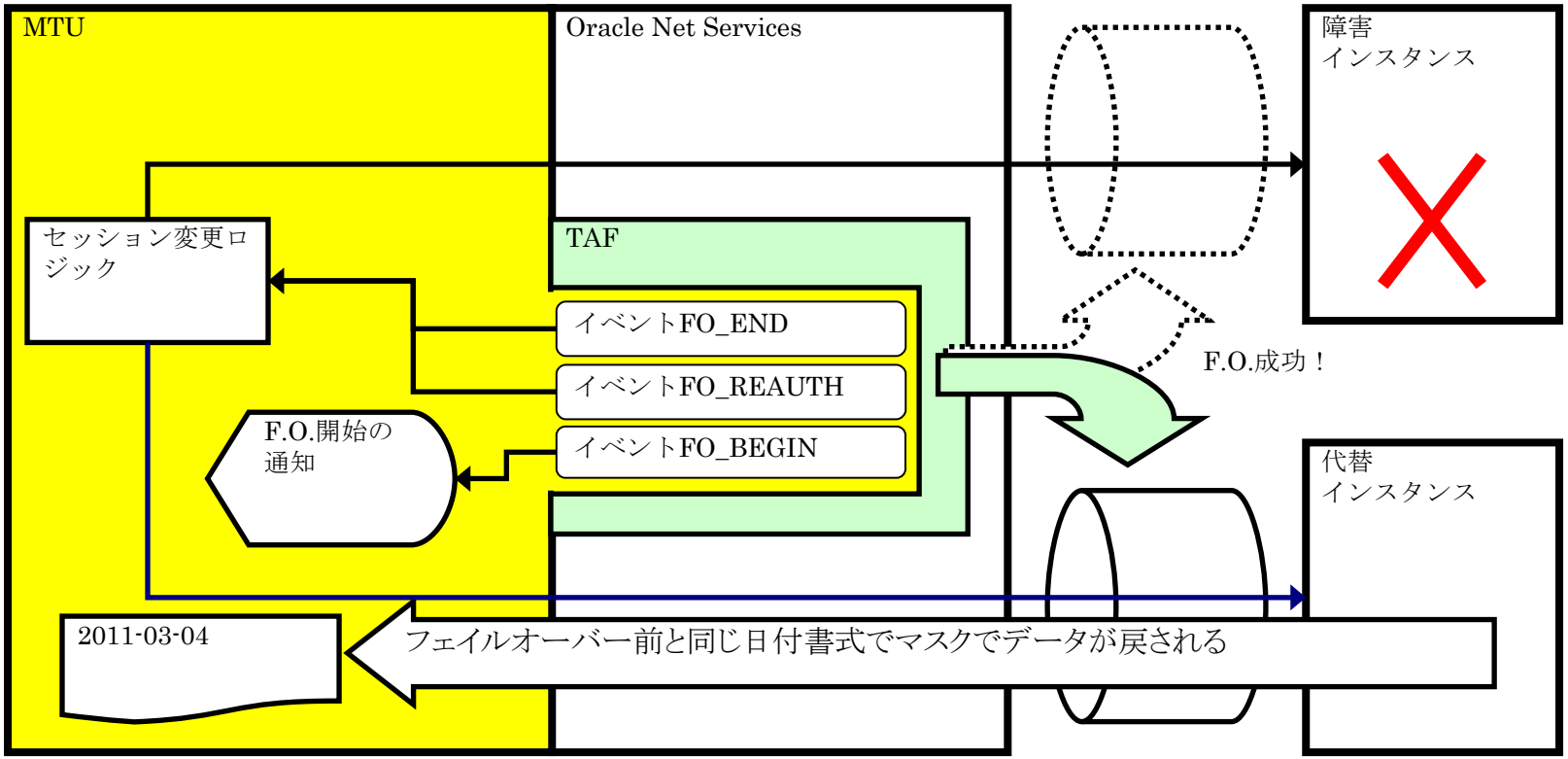
TAFイベント対応

- 不完全対応アプリ



TAFイベント対応

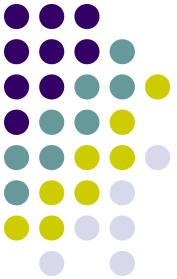
- 完全対応のMTU



TAFイベント対応

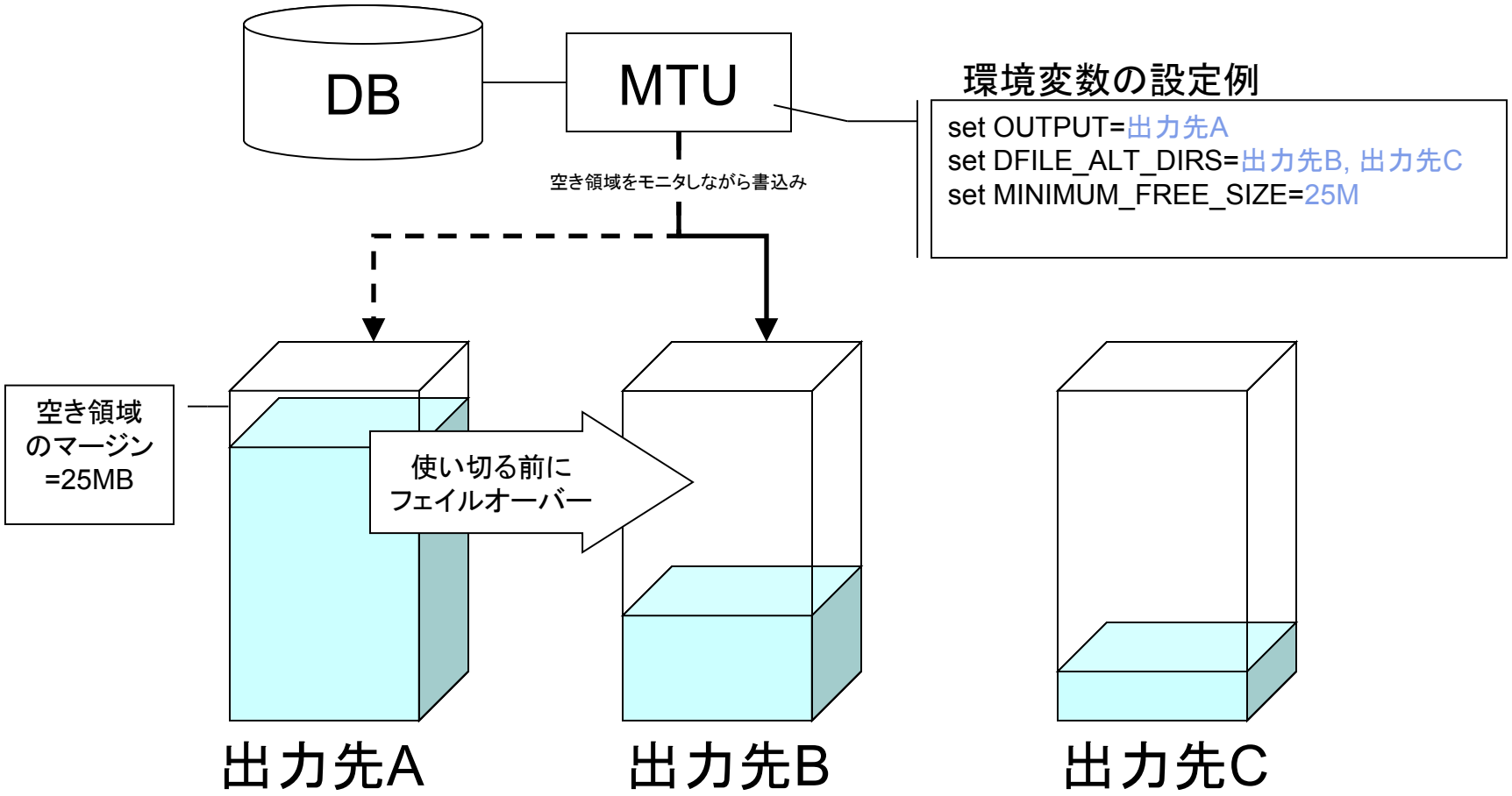
- MTUはOracle Databaseの高可用性オプションを効果的に応用できる日本初の高速アンローダーです。
- データ読取中に接続先インスタンスがダウンしても影響を回避するよう回復動作を試みます。フェイルオーバーに成功すれば読取を再開するのでリランの必要がありません。
- 停電や防災訓練、ファームウエアのアップグレードなどサーバが停止するイベント毎にジョブ・オペレータの介入が必要な状況（MTUの実行時期を調節など）を圧倒的に少なくすることが出来ます。

TAFイベント対応

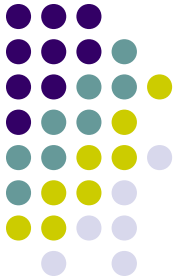


- クライアント・プログラムがTAFを有効に利用するにはFAILOVER_MODEサブパラメータを含む接続記述子(tnsnames.ora)を構成することの他に、非同期に発生する何種類かのTAFイベントを適切にハンドリングするコールバック関数をOracle Net Servicesへフックしておく必要があります。
- MTUの実装では、フェイルオーバーが開始されると直ちに画面へ通知されるのでユーザーはハングと区別することが出来ます。
- フェイルオーバー前に有効だったalter session set nls_date_format等のセッション・パラメータをフェイルオーバー成功後に再送出するので、出力されたデータの日付書式がフェイルオーバーを挟んでデフォルトに戻ってしまう問題は発生しません。

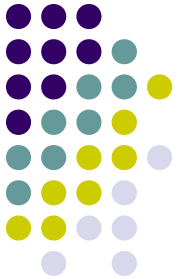
領域あふれ時のフェイルオーバー



領域あふれ時のフェイルオーバー



- 旧版では、ディスクが一杯になるまで書き込み続けるので、領域不足で他のアプリケーションへ悪影響を与える可能性がありました。
- 新しい版では、ディスクの空き領域不足に備えて複数の代替出力先を用意しておくことが可能になりました。
- ディスクを使い切ってからではなく、予め定めたマージン量を下回った時にフェイルオーバーさせることが出来ます。
- 制御ファイルのINFILE句のパスはフェイルオーバー後の名前へ自動的に書き換えられます。
- すべての代替出力先がマージン量以下になった時には自ら処理を中断しフェイルセーフします。
- 領域不足を理由とするダウンタイムを削減できます。

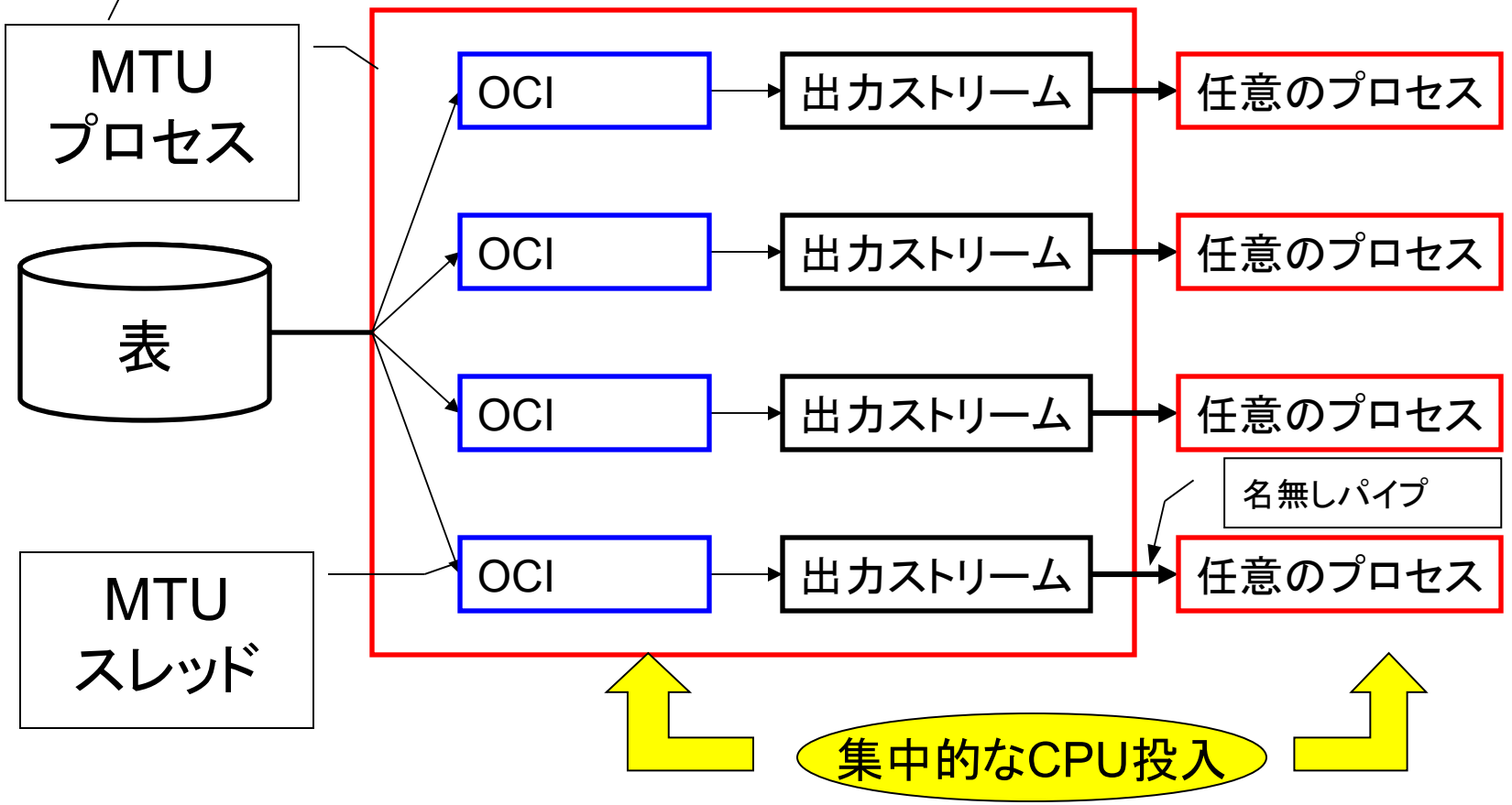


高性能化

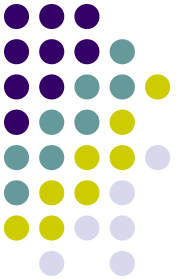
並列化可能なパイプ処理

環境変数の設定例

```
set STDOUT=0x0001
set STREAM_LOCATOR=ipc_pipe://zip "{O}\{C}_{D=yyyyMMdd}_{W=HHmmss}_{E=SRC_USER}_{X}" -v -
```

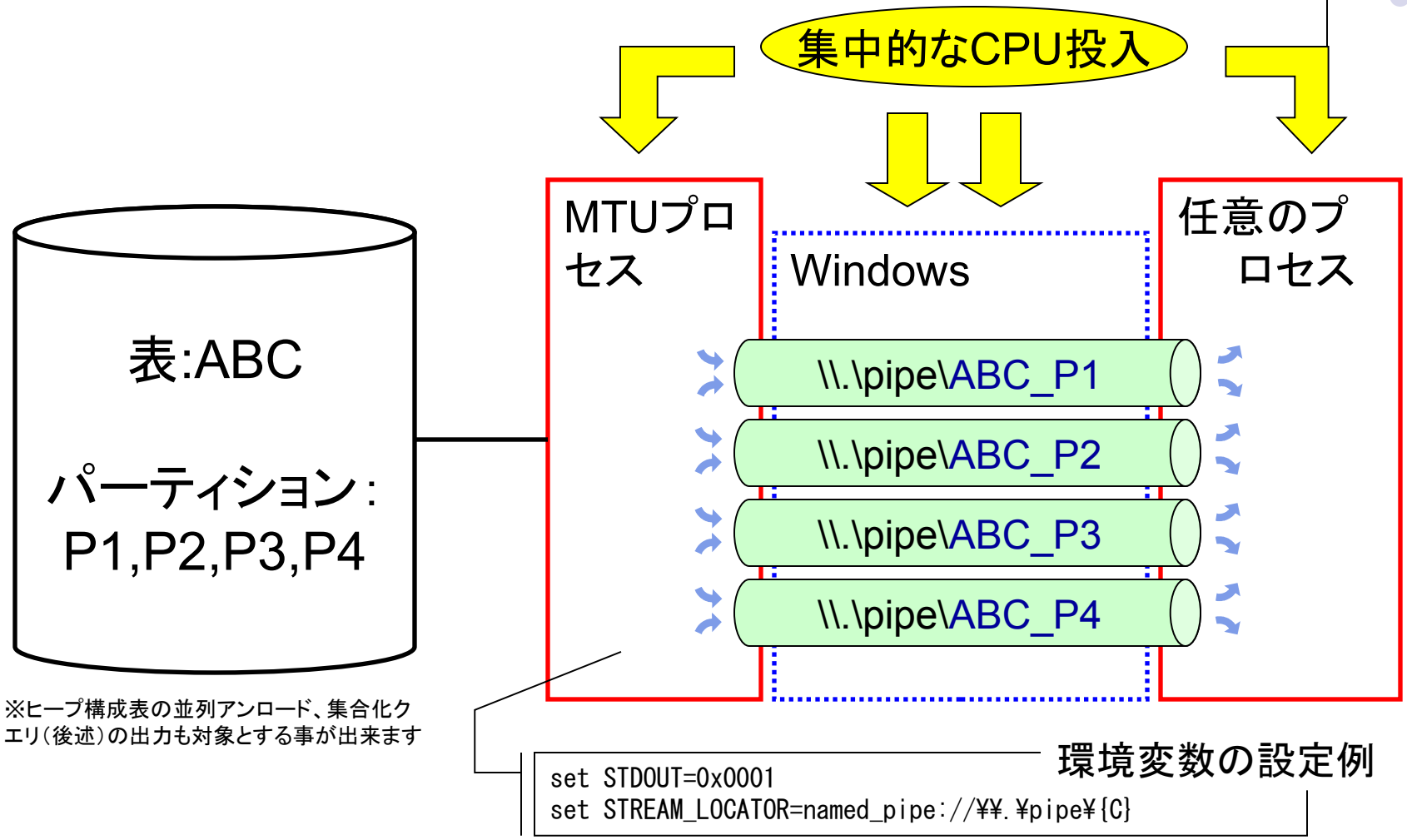


並列化可能なパイプ処理

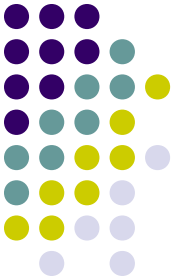


- 旧版では、MTUのプロセス全体で1つの標準出力（コンソール）へデータを出力することが出来ました。
- パイプを使ってテキストデータを他のプロセスの標準入力へリダイレクトできるので、中間ファイルの為のディスク領域やI/O待機が削減できるメリットがあります。
- しかしこの方法ではプロセス当たり一つしかないコンソールへの入出力がボトルネックになる為、並列処理の恩恵を得ることが出来ませんでした。
- 新しい版ではMTUのスレッド毎に任意のプロセスを子プロセスとして起動してそれぞれの標準入力へMTUスレッドの出力ストリームを名無しパイプを使ってリダイレクトすることが出来るようになりました。
- データの読み取りから他のプロセスへデータを受け渡すまでの経路を完全にパイプライン化することが可能になり、CPUの集中投入によるスループット向上を狙い易くなりました。
- 並列化可能なパイプ処理はUnload／Queryのいずれでも使用できます。

名前付きパイプ

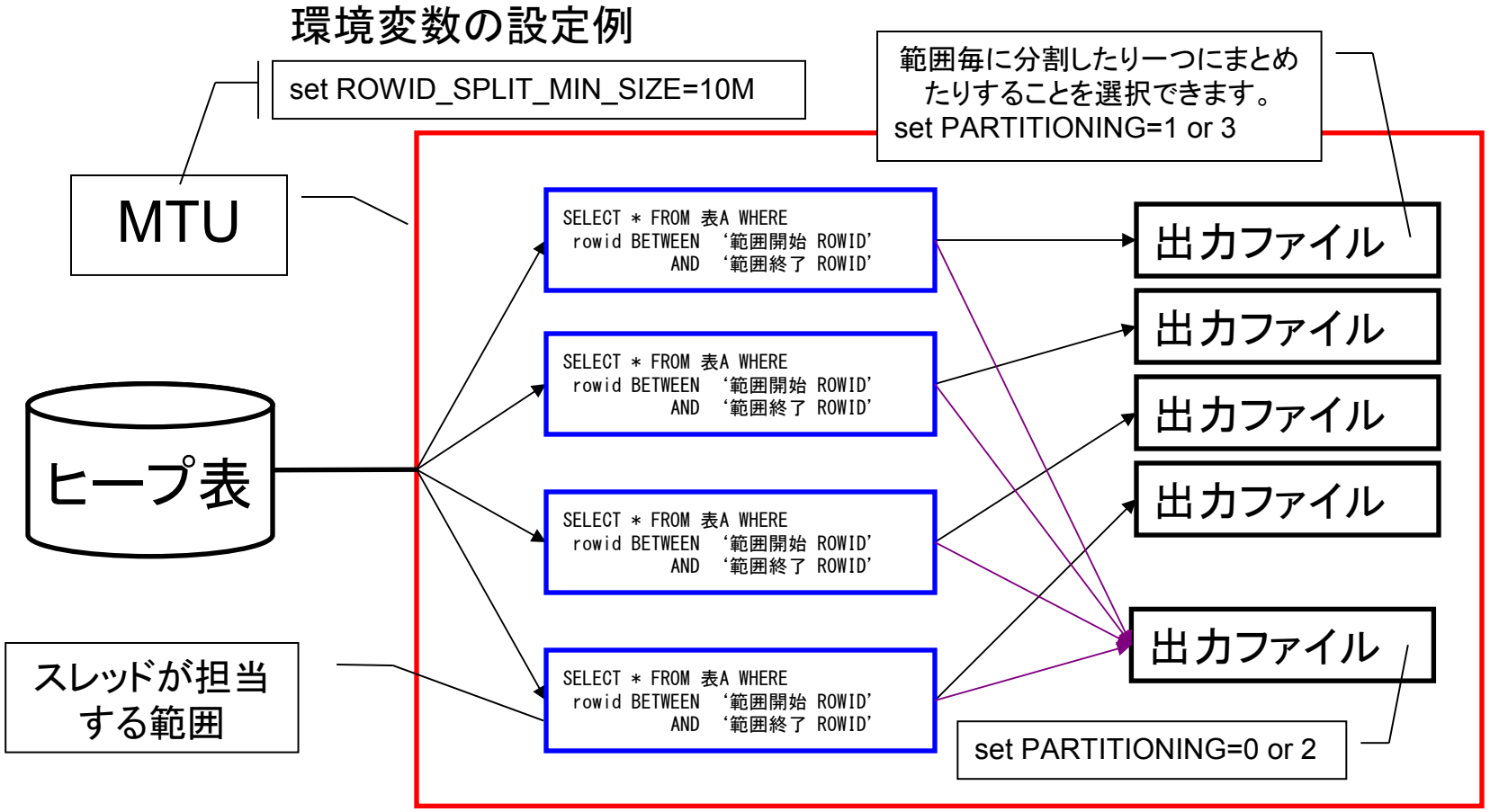

集中的なCPU投入
MTUプロセス
Windows
任意のプロセス
表:ABC
パーティション:
P1,P2,P3,P4
\\.\pipe\ABC_P1
\\.\pipe\ABC_P2
\\.\pipe\ABC_P3
\\.\pipe\ABC_P4
※ヒープ構成表の並列アンロード、集合化クエリ(後述)の出力も対象とする事が出来ます
環境変数の設定例
set STDOUT=0x0001
set STREAM_LOCATOR=named_pipe://¥¥.¥pipe¥{C}

名前付きパイプ

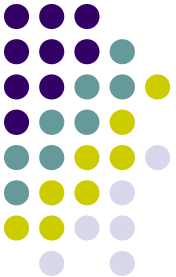


- 「名前付きパイプ」と呼ばれるOSの資源を利用できるようになりました。
- 名前付きパイプはIPC（Interprocess Communication）実現する手段の1つで、これを介してプログラム同士を直結する事が出来ます。
- 名無しパイプとは異なりプロセスは任意の数の名前付きパイプを利用する事が出来ます。従いましてH/Wプラットホームのスケーラビリティーに適した並列度を選択してチューニングを図れます。
- 名前付きパイプの名前は「\\.\pipe\<名前>」という表現です。様々なプログラミング言語からこの名前を与えてストリームを構築できます。
- 例えば javaの場合は java.io.File クラスのコンストラクタ引数に与える事ができるので任意プロセスのプログラミングは容易です。
- MTU、任意プロセスが起動する順序に特別な考慮は不要です。MTUが先に起動しても任意プロセスがコネクトするまで待機します。
- 名前付きパイプはUnload／Queryのいずれでも使用できます

ヒープ構成表の並列アンロード

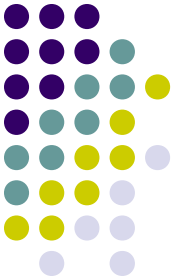


ヒープ構成表の並列アンロード



- 旧版では、非パーティション表の並列アンロードは未対応でした（競合他社製品にはこれが可能なものがあります）。
- その為、アンロード対象の表の中に、ひと際大容量の非パーティション表がある場合、CPUコアひとつ分で処理速度が頭打ちになっていました。
- 新しい版では、非パーティション表の中でもヒープ構成表（普通の表）について並列アンロードが可能になりました。
- ヒープ構成表をROWIDをベースとする複数の範囲に分割してそれぞれの断片化された部分の問合せを並列処理します。
- 索引構成表、クラスタ化表は並列アンロードの対象外です。

集合化クエリ



- 旧版では、問合せファイルにクエリを一つしか記述できなかったので、CPUコアひとつ分で処理速度が頭打ちになっていました。
- 集合化クエリは、選択列リストが同一である複数のクエリを一つの問合せファイルに記述したものです。
- 新しい版では集合化クエリに含まれる複数のクエリを並列処理することが出来ます。
- CPUコア数、及び並列度に対してクエリの数が十分多い場合、CPU資源を集中的に投入して、所要時間の短縮を図ることが可能になりました。

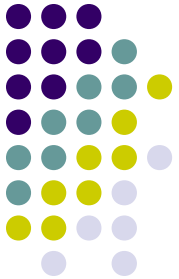
集合化クエリ

集合化クエリの例（単一ファイルに複数のクエリを含めます）

```
select e.empno, e.ename, e.sal, d.dname, d.loc
    from emp e, dept d where e.deptno = d.deptno and d.dname = 'ACCOUNTING';
select e.empno, e.ename, e.sal, d.dname, d.loc
    from emp e, dept d where e.deptno = d.deptno and d.dname = 'RESEARCH';
select e.empno, e.ename, e.sal, d.dname, d.loc
    from emp e, dept d where e.deptno = d.deptno and d.dname = 'SALES';
select e.empno, e.ename, e.sal, d.dname, d.loc
    from emp e, dept d where e.deptno = d.deptno and d.dname = 'OPERATIONS';
```

- 分割には外部キーなど索引付けされた、CLUSTERING_FACTORの低いキーを等価演算子で選択する述語を使うのが性能を引き出すコツです。(上記の例では e.deptno 列)
- 集合化クエリでもバインド変数を含むことができます。
- クエリひとつだけが含まれる問合せファイルも従来通りサポートされます。

➢各クエリの実行されるタイミングは実行毎に代わるため、必ずしもクエリが記述された順序でデータが出力されるとは限らないことにご注意ください。
➢order by句を含むクエリは結果の順序性を保証できないので集合化クエリの中で使用しないで下さい。



機能追加

メインフレーム向けデータ移行

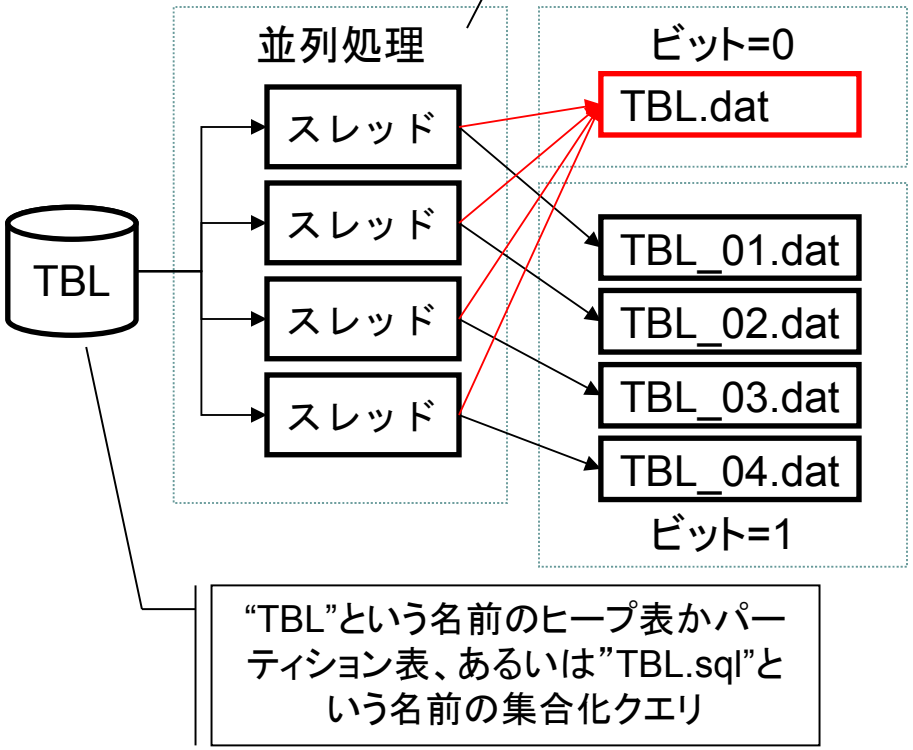
- 旧版は、出力されたファイルをWindowsで使用することが前提で設計されていました。
- Oracle Database → メインフレームという方向のデータ移行に便利な機能を追加しました。
- 追加された新しいパラメータを組み合わせることでSAM (Sequential Access Method；順編成法) ファイル形式へ出力することも出来ます。

	旧版	新しい版で追加
文字エンコーディング方式	ASCII, UTF-8、Shift-JIS	NCHAR/NVARCHAR2をUTF-16BE出力
BOM出力	UTF-8選択時は必ず出力	出力省略を選択可能
数値表現	Oracle 内部データ型の数値をASCII文字列へ変換、ゼロサプレス	パック10進数 (COMP-3), DISPLAY SIGN LEADING SEPARATE
行セパレート方式	可変長、改行付固定長	セパレータ無しの固定長

並列処理データの単一ファイル出力

環境変数の設定例

PARALLELISM=4
ROWID_SPLIT_MIN_SIZE=10M



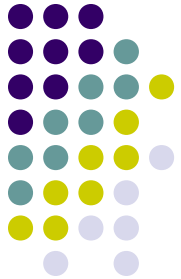
“TBL”という名前のヒープ表かパーティション表、あるいは”TBL.sql”という名前の集合化クエリ

PARTITIONINGの値	32～3ビット目	2ビット目	1ビット目
4以上の値	無効	-	-
3 (=true)	未使用	1	1
2	未使用	1	0
1	未使用	0	1
0	未使用	0	0

ヒープ構成表の並列アンロード

パーティション表の並列アンロード及び集合化クエリ

並列処理データの単一ファイル出力



- 旧版ではパーティション表のアンロードを実行した時、データファイルがパーティション毎に分割出力されていました。
- データファイルが分割されていると、既存のプログラム資産への入力として使う場合に扱いにくい事があります。
- 単一のデータファイルを入手したい場合は、並列処理を諦めるしかありませんでした。
- 新しい版では並列処理、及び単一ファイル出力を一石二鳥で入手できます。
- PARTITIONINGという環境変数が持つ値の特定のビットを0にすると単一ファイル出力を選択できます。
- 単一ファイル出力は、パーティション表のアンロードだけでなく、並列化可能なパイプ処理、名前付きパイプ、ヒープ構成表の並列アンロード、集合化クエリと組み合わせる事も出来ます。
- 単一ファイル出力時であってもTAFイベント、領域あふれ時のフェイル・オーバーに対応します。

➢集中的な書込みの競合が生じる為、ファイル分割時に比較すると5%程度のスローダウンが発生します。
➢並列度1では使用できません。2以上を選択してください。
➢各スレッドが行を操作する時期はOSによりスケジューリングされるので、行の出力順序は実行毎に入れ替る場合がございます。

ありがとうございました

- マルチ・スレッド・アンローダー、Multi-threaded Unloaderは株式会社プラムシックスの登録商標です。
- Oracle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。
- Microsoft, Windows Server, Windows Vista,Excel および Windows は、米国 Microsoft Corporation の、米国、日本およびその他の国における登録商標または商標です。