Oracle Database Technology Night
～ 集え！オラクルの力（チカラ）～

18c ORACLE Database

Oracle Database 18c テクノロジーシリーズ 4
「Development と Performance 関連の機能強化」
～ Performance ～

日本オラクル株式会社
ソリューション・エンジニアリング統括
クラウド・インフラストラクチャー本部
津島 浩樹／纐纈 貴紀

ORACLE

- 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。

アジェンダ

1. Database In-Memoryの機能拡張
2. MemOptimizeプール
3. Optimizerの機能拡張
4. その他の機能

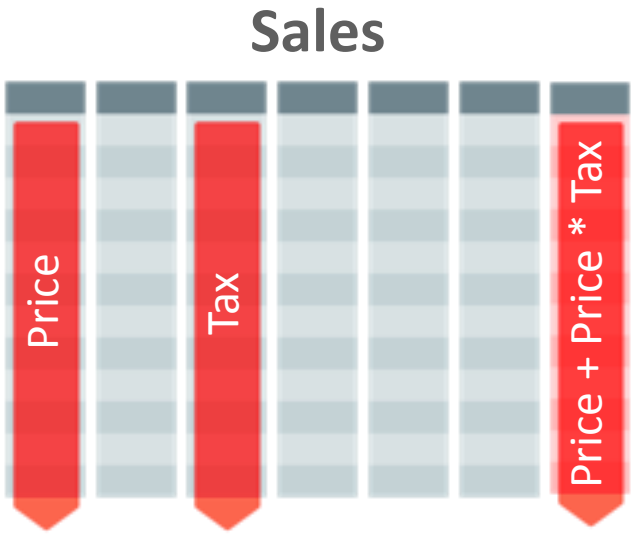
Database In-Memoryの拡張機能

- In-Memory Expressionsの動的キャプチャ・ウィンドウ
- 自動インメモリ管理
- 外部表に対するIn-Memory
- In-Memory ダイナミック・スキャン
- In-Memoryでの算術最適化
- その他の拡張機能

In-Memory Expressionsの動的キャプチャ・ウィンドウ

必要なエディション：EE + DBIM Option, EE-ES + DBIM Option, DBCS EE-EP, ExaCS

In-Memory Expressions（IME）とは？



Price

Tax

それぞれをIn-Memory化して検索を高速化

しかし、PriceとTaxを用いた**演算**が必要になることも・・・
*（例）Price + Price * Tax*

12.2で In-Memory Expressions（IME）が登場
計算式に基づく結果を**仮想列**としてIn-Memory化
※従来の仮想列をIn-Memory化できるようになった

計算式（仮想列）は**明示的な作成**、または
自動キャプチャされた式の利用が可能

12.2での In-Memory Expressionsの課題

- BIツールのようにSQLが自動生成される場合はIMEの明示作成が困難
- IMEの自動キャプチャは意図した式がキャプチャされづらい
  - DB作成後全部または過去24時間がキャプチャ範囲
  - TOP20がキャプチャ対象のため、こぼれ落ちる式が存在する
  - 不要な時間帯のIMEまでキャプチャされ非効率

IMEの使いづらさを改善したい！！

18c In-Memory Expressionsの動的キャプチャ・ウィンドウ

- IMEの自動キャプチャ時に有用
- 自動キャプチャする範囲（ウィンドウ）を任意に定義可能
- ウィンドウ内で発生する式だけを仮想列としてIn-Memory化できる
- ワークロード間隔が分かっている場合に特に効果あり

  利用例）
  - パッケージアプリケーションの特定の機能で使われる式
  - ある特定の時期に集中して実行される分析処理

IME動的キャプチャ・ウィンドウ 設定方法

- 設定手順
  1. 初期化パラメータ INMEMORY_EXPRESSIONS_USAGE をDISABLE以外の値に設定
     （その他 INMEMORY_SIZE等でインメモリが使える設定にしておく）
  2. 管理者権限でSQL*PlusまたはSQL DeveloperにてDBにログイン
  3. IME考慮範囲をユーザー指定範囲に設定 `EXEC DBMS_INMEMORY_ADMIN.IME_CAPTURE_EXPRESSIONS('WINDOW');`
     - CUMULATIVE：データベースの作成後のすべての式統計を考慮します。
     - CURRENT：過去24時間の式統計のみを考慮します。
     - WINDOW：ユーザーが指定した式キャプチャウィンドウで追跡された式を仮想列として追加します。
  4. ウィンドウのオープン `EXEC DBMS_INMEMORY_ADMIN.IME_OPEN_CAPTURE_WINDOW();`
  5. （任意のセッション／アプリケーションにて）ワークロードの実行＝式のキャプチャ
  6. ウィンドウのクローズ `EXEC DBMS_INMEMORY_ADMIN.IME_CLOSE_CAPTURE_WINDOW();`

     ※ウィンドウが開いているかどうかはDBMS_INMEMORY_ADMIN.IME_GET_CAPTURE_STATEで確認可能
  7. キャプチャされた式のポピュレーション `EXEC DBMS_INMEMORY_ADMIN.IME_POPULATE_EXPRESSIONS();`
- 確認
  - SELECT ＊ FROM DBA_EXPRESSION_STATISTICS WHERE SNAPSHOT = ‘WINDOW‘; -- キャプチャされた式の確認
  - SELECT ＊ FROM DBA_IM_EXPRESSIONS; -- ポピュレーションされた式の確認

IME動的キャプチャ・ウィンドウ 実行例

```
--*** キャプチャ設定 ***
SQL> EXEC DBMS_INMEMORY_ADMIN.IME_CAPTURE_EXPRESSIONS('WINDOW');

PL/SQL procedure successfully completed.

SQL> EXEC DBMS_INMEMORY_ADMIN.IME_OPEN_CAPTURE_WINDOW();

PL/SQL procedure successfully completed.
```

この間にワークロードを実行

```
SQL> EXEC DBMS_INMEMORY_ADMIN.IME_CLOSE_CAPTURE_WINDOW();

PL/SQL procedure successfully completed.

SQL> EXEC DBMS_INMEMORY_ADMIN.IME_POPULATE_EXPRESSIONS();

PL/SQL procedure successfully completed.
```

```
--*** ウィンドウ状態の確認 ***
SQL> EXECUTE
DBMS_INMEMORY_ADMIN.IME_GET_CAPTURE_STATE(:b_state, :b_time)

PL/SQL procedure successfully completed.

SQL> PRINT b_state b_time
B_STATE
--------------------------------------------------------------------------------
CLOSE

B_TIME
--------------------------------------------------------------------------------
21-MAY-18 03.37.28.218495 PM
```

```
--*** ポピュレートされた式の確認 ***
SQL> select TABLE_NAME,SQL_EXPRESSION from dba_im_expressions where owner='TEST1';

TABLE_NAME                     SQL_EXPRESSION
------------------------------ --------------------------------------------------
BIGEMP                         ROUND("SAL"*10/52,2)
BIGEMP                         10*(NVL("COMM",0)+"SAL")
```

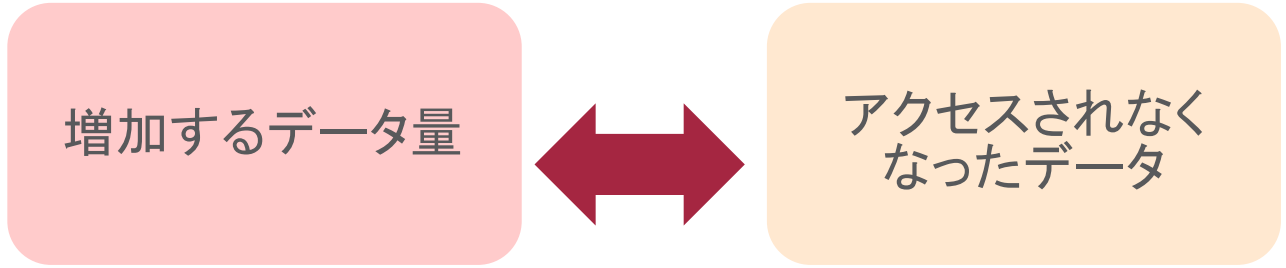
ワークロード中に実行された計算式がポピュレートされている

自動インメモリ管理

必要なエディション：EE-ES + DBIM Option, DBCS EE-EP, ExaCS

これまでのインメモリ管理（～12.2）

- 12.1でのIn-Memory領域管理の不便さ

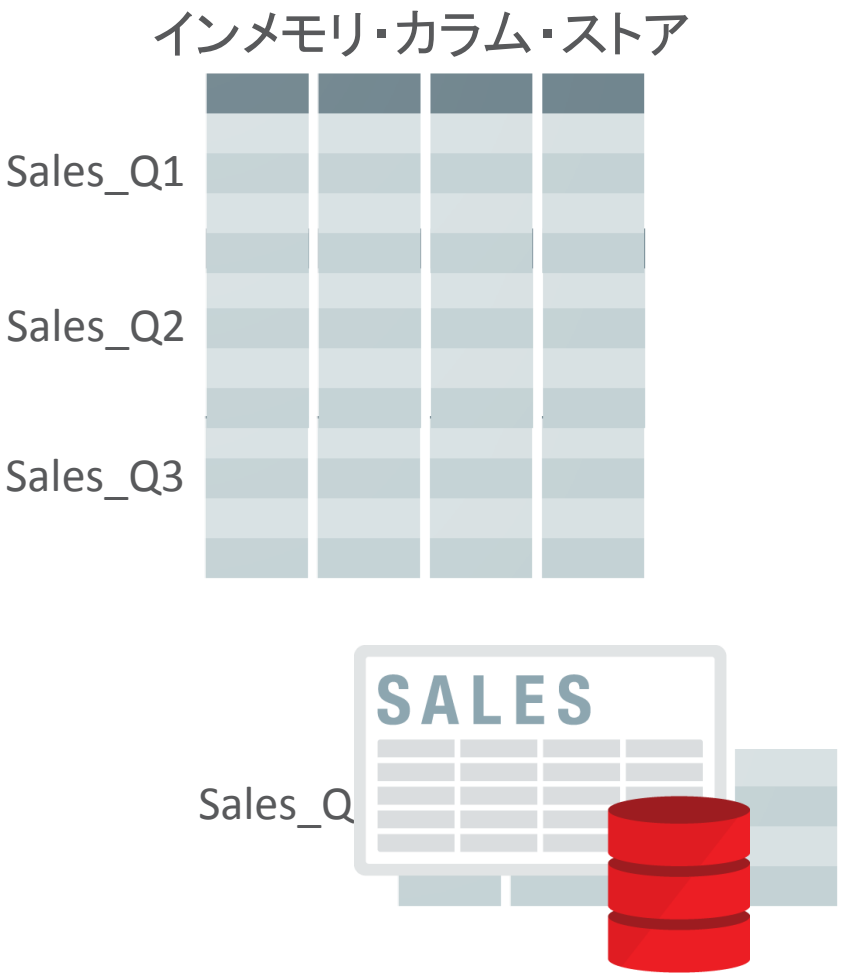


In-Memory領域のコントロールを手動で行わなければならない

- 12.2 ADOインメモリポリシーにて自動的なインメモリ化/インメモリ退去が可能に
  ただし、下記の制約にて利用方法が煩雑
  - セグメントごとにポリシー設定が必要
  - インメモリ化／インメモリ退去のポリシーはそれぞれ個別に設定が必要
  - ポリシー評価・発動の条件（日数等）を予め決めて定義しておく必要がある

本機能はEngineered Systems／DBCS EE-Extreme Performance／ExadataCSにて利用可能です

自動インメモリ管理（18c～）



- 初期化パラメータ：INMEMORY_AUTOMATIC_LEVEL にて制御可能
- ヒート・マップ のアクセス統計を利用し、インメモリ・テーブルおよびパーティションを自動的にランク付け
- 新しいセグメントをインメモリ化する際に、メモリ不足の場合はアクセス頻度の低いインメモリデータを自動的に除去

（利用例）
時系列でパーティション化されているテーブルの過去パーティションを除去して新規パーティションをIn-Memory化

自動インメモリ管理の設定

- 初期化パラメータ：INMEMORY_AUTOMATIC_LEVEL
  - OFF（デフォルト）：自動インメモリ管理は無効。Oracle Database 12gR2（12.2.0.1）と同じ動作です。
  - LOW：メモリ不足の場合、IM列ストアからコールドセグメントを削除した上で新規セグメントをポピュレートします。（ただしADOインメモリポリシー条件が有効なセグメントは削除対象外）
  - MEDIUM：LOWの動作に加え、以前にメモリ不足でポピュレート未完了だったホットセグメントが先にポピュレートされるよう最適化されます。（ADOインメモリポリシーを併用している場合、コールドセグメントであっても削除されず、十分なメモリを確保できない状況が生じます。）
- インメモリPRIORITYがNONEのセグメントのみ自動削除の対象となります。
- セグメント毎の細かな制御が必要な場合、従来のADOインメモリポリシーとの併用も可能です。この場合、ADOインメモリポリシーに定義された条件が優先されます。

（参考）
自動インメモリ管理がヒートマップを評価する期間はデフォルトで31日分です。
評価期間を変更する場合は下記のプロシージャを使います。
DBMS_INMEMORY_ADMIN.AIM_SET_PARAMETER
例）統計ウィンドウ（AIM_STATWINDOW_DAYS定数）を7日間に設定する場合

```
EXEC DBMS_INMEMORY_ADMIN.AIM_SET_PARAMETER ( DBMS_INMEMORY_ADMIN.AIM_STATWINDOW_DAYS, 7 );
```

自動インメモリ管理 実行例

```
SQL> select segment_name,partition_name,bytes_not_populated,populate_status from v$im_segments;

SEGMENT_NAME                   PARTITION_NAME                 BYTES_NOT_POPULATED POPULATE_STAT
------------------------------ ------------------------------ ------------------- -------------
NUM_PART                       PARTITION03                          259031040 OUT OF MEMORY
NUM_PART                       PARTITION02                                  0 COMPLETED
NUM_PART                       PARTITION01                                  0 COMPLETED

SQL> alter system set INMEMORY_AUTOMATIC_LEVEL=low scope=memory;

System altered.

SQL> select count(*) from num_part partition(partition04);

  COUNT(*)
----------
  85196850

SQL> select segment_name,partition_name,bytes_not_populated,populate_status from v$im_segments;

SEGMENT_NAME                   PARTITION_NAME                 BYTES_NOT_POPULATED POPULATE_STAT
------------------------------ ------------------------------ ------------------- -------------
NUM_PART                       PARTITION04                                  0 COMPLETED
NUM_PART                       PARTITION01                                  0 COMPLETED
```

3つ目のパーティションがメモリ不足により全てインメモリ化できず（4つ目以降のパーティションもインメモリ化されない）

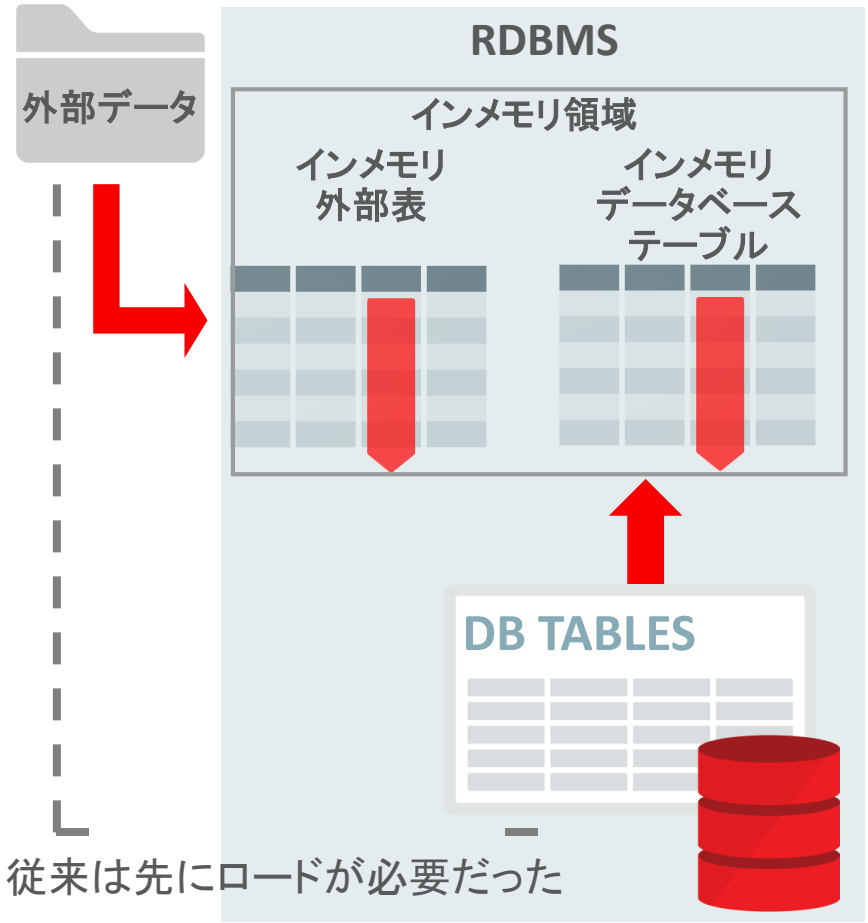
自動管理を有効にしたところ、メモリ不足とならず既存の（アクセス頻度の低い）パーティションを追い出してインメモリ化

外部表に対するIn-memory

必要なエディション：EE-ES（ODAを除く）+ DBIM Option, DBCS EE-EP, ExaCS

本機能はEngineered Systems／DBCS EE-Extreme Performance／ExadataCSにて利用可能です
ただしOracle Database Applianceでは利用できません

外部表に対するIn-Memory



- 従来は外部表データを頻繁に分析に利用したいときは、一旦RDBにロードしてインメモリ化する必要がありました
- 18cより外部表によりデータベース外のデータへの透過的なアクセスが可能となりました
- 18cでの外部表サポート
  - ORACLE_LOADER
  - ORACLE_DATAPUMP

※ORACLE_HDFSおよびORACLE_HIVEは将来サポート予定

In-Memory 外部表 利用方法

- 外部表作成時にINMEMORY句を指定します。
- 対象セグメントを手動でポピュレートします。
- インメモリの外部表を問い合せるセッションでは、初期化パラメータQUERY_REWRITE_INTEGRITYをstale_toleratedに設定する必要があります。

```
SQL> CREATE TABLE TAB_IM_EXT
  2    (COL1  VARCHAR2(100), COL2 VARCHAR2(100))
  3  ORGANIZATION EXTERNAL
        （中略） --通常の外部表定義と同じ
 18  INMEMORY;

Table created.
```

```
SQL> select sum(COL1) SC from TAB_IM_EXT;

Execution Plan
----------------------------------------------------------------------------
| Id  | Operation                    | Name       | Rows  | Bytes | Cost
----------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |            |     1 |     5 |
|   1 |  SORT AGGREGATE              |            |     1 |     5 |
|   2 |   EXTERNAL TABLE ACCESS FULL| TAB_IM_EXT |  1180 |  5900 |
----------------------------------------------------------------------------

SQL> alter session set query_rewrite_integrity=stale_tolerated;

Session altered.

SQL> select sum(COL1) SC from TAB_IM_EXT;

Execution Plan
----------------------------------------------------------------------------
| Id  | Operation                             | Name       | Rows  |
----------------------------------------------------------------------------
|   0 | SELECT STATEMENT                      |            |     1 |
|   1 |  SORT AGGREGATE                       |            |     1 |
|   2 |   EXTERNAL TABLE ACCESS INMEMORY FULL| TAB_IM_EXT |  1180 |
----------------------------------------------------------------------------
```

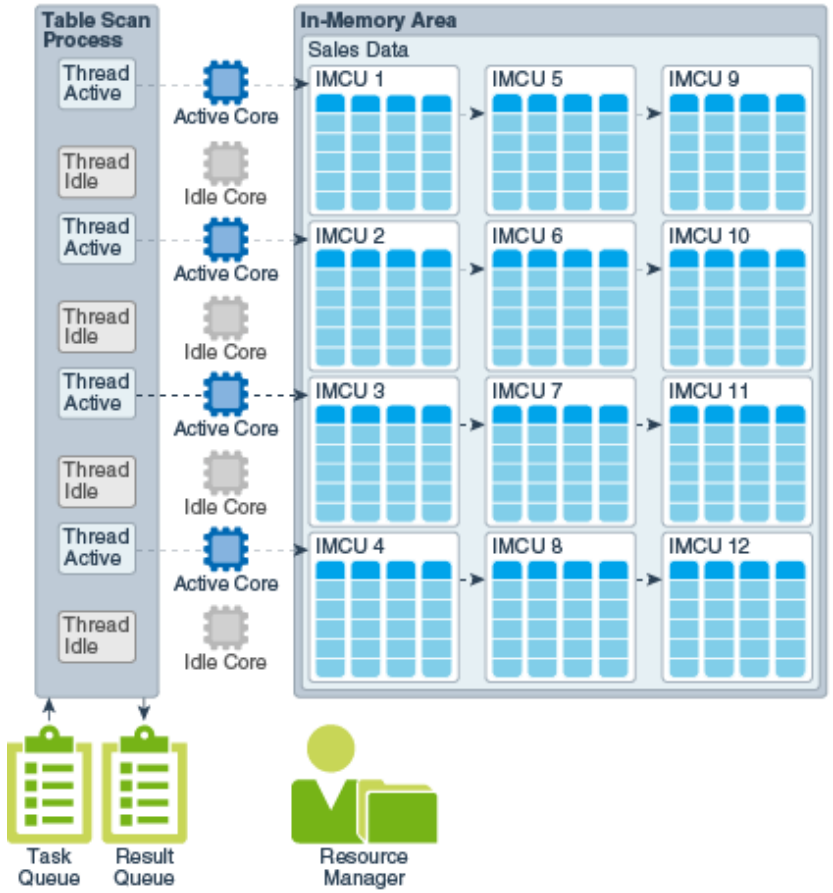
外部表に対するIn-Memory 制限事項

- 自動ポピュレーションおよび自動再ポピュレーションはサポートされません。明示的にDBMS_INMEMORY.POPULATEまたはDBMS_INMEMORY.REPOPULATEを使用する必要があります。
- column句、distribute句、priority句など一部のINMEMORY副次句は無効です。
- ORACLE_LOADERおよびORACLE_DATAPUMP アクセスドライバのみサポートされています。（ORACLE_HDFSおよびORACLE_HIVEは将来サポート予定）
- パーティション化、結合グループ、In-Memory Expressions、In-Memory算術最適化はサポートされません。
- パラレル実行はできません。

In-memoryダイナミック・スキャン

必要なエディション：EE + DBIM Option, EE-ES + DBIM Option, DBCS EE-EP, ExaCS

In-Memory ダイナミック・スキャン 概要



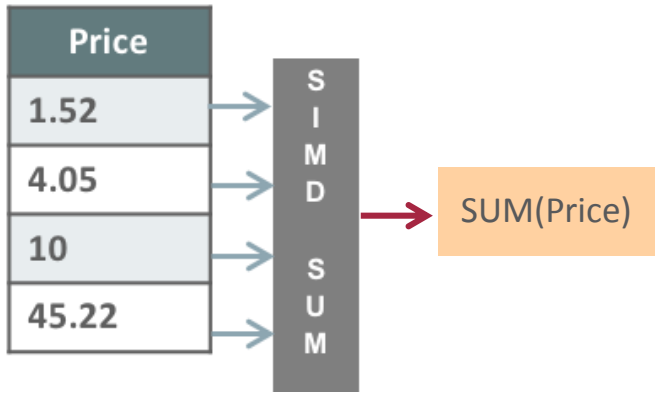
- In-Memory ダイナミック・スキャンは、軽量プロセススレッドを使用してテーブルスキャンを自動的かつ透過的に並列化します。
- Resource Managerが有効な場合にのみ有効になります。
- In-Memory ダイナミック・スキャンはResource Managerにより透過的に制御されるためアプリケーションの変更は必要ありません。
- アイドル状態のCPUリソースを使用してIMCUを並列にスキャンし、CPU使用率を最大化します。スキャンは動的なので、既存のワークロードに影響を与えずに余剰分のCPU帯域幅を使用できます。

In-memoryでの算術最適化

必要なエディション：EE + DBIM Option, EE-ES + DBIM Option, DBCS EE-EP, ExaCS

In-Memoryでの算術最適化

- QUERY LOWで圧縮されたNUMBER列に対し、ハードウェアでのネイティブ計算を可能にする形式でエンコードできるようになりました。
- この機能を有効化するには初期化パラメータINMEMORY_OPTIMIZED_ARITHMETICをENABLEに設定します。（デフォルト: DISABLE）
- SIMDベクトル処理により、単純な集約、GROUP BY集約、および算術演算の高速化が期待できます。
- すべての処理で最適化形式が利用できるわけではないため、IM列ストアには従来のNUMBERデータ型と最適化型の両方が格納されます。（約15%～20%程度のオーバーヘッド）



Database In-Memoryの拡張機能
その他の拡張機能

- 結合グループの拡張
  - 結合グループ（12.2から）
    - 結合でもディレクトリ圧縮（圧縮した状態での処理）が可能
  - 自己結合（単一列の結合グループ）をサポート
    - INMEMORY JOIN GROUP <結合グループ> (<表名> (<列名>))
- LOBのパフォーマンス向上
  - インラインLOBに対してSIMDベクター処理が可能
- ExadataフラッシュキャッシュのIn-Memoryサポート
  - 非圧縮表とOLTP圧縮表をサポート（12.2はHCCのみ）
    - ALTER TABLE <表名> [NO] CELLMEMORY [<インメモリ圧縮>]
    - CAPACITY LOW がデフォルト

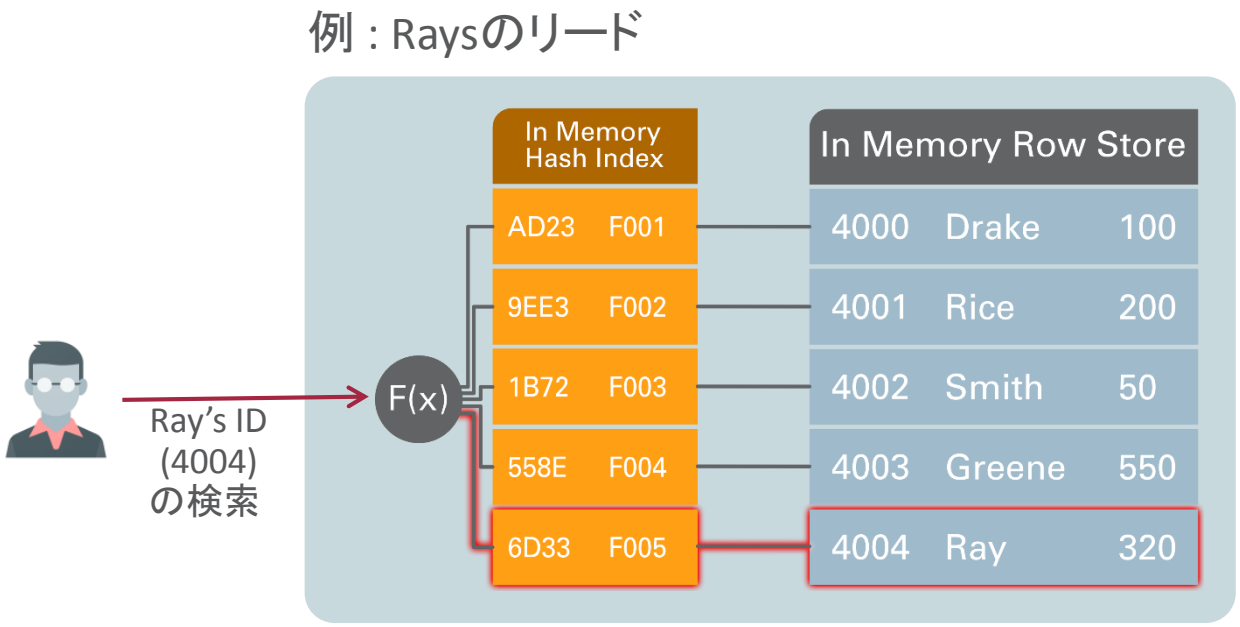
アジェンダ

1. Database In-Memoryの機能拡張
2. MemOptimizeプール
3. Optimizerの機能拡張
4. その他の機能

MemOptimizeプール

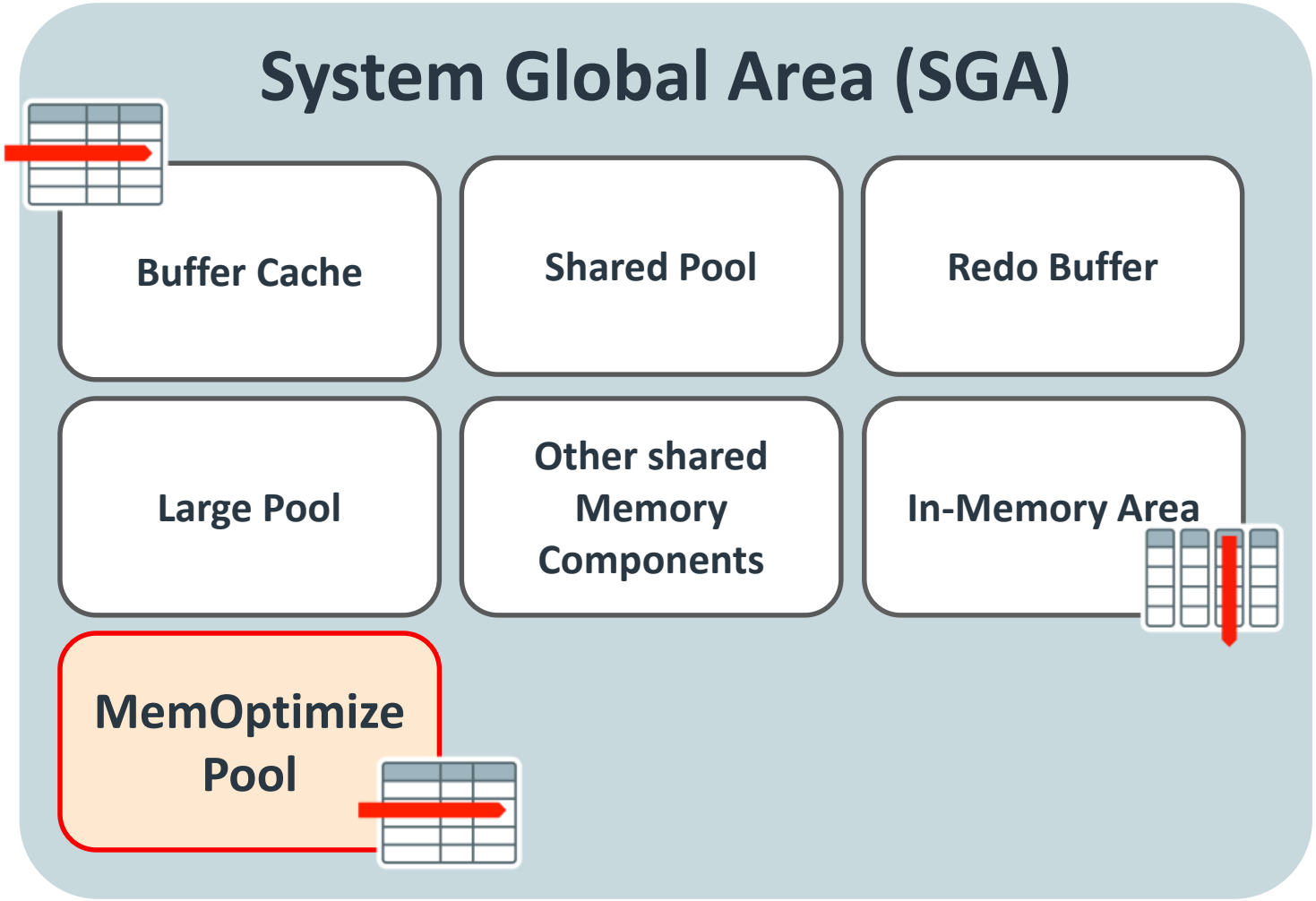
必要なエディション：EE-ES（Exadataのみ）, DBCS EE-EP, ExaCS

MemOptimizeプール



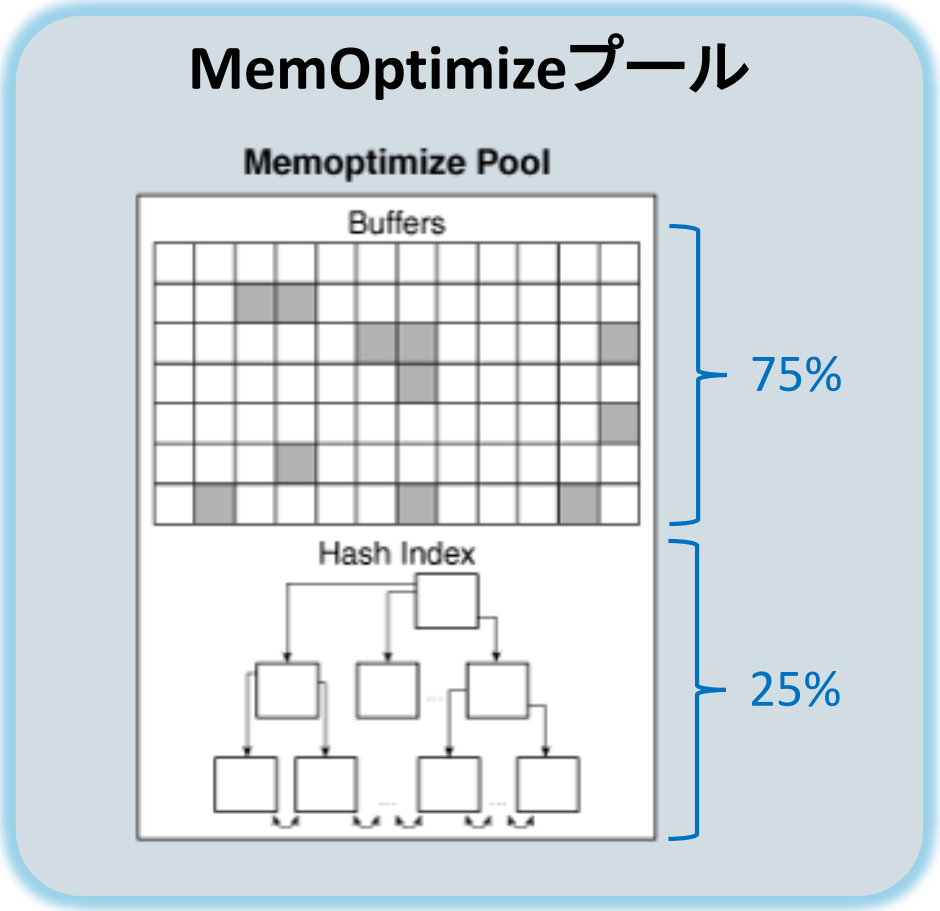
- 超高速な主キー・ベースの検索
  - ハッシュ関数を使用した行アクセス
    - 新たなインメモリ・ハッシュ索引を利用
    - 単一表ハッシュ・クラスタより高速
      - ブロック数が増える（全表スキャンが遅い）などがない
- 新しい低レイテンシ・クライアント・プロトコルによるダイレクト・アクセス
  - 12.2からのExadirectプロトコル
    - RDMA(Remote Direct Memory Access)を使用
- パフォーマンス上のメリット：
  - 主キー・ベースの検索に比べ、最大 x4 のスループット増
  - レスポンス・タイムは最大 50% 減

MemOptimize行ストアの基本構成



- SGA内の新たな領域としてMemOptimizeプールを追加
- MEMOPTIMIZE_POOL_SIZEパラメータによりサイズ設定
  - 最小値は100Mバイト
- SGA_TARGETも十分に大きな値の設定が必要
- インメモリ領域と同じように静的プールとして確保
  - 自動メモリー管理されない

MemOptimizeプール領域: 構成



- 2つのパートで構成:
  - バッファ領域: 75%
    - DBバッファ・キャッシュ内のバッファと同じ構造
  - ハッシュ索引: 25%
    - 複数の非連続メモリー単位としてハッシュ索引を割り当てる
    - 各単位には、複数のハッシュ・バケットを使用し、個別のマップ構造によりメモリー単位と主キーを関連付けられる

MemOptimizeプールの導入

1. 初期化パラメータを設定
   - alter system set MEMOPTIMIZE_POOL_SIZE=<サイズ> scope=spfile;
2. データベースの再起動
   - 領域はV$SGAでは確認できない（show parameterで確認）
3. 格納するデータに対し属性を追加
   - alter table <テーブル名> MEMOPTIMIZE FOR READ ;
     - パーティションは指定できない
     - USER_TABLES.MEMOPTIMIZE_READなどで確認（'enabled'/'disabled'）
   - DISABLEにするときには'NO MEMOPTIMIZE FOR READ'を指定
4. プールにポピュレートする
   - exec DBMS_MEMOPTIMIZE.POPULATE('<スキーマ名>', '<表名>', ['<パーティション名>']) ;
     - これが確認できない（領域不足でもエラーにならない）
   - 削除はDBMS_MEMOPTIMIZE.DROP_OBJECTプロシージャ

MemOptimizeプール
主要統計名（v$sysstat/v$mystat)

統計の確認方法）v$sysstat/v$mystatビューとv$statnameで確認する

```
確認SQL例）
select display_name, value from v$mystat m, v$statname n
where m.STATISTIC#=n.STATISTIC# and display_name like 'memopt r%'
order by 1;
```

統計名	説明
memopt r lookup	ハッシュ索引での合計参照数
memopt r hits	ハッシュ索引での合計ヒット数
memopt r misses	ハッシュ索引での合計ミス数（見つからなかった数）
memopt r rows populated	ポピュレートした行数
memopt r blocks populated	ポピュレートしたブロック数
...	

MemOptimizeプール
メモリ・ハッシュ索引アクセス（主キーに対する等価条件検索）

```
SQL> alter table tab01 memoptimize for read;

…

SQL> select * from tab01 where col1=1000;

-------------------------------------------------------------------------------------------
| Id | Operation                                | Name     | Rows | Bytes |Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|  0 | SELECT STATEMENT                         |          |    1 |   121 |    2   (0)| 00:00:01 |
|  1 |  TABLE ACCESS BY INDEX ROWID READ OPTIM| TAB01    |    1 |   121 |    2   (0)| 00:00:01 |
|* 2 |   INDEX UNIQUE SCAN READ OPTIM           | TAB01_PK |    1 |       |    1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

※ ポピュレートされないときでもこの実行計画になる（'memopt r hits'を確認する）

```
SQL> select display_name,value from v$mystat m,v$statname n
  2   where m.STATISTIC#=n.STATISTIC#
  3     and display_name in ('memopt r lookups','memopt r hits','memopt r misses');

DISPLAY_NAME                        VALUE
---------------------------- ----------
memopt r lookups                     1
memopt r hits                        1
memopt r misses                      0
```

MemOptimizeプール
メモリ・ハッシュ索引を使用しない例

```
SQL> select * from tab01 where col1 BETWEEN 100 AND 200;

-----------------------------------------------------
| Id | Operation                          | Name     |
-----------------------------------------------------
|  0 | SELECT STATEMENT                   |          |
|  1 |  TABLE ACCESS BY INDEX ROWID BATCHED | TAB01    |
|* 2 |   INDEX RANGE SCAN                 | TAB01_PK |
-----------------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

2 - access("COL1">=100 AND "COL1"<=200)
```

```
SQL> select * from tab01 where col1=100 AND col2=200;

-----------------------------------------------
| Id | Operation                    | Name     |
-----------------------------------------------
|  0 | SELECT STATEMENT             |          |
|* 1 |  TABLE ACCESS BY INDEX ROWID | TAB01    |
|* 2 |   INDEX RANGE SCAN           | TAB01_PK |
-----------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

1 - filter("COL2"=200)
2 - access("COL1"=100)
```

```
SQL> select * from tab01 where col1 IN (100,200);

-----------------------------------------------
| Id | Operation                    | Name     |
-----------------------------------------------
|  0 | SELECT STATEMENT             |          |
|  1 |  INLIT ITERATOR              | TAB01    |
|  2 |   TABLE ACCESS BY INDEX ROWID | TAB01    |
|* 3 |    INDEX UNIQUE SCAN         | TAB01_PK |
-----------------------------------------------

Predicate Information (identified by operation id):
---------------------------------------------------

3 - access("COL1"=100 OR "COL1"=200)
```

MemOptimizeプール
キャッシュ・ミスになる場合

- memopt r missesがアップする場合
  - ポピュレートしていない表
  - drop_objectを行った表
  - すべてポピュレートされていない表のデータ(MemOptimizeプールに入り切らないデータ)
  - ポピュレートしていないパーティションへのアクセス
  - ポピュレートしていないインスタンスでのアクセス
  - 更新された行へのアクセス
    - consistent getsがアップする

```
SQL> alter table tab01 memoptimize for read;

SQL> select * from tab01 where col1=1000;

-------------------------------------------------------------
| Id | Operation                                 | Name     |
-------------------------------------------------------------
|  0 | SELECT STATEMENT                          |          |
|  1 |  TABLE ACCESS BY INDEX ROWID READ OPTIM| TAB01    |
|* 2 |   INDEX UNIQUE SCAN READ OPTIM         | TAB01_PK |
-------------------------------------------------------------

SQL> select display_name,value from v$mystat m,v$statname n
  2   where m.STATISTIC#=n.STATISTIC#
  3     and display_name in ('memopt r lookups',
  4                          'memopt r hits',
  5                          'memopt r misses');

DISPLAY_NAME                        VALUE
------------------------------ ----------
memopt r lookups                        1
memopt r hits                           0
memopt r misses                         1
```

MemOptimizeプール
ポピュレートの監視

- ポピュレートの統計が更新しなくなるのをチェック
  - ポピュレートするセグメントの行数がカウントアップされたのを確認

```
SQL> select display_name,value from v$sysstat m,v$statname n
  2   where m.STATISTIC#=n.STATISTIC#
  3     and display_name in ('memopt r rows populated',
  4                          'memopt r blocks populated');

DISPLAY_NAME                          VALUE
----------------------------- ----------
memopt r rows populated              312613
memopt r blocks populated             44659
```

MemOptimizeプールの制限

- ヒープ表（通常の表）のみ使用可

```
alter table iot_01 memoptimize for read;
*
ERROR at line 1:
ORA-62148: MEMOPTIMIZE FOR READ feature cannot be enabled on IOTs.
```

- 圧縮表には使用できない

```
alter table tab_01 memoptimize for read;
*
ERROR at line 1:
ORA-62141: MEMOPTIMIZE FOR READ feature cannot be used with COMPRESS option.
```

- 主キー制約が必須
  - ローカル索引の主キーは使用できない

※ RAC環境でのデータ分散は行われない

アジェンダ

1. Database In-Memoryの機能拡張
2. MemOptimizeプール
3. Optimizerの機能拡張
4. その他の機能

Optimizerの拡張機能

- 近似Top-N問合せ（Developmentを参照）
- パフォーマンス・フィードバック
- Oracle Database Standard Edition(SE)のプラン・スタビリティ
- パーティション・ワイズ操作の強化
- Autonomous Databaseのオプティマイザ

Optimizerの拡張機能
パフォーマンス・フィードバック

- パフォーマンス・フィードバックは“自動DoP”と連携する最適機能
  - 問合せの実行時間に基づいて選択した並列度を調整する
    - 収集した統計（CPU Timeなど）を使用してパラレル度を改善（自動再最適化）
- Oracle Database 12cR2
  - optimizer_adaptive_statisticsによって制御される
- Oracle Database 18c
  - parallel_degree_policy=adaptiveのみで有効
  - optimizer_adaptive_statisticsを使用して制御されなくなった
    - オーバーヘッドが大きい

Optimizerの拡張機能
Oracle Database SEのプラン・スタビリティ（ストアド・アウトライン）

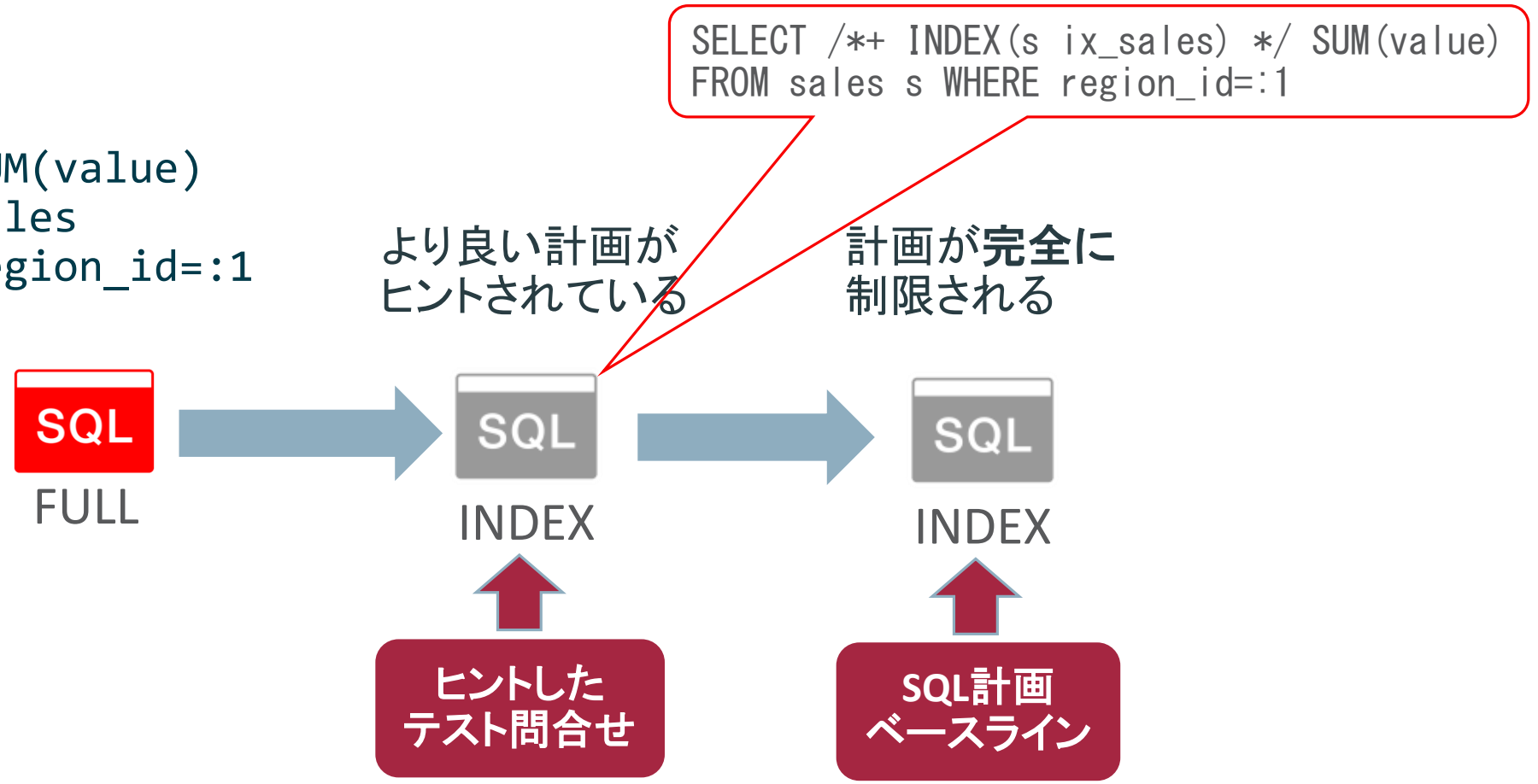
- SQL実行計画を制御するためにSEで使用されることが多い
- Oracle Database 11gR1で非推奨
  - Oracle Database 18cでも引き続き機能する

Optimizerの拡張機能
Oracle Database SEのプラン・スタビリティ（SPMのサブセット）

- これまではSPM（SQL計画管理）はEEのみの機能
- Oracle Database 18cからSEにSPM機能の**サブセット** が可能に
  - ストアド・アウトラインと同様の機能
  - 複数から最適な実行計画を使用することはできない
    - 安定化するだけ（実行計画を変化させない）
- ストアド・アウトラインからSQL計画ベースラインへの移行は簡単
  - `DBMS_SPM.MIGRATE_STORED_OUTLINE(attribute_name => 'all' )`
- ストアド・アウトラインではなくSPMを使用した計画の管理…

Optimizerの拡張機能
SEのSPM使用例（もうストアド・アウトラインは必要ない）

```
SELECT SUM(value)
FROM   sales
WHERE  region_id=:1
```



Optimizerの拡張機能
SEにおけるSPMのサブセット（機能はストアド・アウトラインと似ている）

1つのSQL文で1つのSQL計画ベースラインのみ許可され、SQL計画の展開は無効

	Standard Edition	Enterprise Edition
SQL文に対する複数のSQL計画ベースライン		●
計画の展開（Evolution）		●
自動取得	●	●
SQLチューニング・セットからの取得		●
カーソル・キャッシュからの取得	●	●
AWRからの取得（12.2から）		●
未使用SQL計画ベースラインの自動削除		●
Export/Import (pack/unpack)	●	●
ストアド・アウトラインからSQL計画ベースラインに変換	●	●

Optimizerの拡張機能
パーティション・ワイズ操作の強化

- パラレル実行でのパーティション・ワイズ分析関数（ウィンドウ関数）が可能に
  - TEMP領域の削減にも効果（「津島博士のパフォーマンス講座」第45回）
    - 12.1まで（結合、Group By）、DISTINCT（12.2から）
  - ヒント（USE_PARTITION_WISE_WIF / NO_USE_PARTITION_WISE_WIF）

```
SQL> select /*+ parallel(2) */ tab1.*, row_number() over (partition by c3 order by null) from tab1;
```

実行計画(12.2)

```
-------------------------------------------------------------------
| Id  | Operation                | Name     |
-------------------------------------------------------------------
|   0 | SELECT STATEMENT         |          |
|   1 |  PX COORDINATOR          |          |
|   2 |   PX SEND QC (RANDOM)    | :TQ10001 |
|   3 |    WINDOW SORT           |          |
|   4 |     PX RECEIVE           |          |
|   5 |      PX SEND HASH        | :TQ10000 |
|   6 |       PX BLOCK ITERATOR  |          |
|   7 |        TABLE ACCESS FULL | TAB02    |
-------------------------------------------------------------------
```

実行計画(18c)

```
-------------------------------------------------------------------------------------------------------------
| Id  | Operation                | Name     | Pstart| Pstop |    TQ  |IN-OUT| PQ Distrib |
-------------------------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |          |       |       |        |      |            |
|   1 |  PX COORDINATOR          |          |       |       |        |      |            |
|   2 |   PX SEND QC (RANDOM)    | :TQ10000 |       |       | Q1,00  | P->S |  QC (RAND) |
|   3 |    PX PARTITION RANGE ALL|          |     1 |     4 | Q1,00  | PCWC |            |
|   4 |     WINDOW SORT          |          |       |       | Q1,00  | PCWP |            |
|   5 |      TABLE ACCESS FULL   | TAB1     |     1 |     4 | Q1,00  | PCWP |            |
-------------------------------------------------------------------------------------------------------------
```

Optimizerの拡張機能
Autonomous Databaseのオプティマイザ

- オンライン・オプティマイザ統計収集の強化
  - 従来型DML操作以外はオンライン統計収集
    - 非Autonomous：表が空の場合のバルク・ロードのみを12cR1からサポート
- Statistics-Based Query Transformation
  - オプティマイザ統計を使用した問合せに変換（ビュー名：VW_SQT_xxxx）
    - MIN, MAX, COUNT, APPROXIMATE_COUNT_DISTINCT
- パラレル度（DOP）と同時実行
  - コンシューマ・グループ（HIGH, MEDIUM, LOW）によって自動調整
    - PARALLEL_DEGREE_POLICY＝AUTO （LOWはシリアル実行のみ）
    - パラレルDMLもデフォルトで有効
  - PARALLELヒントによる設定
    - OPTIMIZER_IGNORE_PARALLEL_HINTS=TRUE（デフォルト無視）
    - OPTIMIZER_IGNORE_HINTS=TRUE（その他のヒントもデフォルト無視）

アジェンダ

1. Database In-Memoryの機能拡張
2. MemOptimizeプール
3. Optimizerの機能拡張
4. その他の機能

その他の機能
新しいSQLチューニング・セットのパッケージ

- SQLチューニング・セット（SQL文のセットや実行コンテキストなど）の目的
  - SQLチューニング・アドバイザ、SQLアクセス・アドバイザ、SQLパフォーマンス・アナライザなどの入力
  - データベース間でのSQLの移行（別データベースで診断やチューニングを実施）
- これまでのDBMS_SQLTUNEパッケージ
  - Oracle Tuning Packが必要
- 新しいDBMS_SQLSETパッケージ
  - Oracle Tuning Packが必要ない
  - DBMS_SQLTUNEと同等のサブプログラムを提供

その他の機能
Exadataに対応したSQLチューニング・アドバイザ

- Exadata対応のSQLプロファイル
  - システム統計の補正
    - 失効している場合や収集するとパフォーマンスが向上する場合
    - システム統計のI/O性能（スマート・スキャンのコスト）
      - I/Oシーク時間(ioseektim)、マルチブロック・リード・カウント(mbrc)、I/O転送速度(iotfrspeed)
- システム統計（CPU性能、I/O性能）
  - Exadataでは固有のシステム統計を取得する
    - DBMS_STATS.GATHER_SYSTEM_STATS('EXADATA');

リファレンス
マニュアル・ドキュメント、連載

- Oracle Database In-Memoryガイド, 18*c*
  https://docs.oracle.com/cd/E96517_01/inmem/index.html
- Oracle Databaseパフォーマンス・チューニング・ガイド, 18*c*
  https://docs.oracle.com/cd/E96517_01/tgdba/index.html
- Oracle Database SQLチューニング・ガイド, 18*c*
  https://docs.oracle.com/cd/E96517_01/tgsql/index.html
- 津島博士のパフォーマンス講座
  http://www.oracle.com/technetwork/jp/database/articles/tsushima/

テック・ナイトアーカイブ資料と お役立ち情報

各回テック・ナイトセッション資料 ダウンロードサイト

oracle technight

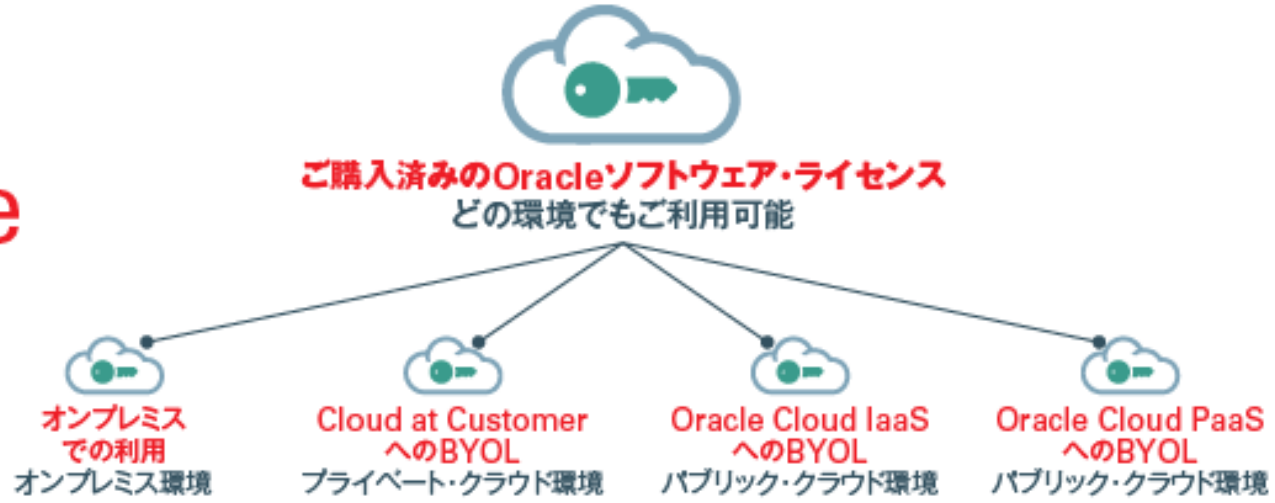
技術コラム しばちょう先生の試して納得！DBAへの道

技術コラム 津島博士のパフォーマンス講座

もしもみなみんがDBをクラウドで動かしてみたら

Bring Your Own License

既存のオラクル・ライセンスを
柔軟にクラウド環境で活用



300ドル分の無料トライアルでOracle Cloudを体験!

https://cloud.oracle.com/ja_JP/tryit

Oracle Cloudでは各種クラウドサービスを300ドル分無料でお試しいただけるトライアルサービスをご提供しております。無料トライアルのお申込み方法の詳細は、左のQRコード、またはURLにアクセスしてください。

Oracle Cloudのユースケース、導入事例、資料、価格などの詳細情報は、下記URLにアクセスしてください。
http://www.oracle.com/jp/cloud/platform/overview/index.html

～ みなさまの投稿をお待ちしております ～

Twitter

#OracleTechNight

こんな時、かけこむ会社が増えています。

Oracle Digitalは、オラクル製品の導入をご検討いただく際の総合窓口。
電話とインターネットによるダイレクトなコニュニケーションで、どんなお問い合わせにもすばやく対応します。
もちろん、無償。どんなことでも、ご相談ください。

Oracle Digital

お問い合わせは電話またはWebフォーム

☎ 0120-155-096
受付時間　月～金　9:00-12:00 / 13:00-17:00
（祝日および年末年始休業日を除きます）

http://www.oracle.com/jp/contact-us

ORACLE

Integrated Cloud

Applications & Platform Services


ORACLE®