IBM® DB2 Universal Database™

システム・モニター ガイドおよびリファレンス

バージョン 8.2

SC88-9157-01
(英文原典：SC09-4847-01)

IBM® DB2 Universal Database™

システム・モニター ガイドおよびリファレンス

バージョン 8.2

SC88-9157-01
(英文原典：SC09-4847-01)

ご注意！

本書および本書で紹介する製品をご使用になる前に、『*特記事項*』に記載されている情報をお読みください。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

`http://www.ibm.com/jp/manuals/main/mail.html`

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

`http://www.ibm.com/jp/manuals/` の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原　典： SC09–4847–01
IBM® DB2 Universal Database™
System Monitor Guide and Reference
Version 8.2

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2004.8

この文書では、平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財）日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

目次

第 1 部 システム・モニター・ガイド

第 1 章 データベース・システム・モニターの紹介

データベース・システム・モニター

データベースのモニターは、データベース管理システムのパフォーマンスと正常性を保守するためにきわめて重要なアクティビティーです。モニターを容易にするために、DB2® はデータベース・マネージャー、データベース、および他の接続されているアプリケーションから情報を収集します。この情報を使用して、例えば以下の事柄を行うことができます。

- データベース使用パターンに基づいたハードウェア要件の予測
- 個々のアプリケーションまたは SQL 照会のパフォーマンスの分析
- 索引および表の使用量の追跡
- システム・パフォーマンス低下の原因の正確な指摘
- 最適化アクティビティー (データベース・マネージャー構成パラメーターの変更、索引の追加、SQL 照会の変更など) の影響の評価

システム・モニター情報にアクセスするための 2 つの主要なツールがあります。それはスナップショット・モニターとイベント・モニターです。これらはそれぞれ別の目的で使用します。スナップショット・モニターでは、特定の時点 (スナップショットが取られる瞬間) のデータベース・アクティビティーの状態のピクチャーをキャプチャーすることができます。イベント・モニターでは、指定されたデータベース・イベントが発生したときにデータをログに記録します。

システム・モニターでは、モニター・データを表示するための方法が複数提供されています。スナップショット・モニターとイベント・モニターのどちらの場合にも、モニター情報をファイルまたは SQL 表に保管したり、 画面に表示したり (宛先を標準出力にする)、またはクライアント・アプリケーションで処理したりするためのオプションがあります。

関連概念:

- 51 ページの『イベント・モニター』
- 5 ページの『カウンターの状況および可視性』
- 4 ページの『データベース・システム・モニターのデータ編成』
- 7 ページの『データベース・システム・モニターのメモリー所要量』
- 21 ページの『スナップショット・モニター』

注:

データベース・システム・モニターのデータ編成

データベース・システム・モニターは、収集した情報を、モニター・エレメント と呼ばれるエンティティーに保管します (モニター・エレメントは、以前にはデータ・エレメントと呼ばれていました)。各モニター・エレメントは、データベース・システムの状態の特定のある局面に関する情報を保管します。さらに、モニター・エレメントはユニークな名前によって識別され、特定のタイプの情報を保管します。

モニター・エレメントがデータを保管できる使用可能なエレメント・タイプには、次のものがあります。

カウンター
あるアクティビティーが起こった回数を数えます。モニター中は、カウンター値は増加します。大部分のカウンター・エレメントはリセットできます。

ゲージ
ある項目の現行値を示します。ゲージ値は、データベース・アクティビティーによって上下します (例えば、保持されているロックの数)。ゲージ・エレメントはリセットできません。

水準点
水準点は、モニターを開始してからエレメントが到達した最高 (最大) または最低 (最小) 値を示します。水準点エレメントはリセットできません。

情報
モニター・アクティビティーの参照タイプの詳細を提供します。これには、パーティション名、別名、およびパス詳細などが含まれます。情報エレメントはリセットできません。

タイム・スタンプ
1970 年の 1 月 1 日以降の経過した秒およびマイクロ秒数を提供することによって、アクティビティーが起こった日時を示します。スナップショット・モニターおよびイベント・モニターの場合、タイム・エレメントの収集は「タイム・スタンプ」モニター・スイッチによって制御されます。このスイッチはデフォルトでは ON ですが、データベース・インスタンスでの CPU 使用率が 100% に近づいた場合には、パフォーマンス上の理由でそれを OFF にする必要があります。タイム・スタンプ・エレメントはリセットできません。

時間
アクティビティーで使用された秒数およびマイクロ秒数を戻します。スナップショット・モニターおよびイベント・モニターの場合、大半時間エレメントの収集は「タイム」モニター・スイッチによって制御されます。このスイッチはデフォルトでは ON ですが、データベース・インスタンスでの CPU 使用率が 100% に近づいた場合には、パフォーマンス上の理由でそれを OFF にする必要があります。時間エレメントのいくつかはリセットできます。

モニター・エレメントは、1 つ以上の論理データ・グループのデータを収集します。論理データ・グループは、データベース・アクティビティーの特定の範囲のデータベース・システム・モニター情報を収集するモニター・エレメントの集合です。モニター・エレメントは、それが提供する情報のレベルに基づいて論理データ・グループにソートされます。例えば、スナップショット・モニター中に、ソー

ト合計時間モニター・エレメントがデータベース (dbase)、アプリケーション (appl)、およびステートメント (stmt) 情報を戻します。したがって、このモニター・エレメントは括弧内に示した論理データ・グループに属します。

多くのモニター・エレメントは、スナップショット・モニターとイベント・モニターの両方によって使用されますが、それらはそれぞれ別個のセットの論理データ・グループを使用します。これは、スナップショットをキャプチャーできるデータベース・アクティビティーの範囲が、イベント・データを収集できるものの範囲とは異なるからです。実際、スナップショット・モニターからアクセス可能なモニター・エレメントのセット全体は、イベント・モニターからアクセス可能なモニター・エレメントのセット全体とは異なっています。

関連概念:

- 3 ページの『データベース・システム・モニター』
- 11 ページの『システム・モニター・スイッチ』
- 51 ページの『イベント・モニター』
- 5 ページの『カウンターの状況および可視性』
- 21 ページの『スナップショット・モニター』

関連資料:

- 「*コマンド・リファレンス*」の『RESET MONITOR コマンド』

カウンターの状況および可視性

データベース・マネージャーによって収集されるモニター・エレメントには、いくつかの累積カウンターが含まれています。カウンターは、データベースまたはデータベース・マネージャーの操作時 (例えば、アプリケーションがトランザクションをコミットするたび) に累積されます。

カウンターが初期設定されるのは、その当該オブジェクトがアクティブになった時点です。例えば、データベース用に読み取られるバッファー・プール・ページ数 (基本モニター・エレメント) は、データベースが活動化されるとゼロに設定されます。

カウンターによっては、モニター・スイッチで制御されるものがあります。あるモニター・スイッチが OFF の場合には、その制御下にあるモニター・エレメントはデータを収集しません。モニター・スイッチが ON になると、関連したすべてのカウンターがゼロにリセットされます。

イベント・モニターによって戻されるカウンターは、イベント・モニターを活動化するとゼロにリセットされます。

イベント・モニターのカウントは、次の 1 つの開始点以後のカウントを表します。

- イベント・モニターの始動、データベース、表スペース、および表に対して。
- イベント・モニターの始動、既存の接続に対して。
- アプリケーション接続、モニターが開始された後で行われた接続に対して。
- モニターが開始された後の、次のトランザクション (作業単位) またはステートメントの開始。
- モニターが開始された後のデッドロックの発生。

イベント・モニターと、モニター・アプリケーション (スナップショット・モニター API を使用するアプリケーション) には、システム・モニター・データに関する独自の論理ビューがあります。したがって、カウンターのリセットや初期設定が行われる際には、そのリセットや初期設定を行うイベント・モニターかアプリケーションだけが関係することになります。イベント・モニター・カウンターをリセットするには、イベント・モニターをいったん OFF にしてから ON にしなければなりません。スナップショットをとるアプリケーションの場合、 RESET MONITOR コマンドを使用するといつでもカウンターのビューをリセットできます。

ステートメントが開始された後でステートメントのイベント・モニターを開始すると、モニターは次の SQL ステートメントの開始時に情報の収集を開始します。その結果、モニターの開始時にデータベース・マネージャーが実行していたステートメントについての情報は、イベント・モニターは戻しません。これはトランザクション情報についても同様です。

関連概念:

- 3 ページの『データベース・システム・モニター』
- 11 ページの『システム・モニター・スイッチ』
- 51 ページの『イベント・モニター』
- 4 ページの『データベース・システム・モニターのデータ編成』
- 21 ページの『スナップショット・モニター』

関連資料:

- 「コマンド・リファレンス」の『RESET MONITOR コマンド』

システム・モニター出力 : 自己記述型データ・ストリーム

モニター・データを画面に表示したり、SQL 表に保管したりすることのほかに、それを処理するクライアント・アプリケーションを開発することができます。システム・モニターは、スナップショット・モニターとイベント・モニターの両方について、自己記述型データ・ストリームを介してモニター・データを戻します。スナップショット・モニター・アプリケーションでは、スナップショット API を呼び出してスナップショットをキャプチャーした後、そのデータ・ストリームを直接処理することができます。

イベント・モニター・データの処理は、これとは異なり、データベース・イベントが発生するペースでイベント・データがアプリケーションに送信されます。パイプ・イベント・モニターの場合は、アプリケーションはイベント・データの到着を待機し、到着時にそれを処理します。ファイル・イベント・モニターの場合は、アプリケーションはイベント・ファイルを解析するため、イベント・レコードをバッチで処理します。

この自己記述型データ・ストリームによって、戻りデータを、一度に 1 つのエレメントずつ解析することができます。このことは、特定のアプリケーションや特定のデータベース状態に関する情報の検索など、モニターに関連した数多くの可能性を実現させるものとなります。

戻されるモニター・データは以下の形式になります。

size モニター・エレメントまたは論理データ・グループに保管されているデータのサイズ (バイト)。論理データ・グループの場合、これは論理グループ内の全データのサイズです。例として、データベース論理グループ (*db*) には、個々のモニター・エレメント (*total_log_used* など) やロールフォワード情報 (*rollforward*) のようなほかの論理データ・グループが含まれます。これには、'size'、'type'、および 'element' 情報に取られるサイズは組み込まれません。

type データに保管されたエレメントのタイプ (例えば、可変長ストリングまたは符号付き 32 ビット数値)。 *header* のエレメント・タイプは、エレメントの論理データ・グループを示します。

element id モニターによってキャプチャーされたモニター・エレメントの ID。論理データ・グループの場合、これはグループの ID です (例えば、 *collected*、*dbase*、または *event_db*)。

data あるモニター・エレメントに対して、モニターによって収集された値。論理データ・グループの場合、データはそれに属するモニター・エレメントから成っています。

モニター・エレメントのすべてのタイム・スタンプは、 2 つの符号なし 4 バイト・モニター・エレメント (秒およびマイクロ秒) で戻されます。これらは GMT (グリニッジ標準時) で 1970 年 1 月 1 日 以降の秒数で表されます。

モニター・エレメント内のストリングのサイズ・エレメントは、ストリング・エレメントの実際のデータ・サイズを表します。このサイズには、NULL 終止符は入っていません。ストリングが NULL で終了することはないからです。

関連概念:

- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』
- 46 ページの『スナップショット・モニター自己記述型データ・ストリーム』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』
- 18 ページの『モニター・スイッチ自己記述型データ・ストリーム』

関連資料:

- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

データベース・システム・モニターのメモリー所要量

データベース・システム・モニター・データを保守するのに必要なメモリーは、モニター・ヒープから割り当てられます。モニター・ヒープのサイズを制御するには mon_heap_sz 構成パラメーターを使用します。モニター・アクティビティーのために必要なメモリーの量は、次の要素によって大きく左右されます。

- モニター・アプリケーションの数
- イベント・モニターの数と性質
- 設定されたモニター・スイッチ
- データベース・アクティビティーのレベル

モニター・コマンドが SQLCODE -973 で失敗するときは、 mon_heap_sz の値を大きくすることを検討してください。

次の公式を使うと、モニター・ヒープに必要なページの概数を求めることができます。

```
(アプリケーションが使用するストレージ              +
 イベント・モニターが使用するストレージ            +
 モニター・アプリケーションが使用するストレージ    +
 ゲートウェイ・アプリケーションが使用するストレージ)       / 4096
```

アプリケーションごとに使用するストレージ:

- 「ステートメント」スイッチが OFFの場合はゼロ。
- 「ステートメント」スイッチが ON の場合:
  - 同時に実行されるステートメント (つまりアプリケーションが持つ可能性があるオープン・カーソルの数) ごとに 400® バイトを追加する。これはアプリケーションが実行するステートメントの累計ではありません。
  - パーティション・データベースの場合は、ステートメントごとに次を追加する。
    - 200 バイト * (サブセクションの平均数)
- アプリケーションが sqleseti() 情報を発行する場合は、ユーザー ID、アプリケーション名、ワークステーション名、およびアカウンティング・ストリングのサイズを追加する。

イベント・モニターごとに使用するストレージ:

- 1300 バイト
- 2 * バッファー・サイズ
- イベント・モニターがファイルに書き込まれる場合は、308 バイトを追加する。
- イベント・モニターのタイプが「データベース」の場合:
  - 2700 バイトを追加
  - ステートメント・キャッシュ内のステートメントごとに 100 バイトを追加
- イベント・モニターのタイプが「表」の場合:
  - 600 バイトを追加
  - アクセスされる表ごとに 75 バイトを追加
- イベント・モニターのタイプが「表スペース」の場合:
  - 10 バイトを追加
  - 表スペースごとに 250 バイトを追加
- イベント・モニターのタイプが「バッファー・プール」の場合:
  - 10 バイトを追加
  - バッファー・プールごとに 250 バイトを追加
- イベント・モニターのタイプが「接続」の場合:
  - 600 バイトを追加
  - 接続されるアプリケーションごとに:

- 600 バイトを追加

– 上記の『アプリケーションごとに使用するストレージ』を追加することを忘れない。

モニター・アプリケーションごとに使用するストレージ:

- 250 バイト
- リセットされるデータベースごとに:
  - 350 バイト
  - リモート・データベースごとに 200 バイトを追加
  - 「ソート」スイッチがステートメントの場合は 25 バイトを追加
  - 「ロック」スイッチが ON の場合は 25 バイトを追加
  - 「表」スイッチが ON の場合:
    - 600 バイトを追加
    - アクセスされる表ごとに 75 バイトを追加
  - 「バッファー・プール」スイッチが ON の場合:
    - 300 バイトを追加
    - アクセスされる表スペースごとに 250 バイトを追加
    - アクセスされるバッファー・プールごとに 250 バイトを追加
  - 「ステートメント」スイッチが ON の場合:
    - 2100 バイトを追加
    - ステートメントごとに 100 バイトを追加
  - データベースに接続されるアプリケーションごとに:
    - 600 バイトを追加
    - アプリケーションの接続先のリモート・データベースごとに 200 バイトを追加
    - 「ソート」スイッチが ON の場合は 25 バイトを追加
    - 「ロック」スイッチが ON の場合は 25 バイトを追加
    - 「バッファー・プール」スイッチが ON の場合は 250 バイトを追加
- リセットされる DCS データベースごとに:
  - そのデータベース用に 200 バイトを追加
  - そのデータベースに接続されるアプリケーションごとに 200 バイトを追加
  - 「ステートメント」スイッチが ON で、伝送レベルのデータがリセットされる場合:
    - データベースごとに、伝送レベルごとに 200 バイトを追加
    - アプリケーションごとに、伝送レベルごとに 200 バイトを追加

ゲートウェイ・アプリケーションが使用するストレージ:

- ホスト・データベースごとに 250 バイト (すべてのスイッチが OFF の場合でも)
- アプリケーションごとに 400 バイト (すべてのスイッチが OFF の場合でも)

- 「ステートメント」スイッチが ON の場合:
  - アプリケーションごとに、同時に実行されるステートメント (つまりアプリケーションが持つ可能性があるオープン・カーソルの数) ごとに 200 バイトを追加する。これはアプリケーションが実行するステートメントの累計ではありません。
  - 伝送レベルのデータは次のように計算する。
    - データベースごとに、伝送レベルごとに 200 バイトを追加
    - アプリケーションごとに、伝送レベルごとに 200 バイトを追加
- 「作業単位」スイッチが ON の場合:
  - アプリケーションごとに 50 バイトを追加
- TMDB (SYNCPOINT TWOPHASE アクティビティー用) を使用するアプリケーションごとに:
  - 20 バイトにさらに XID 自体のサイズを追加
- sqleseti を発行してクライアント名、アプリケーション名、ワークステーション、またはアカウンティングを設定するアプリケーションの場合:
  - 800 バイトにさらにアカウンティング・ストリング自体のサイズを追加

関連概念:

- 3 ページの『データベース・システム・モニター』
- 11 ページの『システム・モニター・スイッチ』

関連タスク:

- 13 ページの『CLP からのモニター・スイッチの設定』

関連資料:

- 「*管理ガイド: パフォーマンス*」の『mon_heap_sz - 「データベース・システム・モニター・ヒープ・サイズ」構成パラメーター』

第 2 章 システム・モニター・スイッチ

システム・モニター・スイッチ

システム・モニター・データを収集すると、データベース・マネージャーの処理オーバーヘッドが発生します。例えば、SQL ステートメントの実行時間を計算するには、すべてのステートメントの実行前後にデータベース・マネージャーがオペレーティング・システムを呼び出して、タイム・スタンプを入手しなければなりません。この種のシステム呼び出しには通常はコストがかかります。システム・モニターによって発生するオーバーヘッドの別の形は、メモリー使用量の増加です。システム・モニターによって追跡されるすべてのモニター・エレメントについて、データベース・マネージャーはメモリーを使用し、収集されたデータを保管します。

モニター情報の保守に関連したオーバーヘッドを最小限にとどめるために、モニター・スイッチを使って、高コストになりかねないデータベース・マネージャーによるデータ収集を制御します。各スイッチには、ON と OFF という 2 つの設定値だけがあります。モニター・スイッチが OFF の場合には、そのスイッチの制御下にあるモニター・エレメントは情報を収集しません。スイッチの制御下にはなく、またスイッチの設定に関係なく常に収集される、基本モニター・データは相当量あります。

それぞれのモニター・アプリケーションには、モニター・スイッチ (およびシステム・モニター・データ) の独自の論理ビューがあります。始動時に、各アプリケーションはそのモニター・スイッチ設定値を、データベース・マネージャー構成ファイル内の dft_monswitches パラメーターから継承します (インスタンス・レベルで)。モニター・アプリケーションは UPDATE MONITOR SWITCHES USING MONSWITCH OFF/ON コマンドを使用して、モニター・スイッチ設定値を変更することができます。 MONSWITCH パラメーターは、下記の「スナップショット・モニター・スイッチ」表の「モニター・スイッチ」欄の値を保持しています。スイッチの設定値をアプリケーション・レベルで変更すると、スイッチを変更したアプリケーションだけに影響が及びます。

インスタンス・レベルのモニター・スイッチの変更は、データベース管理システムを停止せずに行うことができます。これを行うには、UPDATE DBM CFG USING DBMSWITCH OFF/ON コマンドを使用します。 DBMSWITCH パラメーターは、下記の「スナップショット・モニター・スイッチ」表の「DBM パラメーター」欄の値を保持しています。このようにスイッチを動的に更新する際、更新を動的に有効にするためには、更新を実行しているアプリケーションを明示的にインスタンスにアタッチする必要があります。動的な更新を行っても、他の既存のスナップショット・アプリケーションへの影響はありません。新規のモニター・アプリケーションは、更新済みのインスタンス・レベルのモニター・スイッチ設定値を継承します。既存のモニター・アプリケーションが新規のデフォルトのモニター・スイッチ値を継承するには、いったん終了して、そのアタッチメントを再構築する必要があります。データベース・マネージャー構成ファイルのスイッチを更新すると、パーティション・データベース内のすべてのパーティションのスイッチも更新されます。

データベース・マネージャーでは、スナップショット・モニター・アプリケーションとそのスイッチの設定値がすべて追跡されます。 1 つのアプリケーションの構成内でスイッチが ON に設定されている場合は、データベース・マネージャーは必ずモニター・データを収集します。したがって、アプリケーションの構成内で、あるスイッチが OFF になっている場合でも、その同じスイッチが ON になっているアプリケーションが少なくとも 1 つある限り、データベース・マネージャーはデータを収集します。

時間およびタイム・スタンプ・エレメントの収集は、「タイム・スタンプ」スイッチによって制御されます。このスイッチを OFF にすると (デフォルトでは ON)、時間またはタイム・スタンプに関連したモニター・エレメントを判別するときにオペレーティング・システムが呼び出すタイム・スタンプをすべてスキップするよう、データベース・マネージャーに指示されます。 CPU の使用率が 100% に近づいたときには、このスイッチを OFF にすることが重要となります。そのような状況では、タイム・スタンプの発行によって、かなりのパフォーマンス低下が生じます。「タイム・スタンプ」スイッチと他のスイッチによって制御できるモニター・エレメントの場合は、それらのスイッチのいずれかを OFF にするとデータは収集されません。そのため、「タイム・スタンプ」スイッチを OFF にすると、他のモニター・スイッチの制御下にあるデータの全体コストが大幅に減少します。

イベント・モニターは、スナップショット・モニター・アプリケーションと同じような、モニター・スイッチによる影響は受けません。イベント・モニターが定義されると、指定されたイベント・タイプにより必要とされるインスタンス・レベルのモニター・スイッチを、自動的に ON にします。例えば、デッドロック・イベント・モニターは、「ロック」モニター・スイッチを自動的に ON にします。イベント・モニターが活動化されると、必要なモニター・スイッチが ON になります。イベント・モニターが非活動化されると、モニター・スイッチは OFF になります。

「タイム・スタンプ」モニター・スイッチは、イベント・モニターによって自動的に設定されません。これは、イベント・モニターの論理データ・グループに属するモニター・エレメントの収集を制御する、唯一のモニター・スイッチです。「タイム・スタンプ」スイッチが OFF の場合は、イベント・モニターによって収集されるタイム・スタンプおよび時間モニター・エレメントの大半が収集されません。これらのエレメントは、依然として指定された表やファイル、またはパイプに書き込まれますが、その値はゼロとなります。

表 1. スナップショット・モニター・スイッチ

モニター・スイッチ	DBM パラメーター	提供される情報
BUFFERPOOL	DFT_MON_BUFPOOL	読み取りおよび書き込みの数、かかった時間
LOCK	DFT_MON_LOCK	ロック待機時間、デッドロック
SORT	DFT_MON_SORT	使用されたヒープ数、ソート・パフォーマンス
STATEMENT	DFT_MON_STMT	開始/停止時刻、ステートメント識別
TABLE	DFT_MON_TABLE	アクティビティーの程度 (読み取られた/書き込まれた行)

表 1. スナップショット・モニター・スイッチ (続き)

モニター・スイッチ	DBM パラメーター	提供される情報
UOW	DFT_MON_UOW	開始/終了時刻、完了状況
TIMESTAMP	DFT_MON_TIMESTAMP	タイム・スタンプ

関連概念:

- 51 ページの『イベント・モニター』
- 21 ページの『スナップショット・モニター』
- 18 ページの『モニター・スイッチ自己記述型データ・ストリーム』

関連タスク:

- 16 ページの『クライアント・アプリケーションからのモニター・スイッチの設定』
- 13 ページの『CLP からのモニター・スイッチの設定』

CLP からのモニター・スイッチの設定

スナップショットをキャプチャーしたり、イベント・モニターを使用したりする前に、まずデータベース・マネージャーに収集させる必要のあるデータを指定します。次の特殊タイプ・データのいずれかを収集したい場合には、該当するモニター・スイッチを設定してください。

- バッファー・プール・アクティビティー情報
- ロック待機、および時間関連のロック情報
- ソート情報
- SQL ステートメント情報
- 表アクティビティー情報
- 時間およびタイム・スタンプ情報
- 作業単位情報

上記の情報タイプに対応するスイッチは、デフォルトではすべて OFF になっています。ただし、時間およびタイム・スタンプ情報に対応するスイッチだけは、デフォルトで ON になっています。

注: イベント・モニターは、時間およびタイム・スタンプ情報スイッチによってのみ影響を受けます。その他のすべてのスイッチ設定は、イベント・モニターによって収集されるデータには影響しません。

前提条件:

モニター・スイッチの更新を実行するアプリケーションには、インスタンス・アタッチメントがなければなりません。

次のコマンドを使用するには、SYSADM、SYSCTRL、SYSMAINT、または SYSMON のいずれかの権限が必要です。

- UPDATE MONITOR SWITCHES
- GET MONITOR SWITCHES

- GET DATABASE MANAGER MONITOR SWITCHES

UPDATE DBM CFG コマンドを使用するには、SYSADM 権限を持っている必要があります。

手順 (UPDATE MONITOR SWITCHES):

- ローカル・モニター・スイッチを活動化するには、UPDATE MONITOR SWITCHES コマンドを使用する。次の例では、ローカル・モニター・スイッチをすべて ON に更新します。

  ```
  db2 update monitor switches using BUFFERPOOL on LOCK on
      SORT on STATEMENT on TIMESTAMP on TABLE on UOW on
  ```

  アプリケーション (CLP) が切り離すか、または別の UPDATE MONITOR SWITCHES コマンドによってスイッチが非活動化されるまで、スイッチはアクティブのままです。
- ローカル・モニター・スイッチを非活動化するには、UPDATE MONITOR SWITCHES コマンドを使用する。次の例では、ローカル・モニター・スイッチをすべて OFF に更新します。

  ```
  db2 update monitor switches using BUFFERPOOL off, LOCK off,
      SORT off, STATEMENT off, TIMESTAMP off, TABLE off, UOW off
  ```

  以下は、上記の UPDATE MONITOR SWITCH コマンドを発行した後に表示される出力例です。

  ```
            Monitor Recording Switches

  Switch list for db partition number 1
  Buffer Pool Activity Information (BUFFERPOOL) = OFF
  Lock Information                         (LOCK) = OFF
  Sorting Information                      (SORT) = OFF
  SQL Statement Information             (STATEMENT) = OFF
  Table Activity Information                (TABLE) = OFF
  Unit of Work Information                    (UOW) = OFF
  Get timestamp information             (TIMESTAMP) = OFF
  ```
- モニター・スイッチをデータベース・マネージャー・レベルで操作することもできる。これには、UPDATE DBM CFG コマンドを使用して、データベース・マネージャー構成ファイル内の dft_monswitches パラメーターを変更することが必要となります。

  ```
  db2 update dbm cfg using DFT_MON_LOCK on
  ```

  上記の例では、基本情報に加えて、ロック・スイッチによって制御される情報だけが収集されます。

  モニター・アプリケーションが開始されると、必ずデータベース・マネージャーからそのモニター・スイッチ設定を継承します。データベース・マネージャーのモニター・スイッチ設定を変更しても、実行中のモニター・アプリケーションには影響を与えません。モニター・アプリケーションは、自分自身をインスタンスに再度アタッチして、モニター・スイッチ設定の変更を取り出す必要があります。
- パーティション・データベース・システムの場合、特定のパーティションについてモニター・スイッチを固有に設定するか、またはすべてのパーティションについてグローバルに設定することができる。特定のパーティション (例えば、パー

ティション番号 3) に対してモニター・スイッチ (例えば、BUFFERPOOL) を設定するには、次のコマンドを発行します。

```
db2 update monitor switches using BUFFERPOOL on
    at dbpartitionnum 3
```

すべてのパーティションについてモニター・スイッチ (例えば、SORT) を設定するには、次のコマンドを発行します。

```
db2 update monitor switches using SORT on global
```

手順 (GET MONITOR SWITCHES):

- ローカル・モニター・スイッチの状況をチェックするには、GET MONITOR SWITCHES コマンドを使用する。

  ```
  db2 get monitor switches
  ```

- パーティション・データベース・システムの場合、特定のパーティションについてモニター・スイッチ設定を固有に表示するか、またはすべてのパーティションについてグローバルな設定を表示することができる。特定のパーティション (例えば、パーティション番号 2) に対してモニター・スイッチ設定を表示するには、次のコマンドを発行します。

  ```
  db2 get monitor switches at dbpartitionnum 2
  ```

  すべてのパーティションについてのモニター・スイッチ設定を表示するには、次のコマンドを発行します。

  ```
  db2 get monitor switches global
  ```

手順 (GET DATABASE MANAGER MONITOR SWITCHES):

- データベース・マネージャー・レベル (またはインスタンス・レベル) でモニター・スイッチの状況をチェックするには、 GET DATABASE MANAGER MONITOR SWITCHES コマンドを使用する。このコマンドは、モニター対象となっているインスタンスのスイッチ設定全体を表示します。

  ```
  db2 get database manager monitor switches
  ```

  以下は、上記のコマンドを発行した後に表示される出力例です。

  ```
           DBM System Monitor Information Collected

  Switch list for db partition number 1
  Buffer Pool Activity Information (BUFFERPOOL) = OFF
  Lock Information                         (LOCK) = ON    10-25-2001 16:04:39
  Sorting Information                      (SORT) = OFF
  SQL Statement Information             (STATEMENT) = OFF
  Table Activity Information               (TABLE) = OFF
  Unit of Work Information                  (UOW) = OFF
  Get timestamp information             (TIMESTAMP) = OFF
  ```

これで、目的のモニター・スイッチを設定し、スイッチ設定を確認したので、モニター・データのキャプチャーおよび収集の準備ができました。

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 51 ページの『イベント・モニター』
- 21 ページの『スナップショット・モニター』

関連タスク:

• 16 ページの『クライアント・アプリケーションからのモニター・スイッチの設定』

クライアント・アプリケーションからのモニター・スイッチの設定

スナップショットをキャプチャーしたり、イベント・モニターを使用したりする前に、データベース・マネージャーに収集させる必要のあるデータを指定する必要があります。次の特殊タイプ・データのいずれかを収集したい場合には、該当するモニター・スイッチを設定する必要があります。

- バッファー・プール・アクティビティー情報
- ロック、ロック待機、および時間関連のロック情報
- ソート情報
- SQL ステートメント情報
- 表アクティビティー情報
- 時間およびタイム・スタンプ情報
- 作業単位情報

上記の情報タイプに対応するスイッチは、デフォルトではすべて OFF になっています。ただし、時間およびタイム・スタンプ情報に対応するスイッチだけは、デフォルトで ON になっています。

注: イベント・モニターは、時間およびタイム・スタンプ情報スイッチによってのみ影響を受けます。その他のすべてのスイッチ設定は、イベント・モニターによって収集されるデータには影響しません。

前提条件:

モニター・スイッチの更新を実行するアプリケーションには、インスタンス・アタッチメントがなければなりません。

db2MonitorSwitches API を使用するには、SYSADM、SYSCTRL、SYSMAINT、または SYSMON 権限が必要です。

手順:

1. 次の DB2 ライブラリー sqlutil.h および db2ApiDf.h を組み込みます。これらは sqllib の下の include サブディレクトリーにあります。

   ```
   #include <sqlutil.h>
   #include <db2ApiDf.h>
   #include <string.h>
   #include <sqlmon.h>
   ```

2. スイッチ・リストのバッファー単位サイズを 1 KB に設定します。

   ```
   #define SWITCHES_BUFFER_UNIT_SZ 1024
   ```

3. sqlca、db2MonitorSwitches、および sqlm_recording_group 構造体を初期化します。また、スイッチ・リスト・バッファーを含むようにポインターを初期化し、バッファーのサイズを設定します。

   ```
   struct sqlca sqlca;
   memset (&sqlca, '¥0', sizeof(struct sqlca));
   db2MonitorSwitchesData switchesData;
   memset (&switchesData, '¥0', sizeof(switchesData));
   ```

```
struct sqlm_recording_group switchesList[SQLM_NUM_GROUPS];
memset(switchesList, '¥0', sizeof(switchesList));
sqluint32 outputFormat;
static sqluint32 switchesBufferSize = SWITCHES_BUFFER_UNIT_SZ;
char *switchesBuffer;
```

4. スイッチ・リスト出力を保持するバッファーを初期化します。

```
switchesBuffer = (char *)malloc(switchesBufferSize);
memset(switchesBuffer, '¥0', switchesBufferSize));
```

5. ローカル・モニター・スイッチの状態を変更するには、 sqlm_recording_group 構造体内のエレメント (前のステップでは switchesList という名前) を変更します。モニター・スイッチをオンにするには、パラメーター input_state を SQLM_ON に設定する必要があります。モニター・スイッチを OFF にするには、パラメーター input_state を SQLM_OFF に設定する必要があります。

```
switchesList[SQLM_UOW_SW].input_state = SQLM_ON;
switchesList[SQLM_STATEMENT_SW].input_state = SQLM_ON;
switchesList[SQLM_TABLE_SW].input_state = SQLM_ON;
switchesList[SQLM_BUFFER_POOL_SW].input_state = SQLM_OFF;
switchesList[SQLM_LOCK_SW].input_state = SQLM_OFF;
switchesList[SQLM_SORT_SW].input_state = SQLM_OFF;
switchesList[SQLM_TIMESTAMP_SW].input_state = SQLM_OFF;
switchesData.piGroupStates = switchesList;
switchesData.poBuffer = switchesBuffer;
switchesData.iVersion = SQLM_DBMON_VERSION8;
switchesData.iBufferSize = switchesBufferSize;
switchesData.iReturnData = 0;
switchesData.iNodeNumber = SQLM_CURRENT_NODE;
switchesData.poOutputFormat = &outputFormat;
```

   iVersion が SQLM_DBMON_VERSION8 より小さい場合には、SQLM_TIMESTAMP_SW を使用できないことに注意してください。

6. スイッチ設定の変更を実行するには、db2MonitorSwitches() 関数を呼び出します。 db2MonitorSwitches API に対するパラメーターとして、db2MonitorSwitchesData (この例では switchesData) 構造体を渡します。switchesData には、パラメーターとして sqlm_recording_group 構造体が含まれています。

```
db2MonitorSwitches(db2Version810, &switchesData, &sqlca);
```

7. スイッチ・リスト・バッファーからのスイッチ・リスト・データ・ストリームを処理します。
8. スイッチ・リスト・バッファーをクリアします。

```
free(switchesBuffer);
free(pRequestedDataGroups);
```

これで、目的のモニター・スイッチを設定し、スイッチ設定を確認したので、モニター・データのキャプチャーおよび収集の準備ができました。

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 51 ページの『イベント・モニター』
- 21 ページの『スナップショット・モニター』
- 18 ページの『モニター・スイッチ自己記述型データ・ストリーム』

関連タスク:

- 13 ページの『CLP からのモニター・スイッチの設定』

関連資料:

- 「*管理 API リファレンス*」の『db2MonitorSwitches - モニター・スイッチの入手 / 更新』

関連サンプル:

- 『clisnap.c -- Capture a snapshot at the client level (C)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C)』
- 『dbsnap.c -- Capture a snapshot at the database level (C)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C)』
- 『insnap.c -- Capture a snapshot at the instance level (C)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C)』
- 『utilsnap.c -- Utilities for the snapshot monitor samples (C)』
- 『clisnap.C -- Capture a snapshot at the client level (C++)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C++)』
- 『dbsnap.C -- Capture a snapshot at the database level (C++)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C++)』
- 『insnap.C -- Capture a snapshot at the instance level (C++)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C++)』
- 『utilsnap.C -- Utilities for the snapshot monitor samples (C++)』
- 『dbsnap.cbl -- Get a database monitor snapshot (IBM COBOL)』

モニター・スイッチ自己記述型データ・ストリーム

現行のモニター・スイッチ設定値を db2MonitorSwitches API を使って更新または表示すると、この API はスイッチ設定値を自己記述型データ・ストリームとして戻します。 19 ページの図 1 では、パーティション・データベース環境の場合に戻されるスイッチ・リスト情報の構造を示します。

注:

1. これらの例や表では、ID として記述名を使用しています。これらの名前は、実際のデータ・ストリームでは **SQLM_ELM_** という接頭部が付きます。例えば db_event は、イベント・モニター出力では SQLM_ELM_DB_EVENT と表示されます。タイプは、実際のデータ・ストリームでは **SQLM_TYPE_** という接頭部が付きます。例えば、ヘッダーはデータ・ストリームで SQLM_TYPE_HEADER と表示されます。
2. グローバル・スイッチ要求では、それぞれのスイッチ要求ごとに、情報が戻されるパーティションの順番が異なる可能性があります。この場合、パーティション ID がデータ・ストリームに組み込まれます。

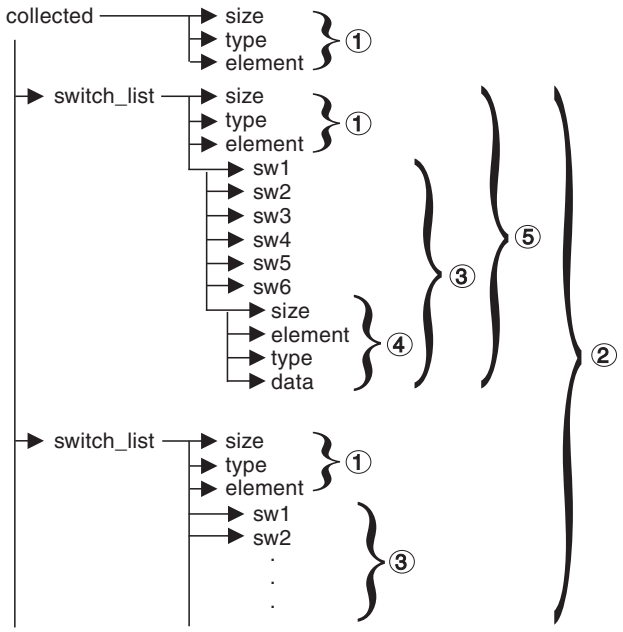

図 1. スイッチ・リスト・モニターのデータ・ストリーム

1. 各論理データ・グループは、サイズと名前を示すヘッダーで始まります。このサイズには、ヘッダー自体に取られるデータ・ボリュームは組み込まれません。
2. 収集したヘッダー内のサイズは、すべてのパーティションのすべてのモニター・スイッチ・リストの合計サイズを戻します。
3. スイッチ・リストのヘッダー内にあるサイズ・エレメントは、そのパーティションのスイッチ・データのサイズを表します。
4. スイッチ情報は、自己記述型です。
5. 非パーティション・データベースでは、独立型パーティションのスイッチ設定値が戻されます。つまり、スイッチ・リストが 1 つだけ戻されます。

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 6 ページの『システム・モニター出力 : 自己記述型データ・ストリーム』

関連タスク:

- 16 ページの『クライアント・アプリケーションからのモニター・スイッチの設定』

関連資料:

- 「*管理 API リファレンス*」の『db2MonitorSwitches - モニター・スイッチの入手 / 更新』

第 3 章 スナップショット・モニターの使用法

スナップショット・モニター

スナップショット・モニターを使用して、データベースおよび特定の時刻に接続しているアプリケーションに関する情報をキャプチャーすることができます。スナップショットは、データベース・システムの状況を判別するのに役立ちます。一定間隔でスナップショットを取れば、傾向の監視や潜在的な問題の予測にも有用です。指定の期間内に起こるすべてのデータベース・アクティビティーについて、モニター・データを取得するには、イベント・モニターを使用できます。

システム・モニターは、データベースがアクティブのときにのみ、それに関する情報を集計します。データベースからすべてのアプリケーションが切断して、データベースが非活動化すると、そのデータベースのシステム・モニター・データは入手不能になります。 ACTIVATE DATABASE コマンドを使用してデータベースを開始するか、データベースに対する永続接続を保守することにより、最後のスナップショットが取られるまでデータベースをアクティブにしておくこともできます。

スナップショット・モニターでは、インスタンス・アタッチメントが必要です。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。インスタンス・アタッチメントは通常、初めてデータベース・システム・モニター API を呼び出す時に、DB2INSTANCE 環境変数で指定されたインスタンスに対して暗黙的に行われます。また、ATTACH TO コマンドを使用して明示的にアタッチすることもできます。一度アプリケーションがアタッチされると、そのアプリケーションが呼び出すシステム・モニター要求は、すべてアタッチ先のインスタンスにあてられます。したがって、クライアント上のインスタンスにアタッチするだけで、そのクライアントからリモート・サーバーをモニターできるようになります。

パーティション・データベース環境では、スナップショットは、インスタンスの任意のパーティションでとることも、単一のインスタンス接続を使用してグローバルにとることもできます。グローバル・スナップショットは、それぞれのパーティションで収集されたデータを集約して単一の値セットを戻します。

スナップショットは CLP または SQL 表関数からキャプチャーしたり、 C または C++ で作成されたスナップショット・モニター API を使用することによってキャプチャーすることができます。さまざまなスナップショットの要求タイプが使用可能になっており、それぞれは特定のタイプのモニター・データを戻します。例えば、バッファー・プール情報だけを戻すスナップショットや、データベース・マネージャー情報を戻すスナップショットをキャプチャーすることができます。スナップショットをキャプチャーする前に、モニター・スイッチの制御下にあるモニター・エレメントからの情報が必要かどうかを考慮してください。あるモニター・スイッチが OFF の場合には、その制御下にあるモニター・エレメントは収集されません。

関連概念:

- 3 ページの『データベース・システム・モニター』
- 11 ページの『システム・モニター・スイッチ』
- 4 ページの『データベース・システム・モニターのデータ編成』
- 44 ページの『サブセクション・スナップショット』
- 45 ページの『パーティション・データベース・システムでのグローバル・スナップショット』

関連タスク:

- 36 ページの『クライアント・アプリケーションからのデータベース・スナップショットのキャプチャー』
- 33 ページの『CLP からのデータベース・スナップショットのキャプチャー』
- 23 ページの『データベース・システム・スナップショットへの SQL アクセス』

関連資料:

- 42 ページの『スナップショット・モニターの出力例』

システム・モニター・データに対するアクセス権: SYSMON 権限

SYSMON データベース・マネージャー・レベルのグループに属するユーザーには、データベース・システム・モニター・データにアクセスできる権限があります。システム・モニター・データにアクセスするには、スナップショット・モニター API、CLP コマンド、または SQL 表関数を使用します。

SYSMON 権限グループは、DB2_SNAPSHOT_NOAUTH レジストリー変数に代わって、システム管理またはシステム制御権限を持たないユーザーがデータベース・システム・モニター・データにアクセスできるようにする手段になります。

スナップショット・モニターを使用してシステム・モニター・データにアクセスする方法としては、 SYSMON 権限以外には、システム管理またはシステム制御権限を持つことしかありません。

SYSMON グループに属するユーザーや、システム管理またはシステム制御権限を持つユーザーは、以下のスナップショット・モニター関数を実行できます。

- CLP コマンド:
  - GET DATABASE MANAGER MONITOR SWITCHES
  - GET MONITOR SWITCHES
  - GET SNAPSHOT
  - LIST ACTIVE DATABASES
  - LIST APPLICATIONS
  - LIST DCS APPLICATIONS
  - RESET MONITOR
  - UPDATE MONITOR SWITCHES
- API:
  - db2GetSnapshot - スナップショットの取得

- db2GetSnapshotSize - *db2GetSnapshot()* 出力バッファーに必要なサイズの見積もり
- db2MonitorSwitches - モニター・スイッチの入手/更新
- db2ResetMonitor - モニターのリセット

- 以前に SYSPROC.SNAPSHOT_FILEW を実行していないスナップショット SQL 表関数

関連概念:

- 21 ページの『スナップショット・モニター』

データベース・システム・スナップショットへの SQL アクセス

スナップショット・モニター SQL 表関数 (*スナップショット表関数* と呼ばれる) を使用して、スナップショット・モニター・データにアクセスするには、以下の 2 とおりの方法があります。

- 直接アクセス
- ファイル・アクセス

直接アクセス

許可ユーザーは、スナップショット表関数で照会を発行し、モニター・データを含む結果セットを受けとることができます。この方法では、スナップショット・モニター・データへのアクセスは、 SYSADM、 SYSCTRL、 SYSMAINT、または SYSMON 権限を持つユーザーにのみ可能です。

直接アクセスを使用してスナップショット情報をキャプチャーするには、以下のようにします。

1. オプション : モニター・スイッチの状況の設定および検査を行います。
2. SQL を使用したデータベース・システム・スナップショットのキャプチャーを行います。

ファイル・アクセス

許可ユーザーは、スナップショット要求タイプ、および影響を受けるパーティションやデータベースを識別して、 SNAPSHOT_FILEW ストアード・プロシージャーを呼び出します。次に、SNAPSHOT_FILEW ストアード・プロシージャーは、データベース・サーバー上のファイルにモニター・データを保管します。

許可ユーザーがSNAPSHOT_FILEWストアード・プロシージャーを呼び出せる各要求タイプについて、すべてのユーザーは、対応するスナップショット表関数を使用して照会を発行することができます。受け取るモニター・データは、SNAPSHOT_FILEW ストアード・プロシージャーによって生成されたファイルからプルされます。

これは、すべてのユーザーにスナップショット・モニター・データへのアクセスを提供する安全な方法ですが、この方法には以下の制限があります。

- SNAPSHOT_FILEW ファイルから入手できるスナップショット・モニター・データは、最後に SNAPSHOT_FILEW ストアード・プロシージャーが呼び出されたときと同じくらい新しいものになる。通常のインターバルで SNAPSHOT_FILEW ストアード・プロシージャーへの呼び出しを行うことによって、最新のスナップショット・モニター・データが使用可能で

あることを確認することができます。例えば、UNIX システム上では、これを行うために cron ジョブを設定することができます。

- スナップショット表関数を使用して照会を発行しているユーザーは、モニターするデータベースまたはパーティションを識別することができない。SNAPSHOT_FILEW 呼び出しを発行しているユーザーによって識別されるデータベース名およびパーティション番号が、スナップショット表関数によってアクセス可能なファイルの内容を決定します。
- ユーザーが、対応する SNAPSHOT_FILEW 要求タイプが実行されていないスナップショット表関数を含む SQL 照会を発行すると、現在接続されているデータベースおよびパーティションに対して直接スナップショットが試行されます。この操作は、ユーザーが SYSADM、 SYSCTRL、SYSMAINT、または SYSMON 権限を持っている場合にのみ成功します。

以下のタスクは、データベース・システム・スナップショット情報をファイルにキャプチャーする SYSADM、SYSCTRL、SYSMAINT、または SYSMON ユーザーによって実行されます。

1. スナップショット要求を発行するユーザーの必要を調べます。特に、必要なモニター・データ、収集元のデータベース、および収集が特定のパーティションに限定される必要があるかどうかを判別します。
2. オプション : モニター・スイッチの状況の設定および検査を行います。
3. ファイルへのデータベース・システム・スナップショット情報のキャプチャーを行います。

いったん、SYSADM、SYSCTRL、SYSMAINT、または SYSMON ユーザーが上記のステップを完了したら、すべてのユーザーは、SQL 照会内のスナップショット表関数を使用してデータベース・システム・スナップショット情報にアクセスすることができます。

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 21 ページの『スナップショット・モニター』

関連タスク:

- 13 ページの『CLP からのモニター・スイッチの設定』
- 25 ページの『SQL 照会のスナップショット表関数を使用したデータベース・システムのスナップショットのキャプチャー (直接アクセス使用)』
- 27 ページの『SNAPSHOT_FILEW ストアード・プロシージャーを使用した、データベース・システム・スナップショット情報のファイルへのキャプチャー』
- 29 ページの『SQL 照会のスナップショット表関数を使用したデータベース・システムのスナップショットへのアクセス (ファイル・アクセス使用)』

関連資料:

- 31 ページの『スナップショット・モニター SQL 表関数』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

SQL 照会のスナップショット表関数を使用したデータベース・システムのスナップショットのキャプチャー (直接アクセス使用)

許可ユーザーは、SQL 照会のスナップショット表関数を使用することにより、 DB2 インスタンスに関するモニター情報のスナップショットをキャプチャーできます。例えば、SAMPLE データベースに関する一般アプリケーション情報のスナップショットは、次のようにしてキャプチャーします。

```
SELECT * FROM TABLE(SNAPSHOT_APPL('SAMPLE', -1))
                    AS SNAPSHOT_APPL
```

各スナップショット表関数は、1 つ以上の行があり、各列がモニター・エレメントを表す表を戻します。したがって、各モニター・エレメントの列名は、モニター・エレメントの名前に対応しています。

戻り表から個々のモニター・エレメントを選択することもできます。例えば、次のステートメントの場合は、 agent_id と appl_id のモニター・エレメントだけが戻されます。

```
SELECT agent_id, appl_id FROM TABLE(
                        SNAPSHOT_APPL(CAST (NULL as VARCHAR), -1))
                        AS SNAPSHOT_APPL
```

前提条件:

データベース・スナップショットを使用するには、SYSADM、 SYSCTRL、SYSMAINT、または SYSMON 権限が必要です。

リモート・インスタンスのスナップショットを取得するには、まず、そのインスタンスに属しているローカル・データベースに接続する必要があります。

制約事項:

バージョン 8.1 で導入されたスナップショット・ユーザー定義関数 (UDF) は、LIST DB DIRECTORY コマンドの発行時に**「ディレクトリー項目タイプ (Directory entry type)」** の値が「間接 (Indirect)」または「ホーム」として表示されるデータベースに対して使用することを意図しています。この UDF をリモート・データベースに対して使用すると、以下のエラーで UDF が失敗します。

```
SQL1427N An instance attachment does not exist.
```

この UDF は以下のいずれかと併用できません。

- モニター・スイッチ・コマンド/API
- モニター・リセット・コマンド/API

この制約事項には、以下のコマンドが含まれます。

- GET MONITOR SWITCHES
- UPDATE MONITOR SWITCHES
- RESET MONITOR

この制限の理由は、この種のコマンドは INSTANCE ATTACH を使用するのに対して、スナップショット UDF は DATABASE CONNECT を使用するためです。

手順:

スナップショット表関数を使用してスナップショットをキャプチャーするには、次のようにする必要があります。

1. データベースに接続します。これは、モニターする必要のあるインスタンス内のどのデータベースでもかまいません。スナップショット表関数を使用した SQL 照会は、データベースに接続していなければ発行できません。
2. キャプチャーする必要のあるスナップショットのタイプ、およびモニターする必要のあるデータベースとパーティションを決定します。
3. 該当するスナップショット表関数を使用して照会を発行します。例えば、次の照会では、現在接続しているパーティションの SAMPLE データベースに関するロック情報のスナップショットをキャプチャーします。

   ```
   SELECT * FROM TABLE(SNAPSHOT_LOCK('SAMPLE',-1)) AS SNAPSHOT_LOCK
   ```

   SQL 表関数には、以下の 2 つの入力パラメーターがあります。

   データベース名

   VARCHAR(255)。 NULL が入力された場合は、現在接続しているデータベースの名前が使用されます。

   パーティション番号

   SMALLINT。パーティション番号のパラメーターには、モニターする必要のあるパーティションの番号に対応する整数 (0 から 999 の間の値) を入力します。現在接続しているパーティションのスナップショットをキャプチャーする場合は、値 -1 または NULL を入力します。グローバル・スナップショットをキャプチャーするには、値 -2 を入力します。

   注:

   a. ただし、次に挙げるスナップショット表関数の場合は、現在接続しているデータベースを指定するために NULL を入力すると、インスタンス内のすべてのデータベースに関するスナップショット情報が戻されます。
      SNAPSHOT_DATABASE、 SNAPSHOT_APPL、 SNAPSHOT_APPL_INFO、 SNAPSHOT_LOCKWAIT、 SNAPSHOT_STATEMENT、 SNAPSHOT_AGENT、 SNAPSHOT_SUBSECT、 SNAPSHOT_BP
   b. データベース名パラメーターは、データベース・マネージャー・レベルのスナップショット表関数には適用されません。あるのは、パーティション番号用のパラメーターだけです。

関連概念:

- 21 ページの『スナップショット・モニター』

関連資料:

- 31 ページの『スナップショット・モニター SQL 表関数』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

SNAPSHOT_FILEW ストアード・プロシージャーを使用した、データベース・システム・スナップショット情報のファイルへのキャプチャー

SNAPSHOT_FILEW ストアード・プロシージャーでは、モニター・データのスナップショットをキャプチャーし、この情報をデータベース・サーバー上のファイルに保管することができます。これにより、すべてのユーザーがスナップショット表関数を使用した照会を行い、これらのファイルにあるスナップショット情報にアクセスできるようになります。そのため、スナップショット・モニター・データへのアクセスをオープンにすると、スナップショット表関数の実行権限を持つすべてのユーザーが、接続したユーザーのリストや、それらのユーザーがデータベースにサブミットした SQL ステートメントなどの機密情報にアクセス可能になってしまいます。スナップショット表関数の実行権限は、デフォルトで、PUBLIC に付与されています。

注: データベース上の実データやユーザー・パスワードがスナップショット表関数によって漏えいすることはありません。

SNAPSHOT_FILEW ストアード・プロシージャーへの呼び出しを発行するときは、モニターするデータベースとパーティションを識別することに加えて、*スナップショット要求タイプ* を指定する必要があります。各スナップショット要求タイプは、収集するモニター・データの有効範囲を決定します。各ユーザーが実行する必要のあるスナップショット表関数に基づいて、スナップショット要求タイプを選択してください。次の表は、スナップショット表関数とそれに対応する要求タイプのリストです。

表 2. スナップショット要求タイプ

スナップショット表関数	有効範囲 (すべてのデータベース (all) か特定のデータベースだけ (specific) か)	スナップショット要求タイプの番号
SNAPSHOT_DBM	-	1
SNAPSHOT_FCM	-	1
SNAPSHOT_FCMNODE	-	1
SNAPSHOT_SWITCHES	-	1
SNAPSHOT_DATABASE	all	9
SNAPSHOT_DATABASE	specific	2
SNAPSHOT_APPL	all	10
SNAPSHOT_APPL	specific	6
SNAPSHOT_APPL_INFO	all	10
SNAPSHOT_APPL_INFO	specific	6
SNAPSHOT_LOCKWAIT	all	10
SNAPSHOT_LOCKWAIT	specific	6
SNAPSHOT_STATEMENT	all	10
SNAPSHOT_STATEMENT	specific	6
SNAPSHOT_AGENT	all	10
SNAPSHOT_AGENT	specific	6
SNAPSHOT_SUBSECT	all	10

表 2. スナップショット要求タイプ (続き)

スナップショット表関数	有効範囲 (すべてのデータベース (all) か特定のデータベースだけ (specific) か)	スナップショット要求タイプの番号
SNAPSHOT_SUBSECT	specific	6
SNAPSHOT_TABLE	specific	5
SNAPSHOT_TBREORG	specific	5
SNAPSHOT_LOCK	specific	8
SNAPSHOT_TBS	specific	13
SNAPSHOT_TBS_CFG	specific	13
SNAPSHOT_QUIESCERS	specific	13
SNAPSHOT_CONTAINER	specific	13
SNAPSHOT_RANGES	specific	13
SNAPSHOT_BP	all	23
SNAPSHOT_BP	specific	22
SNAPSHOT_DYN_SQL	specific	36

前提条件:

SNAPSHOT_FILEW ストアード・プロシージャーを使用してデータベース・スナップショットをキャプチャーするには、 SYSADM、SYSCTRL、SYSMAINT、または SYSMON 権限が必要です。

手順:

SNAPSHOT_FILEW ストアード・プロシージャーを使用してスナップショットをファイルにキャプチャーするには、以下を行う必要があります。

1. データベースに接続します。これは、モニターする必要のあるインスタンス内のどのデータベースでもかまいません。ストアード・プロシージャーは、データベースに接続していなければ呼び出せません。
2. スナップショット要求タイプ、およびモニターする必要のあるデータベースとパーティションを決定します。
3. スナップショット要求タイプ、データベース、およびパーティションを指定する適切なパラメーターを設定して、 SNAPSHOT_FILEW ストアード・プロシージャーを呼び出します。例えば、次の呼び出しでは、現在接続しているパーティションの SAMPLE データベースに関するアプリケーション情報のスナップショットをキャプチャーします。

   ```
   CALL SNAPSHOT_FILEW(6,'SAMPLE',-1)
   ```

   SNAPSHOT_FILEW ストアード・プロシージャーには、3 つの入力パラメーターがあります。

   - SMALLINT: スナップショット要求タイプ。下の表『スナップショット要求タイプ』を参照してください。この表は、スナップショット表関数とそれに対応する要求タイプを相互参照させたものです。
   - VARCHAR (128): データベース名。 NULL が入力された場合は、現在接続しているデータベースの名前が使用されます。

注: このパラメーターは、データベース・マネージャー・レベルのスナップショット表関数には適用されません。あるのは、要求タイプとパーティション番号のパラメーターだけです。

- SMALLINT: パーティション番号 (0 から 999 の間の値)。パーティション番号パラメーターの場合は、モニターしたいパーティション番号に対応する整数を入力します。現在接続しているパーティションのスナップショットをキャプチャーする場合は、値 -1 または NULL を入力します。グローバル・スナップショットをキャプチャーするには、値 -2 を入力します。

関連概念:

- 21 ページの『スナップショット・モニター』

関連タスク:

- 29 ページの『SQL 照会のスナップショット表関数を使用したデータベース・システムのスナップショットへのアクセス (ファイル・アクセス使用)』

関連資料:

- 31 ページの『スナップショット・モニター SQL 表関数』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』
- 「*SQL Administrative Routines*」の『SNAPSHOT_FILEW プロシージャー』

SQL 照会のスナップショット表関数を使用したデータベース・システムのスナップショットへのアクセス (ファイル・アクセス使用)

許可ユーザーが SNAPSHOT_FILEW ストアード・プロシージャーを呼び出したすべての要求タイプについては、どのユーザーも、相当するスナップショット表関数を使用した照会を発行できます。ユーザーが受け取るモニター・データは、SNAPSHOT_FILEW ストアード・プロシージャーによって生成されたファイルから取り出されます。

SNAPSHOT_FILEW ファイルのスナップショット・データには、すべてのユーザーが SQL 照会でスナップショット表関数を使用することによってアクセスできます。例えば、SAMPLE データベースに関する一般アプリケーション情報のスナップショットは、次のようにして作成されます。

```
SELECT * FROM TABLE( SNAPSHOT_APPL(CAST(NULL AS VARCHAR(1)),
                                   CAST (NULL AS INTEGER)))
                     as SNAPSHOT_APPL
```

各スナップショット表関数は、1 つ以上の行があり、各列がモニター・エレメントを表す表を戻します。したがって、各モニター・エレメントの列名は、モニター・エレメントの名前に対応しています。

戻り表から個々のモニター・エレメントを選択することもできます。例えば、次のステートメントの場合は、 agent_id と appl_id のモニター・エレメントだけが戻されます。

```
SELECT agent_id, appl_id FROM TABLE(
                            SNAPSHOT_APPL(CAST(NULL AS VARCHAR(1)),
                                          CAST (NULL AS INTEGER)))
                            as SNAPSHOT_APPL
```

前提条件:

SNAPSHOT_FILEW ファイルにアクセスすることを目的として使用されるすべてのスナップショット表関数については、許可ユーザーが、該当するスナップショット要求タイプを設定して SNAPSHOT_FILEW ストアード・プロシージャーを発行している必要があります。

制約事項:

スナップショット表関数を使用して SNAPSHOT_FILEW ファイルのスナップショット・データにアクセスするユーザーには、モニターするデータベースやパーティションは識別できません。 SNAPSHOT_FILEW ファイルの内容は、ユーザーが SNAPSHOT_FILEW 呼び出しを実行して識別するデータベース名とパーティション番号によって決定されます。

SNAPSHOT_FILEW ファイルから得られるスナップショット・モニター・データは、直前に呼び出された SNAPSHOT_FILEW ストアード・プロシージャーでキャプチャーされたスナップショットだけです。

発行された SQL 照会に、該当する SNAPSHOT_FILEW 要求タイプが実行されていないスナップショット表関数が含まれている場合は、現在接続されているデータベースおよびパーティションへの直接のスナップショットが試行されます。この操作は、ユーザーが SYSADM、SYSCTRL、SYSMAINT、または SYSMON 権限を持っている場合にのみ成功します。

手順:

スナップショット表関数を使用して SNAPSHOT_FILEW ファイルのスナップショット・データにアクセスするには、以下を行う必要があります。

1. データベースに接続します。これは、モニターする必要のあるインスタンス内のどのデータベースでもかまいません。スナップショット表関数を使用した SQL 照会は、データベースに接続していなければ発行できません。
2. キャプチャーする必要のあるスナップショットのタイプを決定します。
3. 該当するスナップショット表関数を使用して照会を発行します。例えば、次の照会では、表スペース情報のスナップショットがキャプチャーされます。

```
SELECT * FROM TABLE(SNAPSHOT_TBS(CAST(NULL AS VARCHAR(1)),
                    CAST (NULL AS INTEGER))) AS SNAPSHOT_TBS
```

   注: データベース名やパーティション番号のパラメーターには、NULL 値を入力してください。スナップショットの対象となるデータベース名やパーティションは、 SNAPSHOT_FILEW ストアード・プロシージャーの呼び出しで決定されます。また、データベース名のパラメーターは、データベース・マネージャー・レベルのスナップショット表関数には適用されません。あるのは、パーティション番号のパラメーターだけです。

関連概念:

- 21 ページの『スナップショット・モニター』

関連タスク:

- 23 ページの『データベース・システム・スナップショットへの SQL アクセス』

- 25 ページの『SQL 照会のスナップショット表関数を使用したデータベース・システムのスナップショットのキャプチャー (直接アクセス使用)』
- 27 ページの『SNAPSHOT_FILEW ストアード・プロシージャーを使用した、データベース・システム・スナップショット情報のファイルへのキャプチャー』

関連資料:

- 31 ページの『スナップショット・モニター SQL 表関数』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

スナップショット・モニター SQL 表関数

利用可能なスナップショット・モニター SQL 表関数 (スナップショット*表関数* と呼ばれる) はたくさんの種類があり、それぞれがデータベース・システムの特定の領域に関するモニター･データを戻します。例えば、SNAPSHOT_BP スナップショット表関数はバッファー・プール情報のスナップショットをキャプチャーします。次の表は、利用可能な各スナップショット・モニター表関数をリストしています。

表 3. スナップショット・モニター SQL 表関数

モニター・レベル	SQL 表関数	戻される情報
データベース・マネージャー	SNAPSHOT_DBM	データベース・マネージャー・レベル情報。
データベース・マネージャー	SNAPSHOT_FCM	高速コミュニケーション・マネージャー (FCM) に関するデータベース・マネージャー・レベル情報。
データベース・マネージャー	SNAPSHOT_FCMNODE	高速コミュニケーション・マネージャー (FCM) に関する、あるパーティションのデータベース・マネージャー・レベル情報。
データベース・マネージャー	SNAPSHOT_SWITCHES	データベース・マネージャーのモニター・スイッチの設定値。
データベース	SNAPSHOT_DATABASE	データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
アプリケーション	SNAPSHOT_APPL	パーティション上のデータベースに接続されている各アプリケーションについての汎用アプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SNAPSHOT_APPL_INFO	パーティション上のデータベースに接続されている各アプリケーションについての汎用アプリケーション・レベル識別情報。
アプリケーション	SNAPSHOT_LOCKWAIT	パーティション上のデータベースに接続されているアプリケーションのロック待機についてのアプリケーション・レベル情報。
アプリケーション	SNAPSHOT_STATEMENT	パーティション上のデータベースに接続されているアプリケーションのステートメントについてのアプリケーション・レベル情報。これには、最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。

表 3. スナップショット・モニター SQL 表関数 (続き)

モニター・レベル	SQL 表関数	戻される情報
アプリケーション	SNAPSHOT_AGENT	パーティション上のデータベースに接続されているアプリケーションに関連したエージェントについてのアプリケーション・レベル情報。
アプリケーション	SNAPSHOT_SUBSECT	パーティション上のデータベースに接続されているアプリケーションのアクセス・プランのサブセクションについてのアプリケーション・レベル情報。
表	SNAPSHOT_TABLE	データベースに接続された各アプリケーションのデータベース・レベルとアプリケーション・レベルの表アクティビティー情報。データベースに接続されたアプリケーションが**アクセスした**各表の表レベルの表アクティビティー情報。表スイッチが必要です。
表	SNAPSHOT_TBREORG	データベース内で再編成を実行している各表に関する、表レベルでの表再編成情報。
ロック	SNAPSHOT_LOCK	データベースに接続された各アプリケーションについての、データベース・レベルおよびアプリケーション・レベルのロック情報。ロック・スイッチが必要です。
表スペース	SNAPSHOT_TBS	データベース・レベルの表スペース・アクティビティーに関する情報。また、データベースに接続された各アプリケーションのアプリケーション・レベル、およびデータベースに接続された各アプリケーションがアクセスしている各表スペースの表スペース・レベルの情報も含みます。バッファー・プール・スイッチが必要です。
表スペース	SNAPSHOT_TBS_CFG	表スペース構成に関する情報。
表スペース	SNAPSHOT_QUIESCERS	表スペース・レベルでの静止プログラムに関する情報。
表スペース	SNAPSHOT_CONTAINER	表スペース・レベルでの表スペース・コンテナー構成に関する情報。
表スペース	SNAPSHOT_RANGES	表スペース・マップの範囲に関する情報。
バッファー・プール	SNAPSHOT_BP	指定されたデータベースのバッファー・プール・アクティビティー・カウンター。バッファー・プール・スイッチが必要です。
動的 SQL	SNAPSHOT_DYN_SQL	データベースの SQL ステートメント・キャッシュからのポイント・イン・タイム指定ステートメント情報。

各スナップショット表関数は 1 行の情報を戻します。そこでは各列がモニター・エレメントを表します。

スナップショットをキャプチャーする前に、モニター・スイッチの制御下にあるモニター・エレメントからの情報が必要かどうかを考慮してください。あるモニター・スイッチが OFF の場合には、その制御下にあるモニター・エレメントは収集されません。必要とするエレメントをスイッチで制御できるかどうか判別するには、個々のモニター・エレメントを参照してください。

モニター表関数は、現行セッションで使用されている接続とは異なる、独立したインスタンス接続を使用します。したがって、デフォルトのデータベース・マネージ

ャー・モニター・スイッチのみ有効です。無効なモニター・スイッチには、現行セッションまたはアプリケーションから動的に ON/OFF されるスイッチが含まれます。

関連概念:

- 21 ページの『スナップショット・モニター』

関連タスク:

- 23 ページの『データベース・システム・スナップショットへの SQL アクセス』

関連資料:

- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

CLP からのデータベース・スナップショットのキャプチャー

CLP から GET SNAPSHOT コマンドを使用して、データベース・スナップショットをキャプチャーすることができます。多数の異なるスナップショット要求タイプを使用することができます。それらには、GET SNAPSHOT コマンドに特定のパラメーターを指定することによってアクセスできます。

データベース・スナップショットをキャプチャーするには、インスタンス・アタッチメントがなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンスのスナップショットを取得するには、まずそのインスタンスにアタッチする必要があります。

前提条件:

データベース・スナップショットを使用するには、SYSADM、 SYSCTRL、SYSMAINT、または SYSMON 権限が必要です。

手順:

1. オプション : モニター・スイッチの状況の設定および検査を行います。
2. CLP から、GET SNAPSHOT コマンドを必要なパラメーターを指定して発行します。次の例では、データベース・マネージャー・レベル情報がスナップショットにキャプチャーされます。

   ```
   db2 get snapshot for dbm
   ```

3. パーティション・データベース・システムの場合、特定のパーティションについてデータベース・スナップショットを固有にキャプチャーすることも、すべてのパーティションについてグローバルなスナップショットをキャプチャーすることもできます。特定のパーティション (例えば、パーティション番号 2) 上のすべてのアプリケーションについてのデータベース・スナップショットをキャプチャーするには、次のコマンドを発行します。

   ```
   db2 get snapshot for all applications at dbpartitionnum 2
   ```

   すべてのパーティション上のすべてのアプリケーションについてのデータベース・スナップショットをキャプチャーするには、次のコマンドを発行します。

   ```
   db2 get snapshot for all applications global
   ```

パーティション・データベース上のグローバル・スナップショットの場合、すべてのパーティションからのモニター・データが集約されます。

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 21 ページの『スナップショット・モニター』
- 45 ページの『パーティション・データベース・システムでのグローバル・スナップショット』

関連タスク:

- 13 ページの『CLP からのモニター・スイッチの設定』

関連資料:

- 「コマンド・リファレンス」の『GET SNAPSHOT コマンド』
- 34 ページの『スナップショット・モニター CLP コマンド』
- 42 ページの『スナップショット・モニターの出力例』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

スナップショット・モニター CLP コマンド

次の表に、サポートされているスナップショット要求のタイプをすべてリストします。いくつかの要求タイプの場合、一部の情報は、関連したモニター・スイッチが ON に設定されている場合にだけ戻されます。スイッチで必須エレメントを制御できるかどうか判別するには、個々のモニター・エレメントを参照してください。

表 4. スナップショット・モニター CLP コマンド

モニター・レベル	CLP コマンド	戻される情報
接続リスト	`list applications [show detail]`	スナップショットが取られたノード上の DB2 インスタンスが管理するデータベースに現在接続されている、すべてのアプリケーションのアプリケーション識別情報。
接続リスト	`list applications for database` *`dbname`* `[show detail]`	指定のデータベースに現在接続されている各アプリケーションに関するアプリケーション識別情報。
接続リスト	`list dcs applications`	スナップショットが取られたノード上の DB2 インスタンスが管理するデータベースに現在接続されている、すべての DCS アプリケーションのアプリケーション識別情報。
データベース・マネージャー	`get snapshot for dbm`	インスタンス・レベルのモニター・スイッチの設定値を含む、データベース・マネージャー・レベルの情報。
データベース・マネージャー	`get dbm monitor switches`	インスタンス・レベルのモニター・スイッチの設定値。
データベース	`get snapshot for database on` *`dbname`*	データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	`get snapshot for all databases`	パーティション上でアクティブな各データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	`list active databases`	個々のアクティブなデータベースに対する接続数。ACTIVATE DATABASE コマンドを使用して開始したものの、接続されていないデータベースを含みます。

表 4. スナップショット・モニター CLP コマンド (続き)

モニター・レベル	CLP コマンド	戻される情報
データベース	`get snapshot for dcs database on dbname`	特定の DCS データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	`get snapshot for remote database on dbname`	特定のフェデレーテッド・システム・データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	`get snapshot for all remote databases`	パーティション上でアクティブな各フェデレーテッド・システム・データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
アプリケーション	`get snapshot for application applid appl-id`	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for application agentid appl-handle`	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for applications on dbname`	パーティション上のデータベースに接続されている各アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for all applications`	パーティション上でアクティブな各アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for dcs application applid appl-id`	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for all dcs applications`	パーティション上にあるアクティブな各 DCS アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for dcs application agentid appl-handle`	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for dcs applications on dbname`	パーティション上にあるデータベースに接続されている各 DCS アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for remote applications on dbname`	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	`get snapshot for all remote applications`	パーティション上にあるアクティブな各フェデレーテッド・システム・アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。

表 4. スナップショット・モニター CLP コマンド (続き)

モニター・レベル	CLP コマンド	戻される情報
表	`get snapshot for tables on dbname`	データベースに接続された各アプリケーションのデータベース・レベルとアプリケーション・レベルの表アクティビティー情報。データベースに接続されたアプリケーションが**アクセスした**各表の表レベルの表アクティビティー情報。表スイッチが必要です。
ロック	`get snapshot for locks for application applid appl-id`	アプリケーションによって保持されているロックのリスト。ロック待機情報にはロック・スイッチが必要です。
ロック	`get snapshot for locks for application agentid appl-handle`	アプリケーションによって保持されているロックのリスト。ロック待機情報にはロック・スイッチが必要です。
ロック	`get snapshot for locks on dbname`	データベースに接続された各アプリケーションについての、データベース・レベルおよびアプリケーション・レベルのロック情報。ロック・スイッチが必要です。
表スペース	`get snapshot for tablespaces on dbname`	データベースの表スペースのアクティビティーに関する情報。バッファー・プール・スイッチが必要です。コンテナー、静止プログラム、および範囲に関する情報も含まれます。この情報はスイッチの制御下にはありません。
バッファー・プール	`get snapshot for all bufferpools`	バッファー・プール・アクティビティー・カウンター。バッファー・プール・スイッチが必要です。
バッファー・プール	`get snapshot for bufferpools on dbname`	指定されたデータベースのバッファー・プール・アクティビティー・カウンター。バッファー・プール・スイッチが必要です。
動的 SQL	`get snapshot for dynamic sql on dbname`	データベースの SQL ステートメント・キャッシュからのポイント・イン・タイム指定ステートメント情報。リモート・データ・ソースからの情報である場合もあります。

関連タスク:

- 13 ページの『CLP からのモニター・スイッチの設定』
- 33 ページの『CLP からのデータベース・スナップショットのキャプチャー』

関連資料:

- 「コマンド・リファレンス」の『GET SNAPSHOT コマンド』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

クライアント・アプリケーションからのデータベース・スナップショットのキャプチャー

C、C++、または COBOL アプリケーションでスナップショット・モニター API を使用して、データベース・スナップショットをキャプチャーすることができます。C および C++ では、db2GetSnapshot() に特定のパラメーターを指定することにより、多数の異なるスナップショット要求タイプにアクセスすることができます。

データベース・スナップショットをキャプチャーするには、インスタンス・アタッチメントがなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンスのスナップショットを取得するには、まずそのインスタンスにアタッチする必要があります。

前提条件:

db2MonitorSwitches API を使用するには、SYSADM、SYSCTRL、SYSMAINT、または SYSMON 権限が必要です。

手順:

1. オプション : モニター・スイッチの状況の設定および検査を行います。
2. DB2 ライブラリー、sqlmon.h および db2ApiDf.h を組み込みます。これらは sqllib の下の include サブディレクトリーにあります。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

3. スナップショットのバッファー単位サイズを 100 KB に設定します。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 102400
   ```

4. sqlca、sqlma、db2GetSnapshotData、および sqlm_collected 構造体を宣言します。また、スナップショット・バッファーを含むようにポインターを初期化し、バッファーのサイズを設定します。

   ```
   struct sqlma *pRequestedDataGroups;
   struct sqlca sqlca;
   memset (&sqlca, '¥0', sizeof(struct sqlca));
   struct sqlm_collected collectedData;
   memset (&collectedData, '¥0', sizeof(collectedData));
   db2GetSnapshotData getSnapshotParam;
   memset (&getSnapshotParam, '¥0', sizeof(getSnapshotParam));

   static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
   sqluint32 outputFormat;
   char *snapshotBuffer;
   ```

5. sqlma 構造体を初期化し、キャプチャーするスナップショットがデータベース・マネージャー・レベル情報のものであることを指定します。

   ```
   pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
   memset(pRequestedDataGroups, '¥0', SQLMASIZE(1));
   pRequestedDataGroups->obj_num = 1;
   pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
   ```

6. スナップショット・リスト出力を保持するバッファーを初期化します。

   ```
   snapshotBuffer = (char *)malloc(snapshotBufferSize);
   memset (snapshotBuffer, '¥0', snapshotBufferSize);
   ```

7. db2GetSnapshotData 構造体に、スナップショット要求タイプ (sqlma 構造体から)、バッファー情報、およびスナップショットをキャプチャーするために必要な他の情報を含めます。

   ```
   getSnapshotParam.piSqlmaData = pRequestedDataGroups;
   getSnapshotParam.poCollectedData = &collectedData;
   getSnapshotParam.poBuffer = snapshotBuffer;
   getSnapshotParam.iVersion = SQLM_DBMON_VERSION8;
   getSnapshotParam.iBufferSize = snapshotBufferSize;
   getSnapshotParam.iStoreResult = 0;
   getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
   getSnapshotParam.poOutputFormat = &outputFormat;
   getSnapshotParam.iSnapshotClass = SQLM_CLASS_DEFAULT;
   ```

8. スナップショットをキャプチャーします。 db2GetSnapshotData 構造体を渡します。これには、スナップショットをキャプチャーするのに必要な情報に加えて、スナップショット出力の宛先となるバッファーの参照も含まれています。

   ```
   db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
   ```

9. バッファーのオーバーフローを処理するためのロジックを組み込みます。スナップショットが取られた後、バッファー・オーバーフローについて sqlcode がチェックされます。バッファー・オーバーフローが発生した場合には、バッファーがクリアされて再初期化され、スナップショットが再度取られます。

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
  snapshotBufferSize = snapshotBufferSize +
  SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("¥nMemory allocation error.¥n");
    return 1;
  }
  getSnapshotParam.iBufferSize = snapshotBufferSize;
  getSnapshotParam.poBuffer = snapshotBuffer;
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. スナップショット・モニターのデータ・ストリームを処理します。
11. バッファーをクリアします。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 21 ページの『スナップショット・モニター』
- 6 ページの『システム・モニター出力 : 自己記述型データ・ストリーム』
- 46 ページの『スナップショット・モニター自己記述型データ・ストリーム』

関連タスク:

- 16 ページの『クライアント・アプリケーションからのモニター・スイッチの設定』

関連資料:

- 「*管理 API リファレンス*」の『db2GetSnapshot - スナップショットの入手』
- 「*管理 API リファレンス*」の『db2MonitorSwitches - モニター・スイッチの入手 / 更新』
- 「*管理 API リファレンス*」の『db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり』
- 「*管理 API リファレンス*」の『db2ResetMonitor - モニターのリセット』
- 39 ページの『スナップショット・モニター API 要求タイプ』

関連サンプル:

- 『clisnap.c -- Capture a snapshot at the client level (C)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C)』
- 『dbsnap.c -- Capture a snapshot at the database level (C)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C)』
- 『insnap.c -- Capture a snapshot at the instance level (C)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C)』
- 『utilsnap.c -- Utilities for the snapshot monitor samples (C)』

- 『clisnap.C -- Capture a snapshot at the client level (C++)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C++)』
- 『dbsnap.C -- Capture a snapshot at the database level (C++)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C++)』
- 『insnap.C -- Capture a snapshot at the instance level (C++)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C++)』
- 『utilsnap.C -- Utilities for the snapshot monitor samples (C++)』
- 『dbsnap.cbl -- Get a database monitor snapshot (IBM COBOL)』

スナップショット・モニター API 要求タイプ

次の表に、サポートされているスナップショット要求のタイプをすべてリストします。いくつかの要求タイプの場合、一部の情報は、関連したモニター・スイッチが ON に設定されている場合にだけ戻されます。スイッチで必須エレメントを制御できるかどうか判別するには、個々のモニター・エレメントを参照してください。

表 5. スナップショット・モニター API 要求タイプ

モニター・レベル	API 要求タイプ	戻される情報
接続リスト	SQLMA_APPLINFO_ALL	スナップショットが取られたノード上の DB2 インスタンスが管理するデータベースに現在接続されている、すべてのアプリケーションのアプリケーション識別情報。
接続リスト	SQLMA_DBASE_APPLINFO	指定のデータベースに現在接続されている各アプリケーションに関するアプリケーション識別情報。
接続リスト	SQLMA_DCS_APPLINFO_ALL	スナップショットが取られたノード上の DB2 インスタンスが管理するデータベースに現在接続されている、すべての DCS アプリケーションのアプリケーション識別情報。
データベース・マネージャー	SQLMA_DB2	インスタンス・レベルのモニター・スイッチの設定値を含む、データベース・マネージャー・レベルの情報。
データベース	SQLMA_DBASE	データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	SQLMA_DBASE_ALL	パーティション上でアクティブな各データベースのデータベース・レベル情報およびカウンター。個々のアクティブなデータベースに対する接続数。 ACTIVATE DATABASE コマンドを使用して開始したものの、接続されていないデータベースを含みます。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	SQLMA_DCS_DBASE	特定の DCS データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	SQLMA_DCS_DBASE_ALL	パーティション上でアクティブな各 DCS データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。

表 5. スナップショット・モニター API 要求タイプ (続き)

モニター・レベル	API 要求タイプ	戻される情報
データベース	SQLMA_DBASE_REMOTE	特定のフェデレーテッド・システム・データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	SQLMA_DBASE_REMOTE_ALL	パーティション上でアクティブな各フェデレーテッド・システム・データベースのデータベース・レベル情報およびカウンター。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
アプリケーション	SQLMA_APPL	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_AGENT_ID	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_DBASE_APPLS	パーティション上のデータベースに接続されている各アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_APPL_ALL	パーティション上でアクティブな各アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_DCS_APPL	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_DCS_APPL_ALL	パーティション上にあるアクティブな各 DCS アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_DCS_APPL_HANDLE	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_DCS_DBASE_APPLS	パーティション上にあるデータベースに接続されている各 DCS アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
アプリケーション	SQLMA_DBASE_APPLS_REMOTE	アプリケーション・レベルの情報。累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。

表 5. スナップショット・モニター API 要求タイプ (続き)

モニター・レベル	API 要求タイプ	戻される情報
アプリケーション	SQLMA_APPL_REMOTE_ALL	パーティション上にあるアクティブな各フェデレーテッド・システム・アプリケーションについてのアプリケーション・レベル情報。これには、累積カウンター、状況情報、および最後に実行された SQL ステートメント (ステートメント・スイッチが設定されている場合) が含まれます。
表	SQLMA_DBASE_TABLES	データベースに接続された各アプリケーションのデータベース・レベルとアプリケーション・レベルの表アクティビティー情報。データベースに接続されたアプリケーションが**アクセスした**各表の表レベルの表アクティビティー情報。表スイッチが必要です。
ロック	SQLMA_APPL_LOCKS	アプリケーションによって保持されているロックのリスト。ロック待機情報にはロック・スイッチが必要です。
ロック	SQLMA_APPL_LOCKS_AGENT_ID	アプリケーションによって保持されているロックのリスト。ロック待機情報にはロック・スイッチが必要です。
ロック	SQLMA_DBASE_LOCKS	データベースに接続された各アプリケーションについての、データベース・レベルおよびアプリケーション・レベルのロック情報。ロック・スイッチが必要です。
表スペース	SQLMA_DBASE_TABLESPACES	データベース・レベルの表スペース・アクティビティーに関する情報。また、データベースに接続された各アプリケーションのアプリケーション・レベル、およびデータベースに接続された各アプリケーションがアクセスしている各表スペースの表スペース・レベルの情報も含みます。バッファー・プール・スイッチが必要です。
バッファー・プール	SQLMA_BUFFERPOOLS_ALL	バッファー・プール・アクティビティー・カウンター。バッファー・プール・スイッチが必要です。
バッファー・プール	SQLMA_DBASE_BUFFERPOOLS	指定されたデータベースのバッファー・プール・アクティビティー・カウンター。バッファー・プール・スイッチが必要です。
動的 SQL	SQLMA_DYNAMIC_SQL	データベースの SQL ステートメント・キャッシュからのポイント・イン・タイム指定ステートメント情報。

関連概念:

- 21 ページの『スナップショット・モニター』
- 46 ページの『スナップショット・モニター自己記述型データ・ストリーム』

関連タスク:

- 16 ページの『クライアント・アプリケーションからのモニター・スイッチの設定』
- 36 ページの『クライアント・アプリケーションからのデータベース・スナップショットのキャプチャー』

関連資料:

- 「*管理 API リファレンス*」の『db2GetSnapshot - スナップショットの入手』
- 「*管理 API リファレンス*」の『db2MonitorSwitches - モニター・スイッチの入手 / 更新』
- 「*管理 API リファレンス*」の『db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり』

- 「*管理 API リファレンス*」の『db2ResetMonitor - モニターのリセット』
- 「*管理 API リファレンス*」の『db2ConvMonStream - モニター・ストリームの変換』

スナップショット・モニターの出力例

スナップショット・モニターの性質を示すため、以下に、CLP を使用して取られるスナップショットの例とそれに対応する出力を示します。この例の目的は、SAMPLE データベースに接続されたアプリケーションが保持しているロックのリストを入手することです。行われるステップは次のとおりです。

1. 次のようにしてサンプル・データベースに接続します。

   ```
   db2 connect to sample
   ```

2. 次のように、UPDATE MONITOR SWITCHES コマンドを使用して「ロック」スイッチを ON にし、ロックのために待機した時間を収集します。

   ```
   db2 update monitor switches using LOCK on
   ```

3. データベース・カタログでロックを必要とするコマンドまたはステートメントを発行します。この場合、次のようにカーソルの宣言、オープン、およびフェッチを行います。

   ```
   db2 -c- declare c1 cursor for
                         select * from staff where job='Sales' for update
   db2 -c- open c1
   db2 -c- fetch c1
   ```

4. 次のように GET SNAPSHOT コマンドを使用して、データベース・ロックのスナップショットを取ります。

   ```
   db2 get snapshot for locks on sample
   ```

CLP から GET SNAPSHOT コマンドが発行された後、スナップショット出力が画面に送られます。

```
            Database Lock Snapshot

Database name                               = SAMPLE
Database path                               = C:¥DB2¥NODE0000¥SQL00001¥
Input database alias                        = SAMPLE
Locks held                                  = 5
Applications currently connected            = 1
Agents currently waiting on locks           = 0
Snapshot timestamp                          = 06-05-2002 17:08:25.048027

Application handle                          = 8
Application ID                              = *LOCAL.DB2.0098C5210749
Sequence number                             = 0001
Application name                            = db2bp.exe
CONNECT Authorization ID                    = DB2ADMIN
Application status                          = UOW Waiting
Status change time                          = Not Collected
Application code page                       = 1252
Locks held                                  = 5
Total wait time (ms)                        = 0

List Of Locks
 Lock Name                  = 0x020003000500000000000000052
 Lock Attributes            = 0x00000000
 Release Flags              = 0x00000001
 Lock Count                 = 1
 Hold Count                 = 0
 Lock Object Name           = 5
```

```
Object Type                 = Row
Tablespace Name             = USERSPACE1
Table Schema                = DB2ADMIN
Table Name                  = STAFF
Mode                        = U

Lock Name                   = 0x02000300000000000000000054
Lock Attributes             = 0x00000000
Release Flags               = 0x00000001
Lock Count                  = 1
Hold Count                  = 0
Lock Object Name            = 3
Object Type                 = Table
Tablespace Name             = USERSPACE1
Table Schema                = DB2ADMIN
Table Name                  = STAFF
Mode                        = IX

Lock Name                   = 0x01000000010000000100810056
Lock Attributes             = 0x00000000
Release Flags               = 0x40000000
Lock Count                  = 1
Hold Count                  = 0
Lock Object Name            = 0
Object Type                 = Internal Variation Lock
Mode                        = S

Lock Name                   = 0x41414141414A48520000000041
Lock Attributes             = 0x00000000
Release Flags               = 0x40000000
Lock Count                  = 1
Hold Count                  = 0
Lock Object Name            = 0
Object Type                 = Internal Plan Lock
Mode                        = S

Lock Name                   = 0x434F4E544F4B4E310000000041
Lock Attributes             = 0x00000000
Release Flags               = 0x40000000
Lock Count                  = 1
Hold Count                  = 0
Lock Object Name            = 0
Object Type                 = Internal Plan Lock
Mode                        = S
```

このスナップショットを見ると、現在 1 つのアプリケーションが SAMPLE データベースに接続しており、 5 つのロックを保持していることが分かります。

```
Locks held                              = 5
Applications currently connected        = 1
```

Application status が UOW Waiting になった時刻 (Status change time) は Not Collected として戻されることに注意してください。これは、「作業単位」スイッチが OFF であるためです。

ロック・スナップショットは、このデータベースに接続しているアプリケーションがロックのために待機している、現在までの合計時間も戻します。

```
Total wait time (ms)                    = 0
```

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 21 ページの『スナップショット・モニター』

関連タスク:

- 13 ページの『CLP からのモニター・スイッチの設定』

関連資料:

- 34 ページの『スナップショット・モニター CLP コマンド』

サブセクション・スナップショット

パーティション間並列処理を使用しているシステムでは、SQL コンパイラーによって、 SQL ステートメントのアクセス・プランがサブセクションに区分化されます。個々のサブセクションは別々の DB2® エージェント (または SMP のエージェント) によって実行されます。

コンパイル時に DB2 コード生成プログラムによって生成される SQL ステートメントのアクセス・プランを取得するには、 db2expln コマンドか dynexpln コマンドを使用します。一例として、複数のパーティションにパーティション化されている表の行をすべて選択すると、アクセス・プランは以下の 2 つのサブセクションに分けられます。

1. サブセクション 0。コーディネーター・サブセクション。この役割は、他の DB2 エージェント (サブエージェント) に取り出された行を収集してアプリケーションに戻すことです。
2. サブセクション 1。この役割は、表スキャンを実行して、行をコーディネーター・エージェントに戻すことです。

この単純な例では、サブセクション 1 はすべてのデータベース・パーティションに分散されます。この表が属するデータベース・パーティション・グループの個々の物理パーティションに、このサブセクションを実行するサブエージェントがあります。

データベース・システム・モニターを使用すると、ランタイムの情報とアクセス・プラン (コンパイル時の情報) を相関させることができます。パーティション間並列処理により、モニターは情報をサブセクションのレベルに分類します。例えば、ステートメント・モニターのスイッチが ON になっている場合に、 GET SNAPSHOT FOR APPLICATION を実行すると、ステートメント全体の情報と、このパーティション上で実行される個々のサブセクションに関する情報が戻されます。

アプリケーションのスナップショットを実行すると、以下のサブセクション情報が戻されます。

- 読み書きされた表の行数
- CPU の使用量
- 経過時間
- このステートメントで作業する他のエージェントとの間で送受信される表キューの行数。これにより、スナップショットを連続してとることで、実行時間の長い照会の実行状況を追跡できます。
- サブセクションの状況。サブセクションが WAIT 状態 (別のエージェントがデータを送受信するのを待機している) の場合は、この情報によって、サブセクショ

ンの実行を妨げているパーティションも識別できます。識別した後にそのパーティションでスナップショットを取って状態を調べることができます。

この情報は、ステートメント・イベント・モニターによって、サブセクションごとに実行完了後に記録されます。この情報には CPU 使用量、合計実行時間、および他の複数のカウンターが含まれます。

関連概念:

- 11 ページの『システム・モニター・スイッチ』
- 21 ページの『スナップショット・モニター』
- 45 ページの『パーティション・データベース・システムでのグローバル・スナップショット』

関連資料:

- 「コマンド・リファレンス」の『GET SNAPSHOT コマンド』
- 34 ページの『スナップショット・モニター CLP コマンド』

パーティション・データベース・システムでのグローバル・スナップショット

パーティション・データベース・システムでは、現行パーティション、指定したパーティション、またはすべてのパーティションのスナップショットをとることができます。パーティション・データベースのすべてのパーティションに渡ってグローバル・スナップショットをとる場合は、結果が戻される前にデータが集約されます。

データは、以下のようなさまざまなエレメント・タイプについて集約されます。

- **カウンター、時間、ゲージ**

  インスタンス内のそれぞれのパーティションから収集されたすべての同種値の合計が入っています。例えば、GET SNAPSHOT FOR DATABASE XYZ ON TEST GLOBAL は、パーティション・データベース・インスタンス内のすべてのパーティションについて、データベースから読み取られた行数 (rows_read) を戻します。

- **水準点**

  パーティション・データベース・システム内のパーティションについて検出される最高値 (高位水準点) または最低値 (低位水準点) を戻します。戻された値に注目したい場合は、個々のパーティションのスナップショットを取って、特定のパーティションが利用不能になっているのか、インスタンス全体に問題があるのかを判別することができます。

- **タイム・スタンプ**

  スナップショット・モニター・エージェントがアタッチされているパーティションのタイム・スタンプ値に設定します。すべてのタイム・スタンプ値は、「**タイム・スタンプ**」モニター・スイッチの制御下にあります。

- **情報**

作業を妨害している可能性のあるパーティションについての最も重要な情報を戻します。例えば、エレメント appl_status について、あるパーティションでの状況が「UOW 実行」であり、別のパーティションでは「ロック待機」の場合には、「ロック待機」が戻されます。なぜなら、これはアプリケーションの実行を保留にしている状況だからです。

さらに、カウンターのリセット、モニター・スイッチの設定、パーティション・データベース内の個々のパーティションまたはすべてのパーティションのモニター・スイッチ設定値の検索も行うことができます。

注: グローバル・スナップショットをとる際に、1 つ以上のパーティションでエラーが生じると、データはスナップショットを正常にとることのできたパーティションから収集され、さらに警告 (sqlcode 1629) が戻されます。モニター・スイッチのグローバル取得または更新が失敗したり、 1 つ以上のパーティションでカウンター・リセットが失敗すると、それらのパーティションではスイッチの設定やデータのリセットは行われません。

関連概念:

- 5 ページの『カウンターの状況および可視性』
- 44 ページの『サブセクション・スナップショット』
- 21 ページの『スナップショット・モニター』

関連タスク:

- 36 ページの『クライアント・アプリケーションからのデータベース・スナップショットのキャプチャー』
- 33 ページの『CLP からのデータベース・スナップショットのキャプチャー』
- 23 ページの『データベース・システム・スナップショットへの SQL アクセス』

スナップショット・モニター自己記述型データ・ストリーム

db2GetSnapshot API でスナップショットをキャプチャーした後、スナップショット出力が自己記述型データ・ストリームとして戻されます。 47 ページの図 2 では、データ・ストリームの構造を示し、 48 ページの表 6 では、戻される論理データ・グループおよびモニター・エレメントのいくつかの例を示します。

注: これらの例や表では、ID として記述名を使用しています。これらの名前は、実際のデータ・ストリームでは **SQLM_ELM_** という接頭部が付きます。例えば collected は、スナップショット・モニター出力で SQLM_ELM_COLLECTEDと表示されます。タイプは、実際のデータ・ストリームでは **SQLM_TYPE_** という接頭部が付きます。例えば、ヘッダーはデータ・ストリームで SQLM_TYPE_HEADER と表示されます。

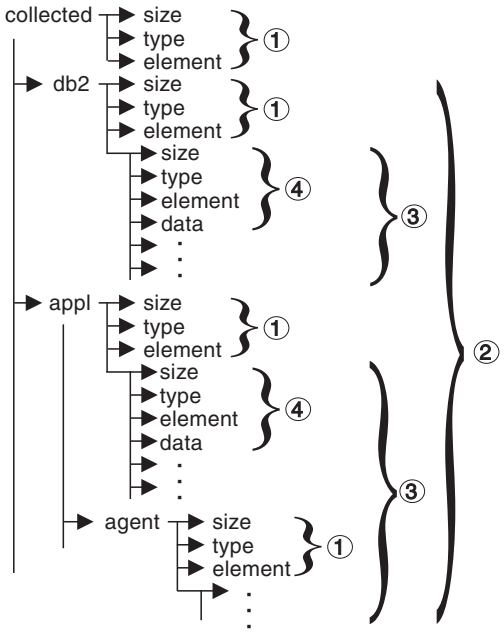

図 2. スナップショット・モニター・データ・ストリーム

1. 各論理データ・グループは、サイズと名前を示すヘッダーで始まります。このサイズには、ヘッダー自体に取られるデータ・ボリュームは組み込まれません。
2. collected ヘッダー内のサイズは、スナップショットの合計サイズを戻します。
3. ほかのヘッダー内のサイズ・エレメントは、その論理データ・グループのデータ全体のサイズを示します (従属のグループをすべて含む)。
4. モニター・エレメント情報が、論理データ・グループ・ヘッダーに続きます。これも自己記述型です。

表 6. スナップショット・データ・ストリームの例

論理データ・グループ	データ・ストリーム	説明
collected	1000	スナップショット・データのサイズ (バイト)。
	header	論理データ・グループが始まることを示す。
	collected	論理データ・グループの名前。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 符号なし 32 ビット数値。
	server_db2_type	収集されたモニター・エレメントの名前。
	sqlf_nt_server	このエレメントに対して収集された値。
	2	このモニター・エレメントに入っているデータのサイズ。
	u16bit	モニター・エレメント・タイプ - 符号なし 16 ビット数値。
	node_number	収集されたモニター・エレメントの名前。
	3	このエレメントに対して収集された値。
db2	200	スナップショットのデータの DB2® レベル部分のサイズ。
	header	論理データ・グループが始まることを示す。
	db2	論理データ・グループの名前。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 符号なし 32 ビット数値。
	sort_heap_allocated	収集されたモニター・エレメントの名前。
	16	このエレメントに対して収集された値。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 符号なし 32 ビット数値。
	local_cons	収集されたモニター・エレメントの名前。
	3	このエレメントに対して収集された値。
	. . .	. . .
appl	100	スナップショットの appl エレメント・データのサイズ。
	header	論理データ・グループが始まることを示す。
	appl	論理データ・グループの名前。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 符号なし 32 ビット数値。
	locks_held	収集されたモニター・エレメントの名前。
	3	このエレメントに対して収集された値。
	. . .	. . .
agent	50	appl 構造のエージェント部分のサイズ。
	header	論理データ・グループが始まることを示す。
	agent	論理データ・グループの名前。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 32 ビット数値。
	agent_pid	収集されたモニター・エレメントの名前。
	12	このエレメントに対して収集された値。
	. . .	. . .

db2GetSnapshot() ルーチンは、自己記述型スナップショット・データを、ユーザー提供のバッファーに戻します。データは、キャプチャーされたスナップショットのタイプに関連する論理データ・グループに分けられて戻されます。

スナップショット要求によって戻される各アイテムには、そのサイズおよびタイプを指定するフィールドが含まれます。サイズを使用して、戻りデータ全体を解析できます。また、フィールド・サイズを使用して、論理データ・グループを読み飛ばすこともできます。例えば、DB2 レコードを読み飛ばすには、データ・ストリーム内のバイト数を判別する必要があります。読み飛ばすバイト数を計算するには、次の公式を使用してください。

```
db2 論理データ・グループのサイズ + sizeof(sqlm_header_info)
```

関連概念:

- 21 ページの『スナップショット・モニター』

関連タスク:

- 36 ページの『クライアント・アプリケーションからのデータベース・スナップショットのキャプチャー』

関連資料:

- 39 ページの『スナップショット・モニター API 要求タイプ』
- 93 ページの『スナップショット・モニター・インターフェースの論理データ・グループへのマッピング』

関連サンプル:

- 『clisnap.c -- Capture a snapshot at the client level (C)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C)』
- 『dbsnap.c -- Capture a snapshot at the database level (C)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C)』
- 『insnap.c -- Capture a snapshot at the instance level (C)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C)』
- 『utilsnap.c -- Utilities for the snapshot monitor samples (C)』
- 『clisnap.C -- Capture a snapshot at the client level (C++)』
- 『clisnap.out -- HOW TO GET A CLIENT LEVEL SNAPSHOT (C++)』
- 『dbsnap.C -- Capture a snapshot at the database level (C++)』
- 『dbsnap.out -- HOW TO GET A DATABASE LEVEL SNAPSHOT (C++)』
- 『insnap.C -- Capture a snapshot at the instance level (C++)』
- 『insnap.out -- HOW TO GET AN INSTANCE LEVEL SNAPSHOT (C++)』
- 『utilsnap.C -- Utilities for the snapshot monitor samples (C++)』

第 4 章 イベント・モニターの使用法

イベント・モニター

イベント・モニターを使用して、指定されたイベントの発生時に、データベースおよび接続されたアプリケーションに関する情報を収集します。イベントは、接続、デッドロック、ステートメント、トランザクションなどの、データベース・アクティビティーの遷移を表します。モニターしたいイベント (1 つ以上) のタイプごとにイベント・モニターを定義することができます。例えば、デッドロック・イベント・モニターは、デッドロックが発生するのを待機します。発生すると、関係するアプリケーションおよび競合するロックに関する情報を収集します。

注: デフォルトでは、すべてのデータベースについて DB2DETAILDEADLOCK という名前のイベント・モニターが定義されています。このモニターは DEADLOCKS WITH DETAILS を追跡します。 DB2DETAILDEADLOCK イベント・モニターは、データベースの開始時に自動的に開始されます。

スナップショット・モニターは一般に、予防的な保守および問題分析のために使用されますが、イベント・モニターは、現時点の問題について管理者に警告し、また今にも起こりそうな問題を追跡するするために使用されます。

イベント・モニターを作成するには、CREATE EVENT MONITOR SQL ステートメントを使用します。イベント・モニターは、それらがアクティブなときにだけイベント・データを収集します。イベント・モニターを活動化または非活動化するには、 SET EVENT MONITOR STATE SQL ステートメントを使用します。イベント・モニターの状況 (アクティブか非アクティブか) は、 SQL 関数 EVENT_MON_STATE によって判別することができます。

CREATE EVENT MONITOR SQL ステートメントを実行すると、それが作成するイベント・モニターの定義が、以下のデータベース・システム・カタログ表に保管されます。

- SYSCAT.EVENTMONITORS: データベースについて定義されたイベント・モニター
- SYSCAT.EVENTS: データベースについてモニターされるイベント
- SYSCAT.EVENTTABLES: 表イベント・モニターのためのターゲット表

それぞれのイベント・モニターには、モニター・エレメント内のインスタンスのデータの、独自の専用論理ビューがあります。特定のイベント・モニターが非活動化された後、再活動化されると、これらのカウンターのビューがリセットされます。リセットは、新たに活動化されたイベント・モニターだけで行われます。他のすべてのイベント・モニターは、引き続きカウンター値の独自のビューを使用し続けます (追加があればそのカウンター値に追加します)。

イベント・モニターの出力は、SQL 表、ファイル、または名前付きパイプに送ることができます。

関連概念:

- 3 ページの『データベース・システム・モニター』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』
- 55 ページの『イベント・モニターの作成』

関連資料:

- 75 ページの『イベント・モニターの出力例』
- 52 ページの『イベント・タイプ』

イベント・タイプ

イベント・モニターは、 CREATE EVENT MONITOR ステートメントで指定されたイベント・タイプに関する情報を戻します。それぞれのイベント・タイプごとに、特定の時点でモニター情報が収集されます。以下の表は、使用可能なイベント・タイプ、モニター・データが収集される時点、およびそれぞれのイベント・タイプについて入手できる情報をリストしています。最初の列にある使用可能なイベント・タイプは、 CREATE EVENT MONITOR ステートメントで使用されるキーワードに対応しています。ここでイベント・タイプが定義されます。

データが発生するように定義されたイベントに加えて、 FLUSH EVENT MONITOR SQL ステートメントを使用してイベントを生成することもできます。この方式によって生成されたイベントは、フラッシュされたイベント・モニターに関連したすべてのモニター・タイプ (DEADLOCKS および DEADLOCKS WITH DETAILS を除く) に関する現行のデータベース・モニター値とともに書き込まれます。

表 7. イベント・タイプ

イベント・タイプ	データが収集される時点	入手できる情報
DEADLOCKS	デッドロック検出時	関係しているアプリケーション、および競合しているロック。
DEADLOCKS WITH DETAILS	デッドロック検出時	関係するアプリケーションについての広範囲の情報。関係するステートメント (およびステートメント・テキスト) の識別や保持されているロックなど。 DEADLOCKS イベント・モニターではなく DEADLOCKS WITH DETAILS イベント・モニターを使用すると追加の情報が収集されるため、デッドロックが発生したときにパフォーマンス・コストの負担になります。
STATEMENTS	SQL ステートメント終了時	ステートメントの開始/停止時刻、使用されている CPU、動的 SQL のテキスト、 SQLCA (SQL ステートメントの戻りコード)、 およびその他のメトリック (取り出しカウントなど)。 **注:** ステートメントの開始/停止時刻は、「タイム・スタンプ」スイッチが OFF のときは使用できません。
	サブセクション終了時	パーティション・データベースの場合、使用されている CPU、実行時間、表、および表キューに関する情報。
TRANSACTIONS	作業単位の終了時	作業単位の作業開始/停止時刻、直前の UOW 時刻、使用されている CPU、ロッキング、およびロギングのメトリック。 XA で実行している場合は、トランザクション・レコードは生成されません。
CONNECTIONS	接続終了時	すべてのアプリケーション・レベルのカウンター。
DATABASE	データベース非活動化時	すべてのデータベース・レベルのカウンター。

表 7. イベント・タイプ (続き)

イベント・タイプ	データが収集される時点	入手できる情報
BUFFERPOOLS	データベース非活動化時	バッファー・プールのカウンター、プリフェッチ機能、ページ・クリーナー、および個々のバッファー・プールの直接 I/O。
TABLESPACES	データベース非活動化時	バッファー・プールのカウンター、プリフェッチ機能、ページ・クリーナー、および個々の表スペースの直接 I/O。
TABLES	データベース非活動化時	個々の表の、読み取り/書き込みが行われる行。

注: 「デッドロックの詳細」イベント・モニターが、新規に作成されるデータベースごとに作成されます。このイベント・モニターは DB2DETAILDEADLOCK という名前で、データベースが活動化されると開始され、データベース・ディレクトリー内のファイルに書き込みます。このイベント・モニターによるオーバーヘッドは、これをドロップすることによって回避することができます。

関連概念:

- 51 ページの『イベント・モニター』
- 5 ページの『カウンターの状況および可視性』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』
- 55 ページの『イベント・モニターの作成』

関連資料:

- 「*SQL リファレンス 第 2 巻*」の『DROP ステートメント』
- 75 ページの『イベント・モニターの出力例』

データベース・システム・イベントからの情報の収集

イベント・モニターはデータベース・オブジェクトであり、 SQL データ定義言語 (SQL DDL) ステートメントを使用して作成され、操作されます。以下にリストするステップは、イベント・モニターの典型的なライフ・サイクルを表しています。これらのステップは、必ずしも示されている順序で実行する必要があるとは限りません。例えば、使用法に応じて、イベント・モニターがドロップされなかったり、非活動化されない場合もあります。

しかし、イベント・モニターのライフ・サイクルでは、以下の 2 つの事柄は必ず行われます。

1. 最初のステップは常にイベント・モニターの作成です。
2. 最後のステップは常にイベント・モニターの削除です。

前提条件:

イベント・モニターを作成し、操作するには、DBADM 権限が必要です。

手順:

1. イベント・モニターを作成します。

2. ファイルおよびパイプ・イベント・モニターの場合のみ: イベント・レコードを受け取るディレクトリーまたは名前付きパイプが存在することを確認します。存在しない場合は、イベント・モニターは活動化されません。

   AIX では、mkfifo コマンドを使用して名前付きパイプを作成することができます。 Linux および他の UNIX タイプ (Solaris オペレーティング環境など) では、pipe() ルーチンを使用します。

   Windows NT または Windows 2000 では、CreateNamedPipe() ルーチンを使用して名前付きパイプを作成することができます。
3. パイプ・イベント・モニターの場合のみ: イベント・モニターを活動化する前に名前付きパイプを開きます。これは、次のオペレーティング・システムの機能を使って行うことができます。
   - UNIX の場合: `open()`
   - Windows NT または Windows 2000 の場合: `ConnectNamedPipe()`

   また、次のように db2evmon 実行可能プログラムを使って行うこともできます。

   ```
   db2evmon -db databasename
            -evm eventmonname
   ```

   `databasename` は、モニター対象のデータベースの名前を表します。

   `evmonname` は、イベント・モニターの名前を表します。
4. 新しく作成したイベント・モニターを活動化して、それが情報を収集できるようにします。

   ```
   SET EVENT MONITOR evmonname STATE 1;
   ```

   イベント・モニターを開始すると、イベント・モニターによって SYSCAT.EVENTMONITORS カタログ表の evmon_activates 列が更新されます。この変更はログに記録されるため、DATABASE CONFIGURATION によって次が表示されます。

   ```
   Database is consistent                                  = NO
   ```

   イベント・モニターが AUTOSTART オプションを使用して作成されている場合、最初のユーザーがデータベースに接続してデータベースを非活動化するために即時に切断すると、ログ・ファイルが作成されます。
5. イベント・モニターがアクティブか非アクティブかを調べるために、次のように表 SYSCAT.EVENTMONITORS に対する照会で SQL 関数 EVENT_MON_STATE を発行します。

   ```
   SELECT evmonname, EVENT_MON_STATE(evmonname) FROM
          syscat.eventmonitors;
   ```

   存在するすべてのイベント・モニターのリストとそれらの状況が表示されます。戻り値 0 は、指定されたイベント・モニターが非アクティブであり、 1 は、アクティブであることを示します。
6. イベント・モニター出力を読み取ります。表書き込みイベント・モニターの場合はターゲット表で確認する必要があります。 CLP からファイルまたはパイプ・

イベント・モニターのデータにアクセスするには、関連タスク『コマンド行からのファイルまたはパイプ・イベント・モニター出力のフォーマット』を参照してください。

7. イベント・モニターを非活動化する、つまりイベント・モニターを OFF にするには、次のように SET EVENT MONITOR ステートメントを使用します。

   ```
   SET EVENT MONITOR evmonname STATE 0
   ```

   イベント・モニターを非活動化しても、それが削除されることはありません。イベント・モニターは休止データベース・オブジェクトとして存在します。イベント・モニターを非活動化すると、その内容がすべてフラッシュされます。そのため、非活動化されているイベント・モニターを再活動化する場合には、再活動化以降に収集された情報だけが含まれます。

   パイプ・イベント・モニターを非活動化した後、対応する名前付きパイプを閉じます。 UNIX では `close()` 関数を使用し、Windows NT または Windows 2000 では `DisconnectNamedPipe()` 関数を使用します。

8. イベント・モニター・オブジェクトをドロップするには、次のように DROP EVENT MONITOR ステートメントを使用します。

   ```
   DROP EVENT MONITOR evmonname
   ```

   イベント・モニターが非アクティブの場合にのみ、それをドロップできます。

   パイプ・イベント・モニターをドロップした後、対応する名前付きパイプを削除します。 UNIX では `unlink()` 関数を使用し、Windows NT または Windows 2000 では `CloseHandle()` 関数を使用します。

   表書き込みイベント・モニターをドロップするときに、関連したターゲット表はドロップされません。同様に、ファイル・イベント・モニターをドロップするときに、関連したファイルは削除されません。

関連概念:

- 51 ページの『イベント・モニター』
- 85 ページの『イベント・レコードとそれに対応するアプリケーション』

関連タスク:

- 55 ページの『イベント・モニターの作成』
- 71 ページの『パーティション・データベース用のイベント・モニターの作成』
- 74 ページの『コマンド行からのファイルまたはパイプ・イベント・モニター出力のフォーマット』

関連資料:

- 75 ページの『イベント・モニターの出力例』

イベント・モニターの作成

イベント・モニターのライフ・サイクルにおける最初のステップは、イベント・モニターの作成です。イベント・モニターを作成する前に、イベント・レコードの宛先を決定する必要があります。それは、SQL 表、ファイル、または名前付きパイプを介して送信します。それぞれのイベント・レコードの宛先ごとに、 CREATE

EVENT MONITOR SQL ステートメントで指定される特定のオプションがあります。パーティション・データベースでのイベントのモニターでは特別な注意も必要です。

前提条件:

イベント・モニターを作成するには、DBADM 権限が必要です。

手順:

- 表イベント・モニターを作成する。
- ファイル・イベント・モニターを作成する。
- パイプ・イベント・モニターを作成する。
- パーティション・データベース用のイベント・モニターを作成する。

イベント・モニターが作成されて活動化されると、指定されたイベントが発生するたびに、モニター・データを記録します。

関連概念:

- 66 ページの『イベント・モニターのファイル管理』
- 70 ページの『イベント・モニターの名前付きパイプ管理』
- 51 ページの『イベント・モニター』
- 60 ページの『イベント・モニターの表管理』

関連タスク:

- 56 ページの『表イベント・モニターの作成』
- 63 ページの『ファイル・イベント・モニターの作成』
- 69 ページの『パイプ・イベント・モニターの作成』
- 71 ページの『パーティション・データベース用のイベント・モニターの作成』

関連資料:

- 「*SQL リファレンス 第 2 巻*」の『CREATE EVENT MONITOR ステートメント』
- 75 ページの『イベント・モニターの出力例』
- 52 ページの『イベント・タイプ』

表イベント・モニターの作成

イベント・モニターの作成時に、それが収集した情報の保管場所を決定する必要があります。表イベント・モニターは SQL 表にイベント・レコードを流します。これは、ファイルおよびパイプ・イベント・モニターの簡単な代替手段で、イベント・モニター・データのキャプチャー、解析、および管理を、より簡単に行えるようにします。イベント・モニターが収集するすべてのイベント・タイプで、関連するそれぞれの論理データ・グループについてのターゲット表が作成されます。

表イベント・モニターの各種オプションは、CREATE EVENT MONITOR ステートメントで設定されます。表書き込みイベント・モニター用の CREATE EVENT MONITOR SQL ステートメントの生成をより簡単に行うために、 db2evtbl コマン

ドを使用することができます。イベント・モニターの名前および目的のイベント・タイプ (1 つ以上) を指定するだけで、 CREATE EVENT MONITOR ステートメントが生成され、すべてのターゲット表のリストが完成します。次に、生成されたステートメントをコピーして、変更を加えれば、 CLP からステートメントを実行することができます。

前提条件:

表イベント・モニターを作成するには、DBADM 権限が必要です。

手順:

1. イベント・モニター・データが表 (または表のセット) に収集されることを指定します。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO TABLE
   ```

   dlmon はイベント・モニターの名前です。
2. モニター対象のイベントのタイプを指定します。単一イベント・モニターで、1 つ以上のイベント・タイプをモニターすることができます。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE
   ```

   このイベント・モニターは、CONNECTIONS および DEADLOCKS WITH DETAILS イベント・タイプをモニターします。上記のステートメントがユーザー 'riihi' によって発行され、ターゲット表の派生名および表スペースが以下のようになっているとします。

   - riihi.connheader_dlmon
   - riihi.conn_dlmon
   - riihi.deadlock_dlmon
   - riihi.dlconn_dlmon
   - riihi.dllock_dlmon
   - riihi.control_dlmon
3. BUFFERSIZE 値を調整することによって、表イベント・モニター・バッファーのサイズを指定します (4K ページ単位)。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO TABLE BUFFERSIZE 8
   ```

   8 は、2 つのイベント表バッファーを合わせた容量です (4K ページ単位)。それぞれのバッファーが 16K で、合計 32K のバッファー・スペースになります。

   BUFFERSIZE のデフォルト (および最小) 値は 4 ページです。パフォーマンス上の理由で、アクティブ率の高いイベント・モニターには、アクティブ率が比較的低いものよりも大きなバッファーが必要です。
4. ユーザーをブロック化または非ブロック化する必要があるかどうかを指定します。ブロック化イベント・モニターの場合、イベントを生成する各エージェントは、イベント・バッファーがいっぱいであれば、それが表に書き込まれるまで待機します。これは、データベースのパフォーマンスを低下させることがあります。なぜなら、バッファーがクリアされるまで、延期されたエージェントおよび

従属エージェントが実行できなくなるからです。イベント・データが脱落しないようにするには、BLOCKED 文節を使用します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 BLOCKED
```

イベント・モニターはデフォルトではブロック化されます。

すべての単一イベント・レコードを収集することよりも、データベース・パフォーマンスのほうが重要な場合には、非ブロック化イベント・モニターを使用してください。その場合、イベントを生成する各エージェントは、イベント・バッファーがいっぱいのときに、それが表に書き込まれるのを待機しません。結果として、非ブロック化イベント・モニターは、アクティブ率の高いシステムではデータの脱落という影響を受けます。イベント・モニターによって生じるパフォーマンス上のオーバーヘッドを最小限に抑えるには、NONBLOCKED 文節を使用します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
```

5. イベント・レコードの収集元の論理データ・グループを指定します。イベント・モニターはそれぞれの論理データ・グループからのデータを、対応する表に保管します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE CONN, DLCONN, DLLOCK
                    BUFFERSIZE 8 NONBLOCKED
```

CONN、DLCONN、および DLLOCK 論理データ・グループが選択されます。その他の選択可能な論理データ・グループ CONNHEADER、DEADLOCK、または CONTROL については言及されていません。 CONNHEADER、DEADLOCK、または CONTROL に関係するデータは、 dlmon イベント・モニターには保管されません。

6. データを収集する必要のあるモニター・エレメントを指定します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
             WRITE TO TABLE CONN,
             DLCONN (EXCLUDES(agent_id, lock_wait_start_time)),
             DLLOCK (INCLUDES(lock_mode, table_name))
             BUFFERSIZE 8 NONBLOCKED
```

CONN のすべてのモニター・エレメントがキャプチャーされます (これがデフォルトの動作です)。 DLCONN の場合は、 agent_id および lock_wait_start_time 以外のすべてのモニター・エレメントがキャプチャーされます。最後に、 DLLOCK の場合は、モニター・エレメント lock_mode、 table_name だけがキャプチャーされます。

7. 作成される表の名前を指定し、表スペースを指定します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
             WRITE TO TABLE CONN,
             DLCONN (TABLE mydept.dlconnections,
                     EXCLUDES(agent_id, lock_wait_start_time)),
             DLLOCK (TABLE dllocks, IN mytablespace,
                     INCLUDES(lock_mode, table_name))
             BUFFERSIZE 8 NONBLOCKED
```

上記のステートメントがユーザー riihi によって発行され、ターゲット表の派生名および表スペースが以下のようになっているとします。

- CONN: riihi.conn_dlmon (デフォルトの表スペースで)
- DLCONN: mydept.dlconnections (デフォルトの表スペースで)
- DLLOCK: riihi.dllocks (MYTABLESPACE 表スペースで)

デフォルトの表スペースは、イベント・モニター定義プログラムに USE 特権があれば、 IBMDEFAULTGROUP から割り当てられます。定義プログラムがこの表スペースに対する USE 特権を有していない場合には、定義プログラムが USE 特権を有する表スペースが割り当てられます。

8. 表スペースがどの程度いっぱいになれば、イベント・モニターが自動的に非活動化するのかを指定します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE DLCONN PCTDEACTIVATE 90
                    BUFFERSIZE 8 NONBLOCKED
```

表スペースが 90% の容量に達すれば、dlmon イベント・モニターが自動的にシャットオフします。 DMS 表スペースには PCTDEACTIVATE 文節のみを使用できます。

9. データベースを開始するたびに、イベント・モニターが自動的に活動化されるかどうかを指定します。デフォルトでは、データベースの開始時にイベント・モニターは自動的には活動化されません。
   - データベースの開始時に自動的に開始するイベント・モニターを作成するには、次のステートメントを発行する。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                    AUTOSTART NONBLOCKED
```

   - データベースの開始時に自動的に開始しないイベント・モニターを作成するには、次のステートメントを発行する。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
                    MANUALSTART
```

   イベント・モニターを活動化または非活動化するには、SET EVENT MONITOR STATE ステートメントを使用します。

表イベント・モニターが作成されて活動化されると、指定されたイベントが発生するたびに、モニター・データを記録します。

関連概念:

- 51 ページの『イベント・モニター』
- 60 ページの『イベント・モニターの表管理』
- 85 ページの『イベント・レコードとそれに対応するアプリケーション』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』

関連資料:

- 「*SQL リファレンス 第 2 巻*」の『CREATE EVENT MONITOR ステートメント』

- 52 ページの『イベント・タイプ』

- 「コマンド・リファレンス」の『db2evtbl - イベント・モニターのターゲット表定義の生成コマンド』

イベント・モニターの表管理

SQL 表にイベント・レコードを保管するように、イベント・モニターを定義することができます。これを行うには、CREATE EVENT MONITOR ステートメントを WRITE TO TABLE 文節を付けて使用します。

表書き込みイベント・モニターを作成すると、データベースは、データを戻す論理データ・グループのそれぞれについて、レコードを保管するためのターゲット表 を作成します。デフォルトでは、データベースはイベント・モニターの作成プログラムのスキーマで表を作成し、それに対応する論理データ・グループおよびイベント・モニター名に従って表に名前を付けます。それぞれの表において、列名はそれが表すモニター・エレメント名と一致しています。

例えば、ユーザー riihi が STATEMENTS イベントをキャプチャーするイベント・モニターを作成しているとします。

```
CREATE EVENT MONITOR foo FOR STATEMENTS WRITE TO TABLE
```

STATEMENTS イベント・タイプを使用するイベント・モニターは、event_connheader、event_stmt、および event_subsection 論理データ・グループからデータを収集します。データベースは以下の表を作成しました。

- riihi.connheader_foo
- riihi.stmt_foo
- riihi.subsection_foo
- riihi.control_foo

個々のイベント・タイプに固有の論理データ・グループを表す表に加えて、すべての表書き込みイベント・モニターについてコントロール表が作成されます。この表は上記の riihi.control_foo のことです。コントロール表には、イベント・モニター・メタデータ、特に event_start、event_db_header (conn_time モニター・エレメントのみ)、および event_overflow 論理データ・グループからのメタデータが含まれています。

ターゲット表の各列名は、イベント・モニターのエレメント ID と一致しています。対応するターゲット表の列がないイベント・モニター・エレメントは無視されます。

表書き込みイベント・モニターのターゲット表は、手動で整理する必要があります。アクティブ量の非常に多いシステムでは、イベント・モニターが大容量のデータを記録するため、マシン・スペースをすぐに使い尽くしてしまうことがあります。ファイルや名前付きパイプに書き込むイベント・モニターとは異なり、表書き込みイベント・モニターは、特定の論理データ・グループまたはモニター・エレメントだけを記録するように定義することができます。この機能により、ユーザーの目的にかなうデータだけを収集することができ、イベント・モニターによって生成されるデータのボリュームを削減することができます。例えば、次のステートメン

トは、 event_xact 論理データ・グループだけから TRANSACTIONS イベントをキャプチャーし、 lock_escal モニター・エレメントだけを含めるように、イベント・モニターを定義します。

```
CREATE EVENT MONITOR foo_lite FOR TRANSACTIONS WRITE TO TABLE
                     XACT(INCLUDES(lock_escal))
```

イベント・モニターのターゲット表をデフォルトの表スペース内のデフォルトのスキーマに、デフォルトの表名で常駐させるのが望ましくない場合があります。例えば、モニター・データのボリュームが大きくなることが予想されるために、ターゲット表を独自の表スペースに存在させたい場合があります。

スキーマ、表、および表スペース名は CREATE EVENT MONITOR ステートメントで指定することができます。スキーマ名は表名とともに提供され、表の派生名を形成します。ターゲット表を使用できるのは、単一イベント・モニターに限られます。ターゲット表がすでに別のイベント・モニターについて定義されているのが検出されたか、または他の理由で作成できない場合には、CREATE EVENT MONITOR ステートメントは失敗します。表スペース名は表名の後に、オプションの IN 文節を付けて追加することができます。 DB2® が作成するターゲット表とは異なり、表スペースがイベント・モニター定義に組み込まれている場合には、それが存在していなければなりません。デフォルトの表スペースは、イベント・モニター定義プログラムに USE 特権があれば、 IBMDEFAULTGROUP から割り当てられます。定義プログラムがこの表スペースに対する USE 特権を有していない場合には、定義プログラムが USE 特権を有する表スペースが割り当てられます。

イベント・モニター・データの検索時のパフォーマンスを向上させるために、イベント表に索引を作成することができます。また、トリガー、関係保全、制約など、他の表属性を追加することもできます。イベント・モニターはそれらを無視します。

例えば、次のステートメントは、event_connheader、event_stmt、および event_subsection 論理データ・グループを使用して、 STATEMENTS イベントをキャプチャーするイベント・モニターを定義します。 3 つのターゲット表のそれぞれには、異なるスキーマ、表、および表スペースの組み合わせがあります。

```
CREATE EVENT MONITOR foo FOR STATEMENTS
             WRITE TO TABLE CONNHEADER,
             STMT (TABLE mydept.statements),
             SUBSECTION (TABLE subsections IN mytablespace)
```

上記のステートメントがユーザー 'riihi' によって発行され、ターゲット表の派生名および表スペースが以下のようになっているとします。

- CONNHEADER: riihi.connheader_foo (デフォルトの表スペースで)
- STMT: mydept.statements (デフォルトの表スペースで)
- SUBSECTION: riihi.subsections (MYTABLESPACE 表スペースで)

イベント・モニターを活動化しているときにターゲット表が存在しない場合には、活動化が続行し、ターゲット表が存在する場合にそこに挿入されるはずのデータは無視されます。また、ターゲット表にモニター・エレメント専用の列がない場合には、そのモニター・エレメントは無視されます。

表書き込みイベント・モニターがアクティブな場合、イベント・レコードを保管する表スペースがその限界に達する可能性があります。 DMS 表スペースについてこ

のリスクを制御するために、イベント・モニターが非活動化されるときの表スペース容量のパーセンテージを定義することができます。これは、CREATE EVENT MONITOR ステートメントの PCTDEACTIVATE 文節で宣言することができます。

非パーティションのデータベース環境では、最後のアプリケーションが終了すると(それまでにデータベースが明示的に活動化されていないと)、表イベント・モニターへの書き込みはすべて非活動化されます。パーティション・データベース環境では、カタログ・パーティションが非活動化されると表イベント・モニターへの書き込みが非活動化されます。

次の表は、ターゲット表が戻されるイベント・タイプごとの、デフォルトのターゲット表名を示しています。

表 8. 表書き込みイベント・モニターのターゲット表

イベント・タイプ	ターゲット表名	入手できる情報
DEADLOCKS	CONNHEADER	接続メタデータ
	DEADLOCK	デッドロック・データ
	DLCONN	デッドロックに関係しているアプリケーションとロック
	CONTROL	イベント・モニター・メタデータ
DEADLOCKS WITH DETAILS	CONNHEADER	接続メタデータ
	DEADLOCK	デッドロック・データ
	DLCONN	デッドロックに関係しているアプリケーション
	DLLOCK	デッドロックに関係しているロック
	CONTROL	イベント・モニター・メタデータ
STATEMENTS	CONNHEADER	接続メタデータ
	STMT	ステートメント・データ
	SUBSECTION	サブセクションに固有のステートメント・データ
	CONTROL	イベント・モニター・メタデータ
TRANSACTIONS	CONNHEADER	接続メタデータ
	XACT	トランザクション・データ
	CONTROL	イベント・モニター・メタデータ
CONNECTIONS	CONNHEADER	接続メタデータ
	CONN	接続データ
	CONTROL	イベント・モニター・メタデータ
	CONMEMUSE	メモリー・プール・メタデータ
DATABASE	DB	データベース・マネージャー・データ
	CONTROL	イベント・モニター・メタデータ
	DBMEMUSE	メモリー・プール・メタデータ
BUFFERPOOLS	BUFFERPOOL	バッファー・プール・データ
	CONTROL	イベント・モニター・メタデータ
TABLESPACES	TABLESPACE	表スペース・データ
	CONTROL	イベント・モニター・メタデータ
TABLES	TABLE	表データ
	CONTROL	イベント・モニター・メタデータ

次の論理データ・グループは、表書き込みイベント・モニターでは収集されません。

- event_log_stream_header
- event_log_header
- event_dbheader (conn_time モニター・エレメントだけが収集される)

イベント・モニター表の各列のデータ・タイプは、列によって表されるモニター・エレメントのデータ・タイプに対応します。以下は、モニター・エレメントの元のシステム・モニター・データ・タイプ (sqlmon.h にある) を、表列の SQL データ・タイプに対応させた、データ・タイプのマッピングの集合です。

表 9. システム・モニター・データ・タイプのマッピング

システム・モニターのデータ・タイプ	SQL データ・タイプ
SQLM_TYPE_STRING	CHAR[n]、VARCHAR[n]、CLOB[n]
SQLM_TYPE_U8BIT および SQLM_TYPE_8BIT	SMALLINT、INTEGER、または BIGINT
SQLM_TYPE_U16BIT および SQLM_TYPE_16BIT	SMALLINT、INTEGER、または BIGINT
SQLM_TYPE_U32BIT および SQLM_TYPE_32BIT	INTEGER または BIGINT
SQLM_TYPE_U64BIT および SQLM_TYPE_64BIT	BIGINT
SQLM_TIMESTAMP	TIMESTAMP
SQLM_TIME	BIGINT
SQLCA: SQLERRMC	VARCHAR[72]
SQLCA: SQLSTATE	CHAR[5]
SQLCA: SQLWARN	CHAR[11]
SQLCA: 他のフィールド	INTEGER または BIGINT

注:

1. すべて列は NOT NULL です。
2. CLOB 列がある表のパフォーマンスは VARCHAR 列がある表より劣るため、stmt evmGroup (またはデッドロックの詳細を使用するときに evmGroup) を指定するときは、 TRUNC キーワードを使用することを検討してください。

関連概念:

- 51 ページの『イベント・モニター』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』
- 68 ページの『表書き込みイベント・モニターおよびファイル・イベント・モニターのバッファー方式』

関連タスク:

- 56 ページの『表イベント・モニターの作成』

関連資料:

- 408 ページの『event_monitor_name イベント・モニター名』

ファイル・イベント・モニターの作成

イベント・モニターの作成時に、それが収集した情報の保管場所を決定する必要があります。ファイル・イベント・モニターは、 8 桁の番号の付いた、拡張子 ″evt″ のファイル (例えば 00000000.evt、 00000001.evt、00000002.evt) にイベント・レコードを流します。データを小さく分割した場合でも、全体を 1 つの論理ファイルと

見なす必要があります (つまり、データ・ストリームの開始はファイル 00000000.evt の最初のバイトであり、データ・ストリームの終了はファイル nnnnnnnn.evt 内の最後のバイトです)。イベント・モニターでは、 1 つのイベント・レコードが 2 つのファイルにわたって入ることはありません。

ファイル・イベント・モニターとそのオプションは、 CREATE EVENT MONITOR ステートメントによって定義されます。

前提条件:

ファイル・イベント・モニターを作成するには、DBADM 権限が必要です。

手順:

1. イベント・モニター・データがファイル (またはファイルのセット) に収集されることを指定し、イベント・ファイルが保管されるディレクトリーの場所を指定します。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO FILE '/tmp/dlevents'
   ```

   dlmon はイベント・モニターの名前です。

   /tmp/dlevents は、イベント・モニターがイベント・ファイルを書き込むディレクトリー・パス (UNIX 上) の名前です。

2. モニター対象のイベントのタイプを指定します。単一イベント・モニターで、1 つ以上のイベント・タイプをモニターすることができます。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents'
   ```

   このイベント・モニターは、CONNECTIONS および DEADLOCKS WITH DETAILS イベント・タイプをモニターします。

3. BUFFERSIZE 値を調整することによって、ファイル・イベント・モニター・バッファーのサイズを指定します (4K ページ単位)。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
   ```

   8 は、2 つのイベント・ファイル・バッファーの容量です (4K ページ単位)。

   BUFFERSIZE のデフォルト (および最小) 値は 4 です。パフォーマンス上の理由で、アクティブ率の高いイベント・モニターには、アクティブ率が比較的低いものよりも大きなバッファーが必要です。

4. ユーザーをブロック化または非ブロック化する必要があるかどうかを指定します。ブロック化イベント・モニターの場合、イベントを生成する各エージェントは、イベント・バッファーがいっぱいであれば、それがファイルに書き込まれるまで待機します。これは、データベースのパフォーマンスを低下させることがあります。なぜなら、バッファーがクリアされるまで、延期されたエージェントおよび従属エージェントが実行できなくなるからです。イベント・データが脱落しないようにするには、BLOCKED 文節を使用します。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       BLOCKED
   ```

イベント・モニターはデフォルトではブロック化されます。

すべての単一イベント・レコードを収集することよりも、データベース・パフォーマンスのほうが重要な場合には、非ブロック化イベント・モニターを使用してください。その場合、イベントを生成する各エージェントは、イベント・バッファーがいっぱいのときに、それがファイルに書き込まれるのを待機しません。結果として、非ブロック化イベント・モニターは、アクティブ率の高いシステムではデータの脱落という影響を受けます。イベント・モニターによって生じるパフォーマンス上のオーバーヘッドを最小限に抑えるには、NONBLOCKED 文節を使用します。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED
```

5. イベント・モニターについて収集できるイベント・ファイルの最大数を指定します。この限度に達すると、イベント・モニターは自分自身を非活動化します。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES 5
   ```

   5 は、作成されるイベント・ファイルの最大数です。

   イベント・モニターが作成できるイベント・ファイルの数に制限がないことを指定することもできます。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES NONE
   ```

6. イベント・モニターによって作成されるそれぞれのイベント・ファイルごとに、最大サイズを指定します (4K ページ単位)。この限度に達すると、新規ファイルが作成されます。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES 5 MAXFILESIZE 32
   ```

   32 は、1 つのイベント・ファイルに入れることができる 4K ページ単位の最大数です。

   この値は、BUFFERSIZE パラメーターによって指定された値よりも大きくなければなりません。また、イベント・ファイルのサイズに制限がないことを指定することもできます。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES NONE MAXFILESIZE NONE
   ```

7. データベースを開始するたびに、イベント・モニターが自動的に活動化されるかどうかを指定します。デフォルトでは、データベースの開始時にイベント・モニターは自動的には活動化されません。

   - データベースの開始時に自動的に開始するイベント・モニターを作成するには、次のステートメントを発行する。

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED AUTOSTART
     ```

- データベースの開始時に自動的に開始しないイベント・モニターを作成するには、次のステートメントを発行する。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                     WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                     NONBLOCKED MANUALSTART
```

イベント・モニターを活動化または非活動化するには、SET EVENT MONITOR STATE ステートメントを使用します。

ファイル・イベント・モニターが作成されて活動化されると、指定されたイベントが発生するたびに、モニター・データを記録します。

関連概念:

- 66 ページの『イベント・モニターのファイル管理』
- 51 ページの『イベント・モニター』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』
- 68 ページの『表書き込みイベント・モニターおよびファイル・イベント・モニターのバッファー方式』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』
- 71 ページの『パーティション・データベース用のイベント・モニターの作成』
- 74 ページの『コマンド行からのファイルまたはパイプ・イベント・モニター出力のフォーマット』

関連資料:

- 75 ページの『イベント・モニターの出力例』
- 52 ページの『イベント・タイプ』

イベント・モニターのファイル管理

ファイル・イベント・モニターにより、イベント・モニターは、イベント・レコードをファイルに保管することができます。イベント・モニターのすべての出力は、CREATE EVENT MONITOR ステートメントの FILE パラメーターで指定されたディレクトリーに入れられます。このディレクトリーが存在しない場合、DB2® はそれを作成しません。したがって、モニターが活動化される前に、そのディレクトリーが存在していなければなりません。そうしない場合、SET EVENT MONITOR コマンドがエラーを戻します。ファイル・イベント・モニターが最初に活動化されるとき、このディレクトリーに db2event.ctl という名前の制御ファイルが作成されます。このファイルは除去したり修正したりしないでください。

デフォルトでは、イベント・モニターはそのトレースを 00000000.evt と呼ばれる単一のファイルに書き込みます。このファイルは、ファイル・システム上にスペースがあるかぎり大きくなります。 CREATE EVENT MONITOR ステートメントの MAXFILESIZE パラメーターでファイル・サイズの限界を指定した場合には、 1 つのファイルがいっぱいになると、出力は自動的に次の番号のファイルに書き込まれます。したがって、アクティブ・ファイルは、番号の一番大きいファイルとなります。

また、CREATE EVENT MONITOR ステートメントの MAXFILES パラメーターを使って、イベント・モニター・トレース全体の最大サイズを制限できます。ファイルの数が MAXFILES で定義された最大値を超えると、イベント・モニターは自らを非活動化し、以下のメッセージが管理通知ログに書き込まれます。

```
DIA1601I Event Monitor monitor-name was deactivated when it reached
its preset MAXFILES and MAXFILESIZE limit.
```

すべてのファイルを除去することにより、この状況を避けることができます。イベント・モニターがまだ実行している間に、アクティブ・ファイルを除くすべてのイベント・ファイルを除去することができます。

ファイル・イベント・モニターがディスク・スペースを使い尽くした場合、システム・エラー・レベル・メッセージを管理通知ログに記録した後、自動的にシャットダウンします。

ファイル・イベント・モニターを再始動する場合、既存のデータを消去するか、既存のデータの後に新規データを追加することができます。このオプションは、CREATE EVENT MONITOR ステートメントで指定されます。そこでは、APPEND モニターまたは REPLACE モニターのどちらかを作成することができます。APPEND がデフォルトのオプションです。 APPEND イベント・モニターは、最後に使用していたファイルの終わりから書き込みを開始します。そのファイルを除去すると、次の順番のファイル番号が使用されます。追加のイベント・モニターが再始動されると、start_event だけが生成されます。イベント・ログ・ヘッダーとデータベース・ヘッダーは、最初の活動化のときに生成されます。 REPLACE イベント・モニターは常に既存のイベント・ファイルを削除し、00000000.evt から書き込みを開始します。

イベント・モニターがアクティブになっているときにモニター・データを処理したい場合があります。そのことは可能です。さらに、ファイルの処理を終えたとき、そのファイルを削除し、次のモニター・データのためにスペースを解放することができます。イベント・モニターを停止して再始動しないかぎり、次のファイルに強制的に切り替えることはできません。また、APPEND モードでなければなりません。アクティブ・ファイル内でどのイベントの処理が終わったかを把握しておくために、処理された最後のレコードのファイル番号およびファイル場所だけを追跡するアプリケーションを作成することができます。次回トレースを処理するときには、アプリケーションはそのファイルの位置を検索するだけで済みます。

関連概念:

- 51 ページの『イベント・モニター』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』
- 68 ページの『表書き込みイベント・モニターおよびファイル・イベント・モニターのバッファー方式』
- 6 ページの『システム・モニター出力 : 自己記述型データ・ストリーム』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』
- 63 ページの『ファイル・イベント・モニターの作成』

関連資料:

- 75 ページの『イベント・モニターの出力例』

表書き込みイベント・モニターおよびファイル・イベント・モニターのバッファー方式

イベント・モニター処理は、レコードをファイルまたは表に書き出す前に、 2 つの内部バッファーを使ってそれをバッファーに入れます。バッファーがいっぱいになると、自動的にレコードの書き込みが行われます。そのため、より大きなバッファーを指定してディスク・アクセスの回数を減らせば、大量のスループットのあるイベント・モニターのモニター・パフォーマンスを改善することができます。イベント・モニターにそのバッファーをフラッシュさせるには、それを非活動化するか、または FLUSH EVENT MONITOR ステートメントを使用してバッファーを空にする必要があります。

ブロック化されたイベント・モニターは、両方のバッファーがいっぱいのときに、モニター・データを送信するデータベース処理を延期します。これは、ブロック化されたイベント・モニターがアクティブのときに、イベント・レコードが廃棄されないようにするためです。延期されたデータベース処理とその結果として付随するデータベース処理は、バッファーが書き込まれるまでは実行できません。これは、ワークロードの種類や入出力装置の速度によっては、パフォーマンス上の大きなオーバーヘッドとなることがあります。イベント・モニターはデフォルトではブロック化されます。

ブロック化されていないイベント・モニターは、イベント・モニターがデータを書き込むことのできる速度よりも速くデータが到着するとき、エージェントから来るモニター・データを廃棄します。これにより、イベント・モニターが、他のデータベース・アクティビティーにパフォーマンス負荷を与えないようにします。

イベント・レコードを廃棄したイベント・モニターは、オーバーフロー・イベントを生成します。これは、モニターがイベントを廃棄する開始時刻と停止時刻、およびその期間に廃棄されたイベントの数を指定します。イベント・モニターは、ペンディングのオーバーフローがあることを報告して、終了または非活動化することがあります。その場合、次のメッセージが db2diag.log に書き込まれます。

```
DIA2503I Event Monitor monitor-name had a pending overflow record
when it was deactivated.
```

イベント・モニター・データの消失は、個々のイベント・レコードについても生じる可能性があります。イベント・レコードの長さがイベント・バッファリング・サイズを超えると、バッファー内に収まらないデータは切り捨てられます。例えば、stmt_text モニター・エレメントのキャプチャー中に、モニターされているデータベースにアタッチしたアプリケーションが長い SQL ステートメントを発行すると、この状態が生じます。イベント・レコード情報全体をキャプチャーする必要がある場合には、より大きなバッファーを指定してください。より大きなバッファーを指定すれば、ファイルまたは表への書き込み頻度が減ることに留意してください。

関連概念:

- 66 ページの『イベント・モニターのファイル管理』

- 51 ページの『イベント・モニター』
- 60 ページの『イベント・モニターの表管理』

関連タスク:

- 63 ページの『ファイル・イベント・モニターの作成』
- 56 ページの『表イベント・モニターの作成』

パイプ・イベント・モニターの作成

イベント・モニターの作成時に、それが収集した情報の保管場所を決定する必要があります。パイプ・イベント・モニターは、イベント・モニターから名前付きパイプに直接イベント・レコードを流します。イベント・モニターがイベント・データを書き込んですぐにデータをパイプから読み取るのは、モニター・アプリケーションの責任です。イベント・モニターがデータをパイプに書き込めない場合 (例えばパイプがいっぱいになっている場合) は、モニター・データは失われます。

パイプ・イベント・モニターは、CREATE EVENT MONITOR ステートメントで定義されます。

前提条件:

パイプ・イベント・モニターを作成するには、DBADM 権限が必要です。

手順:

1. イベント・モニター・データが名前付きパイプに送られることを指定します。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO PIPE '/home/riihi/dlevents'
   ```

   dlmon はイベント・モニターの名前です。

   /home/riihi/dlevents は、イベント・モニターがイベント・レコードを送る宛先の名前付きパイプ (UNIX 上) の名前です。 CREATE EVENT MONITOR ステートメントは、UNIX および Windows パイプ名前構文をサポートします。

   CREATE EVENT MONITOR ステートメントで指定される名前付きパイプは、イベント・モニターを活動化するときに存在し、開いている必要があります。イベント・モニターが自動的に開始するように指定する場合には、イベント・モニターの作成前に名前付きパイプが存在している必要があります。
2. モニター対象のイベントのタイプを指定します。単一イベント・モニターで、1 つ以上のイベント・タイプをモニターすることができます。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO PIPE '/home/riihi/dlevents'
   ```

   このイベント・モニターは、CONNECTIONS および DEADLOCKS WITH DETAILS イベント・タイプをモニターします。
3. データベースを開始するたびに、イベント・モニターが自動的に活動化されるかどうかを指定します。デフォルトでは、データベースの開始時にイベント・モニターは自動的には活動化されません。

- データベースの開始時に自動的に開始するイベント・モニターを作成するには、次のステートメントを発行する。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO PIPE '/home/riihi/dlevents'
                    AUTOSTART
```

- データベースの開始時に自動的に開始しないイベント・モニターを作成するには、次のステートメントを発行する。

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO PIPE '/home/riihi/dlevents'
                    MANUALSTART
```

イベント・モニターを活動化または非活動化するには、SET EVENT MONITOR STATE ステートメントを使用します。

パイプ・イベント・モニターが作成されて活動化されると、指定されたイベントが発生するたびに、モニター・データを記録します。

関連概念:

- 70 ページの『イベント・モニターの名前付きパイプ管理』
- 6 ページの『システム・モニター出力 : 自己記述型データ・ストリーム』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』

関連資料:

- 75 ページの『イベント・モニターの出力例』

イベント・モニターの名前付きパイプ管理

パイプ・イベント・モニターでは、名前付きパイプによって、イベント・モニターのデータ・ストリームを処理することができます。パイプ・イベント・モニターの使用は、リアルタイムでイベント・レコードを処理する必要がある場合に有用です。別の利点として、アプリケーションが不要データを読み取った場合にパイプを介さずにそのデータを無視できるので、ストレージ要件を相当減らせる可能性があります。

AIX® では、mkfifo コマンドを使用して、名前付きパイプを作成することができます。 Linux および他の UNIX® タイプ (Solaris™ オペレーティング環境など) では、pipe() ルーチンを使用します。 Windows® NT または Windows 2000 では、CreateNamedPipe() ルーチンを使用して、名前付きパイプを作成することができます。

データをパイプに送ると、入出力は必ずブロック化され、バッファリングだけがパイプによって実行されます。イベント・モニターがイベント・データを書き込んですぐにデータをパイプから読み取るのは、モニター・アプリケーションの責任です。イベント・モニターがデータをパイプに書き込めない場合 (例えばパイプがいっぱいになっている場合) は、モニター・データは失われます。

さらに、名前付きパイプには、着信するイベント・レコードを処理するための十分なスペースがなければなりません。アプリケーションが十分な速さで名前付きパイ

プからデータを読み取らないと、パイプは満杯になり、オーバーフローが発生します。パイプ・バッファーが小さいほど、オーバーフローの発生する可能性が大きくなります。

パイプのオーバーフローが発生すると、モニターはオーバーフローが発生していることを示すオーバーフロー・イベント・レコードを作成します。イベント・モニターは OFF になりませんが、モニター・データは失われます。モニターが非活動化されるときに未解決のオーバーフロー・イベント・レコードがある場合は、診断メッセージが記録されます。それ以外の場合、オーバーフロー・イベント・レコードは、可能なときにパイプに書き込まれます。

オペレーティング・システムでパイプ・バッファー・サイズを定義できる場合は、少なくとも 32K のパイプ・バッファーを使用してください。大ボリュームのイベント・モニターの場合、モニター・アプリケーションの処理優先順位を、エージェント処理優先順位以上の値に設定する必要があります。

関連概念:

- 51 ページの『イベント・モニター』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』
- 6 ページの『システム・モニター出力 : 自己記述型データ・ストリーム』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』
- 55 ページの『イベント・モニターの作成』

関連資料:

- 75 ページの『イベント・モニターの出力例』

パーティション・データベース用のイベント・モニターの作成

パーティション・データベース・システムでファイルまたはパイプ・イベント・モニターを作成するときには、収集したいモニター・データの有効範囲を決定する必要があります。イベント・モニターは、オペレーティング・システムのプロセスまたはスレッドを使用して、イベント・レコードを作成します。このプロセスまたはスレッドを実行しているパーティションのことを、モニター・パーティションといいます。ファイルおよびパイプ・イベント・モニターは、モニター・パーティション上でローカルに起こったイベントをモニターしたり、 DB2 データベース・マネージャーを実行しているすべてのパーティションで起こったイベントをグローバルにモニターしたりします。グローバル・イベント・モニターは、すべてのパーティションのアクティビティーを含むトレースを、モニター・パーティション上に 1 つ作成します。イベント・モニターがローカルまたはグローバルのどちらであるかは、モニター有効範囲として示します。

モニター・パーティションおよびモニター有効範囲のどちらも、CREATE EVENT MONITOR ステートメントで指定します。

表書き込みイベント・モニターについては、ローカルまたはグローバル有効範囲の概念は該当しません。表書き込みイベント・モニターが活動化されているときに

は、イベント・モニターはすべての パーティションで実行されます。 (より正確に言えば、イベント・モニター処理は、ターゲット表があるデータベース・パーティション・グループに属するパーティションに対して実行されます。) また、イベント・モニター処理が実行するそれぞれのパーティションには、同じターゲット表のセットがあります。これらの表のデータは、それがモニター・データの個々のパーティションのビューを表すという点で異なります。それぞれのパーティションのイベント・モニターのターゲット表の中の目的とする値にアクセスする SQL ステートメントを発行することによって、すべてのパーティションの集約値を入手することができます。

各ターゲット表の最初の列は PARTITION_KEY という名前で、これは表のパーティション・キーとして使用されます。この列の値は、各イベント・モニター・プロセスがそのプロセスが実行されるデータベース・パーティションにデータを挿入できるように選択されます。つまり挿入操作は、イベント・モニター・プロセスが実行されるデータベース・パーティションでローカルに実行されます。どのデータベース・パーティションでも、 PARTITION_KEY フィールドには同じ値が含まれます。このことは、データ・パーティションがドロップされてデータの再分散が行われる場合、ドロップされたデータベース・パーティション上のすべてのデータは均等に分散されるのではなく、他の 1 つのデータベース・パーティションに移動することを意味します。そのため、データベース・パーティションを除去する場合は、そのデータベース・パーティションにある表のすべての行を削除することを事前に検討してください。

その他、表ごとに PARTITION_NUMBER という名前の列を定義することができます。この列には、データが挿入されたパーティションの番号が含まれます。 PARTITION_NUMBER 列は PARTITION_KEY 列と違って必須ではありません。

ターゲット表が定義されている表スペースは、イベント・モニター・データが書き込まれるすべてのパーティションに存在しなければなりません。この規則に従わないと、表スペースが存在しない (イベント・モニターがある) ログオン・パーティションにレコードが書き込まれないことになります。ただし、イベントは表スペースが存在するパーティションに書き込まれ、エラーは戻されません。この動作を利用すると、ユーザーは、特定のパーティションにのみ存在する表スペースを作成することにより、モニター用にパーティションのサブセットを選択できることになります。

表書き込みイベント・モニターの活動化の際、 FIRST_CONNECT および EVMON_START に対するコントロール表の各行は、カタログ・データベース・パーティションにのみ挿入されます。そのため、カタログ・データベース・パーティションにコントロール表のための表スペースが存在しなければなりません。そのスペースがカタログ・データベース・パーティションに存在しない場合は、これらの挿入は行われません。表書き込みイベント・モニターが活動化されるときにパーティションがまだアクティブでない場合は、イベント・モニターが活動化される前にそのパーティションが活動化されます。この場合のデータベースの活動化は、 SQL CONNECT ステートメントによってすべてのパーティションのデータベースが活動化されたかのような動作になります。

注: 詳細付きデッドロック接続イベントにおけるロック・リストには、ロックを待機しているパーティション上のアプリケーションによって保留されているロッ

クだけが含まれます。例えば、デッドロックに関係したアプリケーションがノード 20 上でロックを待機している場合、ノード 20 上のそのアプリケーションによって保留されているロックだけがリストに含まれます。

前提条件:

パーティション・データベース用のイベント・モニターを作成するには、DBADM 権限が必要です。

手順:

1. モニター対象のパーティションを指定します。

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                        WRITE TO FILE '/tmp/dlevents'
                        ON PARTITION 3
   ```

   dlmon は、イベント・モニターの名前を表します。

   /tmp/dlevents は、イベント・モニターがイベント・ファイルを書き込むディレクトリー・パス (UNIX 上) の名前です。

   3 は、モニター対象のパーティション番号を表します。
2. イベント・モニター・データをローカル有効範囲で収集するか、またはグローバル有効範囲で収集するかを指定します。すべてのパーティションからのイベント・モニター・レポートを収集するには、次のステートメントを発行します。

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                        WRITE TO FILE '/tmp/dlevents'
                        ON PARTITION 3 GLOBAL
   ```

   デッドロックおよび詳細付きデッドロック・イベント・モニターに限り、GLOBAL として定義することができます。すべてのパーティションは、デッドロック関連のイベント・レコードをパーティション 3 に報告します。

   ローカル・パーティションからのみイベント・モニター・レポートを収集するには、次のステートメントを発行します。

   ```
   CREATE EVENT MONITOR dlmon FOR DEADLOCKS
                        WRITE TO FILE '/tmp/dlevents'
                        ON PARTITION 3 LOCAL
   ```

   これは、パーティション・データベースでのファイルおよびパイプ・イベント・モニターのデフォルトの動作です。表書き込みイベント・モニターの場合、LOCAL および GLOBAL 文節は無視されます。
3. 既存のイベント・モニターのモニター・パーティションおよび有効範囲値を検討することができます。次のステートメントで、SYSCAT.EVENTMONITORS 表を照会して、このことを行います。

   ```
   SELECT EVMONNAME, NODENUM, MONSCOPE FROM SYSCAT.EVENTMONITORS
   ```

イベント・モニターが作成されて活動化されると、指定されたイベントが発生するたびに、モニター・データを記録します。

関連概念:

- 51 ページの『イベント・モニター』

- 5 ページの『カウンターの状況および可視性』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』

関連タスク:

- 63 ページの『ファイル・イベント・モニターの作成』
- 56 ページの『表イベント・モニターの作成』

関連資料:

- 75 ページの『イベント・モニターの出力例』
- 52 ページの『イベント・タイプ』

コマンド行からのファイルまたはパイプ・イベント・モニター出力のフォーマット

ファイルまたはパイプ・イベント・モニターの出力は バイナリー・ストリームの論理データ・グループです。 db2evmon コマンドを使用することによって、このデータ・ストリームをコマンド行からフォーマットすることができます。この生産性向上ツールは、イベント・モニターのファイルまたはパイプからイベント・レコードを読み込んだ後、それらを画面に書き出します (標準出力)。

イベント・ファイルのパスを提供するか、またはデータベースおよびイベント・モニターの名前を提供することによって、どのイベント・モニターの出力をフォーマットするのかを指定することができます。

前提条件:

データベースに接続しているのではない限り、権限は不要です。データベースに接続している場合には、以下のいずれかが必要です。

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM

手順:

イベント・モニター出力をフォーマットするには、次のようにします。

- イベント・モニター・ファイルが格納されているディレクトリーを指定する。

  ```
  db2evmon -path '/tmp/dlevents'
  ```

  `/tmp/dlevents` は (UNIX) パスです。
- データベースおよびイベント・モニター名を指定する。

  ```
  db2evmon -db 'sample' -evm 'dlmon'
  ```

  `sample` は、イベント・モニターが属するデータベースを示します。

  `dlmon` はイベント・モニターを示します。

関連概念:

- 66 ページの『イベント・モニターのファイル管理』

- 70 ページの『イベント・モニターの名前付きパイプ管理』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』

関連タスク:

- 63 ページの『ファイル・イベント・モニターの作成』
- 69 ページの『パイプ・イベント・モニターの作成』

イベント・モニターの出力例

イベント・モニターの性質を示すため、以下にデッドロック・モニターのシナリオの例を記述します。このシナリオをインプリメントするには、3 つの DB2 CLP ウィンドウが必要です。最初の CLP は モニター・セッション、後の 2 つは アプリケーション 1 およびアプリケーション 2 と呼びます。また、DB2 SAMPLE データベースも必要です。

注: 以下のステップのいずれかを実行すると、SAMPLE データベースを作成して、そこにデータを入れることができます。

- UNIX: `sqllib/bin/db2sampl`
- Windows NT または Windows 2000: `sqllib¥bin¥db2sampl.exe`

モニター・セッションから、表データおよびデータベースへの接続の間に発生するデッドロックを記録するイベント・モニターを定義します。

モニター・セッション

```
db2 connect to sample
db2 "create event monitor dlmon for tables, deadlocks with details
     write to file 'c:¥dlmon'"
mkdir c:¥dlmon
db2 "set event monitor dlmon state 1"
```

この時点で、データベースを使用している 2 つのアプリケーションがデッドロック状態になっています。デッドロックとは、それぞれのアプリケーションが処理を続行するのに必要なロックを、それぞれ他方が保持しているという状態です。最終的に、このデッドロックは DB2 のデッドロック検出コンポーネントによって検出され、トランザクションのうち 1 つがロールバックされます。結果として、1 つのアプリケーションだけがトランザクションを正常に完了します。以下に、このシナリオを示します。

アプリケーション 1

```
db2 connect to sample
db2 +c "insert into staff values(26, 'Simpson',
        2, 'Mgr', 13, 35000, 0)"
```

この時点でアプリケーション 1 は、STAFF 表の行に関する排他ロックを保持しています。

注: 上で使用されている +c オプションにより、指定された CLP コマンドの自動コミットが OFF になります。

アプリケーション 2

```
db2 connect to sample
db2 +c "insert into department values('7G', 'Safety',
        '1', 'A00', NULL)"
```

この時点でアプリケーション 2 は、DEPARTMENT 表の行に関する排他ロックを保持しています。

アプリケーション 1

```
db2 +c "select deptname from department"
```

カーソル固定を前提にすると、アプリケーション 1 では DEPARTMENT 表の個々の行を取り出す際にその行に関する共用ロックが必要になりますが、最後の行に関する排他ロックはアプリケーション 2 に保持されているので、その行に関するロックを取得することはできません。アプリケーション 1 はロックが解除されるのを待機するので、 LOCK WAIT 状態になります。

アプリケーション 2

```
db2 +c "select name from staff"
```

アプリケーション 2 は、アプリケーション 1 が STAFF 表の最後の行に関する排他ロックを解除するのを待機するので、同様に LOCK WAIT 状態になります。

この時点で両方のアプリケーションともデッドロック状態になっています。一方が処理を続行するのに必要なリソースを他方が保持している状態になっているので、この待機状態は絶対に解決されません。結局デッドロック検出機能によってデッドロックがチェックされ、ロールバックのビクティムが選択されます。

アプリケーション 2

```
SQLN0991N The current transaction has been
rolled back because of a deadlock or timeout.
Reason code "2". SQLSTATE=40001
```

この時点でイベント・モニターによりデッドロック・イベントが宛先に記録されます。アプリケーション 1 は処理を続行できるようになります。

アプリケーション 1

```
DEPTNAME
----------------------------
PLANNING
INFORMATION CENTER
. . .
SOFTWARE SUPPORT
9 record(s) selected
```

イベント・モニターの出力はバッファーに入れられますが、このシナリオで生成されたイベント・レコードによってバッファーがいっぱいにはなりませんでした。そのため、イベント・モニターの値は強制的にイベント・モニター書き出しプログラムに送られます。表イベント・データ (データベースが非活動化すると生成される) を生成するために、アプリケーション 1、アプリケーション 2、およびモニター・セッションがデータベースから切断します。

アプリケーション 1

```
db2 connect reset
```

アプリケーション 2

```
db2 connect reset
```

モニター・セッション CLP 内でデータベースから切断した後、イベント・トレースがバイナリー・ファイルとして作成されます。この時点で db2evmon ツールを使用してこのトレースを形式設定できます。

モニター・セッション

```
db2 connect reset
db2evmon -path c:¥dlmon
```

イベント・モニターによって使用される論理データ・グループは、 4 つの異なるレベルに従って配列され、表示されます。すなわちモニター、プロローグ、内容、およびエピローグです。

モニター:

モニター・レベルの情報は、すべてのイベント・モニターに対して生成されます。これは、イベント・モニターのメタデータから成っています。

SQLM_DBMON_VERSION6、SQLM_DBMON_VERSION7、または SQLM_DBMON_VERSION8 のバージョンを戻すイベント・モニターのみが、自己

記述型データ・ストリームを使用します。バージョン 6 以前の出力はバージョン 5 の方式で読み取る必要があります。このようなサイズが静的な構造については、sqlmon.h ファイルを参照してください。

プロローグ:

プロローグ情報は、イベント・モニターが活動化されると生成されます。

```
------------------------------------------------------------------------
                            EVENT LOG HEADER
  Event Monitor name: DLMON
  Server Product ID: SQL08020
  Version of event monitor data: 7
  Byte order: LITTLE ENDIAN
  Number of nodes in db2 instance: 1
  Codepage of database: 1252
  Territory code of database: 1
  Server instance name: DB2
------------------------------------------------------------------------

------------------------------------------------------------------------
  Database Name: SAMPLE
  Database Path: D:¥DB2¥NODE0000¥SQL00001¥
  First connection timestamp: 11/25/2003 13:07:32.649113
  Event Monitor Start time:   11/25/2003 13:07:52.292867
------------------------------------------------------------------------
```

内容:

イベント・モニターの指定されたイベント・タイプに固有の情報は、内容セクションに表示されます。このセクションに記録されるイベントには、それらを作成したアプリケーションへの参照が含まれます (アプリケーション・ハンドルまたはアプリケーション ID)。複数のアプリケーションのイベントを追跡している場合には、アプリケーション ID を使用して各種イベントを追跡します。オーバーフロー・イベントは、内容セクションのその他のものとは異なり、特定のイベント・タイプに対応しません。これは脱落したレコードの数を追跡し、システムが (ブロック化されていない) イベント・モニターに追い付けないときに生成されます。この例では、前のステートメントによって生じたデッドロック状態によって作成されたデッドロック・イベントがあります。

```
3) Connection Header Event ...
  Appl Handle: 12
  Appl Id: *LOCAL.DB2.0063C5180732
  Appl Seq number: 0001
  DRDA AS Correlation Token: *LOCAL.DB2.0063C5180732
  Appl Seq number: 0001
  DRDA AS Correlation Token: *LOCAL.DB2.0063C5180732
  Program Name    : db2bp.exe
  Authorization Id: RIIHI
  Execution Id    : RIIHI
  Codepage Id: 1252
  Territory code: 1
  Client Process Id: 312
  Client Database Alias: SAMPLE
  Client Product Id: SQL08020
  Client Platform: Unknown
  Client Communication Protocol: Local
  Client Network Name:
  Connect timestamp: 11/25/2003 13:07:32.649113

4) Connection Header Event ...
  Appl Handle: 13
```

```
  Appl Id: *LOCAL.DB2.00FE05180800
  Appl Seq number: 0001
  DRDA AS Correlation Token: *LOCAL.DB2.00FE05180800
  Program Name    : db2bp.exe
  Authorization Id: RIIHI
  Execution Id    : RIIHI
  Codepage Id: 1252
  Execution Id    : RIIHI
  Codepage Id: 1252
  Territory code: 0
  Client Process Id: 2144
  Client Database Alias: SAMPLE
  Client Product Id: SQL08020
  Client Platform: Unknown
  Client Communication Protocol: Local
  Client Network Name:
  Connect timestamp: 11/25/2003 13:08:00.897979

5) Deadlock Event ...
  Deadlock ID:   1
  Number of applications deadlocked: 2
  Deadlock detection time: 11/25/2003 13:09:02.831833
  Rolled back Appl participant no: 2
  Rolled back Appl Id: *LOCAL.DB2.00FE05180800
  Rolled back Appl seq number: : 0001

6) Deadlocked Connection ...
  Deadlock ID:   1
  Participant no.: 2
  Participant no. holding the lock: 1
  Appl Id: *LOCAL.DB2.00FE05180800
  Participant no. holding the lock: 1
  Appl Id: *LOCAL.DB2.00FE05180800
  Appl Seq number: 0001
  Appl Id of connection holding the lock: *LOCAL.DB2.00FE05180800
  Seq. no. of connection holding the lock: 0001
  Lock wait start time:
  Lock Name       : 0x020003002800000000000000052
  Lock Attributes : 0x00000000
  Release Flags   : 0x00000001
  Lock Count      : 1
  Hold Count      : 0
  Current Mode    : none
  Deadlock detection time: 11/25/2003 13:09:02.832048
  Table of lock waited on      : STAFF
  Schema of lock waited on     : RIIHI

  Tablespace of lock waited on : USERSPACE1
  Type of lock: Row
  Mode of lock: X   - Exclusive
  Mode application requested on lock: NS  - Share (and Next Key Share)
  Node lock occured on: 0
  Lock object name: 40
  Application Handle: 13
  Deadlocked Statement:
    Type     : Dynamic
    Operation: Fetch
    Section  : 201
    Creator  : NULLID
    Package  : SQLC2E03
    Cursor   : SQLCUR201
    Cursor was blocking: FALSE
    Text     : select name from staff
  List of Locks:
      Lock Name                  : 0x010000000100000001004C0056
      Lock Attributes            : 0x00000000
      Release Flags              : 0x40000000
```

```
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Object Name           : 0
                    Object Type                : Internal - Variation
                    Mode                       : S   - Share

                    Lock Name                  : 0x020004000D0000000000000052
                    Lock Attributes            : 0x00000000
                    Release Flags              : 0x40000000
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Object Name           : 13
                    Object Type                : Row
                    Tablespace Name            : USERSPACE1
                    Table Schema               : RIIHI
                    Table Name                 : DEPARTMENT
                    Mode                       : X   - Exclusive

                    Lock Name                  : 0x94928D848F9F949E7B89505241
                    Lock Attributes            : 0x00000000
                    Release Flags              : 0x40000000
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Object Name           : 0
                    Object Type                : Internal - Plan
                    Mode                       : S   - Share

                    Lock Name                  : 0x96A09A989DA09A7D8E8A6C7441
                    Lock Attributes            : 0x00000000
                    Release Flags              : 0x40000000
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Object Name           : 0
                    Hold Count                 : 0
                    Lock Object Name           : 0
                    Object Type                : Internal - Plan
                    Mode                       : S   - Share

                    Lock Name                  : 0x020004000000000000000000054
                    Lock Attributes            : 0x00000000
                    Release Flags              : 0x40000000
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Object Name           : 4
                    Object Type                : Table
                    Tablespace Name            : USERSPACE1
                    Table Schema               : RIIHI
                    Table Name                 : DEPARTMENT
                    Mode                       : IX  - Intent Exclusive

                    Lock Name                  : 0x020003000000000000000000054
                    Lock Attributes            : 0x00000000
                    Release Flags              : 0x00000001
                    Lock Count                 : 1
                    Hold Count                 : 0
                    Lock Object Name           : 3
                    Object Type                : Table
                    Lock Object Name           : 3
                    Object Type                : Table
                    Tablespace Name            : USERSPACE1
                    Table Schema               : RIIHI
                    Table Name                 : STAFF
                    Mode                       : IS  - Intent Share

                Locks Held:    6
```

```
  Locks in List: 6

7) Deadlocked Connection ...
  Deadlock ID:   1
  Participant no.: 1
  Participant no. holding the lock: 2
  Appl Id: *LOCAL.DB2.0063C5180732
  Appl Seq number: 0002
  Appl Id of connection holding the lock: *LOCAL.DB2.0063C5180732
  Seq. no. of connection holding the lock: 0002
  Lock wait start time:
  Lock Name       : 0x020004000D0000000000000052
  Lock Attributes : 0x00000000
  Release Flags   : 0x00000001
  Lock Count      : 1
  Hold Count      : 0
  Lock Count      : 1
  Hold Count      : 0
  Current Mode    : none
  Deadlock detection time: 11/25/2003 13:09:02.968324
  Table of lock waited on      : DEPARTMENT
  Schema of lock waited on     : RIIHI
  Tablespace of lock waited on : USERSPACE1
  Type of lock: Row
  Mode of lock: X   - Exclusive
  Mode application requested on lock: NS  - Share (and Next Key Share)
  Node lock occured on: 0
  Lock object name: 13
  Application Handle: 12
  Deadlocked Statement:
    Type     : Dynamic
    Operation: Fetch
    Section  : 201
    Creator  : NULLID
    Package  : SQLC2E03
    Cursor   : SQLCUR201
    Cursor was blocking: FALSE
    Text     : select deptname from department
  List of Locks:
      Lock Name                   : 0x020004000D0000000000000052
      Lock Attributes             : 0x00000000
      Release Flags               : 0x00000001
      Lock Count                  : 1
      Hold Count                  : 0
      Lock Object Name            : 13
      Object Type                 : Row
      Tablespace Name             : USERSPACE1
      Table Schema                : RIIHI
      Table Name                  : DEPARTMENT
      Mode                        : NS  - Share (and Next Key Share)

      Lock Name                   : 0x0100000001000000100640056
      Lock Attributes             : 0x00000000
      Release Flags               : 0x40000000
      Lock Count                  : 1
      Hold Count                  : 0
      Lock Object Name            : 0
      Object Type                 : Internal - Variation
      Mode                        : S   - Share

      Lock Name                   : 0x02000300280000000000000052
      Lock Attributes             : 0x00000000
      Lock Name                   : 0x02000300280000000000000052
      Lock Attributes             : 0x00000000
      Release Flags               : 0x40000000
      Lock Count                  : 1
      Hold Count                  : 0
```

```
      Lock Object Name           : 40
      Object Type                : Row
      Tablespace Name            : USERSPACE1
      Table Schema               : RIIHI
      Table Name                 : STAFF
      Mode                       : X   - Exclusive

      Lock Name                  : 0x94928D848F9F949E7B89505241
      Lock Attributes            : 0x00000000
      Release Flags              : 0x40000000
      Lock Count                 : 1
      Hold Count                 : 0
      Lock Object Name           : 0
      Object Type                : Internal - Plan
      Mode                       : S   - Share

      Lock Name                  : 0x020004000000000000000000054
      Lock Attributes            : 0x00000000
      Release Flags              : 0x00000001
      Lock Attributes            : 0x00000000
      Release Flags              : 0x00000001
      Lock Count                 : 1
      Hold Count                 : 0
      Lock Object Name           : 4
      Object Type                : Table
      Tablespace Name            : USERSPACE1
      Table Schema               : RIIHI
      Table Name                 : DEPARTMENT
      Mode                       : IS  - Intent Share

      Lock Name                  : 0x020003000000000000000000054
      Lock Attributes            : 0x00000000
      Release Flags              : 0x40000000
      Lock Count                 : 1
      Hold Count                 : 0
      Lock Object Name           : 3
      Object Type                : Table
      Tablespace Name            : USERSPACE1
      Table Schema               : RIIHI
      Table Name                 : STAFF
      Mode                       : IX  - Intent Exclusive

  Locks Held:    6
  Locks in List: 6
```

エピローグ:

エピローグ情報は、データベースが非活動化状態にあるとき (最後のアプリケーションが切断を終了するとき) に生成されます。

```
8) Table Event ...
  Table schema: SYSIBM
  Table name: SYSBOOT

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 1
  Rows read: 1
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101214

9) Table Event ...
  Table schema: SYSIBM
```

```
  Table name: SYSTABLES

  Record is the result of a flush: FALSE
  Table type: Catalog
  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 28
  Index object pages: 17
  Lob object pages: 256
  Rows read: 2
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101265

10) Table Event ...
  Table schema: SYSIBM
  Table name: SYSPLAN

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 9
  Index object pages: 5
  Lob object pages: 320
  Rows read: 1
  Rows written: 0
  Overflow Accesses: 0
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101303

11) Table Event ...
  Table schema: SYSIBM
  Table name: SYSDBAUTH

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 1
  Index object pages: 3
  Rows read: 3
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101337

12) Table Event ...
  Table schema: SYSIBM
  Table name: SYSEVENTMONITORS
  Table schema: SYSIBM
  Table name: SYSEVENTMONITORS

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 1
  Index object pages: 3
  Lob object pages: 64
  Rows read: 3
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101382
```

```
13) Table Event ...
  Table schema: SYSIBM
  Table name: SYSTABLESPACES

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 1
  Index object pages: 7
  Rows read: 3
  Index object pages: 7
  Rows read: 3
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101412

14) Table Event ...
  Table schema: SYSIBM
  Table name: SYSBUFFERPOOLS

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 1
  Index object pages: 4
  Rows read: 1
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101452

15) Table Event ...
  Table schema: SYSIBM
  Table name: SYSVERSIONS

  Record is the result of a flush: FALSE
  Table type: Catalog
  Data object pages: 1
  Index object pages: 3
  Rows read: 1
  Rows written: 0
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 0
  Table event timestamp: 11/25/2003 13:09:23.101541

16) Table Event ...
  Table schema: RIIHI
  Table name: STAFF

  Record is the result of a flush: FALSE
  Table type: User
  Data object pages: 1
  Rows read: 36
  Data object pages: 1
  Rows read: 36
  Rows written: 1
  Overflow Accesses: 0
  Page reorgs: 0
  Tablespace id: 2
  Table event timestamp: 11/25/2003 13:09:23.101890

17) Table Event ...
  Table schema: RIIHI
  Table name: DEPARTMENT
```

```
Record is the result of a flush: FALSE
Table type: User
Data object pages: 1
Rows read: 9
Rows written: 1
Overflow Accesses: 0
Page reorgs: 0
Tablespace id: 2
Table event timestamp: 11/25/2003 13:09:23.101918
```

関連概念:

- 51 ページの『イベント・モニター』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』

関連資料:

- 52 ページの『イベント・タイプ』

イベント・レコードとそれに対応するアプリケーション

多数のアプリケーションがアタッチされたアクティブ・データベースのイベント・トレースでは、特定のアプリケーションに関連したイベント・レコードを追跡するのは面倒な場合があります。トレースを行いやすくするために、それぞれのイベント・レコードには、アプリケーション・ハンドルとアプリケーション ID が含まれています。これらにより、各レコードを、イベント・レコードが生成されたアプリケーションに関連付けることができます。

アプリケーション・ハンドル (**agent_id**) は、アプリケーションの使用期間中はシステム間でユニークです。ただし、このハンドルは再利用されます (16 ビットのカウンターを使ってこの ID を生成します - パーティション・データベース・システムでは、この ID は、調整パーティション番号と 16 ビットのカウンターから成っています)。ほとんどの場合、この再利用は問題になりません。トレースからレコードを読み取るアプリケーションが、終了した接続を検出できるからです。例えば、(トレースで) 既知の agent_ID を持つ接続ヘッダーを見つけたということは、この agent_ID を使っていた前の接続が終了したということです。

アプリケーション ID はタイム・スタンプを含んでいる ID で、データベース・マネージャーを停止して再始動した後であっても必ずユニークなままになります。

特定のアプリケーションのイベント・レコードの検出は、特に表書き込みイベント・モニターでは簡単です。イベント・モニター表では、各行がイベント・レコードに対応しており、アプリケーション・ハンドルおよびアプリケーション ID が、デフォルトの列値となっています。指定されたアプリケーションのすべてのイベント・レコードを検出するには、特定のアプリケーション ID に対応するすべてのイベント・レコードについて、 SQL SELECT ステートメントを発行するだけです。

関連概念:

- 51 ページの『イベント・モニター』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』

関連資料:

- 75 ページの『イベント・モニターの出力例』

イベント・モニター自己記述型データ・ストリーム

イベント・モニターの出力は バイナリー・ストリームの論理データ・グループで、パイプ・イベント・モニターでもファイル・イベント・モニターでも全く同じです。データ・ストリームのフォーマットは、db2evmon コマンドを使用するか、またはクライアント・アプリケーションを開発することによって行えます。このデータ・ストリームは、自己記述型フォーマットで表示されます。 図 3 では、データ・ストリームの構造を示し、 87 ページの表 10 では、戻される論理データ・グループおよびモニター・エレメントのいくつかの例を示します。

注: これらの例や表では、ID として記述名を使用しています。これらの名前は、実際のデータ・ストリームでは **SQLM_ELM_** という接頭部が付きます。例えば db_event は、イベント・モニター出力では SQLM_ELM_DB_EVENT と表示されます。タイプは、実際のデータ・ストリームでは **SQLM_TYPE_** という接頭部が付きます。例えば、ヘッダーはデータ・ストリームで SQLM_TYPE_HEADER と表示されます。

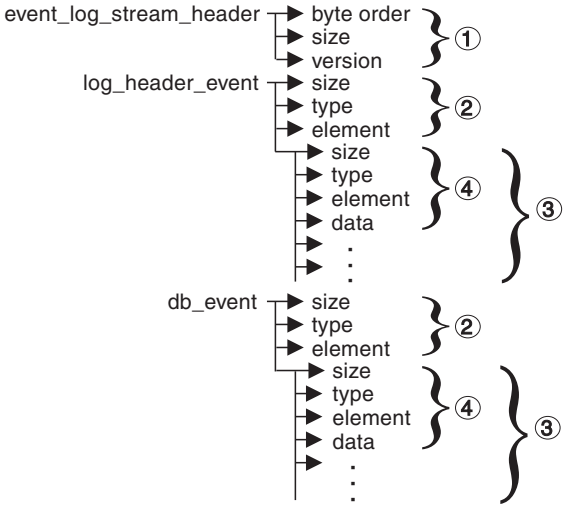

図 3. イベント・モニター・データ・ストリーム

1. sqlm_event_log_data_stream_header の構造は、データ・ストリーム内のほかのヘッダーとは異なります。バージョン・フィールドにより、出力をバージョン 8 のデータ・ストリームとして処理できるかどうかが決まります。

   このヘッダーのサイズとタイプは、バージョン 6 以前のイベント・モニター・ストリームと同じです。これにより、アプリケーションは、イベント・モニター出力が自己記述型か、バージョン 6 より前の静的形式かを判別できます。

注: このモニター・エレメントは、データ・ストリームから sizeof(sqlm_event_log_data_stream) のバイトを読み取ることにより抽出されます。

2. 各論理データ・グループは、サイズとエレメント名を示すヘッダーで始まります。これは、 event_log_stream_header にはあてはまりません。そのサイズ・エレメントには、逆方向互換性を保持するためのダミー値が含まれるからです。
3. ヘッダーのサイズ・エレメントは、論理データ・グループのデータ全体のサイズを示します。
4. モニター・エレメント情報が、論理データ・グループ・ヘッダーに続きます。これも自己記述型です。

表 10. イベント・データ・ストリームの例

論理データ・グループ	データ・ストリーム	説明
event_log_stream_header	sqlm_little_endian	使用されない (以前のリリースとの互換性のため)。
	200	使用されない (以前のリリースとの互換性のため)。
	sqlm_dbmon_version8	データを戻したデータベース・マネージャーのバージョン。自己記述型の形式でデータを記述できるのは、バージョン 6、バージョン 7、およびバージョン 8 のモニターだけです。
log_header_event	100	論理データ・グループのサイズ。
	header	論理データ・グループが始まることを示す。
	log_header	論理データ・グループの名前。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 32 ビット数値。
	byte_order	収集されたモニター・エレメントの名前。
	little_endian	このエレメントに対して収集された値。
	2	このモニター・エレメントに入っているデータのサイズ。
	u16bit	モニター・エレメント・タイプ - 符号なし 16 ビット数値。
	codepage_id	収集されたモニター・エレメントの名前。
	850	このエレメントに対して収集された値。
db_event	100	論理データ・グループのサイズ。
	header	論理データ・グループが始まることを示す。
	db_event	論理データ・グループの名前。
	4	このモニター・エレメントに入っているデータのサイズ。
	u32bit	モニター・エレメント・タイプ - 符号なし 32 ビット数値。
	lock_waits	収集されたモニター・エレメントの名前。
	2	このエレメントに対して収集された値。

event_log_stream_header は、データを戻したデータベース・マネージャーのバージョンを示します。データを自己記述形式で書き込むのは、バージョン 6、バージョン 7、およびバージョン 8 のモニターだけです。これらのバージョンのいずれかのモニターで作業している場合、自己記述型データ・ストリームの処理を開始できます。スナップショット・モニターと違って、イベント・モニターにはトレースの合計サイズを戻す *size* エレメントがありません。 event_log_stream_header に示された数値は、逆方向互換性のために示されたダミー値です。イベント・トレースの合計サイズは、 event_log_stream_header が書き込まれるときには不明です。通常は、ファイルまたはパイプの終わりに達するまで、イベント・モニター・トレースを読み取ることになります。

ログ・ヘッダーではトレースの特性を記述します。これには、トレースが収集されたサーバーのメモリー・モデル (例えばリトル・エンディアン)、およびデータベー

スのコード・ページなどの情報が含まれています。トレースを読み取るシステムのメモリー・モデルがサーバーとは異なる場合 (例えば、Windows® 2000 システム上の UNIX® サーバーからトレースを読み取る場合)、数値に関してバイトのスワッピングを行う必要があります。データベースが、トレースを読み取るマシンとは異なる言語で構成されている場合、コード・ページ変換を行う必要が生じることもあります。トレースを読み取っているとき、 *size* エレメントを使用して、トレース内の論理データ・グループを読み飛ばせます。

関連概念:

- 66 ページの『イベント・モニターのファイル管理』
- 70 ページの『イベント・モニターの名前付きパイプ管理』
- 123 ページの『イベント・タイプの論理データ・グループへのマッピング』

関連タスク:

- 88 ページの『システム間でのイベント・モニター・データの転送』

関連資料:

- 75 ページの『イベント・モニターの出力例』

関連サンプル:

- 『bldevm -- Builds the event monitor program, evm, on AIX (C)』
- 『bldevm.bat -- Builds event monitor program, evm, on Windows』
- 『evm.c -- Process event monitor data on AIX (C)』
- 『evm.c -- Process event monitor data on Windows (C)』
- 『evmprint.c -- Prints all events generated by an event monitor on AIX (C)』
- 『evmprint.c -- Prints all events generated by an event monitor on Windows (C)』
- 『evmread.c -- Read the event monitor self describing data stream on AIX (C)』
- 『evmread.c -- Read the event monitor self describing data stream on Windows (C)』

システム間でのイベント・モニター・データの転送

数値を保管するための規則が異なるシステム間でイベント・モニター情報を転送するときには、変換を行う必要があります。 UNIX プラットフォーム上の情報はリトル・エンディアン・バイト・オーダーで保管され、 Windows プラットフォーム上の情報はビッグ・エンディアン・バイト・オーダーで保管されます。リトル・エンディアン・ソースからのイベント・モニター・データが、ビッグ・エンディアン・プラットフォーム上で読み取られる場合、またはその逆の場合には、バイト変換が必要です。

手順:

論理データ・グループ・ヘッダーおよびモニター・エレメント内の数値を変換するには、以下のロジックを使用します (C で表記)。

```
#include sqlmon.h
#define SWAP2(s) ((((s) >> 8) & 0xFF) | (((s) << 8) & 0xFF00))

#define SWAP4(l) ((((l) >> 24) & 0xFF) | ((((l) & 0xFF0000) >> 8) & 0xFF00) ¥
                    | (((l) & 0xFF00) << 8) | ((l) << 24))
```

```
#define SWAP8( where )                                                          ¥
 {                                                                              ¥
    sqluint32 temp;                                                             ¥
    temp = SWAP4(*(sqluint32 *) (where));                                       ¥
    * (sqluint32 *) (where) = SWAP4(* (((sqluint32 *) (where)) + 1));           ¥
    * (((sqluint32 *) (where)) + 1) = temp;                                     ¥
 }

int HeaderByteReverse( sqlm_header_info * pHeader)
{   int rc = 0;

    pHeader->size = SWAP4(pHeader->size);
    pHeader->type = SWAP2(pHeader->type);
    pHeader->element = SWAP2(pHeader->element);

    return rc;
}

int DataByteReverse( char * dataBuf, sqluint32 dataSize)
{   int rc = 0;

    sqlm_header_info * pElemHeader = NULL;
    char * pElemData = NULL;
    sqluint32 dataOffset = 0;
    sqluint32 elemDataSize  = 0;
    sqluint32 elemHeaderSize = sizeof( sqlm_header_info);

    // For each of the elements in the datas tream that are numeric,
    // perform byte reversal.

    while( dataOffset < dataSize)
    {   /* byte reverse the element header */
        pElemHeader = (sqlm_header_info *)
                      ( dataBuf + dataOffset);

        rc = HeaderByteReverse( pElemHeader);
        if( rc != 0) return rc;
        // Remember the element data's size...it will be byte reversed
        // before we skip to the next element.
        elemDataSize = pElemHeader->size;

        /* byte reverse the element data */
        pElemData = (char *)
                     ( dataBuf + dataOffset + elemHeaderSize);

        if(pElemHeader->type == SQLM_TYPE_HEADER)
        {   rc = DataByteReverse( pElemData, pElemHeader->size);
            if( rc != 0) return rc;
        }
        else
 {   switch( pElemHeader->type)
            {   case SQLM_TYPE_16BIT:
                case SQLM_TYPE_U16BIT:
                    *(sqluint16 *)(pElemData) =
                                    SWAP2(*(short *)(pElemData));
                    break;
                case SQLM_TYPE_32BIT:
                case SQLM_TYPE_U32BIT:
                    *(sqluint32 *)(pElemData) =
                                    SWAP4(*(sqluint32 *)(pElemData));
                    break;
                case SQLM_TYPE_64BIT:
                case SQLM_TYPE_U64BIT:
                    SWAP8(pElemData);
                    break;
                default:
```

```
                    // Not a numeric type. Do nothing.
                    break;
                }
            }
            dataOffset = dataOffset + elemHeaderSize + elemDataSize;
        }

        return 0;
} /*  end of DataByteReverse */
```

関連概念:

- 66 ページの『イベント・モニターのファイル管理』
- 70 ページの『イベント・モニターの名前付きパイプ管理』
- 86 ページの『イベント・モニター自己記述型データ・ストリーム』

第 2 部 システム・モニター・リファレンス

第 5 章 システム・モニターの論理データ・グループ

スナップショット・モニター・インターフェースの論理データ・グループへのマッピング

次の表では、スナップショット・モニター・データへのアクセスに使用できるすべての方法をリストしています。すべてのスナップショット・モニター・データは、モニター・エレメント内に保管され、論理データ・グループによってカテゴリー化されます。それぞれの API 要求タイプ、CLP コマンド、および SQL 表関数は、すべての論理データ・グループのサブセットからのモニター・データのみをキャプチャーします。

この表にリストされている、それぞれの API 要求タイプ、CLP コマンド、および SQL 表関数は、右端の列にリストされている論理データ・グループからのモニター・エレメントを戻します。

注:

1. 対応する SQL 表関数がない、いくつかの API 要求タイプおよび CLP コマンドがあります。その他の API 要求タイプおよび CLP コマンドの場合、それぞれの SQL 表関数は、関連した論理データ・グループのサブセットをキャプチャーします。
2. 一部のモニター・エレメントは、関連したモニター・スイッチが ON に設定されている場合にだけ戻されます。スイッチで必須エレメントを制御できるかどうか判別するには、個々のモニター・エレメントを参照してください。

表 11. スナップショット・モニター・インターフェースの論理データ・グループへのマッピング

API 要求タイプ	CLP コマンド	SQL 表関数	論理データ・グループ
SQLMA_APPLINFO_ALL	list applications [show detail]		appl_info
SQLMA_DBASE_APPLINFO	list applications for database *dbname* [show detail]		appl_info
SQLMA_DCS_APPLINFO_ALL	list dcs applications [show detail]		dcs_appl_info
SQLMA_DB2	get snapshot for dbm	SNAPSHOT_DBM	db2
		SNAPSHOT_FCM	fcm
		SNAPSHOT_FCMNODE	fcm_node
			memory_pool、utility_info、progress、progress_info
	get dbm monitor switches	SNAPSHOT_SWITCHES	switch_list
SQLMA_DBASE	get snapshot for database on *dbname*	SNAPSHOT_DATABASE	dbase
			rollforward、tablespace、memory_pool
SQLMA_DBASE_ALL	get snapshot for all databases	SNAPSHOT_DATABASE	dbase
			rollforward、tablespace、memory_pool
	list active databases		dbase

表 11. スナップショット・モニター・インターフェースの論理データ・グループへのマッピング (続き)

API 要求タイプ	CLP コマンド	SQL 表関数	論理データ・グループ
SQLMA_DCS_DBASE	get snapshot for dcs database on *dbname*		dcs_dbase、stmt_transmissions
SQLMA_DCS_DBASE_ALL	get snapshot for all databases		dcs_dbase、stmt_transmissions
SQLMA_DBASE_REMOTE	get snapshot for remote database on *dbname*		dbase_remote
SQLMA_DBASE_REMOTE_ALL	get snapshot for all remote databases		dbase_remote
SQLMA_APPL	get snapshot for application applid *appl-id*		appl、agent、appl_info、lock_wait、stmt、subsection、memory_pool
SQLMA_AGENT_ID	get snapshot for application agentid *appl-handle*		appl、agent、appl_info、lock_wait、stmt、subsection、memory_pool
SQLMA_DBASE_APPLS	get snapshot for applications on *dbname*	SNAPSHOT_APPL	appl、agent
		SNAPSHOT_APPL_INFO	appl_info
		SNAPSHOT_LOCKWAIT	appl、lock_wait
		SNAPSHOT_STATEMENT	appl、stmt
		SNAPSHOT_AGENT	appl、agent、stmt
		SNAPSHOT_SUBSECT	appl、subsection、stmt
			memory_pool
SQLMA_APPL_ALL	get snapshot for all applications	SNAPSHOT_APPL	appl、agent
		SNAPSHOT_APPL_INFO	appl_info
		SNAPSHOT_LOCKWAIT	appl、lock_wait
		SNAPSHOT_STATEMENT	appl、stmt
		SNAPSHOT_AGENT	appl、agent、stmt
		SNAPSHOT_SUBSECT	appl、subsection、stmt
			memory_pool
SQLMA_DCS_APPL	get snapshot for dcs application applid *appl-id*		dcs_appl、dcs_stmt、dcs_appl_info、stmt_transmissions
SQLMA_DCS_APPL_ALL	get snapshot for all dcs applications		dcs_appl、dcs_stmt、dcs_appl_info、stmt_transmissions
SQLMA_DCS_APPL_HANDLE	get snapshot for dcs application agentid *appl-handle*		dcs_appl、dcs_stmt、dcs_appl_info、stmt_transmissions
SQLMA_DCS_DBASE_APPLS	get snapshot for dcs applications on *dbname*		dcs_appl、dcs_stmt、dcs_appl_info、stmt_transmissions
SQLMA_DBASE_APPLS_REMOTE	get snapshot for remote applications on *dbname*		dbase_appl
SQLMA_APPL_REMOTE_ALL	get snapshot for all remote applications		dbase_appl
SQLMA_DBASE_TABLES	get snapshot for tables on *dbname*	SNAPSHOT_TABLE	table
		SNAPSHOT_TBREORG	table_reorg
			table_list
SQLMA_APPL_LOCKS	get snapshot for locks for application applid *appl-id*		appl_lock_list、lock_wait, lock

表 11. スナップショット・モニター・インターフェースの論理データ・グループへのマッピング (続き)

API 要求タイプ	CLP コマンド	SQL 表関数	論理データ・グループ
SQLMA_APPL_LOCKS_AGENT_ID	get snapshot for locks for application agentid *appl-handle*		appl_lock_list、lock_wait、lock
SQLMA_DBASE_LOCKS	get snapshot for locks on *dbname*	SNAPSHOT_LOCK	appl_lock_list、lock
			db_lock_list、lock_wait
SQLMA_DBASE_TABLESPACES	get snapshot for tablespaces on *dbname*	SNAPSHOT_TBS	tablespace
		SNAPSHOT_TBS_CFG	tablespace、tablespace_nodeinfo
		SNAPSHOT_QUIESCERS	tablespace_quiescer、tablespace_nodeinfo
		SNAPSHOT_CONTAINER	tablespace_container、tablespace_nodeinfo
		SNAPSHOT_RANGES	tablespace_ranges、tablespace_nodeinfo
			tablespace_list、tablespace_nodeinfo
SQLMA_BUFFERPOOLS_ALL	get snapshot for all bufferpools	SNAPSHOT_BP	bufferpool
SQLMA_DBASE_BUFFERPOOLS	get snapshot for bufferpools on *dbname*	SNAPSHOT_BP	bufferpool
SQLMA_DYNAMIC_SQL	get snapshot for dynamic sql on *dbname*	SNAPSHOT_DYN_SQL	dynsql
			dynsql_list

以下の図は、論理データ・グループがスナップショット・データ・ストリーム内に現れる順番を示しています。

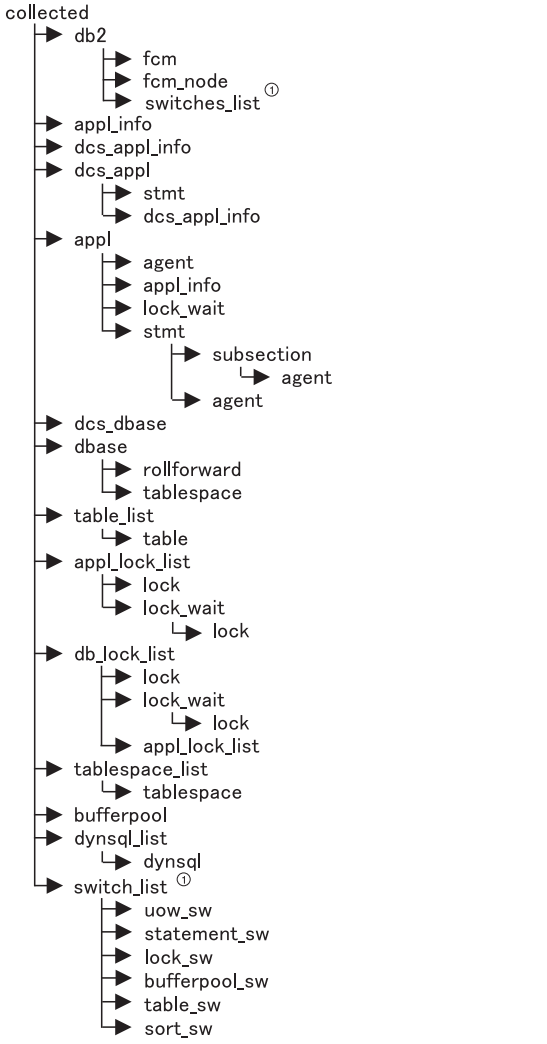

図 4. データ・ストリーム階層

注: 論理データ・グループの一部として、時間が戻されることがあります。

スナップショット・モニターの論理データ・グループおよびモニター・エレメント

次の表は、論理データ・グループと、スナップショット・モニターによって戻されるモニター・エレメントの一覧表です。

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント

スナップショット 論理データ・グループ	モニター・エレメント
agent	179 ページの『agent_pid プロセスまたはスレッド ID』
	313 ページの『lock_timeout_val ロック・タイムアウト』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
appl	359 ページの『acc_curs_blk 受け入れられたブロック・カーソル要求』
	397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
	396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
	191 ページの『agents_stolen スチールされたエージェント』
	173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
	178 ページの『appl_idle_time アプリケーション・アイドル時間』
	170 ページの『appl_priority アプリケーション・エージェント優先順位』
	171 ページの『appl_priority_type アプリケーション優先順位タイプ』
	191 ページの『associated_agents_top 関連エージェント最大数』
	372 ページの『binds_precompiles 試行されたバインド/プリコンパイル』
	264 ページの『cat_cache_inserts カタログ・キャッシュ挿入数』
	263 ページの『cat_cache_lookups カタログ・キャッシュ参照数』
	265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』
	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	174 ページの『conn_complete_time 接続要求完了タイム・スタンプ』
	367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』
	295 ページの『deadlocks デッドロック検出数』
	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
	209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』
	435 ページの『inbound_comm_address インバウンド通信アドレス』
	368 ページの『int_auto_rebinds 内部自動再バインド』
	368 ページの『int_commits 内部コミット数』
	371 ページの『int_deadlock_rollbacks デッドロックによる内部ロールバック』
	369 ページの『int_rollbacks 内部ロールバック数』
	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』
	304 ページの『lock_escalation ロック・エスカレーション』
	302 ページの『lock_timeouts ロック・タイムアウト数』
	313 ページの『lock_timeout_val ロック・タイムアウト』
	311 ページの『lock_wait_time ロック待機中の時間』
	310 ページの『lock_waits ロック待機数』
	294 ページの『locks_held ロック保持数』
	312 ページの『locks_waiting ロックで待機中の現行エージェント』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
appl (続き)	394 ページの『num_agents ステートメントで作動しているエージェントの数』
	359 ページの『open_loc_curs 開かれているローカル・カーソル』
	360 ページの『open_loc_curs_blk 開かれているローカル・ブロック・カーソル』
	357 ページの『open_rem_curs 開かれているリモート・カーソル』
	357 ページの『open_rem_curs_blk 開かれているリモート・ブロック・カーソル』
	269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
	267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	249 ページの『prefetch_wait_time プリフェッチ待ち時間』
	175 ページの『prev_uow_stop_time 直前の作業単位完了タイム・スタンプ』
	276 ページの『priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数』
	278 ページの『priv_workspace_section_inserts 専用ワークスペース・セクション挿入』
	277 ページの『priv_workspace_section_lookups 専用ワークスペース・セクションの参照』
	276 ページの『priv_workspace_size_top 専用ワークスペースの最大サイズ』
	358 ページの『rej_curs_blk リジェクトされたブロック・カーソル要求』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	342 ページの『rows_deleted 削除行数』
	342 ページの『rows_inserted 挿入行数』
	345 ページの『rows_read 読み取り行数』
	343 ページの『rows_selected 選択行数』
	343 ページの『rows_updated 更新行数』
	344 ページの『rows_written 書き込み行数』
	365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』
	274 ページの『shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数』
	275 ページの『shr_workspace_section_inserts 共有ワークスペース・セクション挿入数』
	274 ページの『shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数』
	273 ページの『shr_workspace_size_top 共有ワークスペースの最大サイズ』
	203 ページの『sort_overflows ソート・オーバーフロー』
	371 ページの『sql_reqs_since_commit 最終コミット後の SQL 要求』
	361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
appl (続き)	207 ページの『total_hash_joins ハッシュ結合の合計』
	208 ページの『total_hash_loops ハッシュ・ループの合計』
	202 ページの『total_sort_time ソート時間合計』
	201 ページの『total_sorts ソート合計』
	366 ページの『uid_sql_stmts 実行された更新/挿入/削除 SQL ステートメント』
	249 ページの『unread_prefetch_pages 読み取り不能プリフェッチ・ページ』
	177 ページの『uow_comp_status 作業単位完了状況』
	177 ページの『uow_elapsed_time 最新の作業単位の経過時間』
	313 ページの『uow_lock_wait_time ロック待機中の作業単位の合計時間』
	283 ページの『uow_log_space_used 作業単位ログ・スペース』
	175 ページの『uow_start_time 作業単位開始タイム・スタンプ』
	176 ページの『uow_stop_time 作業単位停止タイム・スタンプ』
	297 ページの『x_lock_escals 排他ロック・エスカレーション数』
appl_id_info	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	158 ページの『appl_name アプリケーション名』
	154 ページの『appl_status アプリケーション状況』
	162 ページの『auth_id 許可 ID』
	164 ページの『client_db_alias アプリケーションで使用するデータベース別名』
	163 ページの『client_nname クライアントの構成 NNAME』
	164 ページの『client_prdid クライアント製品/バージョン ID』
	156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	404 ページの『input_db_alias 入力データベース別名』
	162 ページの『sequence_no シーケンス番号』
	163 ページの『session_auth_id セッション許可 ID』
	157 ページの『status_change_time アプリケーション状況変更時刻』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
appl_info	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	158 ページの『appl_name アプリケーション名』
	272 ページの『appl_section_inserts セクション挿入数 : モニター・エレメント』
	271 ページの『appl_section_lookups セクションの参照回数 : モニター・エレメント』
	154 ページの『appl_status アプリケーション状況』
	162 ページの『auth_id 許可 ID』
	162 ページの『auth_id 許可 ID』
	164 ページの『client_db_alias アプリケーションで使用するデータベース別名』
	163 ページの『client_nname クライアントの構成 NNAME』
	168 ページの『client_pid クライアント・プロセス ID』
	168 ページの『client_platform クライアント・オペレーティング・プラットフォーム』
	164 ページの『client_prdid クライアント製品/バージョン ID』
	169 ページの『client_protocol クライアント通信プロトコル』
	156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
	179 ページの『coord_agent_pid コーディネーター・エージェント』
	173 ページの『coord_node コーディネーター・ノード』
	167 ページの『corr_token DRDA 相関トークン』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	167 ページの『execution_id ユーザー・ログイン ID』
	404 ページの『input_db_alias 入力データベース別名』
	193 ページの『num_assoc_agents 関連したエージェント数』
	162 ページの『sequence_no シーケンス番号』
	157 ページの『status_change_time アプリケーション状況変更時刻』
	169 ページの『territory_code データベース・テリトリー・コード』
	463 ページの『tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング』
	462 ページの『tpmon_client_app TP モニター・クライアント・アプリケーション名』
	461 ページの『tpmon_client_userid TP モニター・クライアント・ユーザー ID』
	462 ページの『tpmon_client_wkstn TP モニター・クライアント・ワークステーション名』
appl_lock_list	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	158 ページの『appl_name アプリケーション名』
	154 ページの『appl_status アプリケーション状況』
	162 ページの『auth_id 許可 ID』
	164 ページの『client_db_alias アプリケーションで使用するデータベース別名』
	156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
	294 ページの『locks_held ロック保持数』
	312 ページの『locks_waiting ロックで待機中の現行エージェント』
	311 ページの『lock_wait_time ロック待機中の時間』
	162 ページの『sequence_no シーケンス番号』
	163 ページの『session_auth_id セッション許可 ID』
	157 ページの『status_change_time アプリケーション状況変更時刻』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
appl_remote	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	466 ページの『create_nickname ニックネーム作成回数』
	471 ページの『create_nickname_time ニックネーム作成応答時間』
	464 ページの『datasource_name データ・ソース名』
	146 ページの『db_name データベース名』
	466 ページの『delete_sql_stmts 削除回数』
	471 ページの『delete_time 削除応答時間』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	465 ページの『insert_sql_stmts 挿入回数』
	469 ページの『insert_time 挿入応答時間』
	472 ページの『passthru_time パススルー時間』
	467 ページの『passthrus パススルー数』
	473 ページの『remote_lock_time リモート・ロック時間』
	468 ページの『remote_locks リモート・ロック数』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	342 ページの『rows_deleted 削除行数』
	342 ページの『rows_inserted 挿入行数』
	343 ページの『rows_selected 選択行数』
	343 ページの『rows_updated 更新行数』
	365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』
	469 ページの『select_time 照会応答時間』
	468 ページの『sp_rows_selected ストアード・プロシージャーによって戻された行数』
	472 ページの『stored_proc_time ストアード・プロシージャー時間』
	467 ページの『stored_procs ストアード・プロシージャー数』
	465 ページの『update_sql_stmts 更新回数』
	470 ページの『update_time 更新応答時間』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
bufferpool	250 ページの『block_ios ブロック入出力要求数』
	248 ページの『bp_name バッファー・プール名』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	260 ページの『direct_read_reqs 直接読み取り要求』
	258 ページの『direct_reads データベースからの直接読み取り』
	261 ページの『direct_read_time 直接読み取り時間』
	260 ページの『direct_write_reqs 直接書き込み要求』
	259 ページの『direct_writes データベースへの直接書き込み』
	262 ページの『direct_write_time 直接書き込み時間』
	236 ページの『files_closed 閉じられたデータベース・ファイル』
	404 ページの『input_db_alias 入力データベース別名』
	252 ページの『pages_from_block_ios ブロック入出力によって読み取られたページ数の合計』
	250 ページの『pages_from_vectored_ios ベクトル化入出力によって読み取られたページ数の合計』
	252 ページの『physical_page_maps 物理ページ・マップ数』
	237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
	238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
	243 ページの『pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求』
	240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
	239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
	241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
	242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
	243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	246 ページの『pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	249 ページの『vectored_ios ベクトル化入出力要求数』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
bufferpool_nodeinfo	257 ページの『bp_cur_buffsz バッファー・プールの現行サイズ』
	257 ページの『bp_new_buffsz 新規バッファー・プール・サイズ』
	257 ページの『bp_pages_left_to_remove 除去残ページ数』
	257 ページの『bp_tbsp_use_count バッファー・プールにマップされている表スペースの数』
	172 ページの『node_number ノード番号』
collected	172 ページの『node_number ノード番号』
	142 ページの『server_db2_type モニター対象 (サーバー) ノードのデータベース・マネージャーのタイプ』
	141 ページの『server_instance_name サーバー・インスタンス名』
	141 ページの『server_nname モニター対象の (サーバー) データベース・パーティションでの NNAME 構成』
	143 ページの『server_prdid サーバー製品/バージョン ID』
	143 ページの『server_version サーバー・バージョン』
	switch_list モニター・スイッチ制御データ
	405 ページの『time_stamp スナップショット時刻』
	146 ページの『time_zone_disp 時間帯変位』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
db2	190 ページの『agents_created_empty_pool - Agents Created Due to Empty Agent Pool』
	189 ページの『agents_from_pool プールから割り当てられたエージェント』
	187 ページの『agents_registered 登録済みエージェント』
	188 ページの『agents_registered_top エージェント最大登録数』
	191 ページの『agents_stolen スチールされたエージェント』
	187 ページの『agents_waiting_on_token トークン待ちエージェント』
	188 ページの『agents_waiting_top エージェント最大待機数』
	192 ページの『comm_private_mem コミット済み専用メモリー』
	185 ページの『con_local_dbases 現行接続を持つローカル・データベース』
	190 ページの『coord_agents_top コーディネーター・エージェント最大数』
	140 ページの『db2start_time データベース・マネージャー開始タイム・スタンプ』
	149 ページの『db_status データベース状況』
	428 ページの『gw_total_cons DB2 Connect の接続試行合計回数』
	428 ページの『gw_cur_cons DB2 Connect の現在の接続数』
	429 ページの『gw_cons_wait_host ホストの応答を待機している接続の数』
	429 ページの『gw_cons_wait_client クライアントの要求送信を待機している接続の数』
	189 ページの『idle_agents アイドル・エージェント数』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』
	183 ページの『local_cons ローカル接続』
	184 ページの『local_cons_in_exec データベース・マネージャーで実行中のローカル接続』
	193 ページの『max_agent_overflows 最大エージェント・オーバーフロー回数』
	194 ページの『num_gw_conn_switches - 接続切り替え回数』
	405 ページの『num_nodes_in_db2_instance パーティション内のノード数』
	200 ページの『piped_sorts_requested 要求されたパイプ・ソート数』
	201 ページの『piped_sorts_accepted 受け入れられたパイプ・ソート』
	207 ページの『post_threshold_hash_joins ハッシュ結合のしきい値』
	199 ページの『post_threshold_sorts ポストしきい値ソート』
	145 ページの『product_name 製品名』
	182 ページの『rem_cons_in データベース・マネージャーへのリモート接続』
	183 ページの『rem_cons_in_exec データベース・マネージャーで実行中のリモート接続』
	144 ページの『service_level サービス・レベル』
	158 ページの『smallest_log_avail_node 使用可能なログ・スペースが最小のノード』
	198 ページの『sort_heap_allocated 割り振られたソート・ヒープの合計』
	205 ページの『sort_heap_top ソート専用ヒープの最高水準点』
	switch_list モニター・スイッチ制御データ
db_lock_list	186 ページの『appls_cur_cons 現在接続されているアプリケーション』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	404 ページの『input_db_alias 入力データベース別名』
	294 ページの『locks_held ロック保持数』
	312 ページの『locks_waiting ロックで待機中の現行エージェント』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dbase	204 ページの『active_sorts アクティブ・ソート』
	394 ページの『agents_top 作成されたエージェントの数』
	157 ページの『appl_id_oldest_xact 最も古いトランザクションを持つアプリケーション』
	272 ページの『appl_section_inserts セクション挿入数 : モニター・エレメント』
	271 ページの『appl_section_lookups セクションの参照回数 : モニター・エレメント』
	186 ページの『appls_cur_cons 現在接続されているアプリケーション』
	186 ページの『appls_in_db2 データベースで現在実行中のアプリケーション』
	372 ページの『binds_precompiles 試行されたバインド/プリコンパイル』
	264 ページの『cat_cache_inserts カタログ・キャッシュ挿入数』
	263 ページの『cat_cache_lookups カタログ・キャッシュ参照数』
	265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』
	266 ページの『cat_cache_size_top カタログ・キャッシュ最高水準点』
	151 ページの『catalog_node カタログ・ノード番号』
	150 ページの『catalog_node_name カタログ・ノード・ネットワーク名』
	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	174 ページの『connections_top 同時接続の最大数』
	190 ページの『coord_agents_top コーディネーター・エージェント最大数』
	148 ページの『db_conn_time データベース活動化タイム・スタンプ』
	279 ページの『db_heap_top 割り振られた最大データベース・ヒープ』
	150 ページの『db_location データベース・ロケーション』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	149 ページの『db_status データベース状況』
	367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』
	295 ページの『deadlocks デッドロック検出数』
	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	236 ページの『files_closed 閉じられたデータベース・ファイル』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dbase (続き)	208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
	209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』
	404 ページの『input_db_alias 入力データベース別名』
	368 ページの『int_auto_rebinds 内部自動再バインド』
	368 ページの『int_commits 内部コミット数』
	371 ページの『int_deadlock_rollbacks デッドロックによる内部ロールバック』
	369 ページの『int_rollbacks 内部ロールバック数』
	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	151 ページの『last_backup 最終バックアップ・タイム・スタンプ』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』
	296 ページの『lock_escals ロック・エスカレーション数』
	295 ページの『lock_list_in_use 使用中のロック・リスト・メモリーの合計』
	302 ページの『lock_timeouts ロック・タイムアウト数』
	311 ページの『lock_wait_time ロック待機中の時間』
	310 ページの『lock_waits ロック待機数』
	294 ページの『locks_held ロック保持数』
	312 ページの『locks_waiting ロックで待機中の現行エージェント』
	285 ページの『log_held_by_dirty_pages ダーティー・ページ別に計算されるログ・スペースの量』
	287 ページの『log_read_time ログ読み取り時間』
	282 ページの『log_reads 読み取られたログ・ページの数』
	286 ページの『log_to_redo_for_recovery リカバリーの場合に再実行されるログの量』
	287 ページの『log_write_time ログ書き込み時間』
	283 ページの『log_writes 書き込まれたログ・ページの数』
	193 ページの『num_assoc_agents 関連したエージェント数』
	310 ページの『num_indoubt_trans 未確定トランザクション数』
	289 ページの『num_log_buffer_full フル・ログ・バッファーの回数』
	290 ページの『num_log_data_found_in_buffer ログ・データがバッファーにある回数』
	289 ページの『num_log_part_page_io 部分ログ・ページ書き込み数』
	288 ページの『num_log_read_io ログ読み取り数』
	288 ページの『num_log_write_io ログ書き込み数』
	269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
	267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』
	269 ページの『pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー』
	270 ページの『pkg_cache_size_top パッケージ・キャッシュの最高水準点』
	243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
	237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
	238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
	243 ページの『pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求』
	240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
	239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
	241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
	242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dbase (続き)	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
	247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
	246 ページの『pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	276 ページの『priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数』
	278 ページの『priv_workspace_section_inserts 専用ワークスペース・セクション挿入』
	277 ページの『priv_workspace_section_lookups 専用ワークスペース・セクションの参照』
	276 ページの『priv_workspace_size_top 専用ワークスペースの最大サイズ』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	342 ページの『rows_deleted 削除行数』
	342 ページの『rows_inserted 挿入行数』
	345 ページの『rows_read 読み取り行数』
	343 ページの『rows_selected 選択行数』
	343 ページの『rows_updated 更新行数』
	280 ページの『sec_log_used_top 使用された最大 2 次ログ・スペース』
	282 ページの『sec_logs_allocated 現在割り振られている 2 次ログ』
	365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』
	144 ページの『server_platform サーバーのオペレーティング・システム』
	274 ページの『shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数』
	275 ページの『shr_workspace_section_inserts 共有ワークスペース・セクション挿入数』
	274 ページの『shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数』
	273 ページの『shr_workspace_size_top 共有ワークスペースの最大サイズ』
	198 ページの『sort_heap_allocated 割り振られたソート・ヒープの合計』
	203 ページの『sort_overflows ソート・オーバーフロー』
	205 ページの『sort_shrheap_allocated 現在割り振られているソート共有ヒープ』
	206 ページの『sort_shrheap_top ソート共有ヒープの最高水準点』
	361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dbase (続き)	281 ページの『tot_log_used_top 使用された最大合計ログ・スペース』
	185 ページの『total_cons データベース活動化以降の接続』
	207 ページの『total_hash_joins ハッシュ結合の合計』
	208 ページの『total_hash_loops ハッシュ・ループの合計』
	284 ページの『total_log_available 使用可能なログの合計』
	284 ページの『total_log_used 使用されているログ・スペースの合計』
	192 ページの『total_sec_cons 2 次接続』
	202 ページの『total_sort_time ソート時間合計』
	201 ページの『total_sorts ソート合計』
	366 ページの『uid_sql_stmts 実行された更新/挿入/削除 SQL ステートメント』
	249 ページの『unread_prefetch_pages 読み取り不能プリフェッチ・ページ』
	297 ページの『x_lock_escals 排他ロック・エスカレーション数』
dbase_remote	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	466 ページの『create_nickname ニックネーム作成回数』
	471 ページの『create_nickname_time ニックネーム作成応答時間』
	464 ページの『datasource_name データ・ソース名』
	146 ページの『db_name データベース名』
	466 ページの『delete_sql_stmts 削除回数』
	471 ページの『delete_time 削除応答時間』
	464 ページの『disconnects 切断回数』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	465 ページの『insert_sql_stmts 挿入回数』
	469 ページの『insert_time 挿入応答時間』
	472 ページの『passthru_time パススルー時間』
	467 ページの『passthrus パススルー数』
	473 ページの『remote_lock_time リモート・ロック時間』
	468 ページの『remote_locks リモート・ロック数』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	342 ページの『rows_deleted 削除行数』
	342 ページの『rows_inserted 挿入行数』
	343 ページの『rows_selected 選択行数』
	343 ページの『rows_updated 更新行数』
	365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』
	469 ページの『select_time 照会応答時間』
	468 ページの『sp_rows_selected ストアード・プロシージャーによって戻された行数』
	472 ページの『stored_proc_time ストアード・プロシージャー時間』
	467 ページの『stored_procs ストアード・プロシージャー数』
	185 ページの『total_cons データベース活動化以降の接続』
	465 ページの『update_sql_stmts 更新回数』
	470 ページの『update_time 更新応答時間』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dcs_appl	178 ページの『appl_idle_time アプリケーション・アイドル時間』
	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	456 ページの『elapsed_exec_time ステートメント実行経過時間』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	427 ページの『gw_con_time DB2 Connect ゲートウェイの最初の接続開始』
	429 ページの『gw_exec_time DB2 Connect ゲートウェイ処理の経過時間』
	457 ページの『host_response_time ホスト応答時間』
	435 ページの『inbound_bytes_received 受信されたインバウンド・バイト数』
	437 ページの『inbound_bytes_sent 送信されたインバウンド・バイト数』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』
	440 ページの『max_data_received_128 受信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数』
	441 ページの『max_data_received_256 受信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数』
	442 ページの『max_data_received_512 受信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数』
	443 ページの『max_data_received_1024 受信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数』
	444 ページの『max_data_received_2048 受信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数』
	445 ページの『max_data_received_4096 受信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数』
	446 ページの『max_data_received_8192 受信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数』
	447 ページの『max_data_received_16384 受信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数』
	448 ページの『max_data_received_31999 受信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数』
	449 ページの『max_data_received_64000 受信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数』
	450 ページの『max_data_received_gt64000 受信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dcs_appl (続き)	440 ページの『max_data_sent_128 送信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数』
	441 ページの『max_data_sent_256 送信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数』
	442 ページの『max_data_sent_512 送信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数』
	443 ページの『max_data_sent_1024 送信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数』
	444 ページの『max_data_sent_2048 送信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数』
	445 ページの『max_data_sent_4096 送信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数』
	446 ページの『max_data_sent_8192 送信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数』
	447 ページの『max_data_sent_16384 送信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数』
	448 ページの『max_data_sent_31999 送信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数』
	449 ページの『max_data_sent_64000 送信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数』
	450 ページの『max_data_sent_gt64000 送信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数』
	451 ページの『max_network_time_1_ms ネットワーク時間が 1 ミリ秒以下のステートメント数』
	451 ページの『max_network_time_4_ms ネットワーク時間が 1 から 4 ミリ秒のステートメント数』
	452 ページの『max_network_time_16_ms ネットワーク時間が 4 から 16 ミリ秒のステートメント数』
	452 ページの『max_network_time_100_ms ネットワーク時間が 16 から 100 ミリ秒のステートメント数』
	453 ページの『max_network_time_500_ms ネットワーク時間が 100 から 500 ミリ秒のステートメント数』
	453 ページの『max_network_time_gt500_ms ネットワーク時間が 500 ミリ秒を超えるステートメント数』
	454 ページの『network_time_top ステートメントの最大ネットワーク時間』
	455 ページの『network_time_bottom ステートメントの最小ネットワーク時間』
	432 ページの『open_cursors オープン・カーソル数』
	436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』
	436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
	175 ページの『prev_uow_stop_time 直前の作業単位完了タイム・スタンプ』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	343 ページの『rows_selected 選択行数』
	430 ページの『sql_stmts 試行された SQL ステートメントの数』
	463 ページの『tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング』
	462 ページの『tpmon_client_app TP モニター・クライアント・アプリケーション名』
	461 ページの『tpmon_client_userid TP モニター・クライアント・ユーザー ID』
	462 ページの『tpmon_client_wkstn TP モニター・クライアント・ワークステーション名』
	177 ページの『uow_comp_status 作業単位完了状況』
	177 ページの『uow_elapsed_time 最新の作業単位の経過時間』
	175 ページの『uow_start_time 作業単位開始タイム・スタンプ』
	176 ページの『uow_stop_time 作業単位停止タイム・スタンプ』
	455 ページの『xid トランザクション ID』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dcs_appl_info	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	433 ページの『agent_status DCS アプリケーション・エージェント』
	159 ページの『appl_id アプリケーション ID』
	158 ページの『appl_name アプリケーション名』
	162 ページの『auth_id 許可 ID』
	163 ページの『client_nname クライアントの構成 NNAME』
	168 ページの『client_pid クライアント・プロセス ID』
	168 ページの『client_platform クライアント・オペレーティング・プラットフォーム』
	164 ページの『client_prdid クライアント製品/バージョン ID』
	169 ページの『client_protocol クライアント通信プロトコル』
	156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
	432 ページの『dcs_appl_status DCS アプリケーション状況』
	426 ページの『dcs_db_name DCS データベース名』
	167 ページの『execution_id ユーザー・ログイン ID』
	427 ページの『gw_db_alias ゲートウェイでのデータベース別名』
	433 ページの『host_ccsid ホスト・コード化文字セット ID』
	426 ページの『host_db_name ホスト・データベース名』
	165 ページの『host_prdid ホスト製品/バージョン ID』
	435 ページの『inbound_comm_address インバウンド通信アドレス』
	166 ページの『outbound_appl_id アウトバウンド・アプリケーション ID』
	434 ページの『outbound_comm_address アウトバウンド通信アドレス』
	434 ページの『outbound_comm_protocol アウトバウンド通信プロトコル』
	166 ページの『outbound_sequence_no アウトバウンド・シーケンス番号』
	162 ページの『sequence_no シーケンス番号』
	157 ページの『status_change_time アプリケーション状況変更時刻』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dcs_dbase	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	459 ページの『con_elapsed_time 最新の接続経過時間』
	458 ページの『con_response_time 接続の最新応答時間』
	426 ページの『dcs_db_name DCS データベース名』
	456 ページの『elapsed_exec_time ステートメント実行経過時間』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	460 ページの『gw_comm_error_time 通信エラー時刻』
	459 ページの『gw_comm_errors 通信エラー』
	427 ページの『gw_con_time DB2 Connect ゲートウェイの最初の接続開始』
	427 ページの『gw_connections_top ホスト・データベースへの同時接続の最大数』
	429 ページの『gw_cons_wait_client クライアントの要求送信を待機している接続の数』
	429 ページの『gw_cons_wait_host ホストの応答を待機している接続の数』
	428 ページの『gw_cur_cons DB2 Connect の現在の接続数』
	428 ページの『gw_total_cons DB2 Connect の接続試行合計回数』
	426 ページの『host_db_name ホスト・データベース名』
	457 ページの『host_response_time ホスト応答時間』
	435 ページの『inbound_bytes_received 受信されたインバウンド・バイト数』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』
	440 ページの『max_data_received_128 受信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数』
	441 ページの『max_data_received_256 受信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数』
	442 ページの『max_data_received_512 受信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数』
	443 ページの『max_data_received_1024 受信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数』
	444 ページの『max_data_received_2048 受信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数』
	445 ページの『max_data_received_4096 受信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数』
	446 ページの『max_data_received_8192 受信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数』
	447 ページの『max_data_received_16384 受信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数』
	448 ページの『max_data_received_31999 受信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数』
	449 ページの『max_data_received_64000 受信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数』
	450 ページの『max_data_received_gt64000 受信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
dcs_dbase (続き)	440 ページの『max_data_sent_128 送信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数』
	441 ページの『max_data_sent_256 送信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数』
	442 ページの『max_data_sent_512 送信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数』
	443 ページの『max_data_sent_1024 送信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数』
	444 ページの『max_data_sent_2048 送信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数』
	445 ページの『max_data_sent_4096 送信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数』
	446 ページの『max_data_sent_8192 送信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数』
	447 ページの『max_data_sent_16384 送信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数』
	448 ページの『max_data_sent_31999 送信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数』
	449 ページの『max_data_sent_64000 送信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数』
	450 ページの『max_data_sent_gt64000 送信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数』
	451 ページの『max_network_time_1_ms ネットワーク時間が 1 ミリ秒以下のステートメント数』
	451 ページの『max_network_time_4_ms ネットワーク時間が 1 から 4 ミリ秒のステートメント数』
	452 ページの『max_network_time_16_ms ネットワーク時間が 4 から 16 ミリ秒のステートメント数』
	452 ページの『max_network_time_100_ms ネットワーク時間が 16 から 100 ミリ秒のステートメント数』
	453 ページの『max_network_time_500_ms ネットワーク時間が 100 から 500 ミリ秒のステートメント数』
	453 ページの『max_network_time_gt500_ms ネットワーク時間が 500 ミリ秒を超えるステートメント数』
	455 ページの『network_time_bottom ステートメントの最小ネットワーク時間』
	454 ページの『network_time_top ステートメントの最大ネットワーク時間』
	436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	343 ページの『rows_selected 選択行数』
	430 ページの『sql_stmts 試行された SQL ステートメントの数』
dcs_stmt	460 ページの『blocking_cursor ブロッキング・カーソル』
	378 ページの『creator アプリケーション作成者』
	456 ページの『elapsed_exec_time ステートメント実行経過時間』
	383 ページの『fetch_count 成功した取り出しの数』
	429 ページの『gw_exec_time DB2 Connect ゲートウェイ処理の経過時間』
	457 ページの『host_response_time ホスト応答時間』
	435 ページの『inbound_bytes_received 受信されたインバウンド・バイト数』
	437 ページの『inbound_bytes_sent 送信されたインバウンド・バイト数』
	458 ページの『num_transmissions_group 伝送グループの回数』
	457 ページの『num_transmissions 伝送回数』
	436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』
	436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
	375 ページの『package_name パッケージ名』
	384 ページの『query_card_estimate 行数見積もりの照会』
	385 ページの『query_cost_estimate 照会コストの見積もり』
	377 ページの『section_number セクション番号』
	381 ページの『stmt_elapsed_time 最新のステートメント経過時間』
	374 ページの『stmt_operation/operation ステートメント操作』
	379 ページの『stmt_start ステートメント操作開始タイム・スタンプ』
	379 ページの『stmt_stop ステートメント操作停止タイム・スタンプ』
	381 ページの『stmt_text SQL 動的ステートメント・テキスト』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
detail_log	292 ページの『current_active_log 現行アクティブ・ログ・ファイル番号』
	292 ページの『current_archive_log 現行アーカイブ・ログ・ファイル番号』
	290 ページの『first_active_log 先頭アクティブ・ログ・ファイル番号』
	291 ページの『last_active_log 最終アクティブ・ログ・ファイル番号』
	172 ページの『node_number ノード番号』
dynsql	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	392 ページの『num_compilations ステートメント・コンパイル数』
	392 ページの『num_executions ステートメント実行回数』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	393 ページの『prep_time_best ステートメント最短準備時間』
	393 ページの『prep_time_worst ステートメント最長準備時間』
	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	203 ページの『sort_overflows ソート・オーバーフロー』
	382 ページの『stmt_sorts ステートメント・ソート回数』
	381 ページの『stmt_text SQL 動的ステートメント・テキスト』
	393 ページの『total_exec_time ステートメント実行の経過時間』
	202 ページの『total_sort_time ソート時間合計』
	403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
	403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』
dynsql_list	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
fcm	210 ページの『buff_free 現在空いている FCM バッファー』
	210 ページの『buff_free_bottom 空き FCM バッファーの最小数』
	211 ページの『ce_free 現在空いている接続項目』
	212 ページの『ce_free_bottom 接続項目の最小数』
	211 ページの『ma_free 現在空いているメッセージ・アンカー』
	211 ページの『ma_free_bottom メッセージ・アンカーの最小数』
	172 ページの『node_number ノード番号』
	212 ページの『rb_free 現在空いている要求ブロック』
	212 ページの『rb_free_bottom 要求ブロックの最小数』
fcm_node	213 ページの『connection_status 接続状況』
	172 ページの『node_number ノード番号』
	213 ページの『total_buffers_sent 送信された FCM バッファーの合計』
	214 ページの『total_buffers_rcvd 受信された FCM バッファーの合計』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
hadr	414 ページの『hadr_connect_status HADR 接続状況 : モニター・エレメント』
	415 ページの『hadr_connect_time HADR 接続時刻 : モニター・エレメント』
	416 ページの『hadr_heartbeat HADR ハートビート : モニター・エレメント』
	417 ページの『hadr_local_host HADR ローカル・ホスト : モニター・エレメント』
	417 ページの『hadr_local_service HADR ローカル・サービス : モニター・エレメント』
	422 ページの『hadr_log_gap HADR ログ・ギャップ』
	420 ページの『hadr_primary_log_file HADR 1 次ログ・ファイル : モニター・エレメント』
	421 ページの『hadr_primary_log_lsn HADR 1 次ログ LSN : モニター・エレメント』
	420 ページの『hadr_primary_log_page HADR 1 次ログ・ページ : モニター・エレメント』
	418 ページの『hadr_remote_host HADR リモート・ホスト : モニター・エレメント』
	419 ページの『hadr_remote_instance HADR リモート・インスタンス : モニター・エレメント』
	418 ページの『hadr_remote_service HADR リモート・サービス : モニター・エレメント』
	413 ページの『hadr_role HADR の役割』
	421 ページの『hadr_standby_log_file HADR スタンバイ・ログ・ファイル : モニター・エレメント』
	422 ページの『hadr_standby_log_lsn HADR スタンバイ・ログ LSN : モニター・エレメント』
	422 ページの『hadr_standby_log_page HADR スタンバイ・ログ・ページ : モニター・エレメント』
	413 ページの『hadr_state HADR の状態 : モニター・エレメント』
	414 ページの『hadr_syncmode HADR 同期モード : モニター・エレメント』
	419 ページの『hadr_timeout HADR タイムアウト : モニター・エレメント』
lock	307 ページの『lock_attributes ロック属性』
	308 ページの『lock_count ロック・カウント』
	309 ページの『lock_current_mode 移行前の元のロック・モード』
	304 ページの『lock_escalation ロック・エスカレーション』
	309 ページの『lock_hold_count ロック保留カウント』
	298 ページの『lock_mode ロック・モード』
	307 ページの『lock_name ロック名』
	301 ページの『lock_object_name ロック対象名』
	300 ページの『lock_object_type 待機中のロック対象タイプ』
	308 ページの『lock_release_flags ロック保留解除フラグ』
	299 ページの『lock_status ロック状況』
	172 ページの『node_number ノード番号』
	349 ページの『table_file_id 表ファイル ID』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	320 ページの『tablespace_name 表スペース名』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
lock_wait	314 ページの『agent_id_holding_lock ロックを保持しているエージェント ID』
	315 ページの『appl_id_holding_lk ロックを保持しているアプリケーション ID』
	307 ページの『lock_attributes ロック属性』
	309 ページの『lock_current_mode 移行前の元のロック・モード』
	304 ページの『lock_escalation ロック・エスカレーション』
	298 ページの『lock_mode ロック・モード』
	304 ページの『lock_mode_requested 要求されているロック・モード』
	307 ページの『lock_name ロック名』
	300 ページの『lock_object_type 待機中のロック対象タイプ』
	308 ページの『lock_release_flags ロック保留解除フラグ』
	313 ページの『lock_wait_start_time ロック待機開始タイム・スタンプ』
	172 ページの『node_number ノード番号』
	386 ページの『ss_number サブセクション番号』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	320 ページの『tablespace_name 表スペース名』
memory_pool	172 ページの『node_number ノード番号』
	196 ページの『pool_cur_size メモリー・プールの現行サイズ』
	195 ページの『pool_id メモリー・プール ID』
	197 ページの『pool_config_size メモリー・プールの構成済みサイズ』
	197 ページの『pool_watermark メモリー・プール水準点』
progress	219 ページの『progress_completed_units 完了した進行作業単位』
	218 ページの『progress_description 進行の記述』
	217 ページの『progress_seq_num 進行シーケンス番号』
	218 ページの『progress_start_time 進行開始時刻』
	219 ページの『progress_total_units 合計進行作業単位』
	218 ページの『progress_work_metric 進行作業メトリック』
progress_list	217 ページの『progress_list_current_seq_num 現行の進行リストのシーケンス番号』
rollforward	172 ページの『node_number ノード番号』
	318 ページの『rf_type ロールフォワード・タイプ』
	318 ページの『rf_log_num ロールフォワードされているログ』
	319 ページの『rf_status ログ・フェーズ』
	317 ページの『rf_timestamp ロールフォワード・タイム・スタンプ』
	318 ページの『ts_name ロールフォワードされている表スペース』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
stmt	394 ページの『agents_top 作成されたエージェントの数』
	460 ページの『blocking_cursor ブロッキング・カーソル』
	375 ページの『consistency_token パッケージ整合性トークン』
	378 ページの『creator アプリケーション作成者』
	377 ページの『cursor_name カーソル名』
	395 ページの『degree_parallelism 並列処理の度合い』
	383 ページの『fetch_count 成功した取り出しの数』
	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	394 ページの『num_agents ステートメントで作動しているエージェントの数』
	375 ページの『package_name パッケージ名』
	376 ページの『package_version_id パッケージ・バージョン』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	384 ページの『query_card_estimate 行数見積もりの照会』
	385 ページの『query_cost_estimate 照会コストの見積もり』
	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	377 ページの『section_number セクション番号』
	203 ページの『sort_overflows ソート・オーバーフロー』
	381 ページの『stmt_elapsed_time 最新のステートメント経過時間』
	372 ページの『stmt_node_number ステートメント・ノード』
	374 ページの『stmt_operation/operation ステートメント操作』
	382 ページの『stmt_sorts ステートメント・ソート回数』
	379 ページの『stmt_start ステートメント操作開始タイム・スタンプ』
	379 ページの『stmt_stop ステートメント操作停止タイム・スタンプ』
	399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
	381 ページの『stmt_text SQL 動的ステートメント・テキスト』
	373 ページの『stmt_type ステートメント・タイプ』
	398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
	202 ページの『total_sort_time ソート時間合計』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
stmt_transmissions	456 ページの『elapsed_exec_time ステートメント実行経過時間』
	457 ページの『host_response_time ホスト応答時間』
	440 ページの『max_data_received_128 受信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数』
	441 ページの『max_data_received_256 受信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数』
	442 ページの『max_data_received_512 受信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数』
	443 ページの『max_data_received_1024 受信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数』
	444 ページの『max_data_received_2048 受信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数』
	445 ページの『max_data_received_4096 受信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数』
	446 ページの『max_data_received_8192 受信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数』
	447 ページの『max_data_received_16384 受信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数』
	448 ページの『max_data_received_31999 受信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数』
	449 ページの『max_data_received_64000 受信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数』
	450 ページの『max_data_received_gt64000 受信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数』
	440 ページの『max_data_sent_128 送信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数』
	441 ページの『max_data_sent_256 送信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数』
	442 ページの『max_data_sent_512 送信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数』
	443 ページの『max_data_sent_1024 送信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数』
	444 ページの『max_data_sent_2048 送信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数』
	445 ページの『max_data_sent_4096 送信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数』
	446 ページの『max_data_sent_8192 送信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数』
	447 ページの『max_data_sent_16384 送信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数』
	448 ページの『max_data_sent_31999 送信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数』
	449 ページの『max_data_sent_64000 送信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数』
	450 ページの『max_data_sent_gt64000 送信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数』
	451 ページの『max_network_time_1_ms ネットワーク時間が 1 ミリ秒以下のステートメント数』
	451 ページの『max_network_time_4_ms ネットワーク時間が 1 から 4 ミリ秒のステートメント数』
	452 ページの『max_network_time_16_ms ネットワーク時間が 4 から 16 ミリ秒のステートメント数』
	452 ページの『max_network_time_100_ms ネットワーク時間が 16 から 100 ミリ秒のステートメント数』
	453 ページの『max_network_time_500_ms ネットワーク時間が 100 から 500 ミリ秒のステートメント数』
	453 ページの『max_network_time_gt500_ms ネットワーク時間が 500 ミリ秒を超えるステートメント数』
	454 ページの『network_time_top ステートメントの最大ネットワーク時間』
	455 ページの『network_time_bottom ステートメントの最小ネットワーク時間』
	436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』
	436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
	438 ページの『outbound_bytes_sent_top 送信された最大アウトバウンド・バイト数』
	438 ページの『outbound_bytes_received_top 受信された最大アウトバウンド・バイト数』
	439 ページの『outbound_bytes_sent_bottom 送信された最小アウトバウンド・バイト数』
	439 ページの『outbound_bytes_received_bottom 受信された最小アウトバウンド・バイト数』
	431 ページの『sql_chains 試行された SQL チェーンの数』
	430 ページの『sql_stmts 試行された SQL ステートメントの数』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
subsection	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	387 ページの『ss_exec_time サブセクション実行経過時間』
	386 ページの『ss_node_number サブセクション・ノード番号』
	386 ページの『ss_number サブセクション番号』
	387 ページの『ss_status サブセクションの状況』
	402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
	401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
	389 ページの『tq_cur_send_spills オーバーフローした表キュー・バッファーの現在数』
	391 ページの『tq_id_waiting_on ノード上の表キュー待機』
	391 ページの『tq_max_send_spills 表キュー・バッファー・オーバーフローの最大数』
	388 ページの『tq_node_waited_for 表キュー上のノード待機』
	390 ページの『tq_rows_read 表キューから読み取られた行数』
	390 ページの『tq_rows_written 表キューに書き込まれた行数』
	388 ページの『tq_tot_send_spills オーバーフローした表キュー・バッファーの合計数』
	387 ページの『tq_wait_for_any 表キュー上のノード送信待機』
table	350 ページの『data_object_pages データ・オブジェクト・ページ数』
	351 ページの『index_object_pages 索引オブジェクト・ページ数』
	351 ページの『lob_object_pages LOB オブジェクト・ページ数』
	352 ページの『long_object_pages 長いオブジェクト・ページ数』
	346 ページの『overflow_accesses オーバーフロー・レコードへのアクセス』
	349 ページの『page_reorgs ページ再編成』
	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	349 ページの『table_file_id 表ファイル ID』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	339 ページの『table_type 表タイプ』
table_list	148 ページの『db_conn_time データベース活動化タイム・スタンプ』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	404 ページの『input_db_alias 入力データベース別名』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
table_reorg	355 ページの『reorg_completion 表再編成完了フラグ』
	355 ページの『reorg_current_counter 表再編成の進行状況』
	356 ページの『reorg_end 表再編成終了時刻』
	356 ページの『reorg_index_id 表の再編成に使用される索引』
	355 ページの『reorg_max_counter 表再編成の合計量』
	354 ページの『reorg_max_phase 表再編成の最大フェーズ数』
	354 ページの『reorg_phase 表再編成のフェーズ数』
	354 ページの『reorg_phase_start 表再編成フェーズ開始時刻』
	356 ページの『reorg_start 表再編成開始時刻』
	353 ページの『reorg_status 表再編成の状況』
	356 ページの『reorg_tbspc_id 表が再編成される表スペース』
	353 ページの『reorg_type 表再編成の属性』
tablespace	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	236 ページの『files_closed 閉じられたデータベース・ファイル』
	338 ページの『fs_caching ファイル・システム・キャッシング』
	243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
	237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
	238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
	243 ページの『pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求』
	240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
	239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
	241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
	242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	246 ページの『pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
tablespace (続き)	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	322 ページの『tablespace_content_type 表スペースのコンテンツ・タイプ』
	324 ページの『tablespace_cur_pool_id 現在使用中のバッファー・プール』
	323 ページの『tablespace_extent_size 表スペースのエクステント・サイズ』
	320 ページの『tablespace_id 表スペース ID』
	320 ページの『tablespace_name 表スペース名』
	324 ページの『tablespace_next_pool_id 次の始動時に使用されるバッファー・プール』
	323 ページの『tablespace_page_size 表スペースのページ・サイズ』
	323 ページの『tablespace_prefetch_size 表スペースのプリフェッチ・サイズ』
	326 ページの『tablespace_rebalancer_mode 再平衡化モード』
	321 ページの『tablespace_type 表スペース・タイプ』
tablespace_container	334 ページの『container_accessible コンテナーのアクセス可能性』
	332 ページの『container_id コンテナー ID』
	333 ページの『container_name コンテナー名』
	334 ページの『container_stripe_set ストライプ・セット』
	333 ページの『container_total_pages コンテナー内の合計ページ数』
	333 ページの『container_type コンテナー・タイプ』
	334 ページの『container_usable_pages コンテナー内の使用可能なページ数』
tablespace_list	148 ページの『db_conn_time データベース活動化タイム・スタンプ』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	404 ページの『input_db_alias 入力データベース別名』
	404 ページの『last_reset 最後のリセット・タイム・スタンプ』

表 12. スナップショット・モニターの論理データ・グループおよびモニター・エレメント (続き)

スナップショット論理データ・グループ	モニター・エレメント
tablespace_nodeinfo	326 ページの『tablespace_free_pages 表スペース内のフリー・ページ数』
	332 ページの『tablespace_min_recovery_time ロールフォワードの最小リカバリー時間』
	332 ページの『tablespace_num_containers 表スペース内のコンテナー数』
	329 ページの『tablespace_num_quiescers - 静止プログラム数』
	335 ページの『tablespace_num_ranges 表スペース・マップ内の範囲数』
	326 ページの『tablespace_page_top 表スペース最高水準点』
	326 ページの『tablespace_pending_free_pages 表スペース内のペンディング・フリー・ページ数』
	323 ページの『tablespace_prefetch_size 表スペースのプリフェッチ・サイズ』
	328 ページの『tablespace_rebalancer_extents_processed 再平衡化で処理されたエクステントの数』
	327 ページの『tablespace_rebalancer_extents_remaining 再平衡化で処理されるエクステントの合計数』
	328 ページの『tablespace_rebalancer_last_extent_moved 再平衡化によって最後に移動されたエクステント』
	329 ページの『tablespace_rebalancer_priority 現行の再平衡化優先順位』
	327 ページの『tablespace_rebalancer_restart_time 再平衡化再始動時刻』
	327 ページの『tablespace_rebalancer_start_time 再平衡化開始時刻』
	322 ページの『tablespace_state 表スペースの状態』
	331 ページの『tablespace_state_change_object_id 状態変更オブジェクト ID』
	331 ページの『tablespace_state_change_ts_id 状態変更表スペース ID』
	324 ページの『tablespace_total_pages 表スペース内の合計ページ数』
	325 ページの『tablespace_usable_pages 表スペース内の使用可能ページ数』
	325 ページの『tablespace_used_pages 表スペース内の使用されているページ数』
tablespace_quiescer	330 ページの『quiescer_agent_id 静止プログラム・エージェント ID』
	329 ページの『quiescer_auth_id 静止プログラム・ユーザー許可 ID』
	330 ページの『quiescer_obj_id 静止プログラム・オブジェクト ID』
	331 ページの『quiescer_state 静止プログラムの状態』
	330 ページの『quiescer_ts_id 静止プログラム表スペース ID』
tablespace_range	337 ページの『range_adjustment 範囲調整』
	338 ページの『range_container_id 範囲コンテナー』
	337 ページの『range_end_stripe 終了ストライプ』
	336 ページの『range_max_extent 範囲内の最大エクステント』
	336 ページの『range_max_page_number 範囲内の最大ページ』
	337 ページの『range_num_containers 範囲内コンテナーの数』
	336 ページの『range_number 範囲番号』
	338 ページの『range_offset 範囲オフセット』
	337 ページの『range_start_stripe 開始ストライプ』
	336 ページの『range_stripe_set_number ストライプ・セット番号』
utility_info	215 ページの『utility_dbname ユーティリティーで操作されるデータベース』
	215 ページの『utility_id ユーティリティー ID』
	215 ページの『utility_type ユーティリティー・タイプ』
	216 ページの『utility_priority ユーティリティー優先度』
	216 ページの『utility_start_time ユーティリティー開始時刻』
	216 ページの『utility_description ユーティリティー記述』
	172 ページの『node_number ノード番号』

イベント・タイプの論理データ・グループへのマッピング

イベント・モニターの出力は、番号の付いた一連の論理データ・グループから成っています。どのイベント・モニター・タイプの場合にも、出力レコードには必ず同じ開始論理データ・グループが含まれています。論理データ・グループが示すフレームは、イベント・モニターによって記録されるイベント・タイプによって異なります。

ファイルおよびパイプ・イベント・モニターの場合、イベント・レコードはどの接続についても生成されることがあるため、ストリーム中にいろいろな順序で現れる場合があります。すなわち、接続 1 のトランザクション・イベントの直後に接続 2 の接続イベントを取得することがあるということです。しかし、単一の接続または単一のイベントに属するレコードは、論理順序で現れます。例えば、ステートメント・レコード (ステートメントの終わり) は、トランザクション・レコード (UOW の終わり) があれば、必ずその前に来ます。同様に、デッドロック・イベント・レコードは、必ずデッドロックに関係する各接続のデッドロック接続イベント・レコードの前に来ます。**アプリケーション ID** または**アプリケーション・ハンドル (agent_id)** を使って、レコードと接続を一致させることができます。

接続ヘッダー・イベントは通常、データベースへのそれぞれの接続ごとに書き込まれます。 **詳細付きデッドロック・イベント・モニター**の場合は、デッドロックが生じたときにだけ書き込まれます。この場合、接続ヘッダー・イベントは、デッドロックに関係したものについてのみ書き込まれます。データベースへのすべての接続について書き込まれるわけではありません。

論理データ・グループは、4 つの異なるレベルに従って配列されます。すなわちモニター、プロローグ、内容、およびエピローグです。以下は、各レベルの詳細な記述です。対応するイベント・タイプおよび論理データ・グループも示しています。

モニター:

モニター・レベルの情報は、すべてのイベント・モニターに対して生成されます。これは、イベント・モニターのメタデータから成っています。

表 13. イベント・モニター・データ・ストリーム : モニター・セクション

イベント・タイプ	論理データ・グループ	入手できる情報
モニター・レベル	event_log_stream_header	イベント・モニターのバージョン・レベルおよびバイトの並び順を識別する。アプリケーションはこのヘッダーを使用して、 evmon 出力ストリームを処理できるかどうかを判別できます。

プロローグ:

プロローグ情報は、イベント・モニターが活動化されると生成されます。

表 14. イベント・モニター・データ・ストリーム : プロローグ・セクション

イベント・タイプ	論理データ・グループ	入手できる情報
ログ・ヘッダー	event_log_header	トレースの特性、例えば、サーバーのタイプおよびメモリーのレイアウト。
データベース・ヘッダー	event_db_header	データベース名、パスおよび活動化時間。
イベント・モニター開始	event_start	モニターが開始されるか再始動された時間。
接続ヘッダー	event_connheader	現行接続のヘッダーごとに 1 つ。接続時間とアプリケーション名を含みます。イベント接続ヘッダーが生成されるのは、接続、ステートメント、トランザクション、およびデッドロックの各イベント・モニターに対してのみです。詳細付きデッドロック・イベント・モニターは、デッドロックが生じたときだけ接続ヘッダーを生成します。

内容:

イベント・モニターの指定されたイベント・タイプに固有の情報は、内容セクションに表示されます。

表 15. イベント・モニター・データ・ストリーム : 内容セクション

イベント・タイプ	論理データ・グループ	入手できる情報
ステートメント・イベント	event_stmt	ステートメント・レベル・データ。動的ステートメントのテキストを含む。ステートメント・イベント・モニターは、取り出しのログを取りません。
サブセクション・イベント	event_subsection	サブセクション・レベル・データ。
トランザクション・イベント	event_xact	トランザクション・レベル・データ。
接続イベント	event_conn	接続レベル・データ。
デッドロック・イベント	event_deadlock	デッドロック・レベル・データ。
デッドロック接続イベント	event_dlconn	デッドロックに関係している接続ごとに 1 つ。関係するアプリケーションと競合しているロックを含みます。

表 15. イベント・モニター・データ・ストリーム : 内容セクション (続き)

イベント・タイプ	論理データ・グループ	入手できる情報
詳細付きデッドロック接続イベント	event_detailed_dlconn、 lock	デッドロックに関係している接続ごとに 1 つ。関係するアプリケーション、競合しているロック、現在のステートメント情報、およびアプリケーション競合によって保持された他のロックを含みます。
オーバーフロー	event_overflow	脱落したレコードの数。書き込み装置が (ブロック化されていない) イベント・モニターに追い付かないときに生成されます。

エピローグ:

エピローグ情報は、データベースが非活動化状態にあるとき (最後のアプリケーションが切断を終了したとき) に生成されます。

表 16. イベント・モニター・データ・ストリーム : エピローグ・セクション

イベント・タイプ	論理データ・グループ	入手できる情報
データベース・イベント	event_db	データベース・マネージャー・レベル・データ。
バッファー・プール・イベント	event_bufferpool	バッファー・プール・レベル・データ。
表スペース・イベント	event_tablespace	表スペース・レベル・データ。
表イベント	event_table	表レベル・データ。

関連概念:

- 51 ページの『イベント・モニター』
- 4 ページの『データベース・システム・モニターのデータ編成』

関連タスク:

- 53 ページの『データベース・システム・イベントからの情報の収集』

関連資料:

- 75 ページの『イベント・モニターの出力例』

イベント・モニターの論理データ・グループおよびモニター・エレメント

次の表は、論理データ・グループと、イベント・モニターによって戻されるモニター・エレメントの一覧表です。

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント

イベント論理データ・グループ	モニター・エレメント名
event_bufferpool	248 ページの『bp_name バッファー・プール名』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	409 ページの『event_time イベント時刻』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	409 ページの『evmon_flushes イベント・モニター・フラッシュ回数』
	236 ページの『files_closed 閉じられたデータベース・ファイル』
	408 ページの『partial_record 部分レコード』
	243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
	237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
	238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
	240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
	239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
	241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
	242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_conn	359 ページの『acc_curs_blk 受け入れられたブロック・カーソル要求』
	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	170 ページの『appl_priority アプリケーション・エージェント優先順位』
	171 ページの『appl_priority_type アプリケーション優先順位タイプ』
	272 ページの『appl_section_inserts セクション挿入数 : モニター・エレメント』
	271 ページの『appl_section_lookups セクションの参照回数 : モニター・エレメント』
	171 ページの『authority_lvl ユーザー許可レベル』
	372 ページの『binds_precompiles 試行されたバインド/プリコンパイル』
	264 ページの『cat_cache_inserts カタログ・キャッシュ挿入数』
	263 ページの『cat_cache_lookups カタログ・キャッシュ参照数』
	265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』
	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』
	295 ページの『deadlocks デッドロック検出数』
	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	149 ページの『disconn_time データベース非活動化タイム・スタンプ』
	362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
	209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』
	368 ページの『int_auto_rebinds 内部自動再バインド』
	368 ページの『int_commits 内部コミット数』
	371 ページの『int_deadlock_rollbacks デッドロックによる内部ロールバック』
	369 ページの『int_rollbacks 内部ロールバック数』
	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	304 ページの『lock_escalation ロック・エスカレーション』
	302 ページの『lock_timeouts ロック・タイムアウト数』
	311 ページの『lock_wait_time ロック待機中の時間』
	310 ページの『lock_waits ロック待機数』
	408 ページの『partial_record 部分レコード』
	269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
	267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_conn (続き)	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	249 ページの『prefetch_wait_time プリフェッチ待ち時間』
	276 ページの『priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数』
	278 ページの『priv_workspace_section_inserts 専用ワークスペース・セクション挿入』
	277 ページの『priv_workspace_section_lookups 専用ワークスペース・セクションの参照』
	276 ページの『priv_workspace_size_top 専用ワークスペースの最大サイズ』
	358 ページの『rej_curs_blk リジェクトされたブロック・カーソル要求』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	345 ページの『rows_read 読み取り行数』
	343 ページの『rows_selected 選択行数』
	347 ページの『int_rows_updated 更新された内部行数』
	344 ページの『rows_written 書き込み行数』
	365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』
	162 ページの『sequence_no シーケンス番号』
	274 ページの『shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数』
	275 ページの『shr_workspace_section_inserts 共有ワークスペース・セクション挿入数』
	274 ページの『shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数』
	273 ページの『shr_workspace_size_top 共有ワークスペースの最大サイズ』
	203 ページの『sort_overflows ソート・オーバーフロー』
	361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
	400 ページの『system_cpu_time システム CPU 時間』
	207 ページの『total_hash_joins ハッシュ結合の合計』
	208 ページの『total_hash_loops ハッシュ・ループの合計』
	192 ページの『total_sec_cons 2 次接続』
	202 ページの『total_sort_time ソート時間合計』
	201 ページの『total_sorts ソート合計』
	366 ページの『uid_sql_stmts 実行された更新/挿入/削除 SQL ステートメント』
	249 ページの『unread_prefetch_pages 読み取り不能プリフェッチ・ページ』
	399 ページの『user_cpu_time ユーザー CPU 時間』
	297 ページの『x_lock_escals 排他ロック・エスカレーション数』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_connheader	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	158 ページの『appl_name アプリケーション名』
	162 ページの『auth_id 許可 ID』
	164 ページの『client_db_alias アプリケーションで使用するデータベース別名』
	163 ページの『client_nname クライアントの構成 NNAME』
	168 ページの『client_pid クライアント・プロセス ID』
	168 ページの『client_platform クライアント・オペレーティング・プラットフォーム』
	164 ページの『client_prdid クライアント製品/バージョン ID』
	169 ページの『client_protocol クライアント通信プロトコル』
	156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
	148 ページの『conn_time データベース接続時刻』
	167 ページの『corr_token DRDA 相関トークン』
	167 ページの『execution_id ユーザー・ログイン ID』
	172 ページの『node_number ノード番号』
	162 ページの『sequence_no シーケンス番号』
	169 ページの『territory_code データベース・テリトリー・コード』
event_connmemuse	172 ページの『node_number ノード番号』
	196 ページの『pool_cur_size メモリー・プールの現行サイズ』
	195 ページの『pool_id メモリー・プール ID』
	197 ページの『pool_config_size メモリー・プールの構成済みサイズ』
	197 ページの『pool_watermark メモリー・プール水準点』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_db	272 ページの『appl_section_inserts セクション挿入数 : モニター・エレメント』
	271 ページの『appl_section_lookups セクションの参照回数 : モニター・エレメント』
	372 ページの『binds_precompiles 試行されたバインド/プリコンパイル』
	264 ページの『cat_cache_inserts カタログ・キャッシュ挿入数』
	263 ページの『cat_cache_lookups カタログ・キャッシュ参照数』
	265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』
	266 ページの『cat_cache_size_top カタログ・キャッシュ最高水準点』
	151 ページの『catalog_node カタログ・ノード番号』
	150 ページの『catalog_node_name カタログ・ノード・ネットワーク名』
	363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
	174 ページの『connections_top 同時接続の最大数』
	279 ページの『db_heap_top 割り振られた最大データベース・ヒープ』
	367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』
	295 ページの『deadlocks デッドロック検出数』
	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	149 ページの『disconn_time データベース非活動化タイム・スタンプ』
	362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	409 ページの『evmon_flushes イベント・モニター・フラッシュ回数』
	362 ページの『failed_sql_stmts 失敗したステートメント操作』
	236 ページの『files_closed 閉じられたデータベース・ファイル』
	208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
	209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』
	368 ページの『int_auto_rebinds 内部自動再バインド』
	368 ページの『int_commits 内部コミット数』
	369 ページの『int_rollbacks 内部ロールバック数』
	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	296 ページの『lock_escals ロック・エスカレーション数』
	302 ページの『lock_timeouts ロック・タイムアウト数』
	311 ページの『lock_wait_time ロック待機中の時間』
	310 ページの『lock_waits ロック待機数』
	285 ページの『log_held_by_dirty_pages ダーティー・ページ別に計算されるログ・スペースの量』
	287 ページの『log_read_time ログ読み取り時間』
	282 ページの『log_reads 読み取られたログ・ページの数』
	286 ページの『log_to_redo_for_recovery リカバリーの場合に再実行されるログの量』
	287 ページの『log_write_time ログ書き込み時間』
	283 ページの『log_writes 書き込まれたログ・ページの数』
	288 ページの『num_log_read_io ログ読み取り数』
	288 ページの『num_log_write_io ログ書き込み数』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_db (続き)	408 ページの『partial_record 部分レコード』
	269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
	267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』
	269 ページの『pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー』
	270 ページの『pkg_cache_size_top パッケージ・キャッシュの最高水準点』
	243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
	237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
	238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
	243 ページの『pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求』
	240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
	239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
	241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
	242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
	247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
	246 ページの『pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	249 ページの『prefetch_wait_time プリフェッチ待ち時間』
	276 ページの『priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数』
	278 ページの『priv_workspace_section_inserts 専用ワークスペース・セクション挿入』
	277 ページの『priv_workspace_section_lookups 専用ワークスペース・セクションの参照』
	276 ページの『priv_workspace_size_top 専用ワークスペースの最大サイズ』
	364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
	342 ページの『rows_deleted 削除行数』
	342 ページの『rows_inserted 挿入行数』
	345 ページの『rows_read 読み取り行数』
	343 ページの『rows_selected 選択行数』
	343 ページの『rows_updated 更新行数』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_db (続き)	280 ページの『sec_log_used_top 使用された最大 2 次ログ・スペース』
	365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』
	144 ページの『server_platform サーバーのオペレーティング・システム』
	274 ページの『shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数』
	275 ページの『shr_workspace_section_inserts 共有ワークスペース・セクション挿入数』
	274 ページの『shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数』
	273 ページの『shr_workspace_size_top 共有ワークスペースの最大サイズ』
	203 ページの『sort_overflows ソート・オーバーフロー』
	361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
	281 ページの『tot_log_used_top 使用された最大合計ログ・スペース』
	185 ページの『total_cons データベース活動化以降の接続』
	207 ページの『total_hash_joins ハッシュ結合の合計』
	208 ページの『total_hash_loops ハッシュ・ループの合計』
	202 ページの『total_sort_time ソート時間合計』
	201 ページの『total_sorts ソート合計』
	366 ページの『uid_sql_stmts 実行された更新/挿入/削除 SQL ステートメント』
	249 ページの『unread_prefetch_pages 読み取り不能プリフェッチ・ページ』
	297 ページの『x_lock_escals 排他ロック・エスカレーション数』
event_dbheader	148 ページの『conn_time データベース接続時刻』
	146 ページの『db_name データベース名』
	147 ページの『db_path データベース・パス』
event_dbmemuse	172 ページの『node_number ノード番号』
	196 ページの『pool_cur_size メモリー・プールの現行サイズ』
	195 ページの『pool_id メモリー・プール ID』
	197 ページの『pool_config_size メモリー・プールの構成済みサイズ』
	197 ページの『pool_watermark メモリー・プール水準点』
event_deadlock	304 ページの『deadlock_id デッドロック・イベント ID』
	305 ページの『deadlock_node デッドロック発生場所のパーティション番号』
	303 ページの『dl_conns デッドロックに関係している接続』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	316 ページの『rolled_back_agent_id ロールバックされたエージェント』
	316 ページの『rolled_back_appl_id ロールバック・アプリケーション』
	306 ページの『rolled_back_participant_no ロールバック参加アプリケーション』
	317 ページの『rolled_back_sequence_no ロールバックされたシーケンス番号』
	380 ページの『start_time イベント開始時刻』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_detailed_dlconn	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	315 ページの『appl_id_holding_lk ロックを保持しているアプリケーション ID』
	460 ページの『blocking_cursor ブロッキング・カーソル』
	375 ページの『consistency_token パッケージ整合性トークン』
	378 ページの『creator アプリケーション作成者』
	377 ページの『cursor_name カーソル名』
	304 ページの『deadlock_id デッドロック・イベント ID』
	305 ページの『deadlock_node デッドロック発生場所のパーティション番号』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	304 ページの『lock_escalation ロック・エスカレーション』
	298 ページの『lock_mode ロック・モード』
	304 ページの『lock_mode_requested 要求されているロック・モード』
	302 ページの『lock_node ロック・ノード』
	301 ページの『lock_object_name ロック対象名』
	300 ページの『lock_object_type 待機中のロック対象タイプ』
	313 ページの『lock_wait_start_time ロック待機開始タイム・スタンプ』
	294 ページの『locks_held ロック保持数』
	306 ページの『locks_in_list 報告されたロックの回数』
	375 ページの『package_name パッケージ名』
	376 ページの『package_version_id パッケージ・バージョン』
	305 ページの『participant_no デッドロック内の参加者』
	306 ページの『participant_no_holding_lk アプリケーションが必要とするオブジェクトのロックを保留する参加者』
	377 ページの『section_number セクション番号』
	162 ページの『sequence_no シーケンス番号』
	316 ページの『sequence_no_holding_lk ロックを保持しているシーケンス番号』
	380 ページの『start_time イベント開始時刻』
	374 ページの『stmt_operation/operation ステートメント操作』
	381 ページの『stmt_text SQL 動的ステートメント・テキスト』
	373 ページの『stmt_type ステートメント・タイプ』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	320 ページの『tablespace_name 表スペース名』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_dlconn	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	315 ページの『appl_id_holding_lk ロックを保持しているアプリケーション ID』
	304 ページの『deadlock_id デッドロック・イベント ID』
	305 ページの『deadlock_node デッドロック発生場所のパーティション番号』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	307 ページの『lock_attributes ロック属性』
	308 ページの『lock_count ロック・カウント』
	309 ページの『lock_current_mode 移行前の元のロック・モード』
	304 ページの『lock_escalation ロック・エスカレーション』
	309 ページの『lock_hold_count ロック保留カウント』
	298 ページの『lock_mode ロック・モード』
	304 ページの『lock_mode_requested 要求されているロック・モード』
	307 ページの『lock_name ロック名』
	302 ページの『lock_node ロック・ノード』
	301 ページの『lock_object_name ロック対象名』
	300 ページの『lock_object_type 待機中のロック対象タイプ』
	308 ページの『lock_release_flags ロック保留解除フラグ』
	313 ページの『lock_wait_start_time ロック待機開始タイム・スタンプ』
	305 ページの『participant_no デッドロック内の参加者』
	306 ページの『participant_no_holding_lk アプリケーションが必要とするオブジェクトのロックを保留する参加者』
	162 ページの『sequence_no シーケンス番号』
	316 ページの『sequence_no_holding_lk ロックを保持しているシーケンス番号』
	380 ページの『start_time イベント開始時刻』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	320 ページの『tablespace_name 表スペース名』
event_log_header	407 ページの『byte_order イベント・データのバイト・オーダー』
	156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
	408 ページの『event_monitor_name イベント・モニター名』
	405 ページの『num_nodes_in_db2_instance パーティション内のノード数』
	143 ページの『server_prdid サーバー製品/バージョン ID』
	141 ページの『server_instance_name サーバー・インスタンス名』
	169 ページの『territory_code データベース・テリトリー・コード』
	408 ページの『version モニター・データのバージョン』
event_overflow	406 ページの『count イベント・モニター・オーバーフロー数』
	406 ページの『first_overflow_time 最初のイベント・オーバーフロー時刻』
	407 ページの『last_overflow_time 最後のイベント・オーバーフロー時刻』
	172 ページの『node_number ノード番号』
event_start	380 ページの『start_time イベント開始時刻』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_stmt	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	394 ページの『agents_top 作成されたエージェントの数』
	159 ページの『appl_id アプリケーション ID』
	460 ページの『blocking_cursor ブロッキング・カーソル』
	375 ページの『consistency_token パッケージ整合性トークン』
	378 ページの『creator アプリケーション作成者』
	377 ページの『cursor_name カーソル名』
	383 ページの『fetch_count 成功した取り出しの数』
	347 ページの『int_rows_deleted 削除された内部行数』
	348 ページの『int_rows_inserted 挿入された内部行数』
	347 ページの『int_rows_updated 更新された内部行数』
	375 ページの『package_name パッケージ名』
	376 ページの『package_version_id パッケージ・バージョン』
	408 ページの『partial_record 部分レコード』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	377 ページの『section_number セクション番号』
	162 ページの『sequence_no シーケンス番号』
	203 ページの『sort_overflows ソート・オーバーフロー』
	410 ページの『sql_req_id SQL ステートメントの要求 ID』
	384 ページの『sqlca SQL 連絡域 (SQLCA)』
	380 ページの『start_time イベント開始時刻』
	374 ページの『stmt_operation/operation ステートメント操作』
	381 ページの『stmt_text SQL 動的ステートメント・テキスト』
	373 ページの『stmt_type ステートメント・タイプ』
	380 ページの『stop_time イベント停止時刻』
	400 ページの『system_cpu_time システム CPU 時間』
	202 ページの『total_sort_time ソート時間合計』
	201 ページの『total_sorts ソート合計』
	399 ページの『user_cpu_time ユーザー CPU 時間』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_subsection	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	394 ページの『num_agents ステートメントで作動しているエージェントの数』
	408 ページの『partial_record 部分レコード』
	387 ページの『ss_exec_time サブセクション実行経過時間』
	386 ページの『ss_node_number サブセクション・ノード番号』
	386 ページの『ss_number サブセクション番号』
	402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
	401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
	391 ページの『tq_max_send_spills 表キュー・バッファー・オーバーフローの最大数』
	390 ページの『tq_rows_read 表キューから読み取られた行数』
	390 ページの『tq_rows_written 表キューに書き込まれた行数』
	388 ページの『tq_tot_send_spills オーバーフローした表キュー・バッファーの合計数』
event_table	350 ページの『data_object_pages データ・オブジェクト・ページ数』
	409 ページの『event_time イベント時刻』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	409 ページの『evmon_flushes イベント・モニター・フラッシュ回数』
	351 ページの『index_object_pages 索引オブジェクト・ページ数』
	351 ページの『lob_object_pages LOB オブジェクト・ページ数』
	352 ページの『long_object_pages 長いオブジェクト・ページ数』
	346 ページの『overflow_accesses オーバーフロー・レコードへのアクセス』
	349 ページの『page_reorgs ページ再編成』
	408 ページの『partial_record 部分レコード』
	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	339 ページの『table_type 表タイプ』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_tablespace	260 ページの『direct_read_reqs 直接読み取り要求』
	261 ページの『direct_read_time 直接読み取り時間』
	258 ページの『direct_reads データベースからの直接読み取り』
	260 ページの『direct_write_reqs 直接書き込み要求』
	262 ページの『direct_write_time 直接書き込み時間』
	259 ページの『direct_writes データベースへの直接書き込み』
	409 ページの『event_time イベント時刻』
	410 ページの『evmon_activates イベント・モニター活動化回数』
	409 ページの『evmon_flushes イベント・モニター・フラッシュ回数』
	236 ページの『files_closed 閉じられたデータベース・ファイル』
	408 ページの『partial_record 部分レコード』
	243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
	237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
	238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
	243 ページの『pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求』
	240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
	239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
	241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
	242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
	255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
	224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
	226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
	254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
	228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
	256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』
	229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
	232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
	255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
	233 ページの『pool_index_writes バッファー・プール索引の書き込み』
	246 ページの『pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数』
	235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
	225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
	227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
	231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』
	232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
	236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
	320 ページの『tablespace_name 表スペース名』

表 17. イベント・モニターの論理データ・グループおよびモニター・エレメント (続き)

イベント論理データ・グループ	モニター・エレメント名
event_xact	153 ページの『agent_id アプリケーション・ハンドル (エージェント ID)』
	159 ページの『appl_id アプリケーション ID』
	296 ページの『lock_escals ロック・エスカレーション数』
	311 ページの『lock_wait_time ロック待機中の時間』
	303 ページの『locks_held_top ロック保持最大数』
	408 ページの『partial_record 部分レコード』
	175 ページの『prev_uow_stop_time 直前の作業単位完了タイム・スタンプ』
	345 ページの『rows_read 読み取り行数』
	344 ページの『rows_written 書き込み行数』
	162 ページの『sequence_no シーケンス番号』
	400 ページの『system_cpu_time システム CPU 時間』
	283 ページの『uow_log_space_used 作業単位ログ・スペース』
	175 ページの『uow_start_time 作業単位開始タイム・スタンプ』
	178 ページの『uow_status 作業単位の状況』
	176 ページの『uow_stop_time 作業単位停止タイム・スタンプ』
	399 ページの『user_cpu_time ユーザー CPU 時間』
	297 ページの『x_lock_escals 排他ロック・エスカレーション数』
lock	307 ページの『lock_attributes ロック属性』
	308 ページの『lock_count ロック・カウント』
	309 ページの『lock_current_mode 移行前の元のロック・モード』
	304 ページの『lock_escalation ロック・エスカレーション』
	309 ページの『lock_hold_count ロック保留カウント』
	298 ページの『lock_mode ロック・モード』
	307 ページの『lock_name ロック名』
	301 ページの『lock_object_name ロック対象名』
	300 ページの『lock_object_type 待機中のロック対象タイプ』
	308 ページの『lock_release_flags ロック保留解除フラグ』
	299 ページの『lock_status ロック状況』
	172 ページの『node_number ノード番号』
	349 ページの『table_file_id 表ファイル ID』
	340 ページの『table_name 表名』
	341 ページの『table_schema 表スキーマ名』
	320 ページの『tablespace_name 表スペース名』
sqlca	sqlcabc
	sqlcode
	sqlerrml
	sqlcaid
	sqlerrmc
	sqlerrp
	sqlerrd
	sqlwarn
	sqlstate

第 6 章 モニター・エレメント

データベース・システム・モニターのエレメント

システム・モニターが戻すモニター・エレメントは次のように分類できます。

- モニターされるデータベース・マネージャー、アプリケーション、またはデータベース接続の**識別**。
- システムの**構成**に最初に必要とされるデータ。
- データベース、アプリケーション、表、またはステートメントを含むさまざまなレベルのデータベース・**アクティビティー**。この情報を使用して、アクティビティーのモニター、問題判別、およびパフォーマンス分析を行うことができます。この情報を構成にも使用することができます。
- **DB2 Connect** アプリケーションに関する情報。この中には、ゲートウェイで実行中の DCS アプリケーション、実行中の SQL ステートメント、およびデータベース接続に関する情報が含まれます。
- **フェデレーテッド・データベース・システム**に関する情報。これには、 DB2 フェデレーテッド・システム内で実行中の各アプリケーションからデータ・ソースへのすべてのアクセスに関する情報、およびフェデレーテッド・サーバー・インスタンス内で実行中の特定のアプリケーションからデータ・ソースへのアクセスに関する情報が含まれます。

モニター・エレメントは次の標準形式で記述されます。

エレメント ID
エレメントの名前。データ・ストリームをそのまま構文解析すると、エレメント ID が大文字となり、 SQLM_ELM_ の接頭部が付きます。

エレメント・タイプ
モニター・エレメントが戻す情報のタイプ。例えば db2start_time モニター・エレメントはタイム・スタンプを戻します。

スナップショット・モニター情報
モニター・エレメントがスナップショット・モニター情報を戻す場合は、次のフィールドがある表が記載されています。

- *スナップショット・レベル* : スナップショット・モニターでキャプチャー可能な情報レベル。例えば appl_status モニター・エレメントは、アプリケーション・レベルおよびロック・レベルでの情報を戻します。
- *論理データ・グループ* : キャプチャーされたスナップショット情報が戻される論理データ・グループ。データ・ストリームをそのまま構文解析すると、論理データ・グループ ID が大文字となり、 SQLM_ELM_ の接頭部が付きます。例えば appl_status モニター・エレメントは、 appl_id_info グループおよび appl_lock_list グループの情報を戻します。
- モニター・スイッチ : この情報を取得するために設定が必要なシステム・モニター・スイッチ。スイッチが「基本」の場合は、モニター・エレメントのデータが常に収集されます。

イベント・モニター情報

モニター・エレメントがイベント・モニターによって収集される場合は、次のフィールドがある表が記載されています。

- *イベント・タイプ* : イベント・モニターで収集可能な情報のレベル。この情報を収集するには、このイベント・タイプを使用してイベント・モニターを作成する必要があります。例えば appl_status モニター・エレメントは、「接続」イベント・モニターで収集されます。
- *論理データ・グループ* : キャプチャーされたイベント情報が戻される論理データ・グループ。データ・ストリームをそのまま構文解析すると、論理データ・グループ ID が大文字となり、 SQLM_ELM_ の接頭部が付きます。例えば appl_status モニター・エレメントは、 event_conn グループの情報を戻します。
- モニター・スイッチ : この情報を取得するために設定が必要なシステム・モニター・スイッチ。イベント・モニターの場合、イベント・データの収集を制約できるモニター・スイッチは「タイム・スタンプ」スイッチのみです。このフィールドにダッシュが示されている場合は、モニター・エレメントのデータが常に収集されます。

説明 モニター・エレメントによって収集されるデータの説明。

使用法 データベース・システムをモニターする際の、モニター・エレメントによって収集された情報の使用方法に関する情報。

サーバーの識別および状況

サーバーの識別および状況に関するモニター・エレメント

次のエレメントにより、サーバーに関する識別および状況情報が提供されます。

- db2start_time データベース・マネージャー開始タイム・スタンプ : モニター・エレメント
- server_nname モニター対象の (サーバー) データベース・パーティションでの NNAME 構成 : モニター・エレメント
- server_instance_name サーバー・インスタンス名 : モニター・エレメント
- server_db2_type モニター対象 (サーバー) ノードのデータベース・マネージャーのタイプ : モニター・エレメント
- server_prdid サーバー製品/バージョン ID : モニター・エレメント
- server_version サーバー・バージョン : モニター・エレメント
- service_level サービス・レベル : モニター・エレメント
- server_platform サーバーのオペレーティング・システム : モニター・エレメント
- product_name 製品名 : モニター・エレメント
- db2_status DB2 インスタンス状況 : モニター・エレメント
- time_zone_disp 時間帯変位 : モニター・エレメント

db2start_time データベース・マネージャー開始タイム・スタンプ

エレメント ID db2start_time

エレメント・タイプ タイム・スタンプ

表 18. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 db2start コマンドを使用してデータベース・マネージャーを開始した日時。

使用法 このエレメントと *time_stamp* モニター・エレメントを組み合わせて使用すると、データベース・マネージャーを開始してからスナップショットが取られるまでの経過時間を計算できます。

関連資料:

• 405 ページの『time_stamp スナップショット時刻』

server_nname モニター対象の (サーバー) データベース・パーティションでの NNAME 構成

エレメント ID server_nname

エレメント・タイプ 情報

表 19. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

説明 データベース・システム・モニターでモニターされているデータベース・パーティション名。

使用法 このエレメントは、モニター対象のデータベース・サーバー・パーティションを識別するときに使用します。この情報は、モニター出力を後で解析するためにファイルやデータベースに保管しており、データベース・サーバーのパーティションごとにデータを区別する必要がある場合に役立ちます。このデータベース・パーティション名は、*nname* 構成パラメーターに従って決定されます。

このエレメントは、NetBIOS LAN 環境が存在する Windows 環境にのみ適用されます。

関連資料:

• 163 ページの『client_nname クライアントの構成 NNAME』

server_instance_name サーバー・インスタンス名

エレメント ID server_instance_name

エレメント・タイプ 情報

表 20. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

表 21. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-

説明 スナップショットが取られるデータベース・マネージャー・インスタンスの名前。

使用法 複数のデータベース・マネージャーのインスタンスが同一のシステム上にある場合、このデータ項目は、スナップショット呼び出しが行われたインスタンスを一意的に識別するために使用されます。モニター出力を後で解析するためにファイルやデータベースに保管しており、データベース・マネージャーの各インスタンスからのデータを特定する必要がある場合には、この情報と *server_nname* を併用すると役に立ちます。

関連資料:

- 141 ページの『server_nname モニター対象の (サーバー) データベース・パーティションでの NNAME 構成』

server_db2_type モニター対象 (サーバー) ノードのデータベース・マネージャーのタイプ

エレメント ID server_db2_type

エレメント・タイプ 情報

表 22. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

説明 モニター中のデータベース・マネージャーのタイプを識別します。

使用法 データベース・マネージャーについて、次の構成タイプの 1 つが含まれます。

API シンボリック定数	コマンド行プロセッサー出力
sqlf_nt_server	ローカル・クライアントおよびリモート・クライアントを指定したデータベース・サーバー
sqlf_nt_stand_req	ローカル・クライアントを指定したデータベース・サーバー

API シンボリック定数は、組み込みファイルの *sqlutil.h* に定義されています。

関連資料:

- 141 ページの『server_nname モニター対象の (サーバー) データベース・パーティションでの NNAME 構成』

server_prdid サーバー製品/バージョン ID

エレメント ID server_prdid

エレメント・タイプ 情報

表 23. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

表 24. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-

説明 サーバー上で実行中の製品とバージョン。

使用法 PPPVVRRM の形式になっています。各部分の定義は次のとおりです。

PPP SQL です。

VV 2 桁でバージョン番号を示します (バージョン番号が 1 桁の場合には、高位の桁は 0 になります)。

RR 2 桁でリリース番号を示します (リリース番号が 1 桁の場合には、高位の桁は 0 になります)。

M 1 桁で修正レベルを示します。

関連資料:

- 164 ページの『client_prdid クライアント製品/バージョン ID』

server_version サーバー・バージョン

エレメント ID server_version

エレメント・タイプ 情報

表 25. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

説明 情報を戻しているサーバーのバージョン。

使用法 このフィールドは、データベース・システム・モニター情報を収集しているデータベース・サーバーのレベルを識別します。これにより、アプリケーションは、データを戻しているサーバーのレベルに基づいてデータを解釈することができます。有効な値は次のとおりです。

SQLM_DBMON_VERSION1 データは、DB2 バージョン 1 によって戻されました。

SQLM_DBMON_VERSION2 データは、DB2 バージョン 2 によって戻されました。

SQLM_DBMON_VERSION5 データは、DB2 Universal Database バージョン 5 によって戻されました。

SQLM_DBMON_VERSION5_2 データは、DB2 Universal Database バージョン 5.2 によって戻されました。

SQLM_DBMON_VERSION6 データは、DB2 Universal Database バージョン 6 によって戻されました。

SQLM_DBMON_VERSION7 データは、DB2 Universal Database バージョン 7 によって戻されました。

SQLM_DBMON_VERSION8 データは、DB2 Universal Database バージョン 8 によって戻されました。

関連資料:

• 143 ページの『server_prdid サーバー製品/バージョン ID』

service_level サービス・レベル

エレメント ID service_level

エレメント・タイプ 情報

表 26. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 DB2 インスタンスの現在の修正サービス・レベルを示します。

server_platform サーバーのオペレーティング・システム

エレメント ID server_platform

エレメント・タイプ 情報

表 27. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 28. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 データベース・サーバーが稼働中のオペレーティング・システム。

使用法 このエレメントを使用して、リモート・アプリケーションの問題判別を行えます。このフィールドの値は、ヘッダー・ファイルの *sqlmon.h* にあります。

関連資料:

- 150 ページの『db_location データベース・ロケーション』
- 168 ページの『client_platform クライアント・オペレーティング・プラットフォーム』

product_name 製品名

エレメント ID product_name

エレメント・タイプ 情報

表 29. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 実行中の DB2 インスタンスのバージョンの詳細情報。

db2_status DB2 インスタンス状況

エレメント ID db2_status

エレメント・タイプ 情報

表 30. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 データベース・マネージャーのインスタンスの現在の状況。

使用法 このエレメントを使用して、データベース・マネージャー・インスタンスの状態を判別できます。

このエレメントの値は以下のとおりです。

API 定数	値	説明
SQLM_DB2_ACTIVE	0	データベース・マネージャー・インスタンスはアクティブになっている。
SQLM_DB2_QUIESCE_PEND	1	インスタンス内のインスタンスおよびデータベースは静止ペンディング状態となっている。インスタンス・データベースへの新規接続は許可されず、新規作業単位を開始することはできません。静止要求によっては、アクティブな作業単位の完了が許可される場合と即時ロールバックが行われる場合があります。
SQLM_DB2_QUIESCED	2	インスタンス内のインスタンスおよびデータベースは静止状態となっている。インスタンス・データベースへの新規接続は許可されず、新規作業単位を開始することはできません。

関連資料:

• 149 ページの『db_status データベース状況』

time_zone_disp 時間帯変位

エレメント ID time_zone_disp

エレメント・タイプ 情報

表 31. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

説明 グリニッジ標準時 (GMT) と現地時間帯の差を示す秒数。

使用法 データベース・システム・モニターから報告される時刻はすべて GMT であり、現地時間はこの差に基づいて計算されます。

データベースの識別および状況

データベースの識別および状況に関するモニター・エレメント

次のエレメントにより、データベースに関する識別および状況情報が提供されます。

- db_name データベース名 : モニター・エレメント
- db_path データベース・パス : モニター・エレメント
- db_conn_time データベース活動化タイム・スタンプ : モニター・エレメント
- conn_time データベース接続時刻 : モニター・エレメント
- disconn_time データベース非活動化タイム・スタンプ : モニター・エレメント
- db_status データベース状況 : モニター・エレメント
- catalog_node_name カタログ・ノード・ネットワーク名 : モニター・エレメント
- db_location データベース・ロケーション : モニター・エレメント
- catalog_node カタログ・ノード番号 : モニター・エレメント
- last_backup 最終バックアップ・タイム・スタンプ : モニター・エレメント

db_name データベース名

エレメント ID db_name

エレメント・タイプ 情報

表 32. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl_id_info	基本

表 32. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_remote	基本
表スペース	tablespace_list	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
表	table_list	表
ロック	db_lock_list	基本
動的 SQL	dynsql_list	基本
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl_info	基本

表 33. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbheader	-

説明 情報が収集されるデータベースの実名、またはアプリケーションの接続先のデータベースの実名。これはデータベースの作成時に与えられた名前です。

使用法 このエレメントを使用すると、データが適用される特定のデータベースを識別できます。

ホストへの接続、または AS/400 および iSeries のデータベース・サーバーへの接続で DB2 Connect を使用しないアプリケーションの場合は、このエレメントと *db_path* モニター・エレメントを組み合わせて使用すると、データベースを個別に識別し、モニターが提供する情報の各レベルに関連付けることができます。

関連資料:

- 147 ページの『db_path データベース・パス』
- 164 ページの『client_db_alias アプリケーションで使用するデータベース別名』
- 404 ページの『last_reset 最後のリセット・タイム・スタンプ』
- 404 ページの『input_db_alias 入力データベース別名』

db_path データベース・パス

エレメント ID db_path

エレメント・タイプ 情報

表 34. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl_id_info	基本
表スペース	tablespace_list	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
表	table_list	表
ロック	db_lock_list	基本

表 34. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql_list	基本

表 35. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbheader	-

説明 モニター対象のシステムに保管されているデータベースのロケーションを示す絶対パスです。

使用法 このエレメントと *db_name* モニター・エレメントを組み合わせて使用すると、データが適用される特定のデータベースを識別できます。

関連資料:

- 146 ページの『db_name データベース名』
- 404 ページの『input_db_alias 入力データベース別名』

db_conn_time データベース活動化タイム・スタンプ

エレメント ID db_conn_time

エレメント・タイプ タイム・スタンプ

表 36. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	タイム・スタンプ
表スペース	tablespace_list	バッファー・プール、タイム・スタンプ
表	table_list	タイム・スタンプ

説明 データベースへの接続の日時 (データベース・レベルでは、これはデータベースへの最初の接続)、またはデータベースの活動化が発行された日時。

使用法 このエレメントと disconn_time モニター・エレメントを組み合わせて使用すると、合計接続時間を計算できます。

関連資料:

- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 405 ページの『time_stamp スナップショット時刻』
- 148 ページの『conn_time データベース接続時刻』

conn_time データベース接続時刻

エレメント ID conn_time

エレメント・タイプ タイム・スタンプ

表 37. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbheader	-
接続	event_connheader	-

説明 データベースへの接続の日時 (データベース・レベルでは、これはデータベースへの最初の接続)、またはデータベースの活動化が発行された日時。

使用法 このエレメントと disconn_time モニター・エレメントを組み合わせて使用すると、次の時点以降の経過時間を計算できます。

- データベースがアクティブだったとき (データベース・レベルの情報の場合)。
- 接続がアクティブだったとき (接続レベルの情報の場合)。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 149 ページの『disconn_time データベース非活動化タイム・スタンプ』

disconn_time データベース非活動化タイム・スタンプ

エレメント ID disconn_time

エレメント・タイプ タイム・スタンプ

表 38. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 アプリケーションがデータベースとの接続を切断した日時 (データベース・レベルでは、アプリケーションが最後に切断した日時)。

使用法 このエレメントを使用すると、次の時点からの経過時間を計算できます。

- データベースがアクティブだったとき (データベース・レベルの情報の場合)
- 接続がアクティブだったとき (接続レベルの情報の場合)。

db_status データベース状況

エレメント ID db_status

エレメント・タイプ 情報

表 39. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 データベースの現在の状況。

使用法 このエレメントを使用して、データベースの状態を判別できます。

このフィールドの値は以下のとおりです。

API 定数	値	説明
SQLM_DB_ACTIVE	0	データベースはアクティブになっている。
SQLM_DB_QUIESCE_PEND	1	データベースは静止ペンディング状態となっている。データベースに対する新しい接続は許可されません。新しい作業単位も開始できません。静止要求によっては、アクティブな作業単位の完了が許可される場合と即時ロールバックが行われる場合があります。
SQLM_DB_QUIESCED	2	データベースは静止状態となっている。データベースに対する新しい接続は許可**されません**。新しい作業単位も開始できません。
SQLM_DB_ROLLFWD	3	データベースでロールフォワードが進行中。

関連資料:

• 145 ページの『db2_status DB2 インスタンス状況』

catalog_node_name カタログ・ノード・ネットワーク名

エレメント ID catalog_node_name

エレメント・タイプ 情報

表 40. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 41. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 カタログ・ノードのネットワーク名。

使用法 このエレメントを使用して、データベースのロケーションを判別できます。

db_location データベース・ロケーション

エレメント ID db_location

エレメント・タイプ 情報

表 42. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 アプリケーションに関連したデータベースのロケーション。

使用法 スナップショットをとるアプリケーションに対応する、データベース・サーバーの相対的なロケーションを判別します。次の値があります。

- SQLM_LOCAL
- SQLM_REMOTE

関連資料:

- 144 ページの『server_platform サーバーのオペレーティング・システム』

catalog_node カタログ・ノード番号

エレメント ID catalog_node

エレメント・タイプ 情報

表 43. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 44. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 データベース・カタログ表が保管されているノードのノード番号。

使用法 カタログ・ノードは、すべてのシステム・カタログ表が保管されているノードです。システム・カタログ表へのアクセスは、すべてこのノードを通る必要があります。

関連資料:

- 「コマンド・リファレンス」の『BACKUP DATABASE コマンド』

last_backup 最終バックアップ・タイム・スタンプ

エレメント ID last_backup

エレメント・タイプ タイム・スタンプ

表 45. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	タイム・スタンプ

説明 データベース・バックアップが最後に完了した日時。

使用法 このエレメントを使用すると、バックアップをしばらく行っていないデータベースを識別したり、最新のデータベース・バックアップ・ファイルを識別できます。データベースを一度もバックアップしていない場合は、このタイム・スタンプがゼロに初期化されます。

アプリケーションの識別および状況

アプリケーションの識別および状況に関するモニター・エレメント

次のエレメントにより、データベースおよび関連するアプリケーションに関する情報が提供されます。

- agent_id アプリケーション・ハンドル (エージェント ID) : モニター・エレメント
- appl_status アプリケーション状況 : モニター・エレメント
- codepage_id アプリケーションで使用するコード・ページ ID : モニター・エレメント
- status_change_time アプリケーション状況変更時刻 : モニター・エレメント
- appl_id_oldest_xact 最も古いトランザクションを持つアプリケーション : モニター・エレメント
- smallest_log_avail_node 使用可能なログ・スペースが最小のノード : モニター・エレメント
- appl_name アプリケーション名 : モニター・エレメント
- appl_id アプリケーション ID : モニター・エレメント
- sequence_no シーケンス番号 : モニター・エレメント
- auth_id 許可 ID : モニター・エレメント
- session_auth_id セッション許可 ID : モニター・エレメント
- client_nname クライアントの構成 NNAME : モニター・エレメント
- client_prdid クライアント製品/バージョン ID : モニター・エレメント
- client_db_alias アプリケーションで使用するデータベース別名 : モニター・エレメント
- host_prdid ホスト製品/バージョン ID : モニター・エレメント
- outbound_appl_id アウトバウンド・アプリケーション ID : モニター・エレメント
- outbound_sequence_no アウトバウンド・シーケンス番号 : モニター・エレメント
- execution_id ユーザー・ログイン ID : モニター・エレメント
- corr_token DRDA 相関トークン : モニター・エレメント
- client_pid クライアント・プロセス ID : モニター・エレメント
- client_platform クライアント・オペレーティング・プラットフォーム : モニター・エレメント
- client_protocol クライアント通信プロトコル : モニター・エレメント
- territory_code データベース・テリトリー・コード : モニター・エレメント
- appl_priority アプリケーション・エージェント優先順位 : モニター・エレメント
- appl_priority_type アプリケーション優先順位タイプ : モニター・エレメント
- authority_lvl ユーザー許可レベル : モニター・エレメント
- node_number ノード番号 : モニター・エレメント
- coord_node コーディネーター・ノード : モニター・エレメント
- appl_con_time 接続要求開始タイム・スタンプ : モニター・エレメント
- connections_top 同時接続の最大数 : モニター・エレメント

- conn_complete_time 接続要求完了タイム・スタンプ : モニター・エレメント
- prev_uow_stop_time 直前の作業単位完了タイム・スタンプ : モニター・エレメント
- uow_start_time 作業単位開始タイム・スタンプ : モニター・エレメント
- uow_stop_time 作業単位停止タイム・スタンプ : モニター・エレメント
- uow_elapsed_time 最新の作業単位の経過時間 : モニター・エレメント
- uow_comp_status 作業単位完了状況 : モニター・エレメント
- uow_status 作業単位の状況 : モニター・エレメント
- appl_idle_time アプリケーション・アイドル時間 : モニター・エレメント
- DB2 エージェント情報に関するモニター・エレメント

agent_id アプリケーション・ハンドル (エージェント ID)

エレメント ID agent_id

エレメント・タイプ 情報

表 46. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本
DCS アプリケーション	dcs_appl_info	基本

表 47. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-
ステートメント	event_stmt	-
ステートメント	event_subsection	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 システム全体での、アプリケーションの**ユニーク** ID。パーティションが 1 つだけのデータベースの場合は、この ID は 16 ビット・カウンターで構成されます。パーティションが複数のデータベースでは、この ID はコーディネーター・パーティション番号と 16 ビット・カウンターが連結されて構成されます。さらに、アプリケーションが 2 次接続を行う可能性のあるパーティションには、すべて同一の ID が使用されます。

使用法 アプリケーション・ハンドルを使用すると、アクティブ・アプリケーションを一意的に識別できます (アプリケーション・ハンドルは、エージェント ID と同義です)。

注: *agent_id* モニター・エレメントは、使用する DB2 のバージョンによって動作が異なります。バージョン SQLM_DBMON_VERSION1 または SQLM_DBMON_VERSION2 の DB2 から DB2 Universal Database (バージョン 5 またはそれ以上) のデータベースへスナップショットをとる時、戻される *agent_id* はアプリケーション ID として使用できず、

アプリケーションを提供するエージェントの *agent_pid* として有効です。このような場合、*agent_id* はバックレベルの互換性のために戻されますが、内部的に、DB2 Universal Database サーバーは、値を *agent_id* として認識しません。

この値は、エージェント ID を必要とする GET SNAPSHOT コマンドへの入力として使用できます。

またイベント・トレースを読み取るとき、イベント・レコードを指定のアプリケーションと一致させるために使用できます。

FORCE APPLICATION コマンドまたは API への入力として使用することができます。マルチノード・システムでは、アプリケーションが接続されているノードから、このコマンドを出すことができます。効力範囲はグローバルです。

appl_status アプリケーション状況

エレメント ID appl_status

エレメント・タイプ 情報

表 48. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本

表 49. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 アプリケーションの現在の状況。

使用法 このエレメントは、潜在的なアプリケーション問題の診断に役立ちます。このフィールドの値は以下のとおりです。

API 定数	説明
SQLM_CONNECTPEND	**データベース接続ペンディング：** アプリケーションはデータベース接続を開始していますが、要求は完了していません。
SQLM_CONNECTED	**データベース接続完了：** アプリケーションがデータベース接続を開始し、要求は完了しています。
SQLM_UOWEXEC	**作業単位実行中：** データベース・マネージャーが作業単位に代わって要求を実行中です。
SQLM_UOWWAIT	**作業単位の待機中：** データベース・マネージャーが作業単位のためにアプリケーション内で待機中です。通常、この状況はシステムがアプリケーションのコード内で実行中であることを示します。
SQLM_LOCKWAIT	**ロック待機：** 作業単位がロックを待機中です。ロックが付与されると、状況は直前の値に復帰します。

API 定数	説明
SQLM_COMMIT_ACT	**コミット・アクティブ：** 作業単位が、そのデータベース変更をコミットしています。
SQLM_ROLLBACK_ACT	**ロールバック・アクティブ：** 作業単位がデータベースの変更をロールバックしています。
SQLM_RECOMP	**再コンパイル中：** データベース・マネージャーがアプリケーションに代わってプランを再コンパイル (再バインド) 中です。
SQLM_COMP	**コンパイル中：** データベース・マネージャーはアプリケーションのために SQL ステートメントのコンパイルまたはプランのプリコンパイルを行っています。
SQLM_INTR	**要求が割り込みを受けました：** 要求の割り込みが進行しています。
SQLM_DISCONNECTPEND	**データベース切断ペンディング：** アプリケーションはデータベースの切断を開始していますが、コマンドの実行は完了していません。アプリケーションがデータベース切断コマンドを明示的に実行していない可能性があります。切断せずにアプリケーションが終了すると、データベース・マネージャーがデータベースとの接続を切断します。
SQLM_DECOUPLED	**エージェントから切り離し済み：** アプリケーションはエージェントから切り離されました。
SQLM_TPREP	**トランザクション準備済み：** 作業単位は、2 フェーズ・コミット・プロトコルの準備段階に入っているグローバル・トランザクションの一部です。
SQLM_THCOMT	**トランザクションを経験的にコミット：** 作業単位は、経験的にコミットされたグローバル・トランザクションの一部です。
SQLM_THABRT	**トランザクションを経験的にロールバック：** 作業単位は、経験的にロールバックされたグローバル・トランザクションの一部です。
SQLM_TEND	**トランザクション終了：** 作業単位は、終了したグローバル・トランザクションの一部ですが、 2 フェーズ・コミット・プロトコルの準備段階にはまだ入っていません。
SQLM_CREATE_DB	**データベース作成中：** エージェントは、データベース作成の要求を開始しましたが、要求はまだ完了していません。
SQLM_RESTART	**データベース再始動中：** アプリケーションは、クラッシュ・リカバリーを実行するためにデータベースを再始動しています。
SQLM_RESTORE	**データベースのリストア中：** アプリケーションは、バックアップ・イメージをデータベースにリストアしています。
SQLM_BACKUP	**データベースのバックアップ中：** アプリケーションはデータベースのバックアップを実行しています。

API 定数	説明
SQLM_LOAD	**データの高速ロード：** アプリケーションは、データベースにデータの『高速ロード』を実行中です。
SQLM_UNLOAD	**データの高速アンロード：** アプリケーションは、データベースからデータの『高速アンロード』を実行中です。
SQLM_IOERROR_WAIT	**表スペースを使用不可にするための待機：** アプリケーションが入出力エラーを検出し、特定の表スペースを使用不可にしようとしています。アプリケーションは、表スペースを使用不可にする前に、表スペース上のその他のすべてのアクティブなトランザクションが完了するまで待機する必要があります。
SQLM_QUIESCE_TABLESPACE	**表スペースの静止中：** アプリケーションが表スペース静止要求を実行中です。
SQLM_WAITFOR_REMOTE	**リモート・パーティション待ち：** アプリケーションは、パーティション・データベース・インスタンス内のリモート・パーティションからの応答を待っています。
SQLM_REMOTE_RQST	**リモート要求ペンディング：** アプリケーションはフェデレーテッド・データ・ソースからの結果を待っています。
SQLM_ROLLBACK_TO_SAVEPOINT	**セーブポイントへのロールバック：** アプリケーションがセーブポイントにロールバック中です。

関連資料:

- 157 ページの『status_change_time アプリケーション状況変更時刻』
- 374 ページの『stmt_operation/operation ステートメント操作』

codepage_id アプリケーションで使用するコード・ページ ID

エレメント ID codepage_id

エレメント・タイプ 情報

表 50. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本
DCS アプリケーション	dcs_appl_info	基本

表 51. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-
接続	event_connheader	-

説明 コード・ページ ID。

使用法 スナップショット・モニター・データの場合は、モニター対象のアプリケー

ションが開始されたパーティションでのコード・ページとなります。この ID は、リモート・アプリケーションの問題判別に使用できます。この情報を使用すると、アプリケーションのコード・ページとデータベースのコード・ページ (DRDA ホスト・データベースの場合はホスト CCSID) の間でデータ変換がサポートされるように指定できます。サポート対象のコード・ページについては、「*管理ガイド*」を参照してください。

イベント・モニター・データの場合は、イベント・データを収集したデータベースのコード・ページとなります。このエレメントを使用すると、使用中のイベント・モニター・アプリケーションの実行に使用しているコード・ページとデータベースが使用しているコード・ページが異なるコード・ページかどうかを判別できます。イベント・モニターが書き込むデータには、データベースのコード・ページが使用されます。使用しているイベント・モニター・アプリケーションが別のコード・ページを使用する場合は、データを読み取るのに文字変換が必要になる場合があります。

status_change_time アプリケーション状況変更時刻

エレメント ID status_change_time

エレメント・タイプ タイム・スタンプ

表 52. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	作業単位、タイム・スタンプ
ロック	appl_lock_list	作業単位、タイム・スタンプ
DCS アプリケーション	dcs_appl_info	作業単位、タイム・スタンプ

説明 アプリケーションが現在の状況になった日時。

使用法 このエレメントを使用して、アプリケーションが現在の状況になっている時間を判別できます。同じ状況が長時間にわたり継続している場合は、問題が起きている可能性があります。

関連資料:

- 154 ページの『appl_status アプリケーション状況』

appl_id_oldest_xact 最も古いトランザクションを持つアプリケーション

エレメント ID appl_id_oldest_xact

エレメント・タイプ 情報

表 53. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 最も古いトランザクションを持つアプリケーションのアプリケーション ID (アプリケーション・スナップショットからの *agent_id* 値に対応)。

使用法 このエレメントを使用すると、最も古いアクティブ・トランザクションを持

つアプリケーションを判別できます。このアプリケーションにログ・スペースの解放を強制することができます。ログ・スペースを大量に消費している場合は、アプリケーションを調べて、より頻繁にコミットするよう変更できるかどうか判別してください。

ロギングを保留しているトランザクションがない場合や、最も古いトランザクションがアプリケーション ID を持たない場合 (例えば、未確定トランザクションまたは非アクティブ・トランザクション) もあります。これらの場合には、このアプリケーションの ID はデータ・ストリームに返されません。

関連資料:

- 295 ページの『deadlocks デッドロック検出数』
- 314 ページの『agent_id_holding_lock ロックを保持しているエージェント ID』

smallest_log_avail_node 使用可能なログ・スペースが最小のノード

エレメント ID smallest_log_avail_node

エレメント・タイプ 情報

表 54. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 このエレメントはグローバル・スナップショットの場合にだけ戻され、使用可能なログ・スペースが最も少ない (バイト数) ノードを示します。

使用法 このエレメントと appl_id_oldest_xact を組み合わせて使用すると、データベースに十分なログ・スペースがあることを確認できます。グローバル・スナップショットでは、appl_id_oldest_xact、total_log_used、および total_log_available がこのノードの値に対応します。

関連資料:

- 284 ページの『total_log_used 使用されているログ・スペースの合計』
- 284 ページの『total_log_available 使用可能なログの合計』
- 157 ページの『appl_id_oldest_xact 最も古いトランザクションを持つアプリケーション』

appl_name アプリケーション名

エレメント ID appl_name

エレメント・タイプ 情報

表 55. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本

表 55. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

表 56. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 クライアントで実行中のアプリケーションの名前。データベースまたは DB2 Connect サーバーが識別できる名前です。

使用法 このエレメントと *appl_id* を使用すると、データ項目をアプリケーションに関連付けることができます。

クライアント/サーバー環境において、この名前はデータベース接続を確立する際にクライアントからサーバーに送られます。 CLI アプリケーションは、 SQLSetConnectAttr への呼び出しを使用して SQL_ATTR_INFO_PROGRAMNAME 属性を設定できます。サーバーへの接続が確立される前に SQL_ATTR_INFO_PROGRAMNAME が設定されると、指定された値は実際のクライアント・アプリケーション名をオーバーライドし、 *appl_name* モニター・エレメント中に表示されます。

クライアント・アプリケーションのコード・ページと実行中のデータベース・システム・モニターが使用しているコード・ページが異なる場合は、*appl_name* を変換するときに *codepage_id* を利用できます。

関連資料:

- 156 ページの『codepage_id アプリケーションで使用するコード・ページ ID』
- 159 ページの『appl_id アプリケーション ID』

appl_id アプリケーション ID

エレメント ID appl_id

エレメント・タイプ 情報

表 57. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
DCS アプリケーション	dcs_appl_info	基本
ロック	appl_lock_list	基本

表 58. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
接続	event_connheader	-
ステートメント	event_stmt	-
トランザクション	event_xact	-

表 58. イベント・モニター情報 (続き)

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 アプリケーションがデータベース・マネージャーのデータベースに接続したとき、または DDCS が DRDA データベースへの接続要求を受け取ったときにこの ID が生成されます。

使用法 この ID はクライアントとサーバーの両者により認識されるため、この ID を使用すると、アプリケーションのクライアント部分とサーバー部分を相関させることができます。 DDCS アプリケーションでアプリケーションのクライアント部分とサーバー部分を相関させるには outbound_appl_id も必要です。

この ID は、ネットワーク内ではユニークな ID です。アプリケーション ID にはさまざまな形式があり、データベース・マネージャー、DDCS、またはその両方を実行中のクライアントとサーバー・マシン間の通信プロトコルにより形式が異なります。どの形式の場合もピリオドで区切られた 3 つの部分で構成されます。

1. APPC

   形式 `Network.LU Name.Application instance`

   例 `CAIBMTOR.OSFDBX0.930131194520`

   詳細 このアプリケーション ID は、 APPC の会話が割り振られるとネットワーク上に流れる実 SNA LUWID (作業論理単位 ID) の表示可能な形式です。 APPC が生成するアプリケーション ID は、ネットワーク名、LU 名、および LUWID インスタンス番号が連結されて構成され、これによってクライアント/サーバー・アプリケーションのユニークなラベルが作成されます。ネットワーク名および LU 名に使用できる文字数は、それぞれ最大 8 文字です。アプリケーションのインスタンスは、 12 個の 10 進数文字を使用した LUWID インスタンス番号に対応します。

2. TCP/IP

   形式 `IPAddr.Port.Application instance`

   例 `G91A3955.F33A.02DD18143340`

   詳細 TCP/IP の生成するアプリケーション ID は、 3 つのセクションで構成されます。最初のセクションには IP アドレスが含まれます。 IP アドレスは、最大 8 個の 16 進文字を使用する 32 ビットの数値で表されます。 2 番目のセクションにはポート番号が含まれ、4 つの 16 進文字で表されます。 3 番目のセクションには、このアプリケーションのインスタンスのユニーク ID が含まれます。

注: 16 進数の IP アドレスまたはポート番号が 0 から 9 で始まる場合、それぞれ G から P に変更されます。例えば、″0″ は ″G″ に、″1″ は ″H″ にそれぞれマップされます。

IP アドレス AC10150C.NA04.006D07064947 は以下のように解釈されます。

- IP アドレスは AC10150C のままで、これは 172.16.21.12 に変換される。
- ポート番号は NA04。最初の文字 ″N″ は ″7″ にマップされます。したがって、16 進形式のポート番号は 7A04 となり、これは 10 進形式で 31236 に変換されます。

3. IPX/SPX

   形式 `Netid.nodeid.Application instance`

   例 `C11A8E5C.400011528250.0131214645`

   詳細 IPX/SPX が生成するアプリケーション ID は、文字ネットワーク ID (8 個の 16 進文字)、ノード ID (12 個の 16 進文字)、およびアプリケーション・インスタンスのユニーク ID の連結で構成されています。アプリケーション・インスタンスは、10 個の 10 進数文字を使用して、 `MMDDHHMMSS` の形式によるタイム・スタンプに対応します。

4. NetBIOS

   形式 `*NETBIOS.nname.Application instance`

   例 `*NETBIOS.SBOIVIN.930131214645`

   詳細 パーティションのないデータベース・システムの場合は、NetBIOS アプリケーション ID は、 『*NETBIOS』のストリング、クライアントのデータベース構成ファイルに定義されている nname、およびこのアプリケーションのインスタンスのユニーク ID の連結で構成されています。パーティション・データベース・システムの場合は、NetBIOS アプリケーション ID は、「Nxxx.etc」というストリングで構成されます。「xxx」はアプリケーションがアタッチされているパーティションです。

5. ローカル・アプリケーション

   形式 `*LOCAL.DB2 instance.Application instance`

   例 `*LOCAL.DB2INST1.930131235945`

   詳細 ローカル・アプリケーション用に生成されるアプリケーション ID は、 `*LOCAL` のストリング、DB2 インスタンスの名前、およびこのアプリケーションのインスタンスのユニーク ID の連結で構成されます。

   複数パーティション・インスタンスの場合は、 `LOCAL` は `Nx` に置き換えられます。 `x` は、クライアントからデー

タベースへの接続元のパーティション番号です。例えば、*N2.DB2INST1.0B5A12222841 となります。

client_protocol を使用すると、接続に使用している通信プロトコルを判別できるので、 *appl_id* の形式も判別できます。

関連資料:

- 166 ページの『outbound_appl_id アウトバウンド・アプリケーション ID』
- 169 ページの『client_protocol クライアント通信プロトコル』

sequence_no シーケンス番号

エレメント ID sequence_no

エレメント・タイプ 情報

表 59. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
DCS アプリケーション	dcs_appl_info	基本

表 60. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
接続	event_connheader	-
ステートメント	event_stmt	-
トランザクション	event_xact	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 作業単位が終了するごとに (つまり、 COMMIT または ROLLBACK が作業単位を終了するごとに) この ID が増加します。 appl_id と sequence_no を使用してトランザクションを一意的に識別します。

auth_id 許可 ID

エレメント ID auth_id

エレメント・タイプ 情報

表 61. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本
DCS アプリケーション	dcs_appl_info	基本

表 62. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 モニターされているアプリケーションを呼び出したユーザーの許可 ID。この ID は、DB2 Connect のゲートウェイ・ノード上では、ホスト上にあるユーザーの許可 ID となります。

使用法 このエレメントを使用すると、アプリケーションを呼び出したユーザーを判別できます。

関連資料:

• 158 ページの『appl_name アプリケーション名』

session_auth_id セッション許可 ID

エレメント ID session_auth_id

エレメント・タイプ 情報

表 63. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本

説明 このアプリケーションによって使用されている現在のセッション許可 ID。

使用法 このエレメントは、SQL ステートメントの準備、SQL ステートメントの実行、またはその両方を行うのにどの許可 ID が使用されているかを判別するのに役立ちます。

client_nname クライアントの構成 NNAME

エレメント ID client_nname

エレメント・タイプ 情報

表 64. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
DCS アプリケーション	dcs_appl_info	基本

表 65. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 クライアント・データベース・パーティションのデータベース・マネージャー構成ファイルにある *nname*。

使用法 このエレメントを使用すると、アプリケーションを実行中のクライアント・

データベース・パーティションを識別できます。このエレメントは、NetBIOS LAN 環境が存在する Windows 環境にのみ適用されます。

関連資料:

- 141 ページの『server_nname モニター対象の (サーバー) データベース・パーティションでの NNAME 構成』

client_prdid クライアント製品/バージョン ID

エレメント ID client_prdid

エレメント・タイプ 情報

表 66. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
DCS アプリケーション	dcs_appl_info	基本

表 67. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 クライアント上で実行中の製品とバージョン。

使用法 このエレメントを使用すると、データベース・クライアントの製品とコード・バージョンを識別できます。 PPPVVRRM の形式になっています。各部分の定義は次のとおりです。

- PPP は製品を示し、DB2 製品の場合は 『SQL』 である。
- VV は 2 桁でバージョン番号を示す (バージョン番号が 1 桁の場合には、高位の桁は 0 になります)。
- RR は 2 桁でリリース番号を示す (リリース番号が 1 桁の場合には、高位の桁は 0 になります)。
- M は 1 桁でモディフィケーション・レベルを示す。

関連資料:

- 143 ページの『server_prdid サーバー製品/バージョン ID』

client_db_alias アプリケーションで使用するデータベース別名

エレメント ID client_db_alias

エレメント・タイプ 情報

表 68. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_id_info	基本
ロック	appl_lock_list	基本

表 69. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 アプリケーションがデータベースに接続するときに指定するデータベースの別名。

使用法 このエレメントを使用して、アプリケーションがアクセスしている実際のデータベースを識別できます。この名前と *db_name* 間のマッピングには、クライアント・ノードとデータベース・マネージャーのサーバー・ノードにあるデータベース・ディレクトリーを使用します。

この別名は、データベース接続要求を発行したデータベース・マネージャー内に定義されている別名です。

データベースの別名が異なると認証タイプが異なるので、このエレメントを認証タイプの判別に利用することもできます。

関連資料:

- 146 ページの『db_name データベース名』
- 404 ページの『last_reset 最後のリセット・タイム・スタンプ』
- 404 ページの『input_db_alias 入力データベース別名』

host_prdid ホスト製品/バージョン ID

エレメント ID host_prdid

エレメント・タイプ 情報

表 70. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 サーバー上で実行中の製品とバージョン。

使用法 これを使用して、DRDA ホスト・データベース製品の製品とコード・バージョンを識別できます。 PPPVVRRM の形式になっています。各部分の定義は次のとおりです。

- PPP は、次のホスト DRDA 製品を示す。
  - ARI の場合 : DB2 Server for VSE & VM
  - DSN の場合 : DB2 for OS/390 and z/OS
  - QSQ の場合 : DB2 UDB for AS/400
  - SQL の場合 : 他の DB2 製品
- VV は 2 桁でバージョン番号を示す (バージョン番号が 1 桁の場合には、高位の桁は 0 になります)。
- RR は 2 桁でリリース番号を示す (リリース番号が 1 桁の場合には高位の桁は 0 になります)。
- M は 1 桁でモディフィケーション・レベルを示す。

outbound_appl_id アウトバウンド・アプリケーション ID

エレメント ID outbound_appl_id

エレメント・タイプ 情報

表 71. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 アプリケーションが DRDA ホスト・データベースに接続すると、この ID が生成されます。この ID は、DB2 Connect ゲートウェイをホストに接続するときに使用しますが、 *appl_id* はクライアントを DB2 Connect ゲートウェイに接続するときに使用します。

使用法 このエレメントと *appl_id* を組み合わせて使用すると、アプリケーション情報のクライアントの部分とサーバーの部分を関連付けることができます。

この ID は、ネットワーク内ではユニークな ID です。

ゲートウェイ・コンセントレーターがオンである場合、または DCS アプリケーションが作業論理単位内にない場合に、このエレメントはブランクになります。

形式 `Network.LU Name.Application instance`

例 `CAIBMTOR.OSFDBM0.930131194520`

詳細 このアプリケーション ID は、 APPC の会話が割り振られるとネットワーク上に流れる実 SNA LUWID (作業論理単位 ID) の表示可能な形式です。 APPC が生成するアプリケーション ID は、ネットワーク名、LU 名、および LUWID インスタンス番号が連結されて構成され、これによってクライアント/サーバー・アプリケーションのユニークなラベルが作成されます。ネットワーク名および LU 名に使用できる文字数は、それぞれ最大 8 文字です。アプリケーションのインスタンスは、 12 個の 10 進数文字を使用した LUWID インスタンス番号に対応します。

関連資料:

• 159 ページの『appl_id アプリケーション ID』

outbound_sequence_no アウトバウンド・シーケンス番号

エレメント ID outbound_sequence_no

エレメント・タイプ 情報

表 72. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 ゲートウェイ・コンセントレーターがオンである場合、または DCS アプリケーションが作業論理単位内にない場合に、このエレメントはブランクになります。

execution_id ユーザー・ログイン ID

エレメント ID execution_id

エレメント・タイプ 情報

表 73. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
アプリケーション	appl	基本
DCS アプリケーション	dcs_appl_info	基本

表 74. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 ユーザーがオペレーティング・システムにログインするときに指定した ID。この ID は、ユーザーがデータベースに接続するときに指定する auth_id とは異なります。

使用法 このエレメントを使用すると、モニター対象のアプリケーションを実行している個人のオペレーティング・システム・ユーザー ID を識別できます。

関連資料:

- 162 ページの『auth_id 許可 ID』

corr_token DRDA 相関トークン

エレメント ID corr_token

エレメント・タイプ 情報

表 75. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
アプリケーション	appl	基本

表 76. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 DRDA AS 相関トークン。

使用法 DRDA 相関トークンは、アプリケーション・サーバーとアプリケーション・リクエスターの間の処理の相関を求めるときに使用します。これはエラ

ーが発生したときにログにダンプされる ID であるため、エラーとなった会話を識別するのに利用できます。場合によっては、会話の LUWID を示します。

通信に DRDA を使用していない場合は、このエレメントが *appl_id* を戻します (appl_id 参照)。

データベース・システム・モニター API を使用すると、このエレメントの長さを判別するために API 定数の SQLM_APPLID_SZ が使用されることに注意してください。

client_pid クライアント・プロセス ID

エレメント ID client_pid

エレメント・タイプ 情報

表 77. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
アプリケーション	appl	基本
DCS アプリケーション	dcs_appl_info	基本

表 78. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 データベースへの接続を行ったクライアント・アプリケーションのプロセス ID。

使用法 このエレメントを使用して、CPU および入出力時間などのモニター情報をクライアント・アプリケーションに関連付けることができます。

DRDA AS 接続のとき、このエレメントは 0 に設定されます。

client_platform クライアント・オペレーティング・プラットフォーム

エレメント ID client_platform

エレメント・タイプ 情報

表 79. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
アプリケーション	appl	基本
DCS アプリケーション	dcs_appl_info	基本

表 80. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 クライアント・アプリケーションが稼働中のオペレーティング・システム。

使用法 このエレメントを使用して、リモート・アプリケーションの問題判別を行えます。このフィールドの値は、ヘッダー・ファイルの *sqlmon.h* にあります。

関連資料:

• 144 ページの『server_platform サーバーのオペレーティング・システム』

client_protocol クライアント通信プロトコル

エレメント ID client_protocol

エレメント・タイプ 情報

表 81. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
アプリケーション	appl	基本
DCS アプリケーション	dcs_appl_info	基本

表 82. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-

説明 クライアント・アプリケーションがサーバーとの通信に使用している通信プロトコル。

使用法 このエレメントを使用して、リモート・アプリケーションの問題判別を行えます。このフィールドの値は以下のとおりです。

API 定数	通信プロトコル
SQLM_PROT_UNKNOWN	(注 1)
SQLM_PROT_LOCAL	なし (注 2)
SQLM_PROT_APPC	APPC
SQLM_PROT_TCPIP	TCP/IP
SQLM_PROT_IPXSPX	IPX/SPX
SQLM_PROT_NETBIOS	NETBIOS

注:

1. クライアントは、不明のプロトコルを使用して通信を行っています。この値は、今後のクライアントが下位レベルのサーバーに接続した場合にのみ戻されます。
2. クライアントは、サーバーと同一のノードで実行されており、使用中の通信プロトコルはありません。

territory_code データベース・テリトリー・コード

エレメント ID territory_code

エレメント・タイプ 情報

表 83. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
アプリケーション	appl	基本

表 84. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-
接続	event_connheader	-

説明 モニター・データが収集されるデータベースのテリトリー・コード。このモニター・エレメントは以前は country_code と呼んでいました。

使用法 テリトリー・コード情報は、データベース構成ファイルに記録されています。

DRDA AS 接続の場合、このエレメントは 0 に設定されます。

appl_priority アプリケーション・エージェント優先順位

エレメント ID appl_priority

エレメント・タイプ 情報

表 85. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

表 86. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 このアプリケーションのために作業するエージェントの優先順位。

使用法 このエレメントを使用すると、アプリケーションが所定の優先順位で実行されているかどうかを確認できます。アプリケーションの優先順位は、管理者が設定できます。優先順位は、ガバナー・ユーティリティー (**db2gov**) で変更できます。

DB2 はガバナーを使用して、データベースに使用するアプリケーションの動作のモニターおよび変更を行います。この情報は、アプリケーションのスケジュール処理とシステム・リソースのバランス処理に使用されます。

ガバナー・デーモンはスナップショットをとることにより、アプリケーションの統計データを収集します。さらに、そのデータベース上で実行されるアプリケーションを管理する規則に照らして、この統計内容をチェックします。ガバナーが規則違反を発見すると、適切なアクションを実行します。このような規則やアクションの内容は、ガバナー構成ファイル内にユーザーが指定します。

規則に関連したアクションによりアプリケーションの優先順位が変更される場合、違反が検出されたパーティション内のエージェントの優先順位がガバナーによって変更されます。

関連資料:

- 171 ページの『appl_priority_type アプリケーション優先順位タイプ』

appl_priority_type アプリケーション優先順位タイプ

エレメント ID appl_priority_type

エレメント・タイプ 情報

表 87. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

表 88. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 アプリケーションのために作業しているエージェントの、オペレーティング・システムの優先順位タイプ。

使用法 使用状況に基づいて、動的優先順位がオペレーティング・システムによって再計算されます。静的優先順位は変更されません。

関連資料:

- 170 ページの『appl_priority アプリケーション・エージェント優先順位』
- 385 ページの『query_cost_estimate 照会コストの見積もり』

authority_lvl ユーザー許可レベル

エレメント ID authority_lvl

エレメント・タイプ 情報

表 89. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本
アプリケーション	appl_info	基本

表 90. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 アプリケーションに対して付与されている最高権限レベル。

使用法 アプリケーションで許可される操作は、直接または間接的に付与されます。

sql.h の次の定義を使用して、ユーザーに明示的に付与されている許可を判別できます。

- SQL_SYSADM
- SQL_DBADM
- SQL_CREATETAB
- SQL_BINDADD
- SQL_CONNECT
- SQL_CREATE_EXT_RT
- SQL_CREATE_NOT_FENC
- SQL_SYSCTRL
- SQL_SYSMAINT

sql.h の次の定義を使用して、グループまたは PUBLIC 権限から継承されている間接許可を判別できます。

- SQL_SYSADM_GRP
- SQL_DBADM_GRP
- SQL_CREATETAB_GRP
- SQL_BINDADD_GRP
- SQL_CONNECT_GRP
- SQL_CREATE_EXT_RT_GRP
- SQL_CREATE_NOT_FENC_GRP
- SQL_SYSCTRL_GRP
- SQL_SYSMAINT_GRP

関連概念:

- 「*管理ガイド: インプリメンテーション*」の『特権、権限レベル、およびデータベース権限』

node_number ノード番号

エレメント ID node_number

エレメント・タイプ 情報

表 91. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本
データベース・マネージャー	memory_pool	基本
データベース・マネージャー	fcm	基本
データベース・マネージャー	fcm_node	基本

表 91. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本
データベース	detail_log	基本
バッファー・プール	bufferpool_nodeinfo	バッファー・プール
表スペース	rollforward	基本
ロック	lock	基本
ロック	lock_wait	基本

表 92. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_connheader	-
デッドロック	lock	-
オーバーフロー・レコード	event_overflow	-
データベース	event_dbmemuse	-
接続	event_connmemuse	-

説明 *db2nodes.cfg* ファイル内でノードに割り当てられた番号。

使用法 この値は現在のノード番号を示しており、複数のノードをモニターするときにこの値を利用できます。

coord_node コーディネーター・ノード

エレメント ID coord_node

エレメント・タイプ 情報

表 93. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

表 94. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 マルチノード・システムでは、インスタンスに接続つまりアタッチされたアプリケーションがあるノードのノード番号。

使用法 接続されたアプリケーションは、それぞれ 1 つのコーディネーター・ノードにより処理されます。

appl_con_time 接続要求開始タイム・スタンプ

エレメント ID appl_con_time

エレメント・タイプ タイム・スタンプ

表 95. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	タイム・スタンプ

説明 アプリケーションが接続要求を開始した日時。

使用法 このエレメントを使用すると、アプリケーションがデータベースへの接続要求を開始した日時を判別できます。

関連資料:

- 174 ページの『conn_complete_time 接続要求完了タイム・スタンプ』

connections_top 同時接続の最大数

エレメント ID connections_top

エレメント・タイプ 水準点

表 96. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 97. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 データベースを活動化した時点からの、そのデータベースへの同時接続の最大数。

使用法 このエレメントは、「*管理ガイド*」に記載されている *maxappls* 構成パラメーターの設定値を評価するときに利用できます。

maxappls は、データベース接続数を制限するので、このエレメントの値が *maxappls* パラメーターと同じ場合は、一部のデータベース接続がリジェクトされていることを示します。

次の公式を使用すると、スナップショットを取った時点の接続数を計算できます。

```
rem_cons_in + local_cons
```

関連資料:

- 182 ページの『rem_cons_in データベース・マネージャーへのリモート接続』
- 183 ページの『local_cons ローカル接続』

conn_complete_time 接続要求完了タイム・スタンプ

エレメント ID conn_complete_time

エレメント・タイプ タイム・スタンプ

表 98. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	タイム・スタンプ

説明 接続要求が認可された日時。

使用法 このエレメントを使用すると、データベースへの接続要求が認可された日時を判別できます。

関連資料:

- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』

prev_uow_stop_time 直前の作業単位完了タイム・スタンプ

エレメント ID prev_uow_stop_time

エレメント・タイプ タイム・スタンプ

表 99. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位、タイム・スタンプ
DCS アプリケーション	dcs_appl	作業単位、タイム・スタンプ

説明 作業単位が完了した時刻です。

使用法 このエレメントと *uow_stop_time* を組み合わせて使用すると、 COMMIT と ROLLBACK のポイント間の合計経過時間を計算できます。 *uow_start_time* と組み合わせて使用すると、作業単位間のアプリケーションの合計時間を計算できます。次のいずれかの時刻を示します。

- アプリケーションが作業単位中の場合は、最後に作業単位が完了した時刻を示す。
- アプリケーションが作業単位中ではない場合は (アプリケーションが 1 つの作業単位を完了し、次の作業単位をまだ開始していない場合)、最後に完了した作業単位の直前の作業単位の停止時刻を示す。最後に完了した作業単位の停止時刻は、uow_stop_time で示す。
- アプリケーションが最初の作業単位中の場合は、データベース接続要求完了時刻となる。

関連資料:

- 174 ページの『conn_complete_time 接続要求完了タイム・スタンプ』
- 175 ページの『uow_start_time 作業単位開始タイム・スタンプ』
- 176 ページの『uow_stop_time 作業単位停止タイム・スタンプ』

uow_start_time 作業単位開始タイム・スタンプ

エレメント ID uow_start_time

エレメント・タイプ タイム・スタンプ

表 100. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位、タイム・スタンプ
DCS アプリケーション	dcs_appl	作業単位、タイム・スタンプ

説明 作業単位が最初にデータベース・リソースを要求した日時。

使用法 このリソース要求は、その作業単位で SQL ステートメントを初めて実行したときに発生します。

- 最初の作業単位の場合は、 *conn_complete_time* の後の最初のデータベース要求 (SQL ステートメントの実行) の時刻。
- その後の作業単位の場合は、前回の COMMIT または ROLLBACK の後の最初のデータベース要求 (SQL ステートメントの実行) の時刻。

注: 「*SQL リファレンス*」は、作業単位の境界を COMMIT または ROLLBACK のポイントとして定義します。

データベース・システム・モニターでは、 COMMIT/ROLLBACK とその作業単位定義から出される次の SQL ステートメントまでの経過時間を除外します。この測定方式により、データベース・マネージャーがデータベース要求の処理に要する時間を、その作業単位の最初の SQL ステートメント以前にアプリケーション・ロジック内で要する時間とは切り離して反映します。作業単位の経過時間には、作業単位内で SQL ステートメント間のアプリケーション・ロジックを実行する時間が含まれます。

このエレメントと *uow_stop_time* を組み合わせると、作業単位の合計経過時間を計算できます。 *prev_uow_stop_time* と組み合わせると、作業単位間にアプリケーションで要した時間を計算できます。

uow_stop_time と *prev_uow_stop_time* を組み合わせると、 SQL リファレンスの定義による作業単位の経過時間を計算できます。

関連資料:

- 174 ページの『conn_complete_time 接続要求完了タイム・スタンプ』
- 175 ページの『prev_uow_stop_time 直前の作業単位完了タイム・スタンプ』
- 176 ページの『uow_stop_time 作業単位停止タイム・スタンプ』

uow_stop_time 作業単位停止タイム・スタンプ

エレメント ID uow_stop_time

エレメント・タイプ タイム・スタンプ

表 101. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位、タイム・スタンプ
DCS アプリケーション	dcs_appl	作業単位、タイム・スタンプ

説明 最新の作業単位が完了した日時。これが起こるのはデータベースの変更がコミットまたはロールバックされたときです。

使用法 このエレメントと *prev_uow_stop_time* を組み合わせて使用すると、COMMIT/ROLLBACK ポイント間の合計経過時間を計算できます。*uow_start_time* と組み合わせると、前回の作業単位の経過時間を計算できます。

タイム・スタンプの内容は、次のように設定されます。

- アプリケーションが 1 つの作業単位を完了し、次の作業単位をまだ開始していない場合 (*uow_start_time* の定義で)、このエレメントはゼロ以外の有効なタイム・スタンプになる。
- アプリケーションが作業単位を実行中の場合は、このエレメントにはゼロが含まれる。
- アプリケーションがデータベースに初めて接続すると、このエレメントは *conn_complete_time* に設定される。

新しい作業単位が開始すると、このエレメントの内容は、*prev_uow_stop_time* に移動します。

関連資料:

- 174 ページの『conn_complete_time 接続要求完了タイム・スタンプ』
- 175 ページの『prev_uow_stop_time 直前の作業単位完了タイム・スタンプ』
- 175 ページの『uow_start_time 作業単位開始タイム・スタンプ』

uow_elapsed_time 最新の作業単位の経過時間

エレメント ID uow_elapsed_time

エレメント・タイプ 時間

表 102. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位、タイム・スタンプ
DCS アプリケーション	dcs_appl	作業単位、タイム・スタンプ

説明 最後に完了した作業単位の実行経過時間。

使用法 作業単位の完了にかかる時間の標識として、このエレメントを使用します。

関連資料:

- 459 ページの『gw_comm_errors 通信エラー』
- 460 ページの『gw_comm_error_time 通信エラー時刻』

uow_comp_status 作業単位完了状況

エレメント ID uow_comp_status

エレメント・タイプ 情報

表 103. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位
DCS アプリケーション	dcs_appl	基本

表 104. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
トランザクション	event_xact	-

説明 作業単位の状況およびそれが停止したときの状況。

使用法 このエレメントを使用すると、作業単位が終了した原因がデッドロックによるものか、または異常終了によるものかを判別できます。次の原因が考えられます。

- コミット・ステートメントによりコミットされた。
- ロールバック・ステートメントによりロールバックされた。
- デッドロックによりロールバックされた。
- 異常終了によりロールバックされた。
- アプリケーションの正常終了によりコミットされた。
- 進行中であった作業単位に対する FLUSH EVENT MONITOR コマンドの結果が不明。

注: API ユーザーは、データベース・システム・モニターの定数の定義が含まれているヘッダー・ファイル (*sqlmon.h*) を参照してください。

uow_status 作業単位の状況

エレメント ID uow_status

エレメント・タイプ 情報

表 105. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
トランザクション	event_xact	-

説明 作業単位の状況。

使用法 このエレメントを使用すると、作業単位の状況を判別できます。

appl_idle_time アプリケーション・アイドル時間

エレメント ID appl_idle_time

エレメント・タイプ 情報

表 106. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ステートメント
DCS アプリケーション	dcs_appl	ステートメント

説明 アプリケーションがサーバーに対して何らかの要求を出してから経過した秒

数。これには、トランザクションを終了せずに、例えばコミットまたはロールバックを発行していないアプリケーションが含まれます。

使用法 この情報を使用すると、ユーザーが指定秒数の間アイドル状態になったときに、ユーザーに強制するようなアプリケーションをインプリメントできます。

関連資料:

- 170 ページの『appl_priority アプリケーション・エージェント優先順位』
- 171 ページの『appl_priority_type アプリケーション優先順位タイプ』
- 385 ページの『query_cost_estimate 照会コストの見積もり』

DB2 エージェントの情報

DB2 エージェント情報に関するモニター・エレメント

以下のデータベース・システム・モニター・エレメントにより、エージェントに関する情報が提供されます。

- agent_pid プロセスまたはスレッド ID : モニター・エレメント
- coord_agent_pid コーディネーター・エージェント : モニター・エレメント

agent_pid プロセスまたはスレッド ID

エレメント ID agent_pid

エレメント・タイプ 情報

表 107. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	agent	ステートメント

説明 DB2 エージェントのプロセス ID (UNIX システム) またはスレッド ID (Windows システム)。

使用法 このエレメントを使用すると、データベース・システム・モニター情報をシステム・トレースなどの他の診断情報のソースにリンクできます。また、このエレメントを使用すると、データベース・アプリケーションの処理を行うエージェントがシステム・リソースをどのように使用しているのかをモニターできます。

関連資料:

- 179 ページの『coord_agent_pid コーディネーター・エージェント』

coord_agent_pid コーディネーター・エージェント

エレメント ID coord_agent_pid

エレメント・タイプ 情報

表 108. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本

説明 アプリケーションのコーディネーター・エージェントのプロセス ID (UNIX システム) またはスレッド ID (Windows システム)。

使用法 このエレメントを使用すると、データベース・システム・モニター情報をシステム・トレースなどの他の診断情報のソースにリンクできます。

関連資料:

• 179 ページの『agent_pid プロセスまたはスレッド ID』

データベース・マネージャー構成

データベース・マネージャー構成に関するモニター・エレメント

次のエレメントにより、データベース・マネージャーの構成情報が提供されます。

- エージェントおよび接続に関するモニター・エレメント
- メモリー・プールに関するモニター・エレメント
- ソートに関するモニター・エレメント
- ハッシュ結合に関するモニター・エレメント
- 高速コミュニケーション・マネージャーに関するモニター・エレメント
- ユーティリティーに関するモニター・エレメント

エージェントおよび接続

エージェントおよび接続に関するモニター・エレメント

エージェントは、クライアント・アプリケーションが作成した要求を実行するプロセスまたはスレッドです。接続されたアプリケーションは、それぞれ 1 つの**コーディネーター・エージェント**と、場合によってはさらに 1 組のサブコーディネーター・エージェント、つまり**サブエージェント**のサービスを受けることもあります。サブエージェントは、パーティション・データベース内および SMP マシン上で、並列 SQL の処理に使用されます。エージェントは次のように分類できます。

- **コーディネーター・エージェント**

  ローカル・アプリケーションまたはリモート・アプリケーションが接続する最初のエージェントです。データベース接続またはインスタンスのアタッチごとに専用のコーディネーター・エージェントが 1 つずつあります。パーティションごとのコーディネーター・エージェントの最大数は、 *max_coordagents* 構成パラメーターによりコントロールされます。

- **サブエージェント**

  パーティション・データベースでは、コーディネーター・エージェントがエージェントを追加することによって、 SQL 処理の速度を上げることができます。エージェント・プールからサブエージェントが選択され、処理が終わるとエージェント・プールに戻されます。エージェント・プールのサイズは、 *num_poolagents* 構成パラメーターによってコントロールされます。

- **関連エージェント**

  アプリケーションの処理を実行するコーディネーターまたはサブエージェントは、そのアプリケーションに関連付けられます。アプリケーションの処理が終了すると、関連エージェントとしてエージェント・プールに入ります。アプリケー

ションがさらに処理を実行しようとすると、 DB2 がエージェント・プールの中からアプリケーションにすでに関連付けられているエージェントを検索し、そのエージェントに処理を割り当てます。 1 つも見つからないと、DB2 は、次の処理により、要求を満たすエージェントを探します。

1. アプリケーションに関連付けられていないアイドルのエージェントを選択します。
2. アイドル状態のエージェントがない場合は、エージェントを作成します。
3. ほかのアプリケーションに関連付けられているエージェントを見つけます。例えば、 *maxagents* に達しているためにエージェントを作成できない場合、DB2 はほかのアプリケーションに関連付けられているアイドル状態のエージェントを取ろうとします。これを**スチールされたエージェント**と呼びます。

- **プライム状態のエージェント**

  リモート・データベース上での作業を見込んで DRDA データベースに接続されている DRDA 接続プール内のゲートウェイ・エージェント。

maxagents 構成パラメーターは、タイプとは無関係に、インスタンスに対して存在可能なエージェントの最大数を定義します。 *maxagents* の値では、エージェントは作成されません。 DB2START のときにエージェント・プール内に作成される初期のエージェント数は、 *num_initagents* 構成パラメーターによって決まります。

max_coordagents に達していなければ、アイドル状態のエージェントがないことを想定して、それぞれの接続で新しいエージェントが作成されます。サブエージェントが使用されない場合は、 *max_coordagents* と *maxagents* が等しくなります。サブエージェントが使用されると、コーディネーター・エージェントとサブエージェントの組み合わせによっては *maxagents* に到達するものがあります。

1 つのエージェントに処理が割り当てられると、エージェントはトークンまたは許可を取得してトランザクションを処理しようとします。データベース・マネージャーは *maxcagents* 構成パラメーターを使用して、使用可能なトークン数をコントロールします。使用可能なトークンがない場合は、使用可能なトークンが現れるまでエージェントはスリープ状態となり、使用可能なトークンが現れると要求された処理が実行されます。これにより、ユーザーは *maxcagents* を使用して、サーバーで処理するロード、つまり同時に実行するトランザクション数をコントロールできます。

次のエレメントにより、エージェントおよび接続に関する情報が提供されます。

- rem_cons_in データベース・マネージャーへのリモート接続 : モニター・エレメント
- rem_cons_in_exec データベース・マネージャーで実行中のリモート接続 : モニター・エレメント
- local_cons ローカル接続 : モニター・エレメント
- local_cons_in_exec データベース・マネージャーで実行中のローカル接続 : モニター・エレメント
- con_local_dbases 現行接続を持つローカル・データベース：モニター・エレメント
- total_cons データベース活動化以降の接続 : モニター・エレメント
- appls_cur_cons 現在接続されているアプリケーション : モニター・エレメント
- appls_in_db2 データベースで現在実行中のアプリケーション : モニター・エレメント
- agents_registered 登録済みエージェント : モニター・エレメント

- agents_waiting_on_token トークン待ちエージェント : モニター・エレメント
- agents_registered_top エージェント最大登録数 : モニター・エレメント
- agents_waiting_top エージェント最大待機数 : モニター・エレメント
- idle_agents アイドル・エージェント数 : モニター・エレメント
- agents_from_pool プールから割り当てられたエージェント : モニター・エレメント
- agents_created_empty_pool エージェント・プールが空のために作成されたエージェント : モニター・エレメント
- coord_agents_top コーディネーター・エージェント最大数 : モニター・エレメント
- agents_stolen スチールされたエージェント : モニター・エレメント
- associated_agents_top 関連エージェント最大数 : モニター・エレメント
- comm_private_mem コミット済み専用メモリー : モニター・エレメント
- total_sec_cons 2 次接続 : モニター・エレメント
- num_assoc_agents 関連したエージェント数 : モニター・エレメント
- max_agent_overflows 最大エージェント・オーバーフロー回数 : モニター・エレメント
- num_gw_conn_switches - 接続切り替え回数 : モニター・エレメント

rem_cons_in データベース・マネージャーへのリモート接続

エレメント ID rem_cons_in

エレメント・タイプ ゲージ

表 109. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 モニター中のデータベース・マネージャーのインスタンスに対してリモート・クライアントから開始された接続の現在の数。

使用法 リモート・クライアントからこのインスタンス内のデータベースへの接続数を示します。この値は頻繁に変化するため、システム使用量の現実的な値を得るためには、ある一定の時間の中で特定のインターバルを設けてサンプルを採取する必要があります。この数値には、データベース・マネージャーと同じインスタンスで開始したアプリケーションは含まれません。

これらのエレメントと local_cons モニター・エレメントを組み合わせて使用すると、「*管理ガイド*」に記載されている *max_coordagents* 構成パラメーターの設定値を調整するときに利用できます。

関連資料:

- 183 ページの『rem_cons_in_exec データベース・マネージャーで実行中のリモート接続』
- 183 ページの『local_cons ローカル接続』
- 184 ページの『local_cons_in_exec データベース・マネージャーで実行中のローカル接続』

rem_cons_in_exec データベース・マネージャーで実行中のリモート接続

エレメント ID rem_cons_in_exec

エレメント・タイプ *ゲージ*

表 110. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 モニター中のデータベース・マネージャー・インスタンス内のデータベースに現在接続していて、作業単位を現在処理しているリモート・アプリケーションの数。

使用法 この数値は、データベース・マネージャーで実行中の並行処理のレベルを判別するときに利用できます。この値は頻繁に変化するため、システム使用量の現実的な値を得るためには、ある一定の時間の中で特定のインターバルを設けてサンプルを採取する必要があります。この数値には、データベース・マネージャーと同じインスタンスで開始したアプリケーションは含まれません。

このエレメントと local_cons_in_exec モニター・エレメントを組み合わせて使用すると、「*管理ガイド*」に記載されている *maxcagents* 構成パラメーターの設定値を調整するときに利用できます。

関連資料:

- 182 ページの『rem_cons_in データベース・マネージャーへのリモート接続』
- 183 ページの『local_cons ローカル接続』
- 184 ページの『local_cons_in_exec データベース・マネージャーで実行中のローカル接続』

local_cons ローカル接続

エレメント ID local_cons

エレメント・タイプ *ゲージ*

表 111. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 モニター中のデータベース・マネージャー・インスタンス内のデータベースに現在接続しているローカル・アプリケーションの数。

使用法 この数は、データベース・マネージャーで発生している並行処理のレベルを判別するのに役立ちます。この値は頻繁に変化するため、システム使用量の現実的な値を得るためには、ある一定の時間の中で特定のインターバルを設けてサンプルを採取する必要があります。

この数値に含まれるのは、データベース・マネージャーと同じインスタンスで開始したアプリケーションだけです。アプリケーションは接続されていますが、データベース内で作業単位を実行している場合としていない場合があります。

このエレメントと rem_cons_in モニター・エレメントを組み合わせて使用すると、「*管理ガイド*」に記載されている *maxagents* 構成パラメーターの設定値を調整するときに利用できます。

関連資料:

- 182 ページの『rem_cons_in データベース・マネージャーへのリモート接続』
- 183 ページの『rem_cons_in_exec データベース・マネージャーで実行中のリモート接続』
- 184 ページの『local_cons_in_exec データベース・マネージャーで実行中のローカル接続』

local_cons_in_exec データベース・マネージャーで実行中のローカル接続

エレメント ID local_cons_in_exec

エレメント・タイプ ゲージ

表 112. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 モニター中のデータベース・マネージャー・インスタンス内のデータベースに現在接続していて、作業単位を現在処理しているローカル・アプリケーションの数。

使用法 この数は、データベース・マネージャーで発生している並行処理のレベルを判別するのに役立ちます。この値は頻繁に変化するため、システム使用量の現実的な値を得るためには、ある一定の時間の中で特定のインターバルを設けてサンプルを採取する必要があります。この数値に含まれるのは、データベース・マネージャーと同じインスタンスで開始したアプリケーションだけです。

このエレメントと rem_cons_in_exec モニター・エレメントを組み合わせて使用すると、「*管理ガイド*」に記載されている *maxcagents* 構成パラメーターの設定値を調整するときに利用できます。

関連資料:

- 182 ページの『rem_cons_in データベース・マネージャーへのリモート接続』
- 183 ページの『rem_cons_in_exec データベース・マネージャーで実行中のリモート接続』
- 183 ページの『local_cons ローカル接続』

con_local_dbases 現行接続を持つローカル・データベース

エレメント ID con_local_dbases

エレメント・タイプ ゲージ

表 113. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 アプリケーションが接続されているローカル・データベースの数。

使用法 この数値は、データベース・レベルでデータを収集する際に、予想されるデータベース情報レコードの数を示します。

アプリケーションは、ローカルまたはリモートで実行中ですが、データベース・マネージャー内で作業単位を実行している場合としていない場合があります。

total_cons データベース活動化以降の接続

エレメント ID total_cons

エレメント・タイプ カウンター

表 114. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 115. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 最初の接続、活動化、または最後のリセット (コーディネーター・エージェント) 以降のデータベースへの接続数を示します。

使用法 このエレメントと db_conn_time および db2start_time モニター・エレメントを組み合わせて使用すると、アプリケーションがデータベースに接続する頻度を計算できます。

接続の頻度が少ない場合は、他のアプリケーションに接続する前に、ACTIVATE DATABASE コマンドを使用して、データベースを活動化することもできます。これは、初めてデータベースに接続する際に時折生じる追加のオーバーヘッド (初期バッファー・プール割り振りなど) のためです。こうすると、それ以後の接続処理の速度を上げることができます。

注: このエレメントをリセットすると、値はゼロではなく、現在接続中のアプリケーションの数に設定されます。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 186 ページの『appls_cur_cons 現在接続されているアプリケーション』
- 186 ページの『appls_in_db2 データベースで現在実行中のアプリケーション』
- 192 ページの『total_sec_cons 2 次接続』

appls_cur_cons 現在接続されているアプリケーション

エレメント ID appls_cur_cons

エレメント・タイプ ゲージ

表 116. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
ロック	db_lock_list	基本

説明 現在データベースに接続されているアプリケーションの数を示します。

使用法 このエレメントを使用して、データベース内のアクティビティー・レベルおよび使用中のシステム・リソースの量を確認することができます。

「*管理ガイド*」に記載されている *maxappls* および *max_coordagents* 構成パラメーターの設定値を調整するときに利用できます。例えば、この値が *maxappls* の値と常に同じ場合は、 *maxappls* の値を増やすことができます。詳しくは、『*rem_cons_in*』および『*local_cons*』モニター・エレメントの項を参照してください。

関連資料:

- 182 ページの『rem_cons_in データベース・マネージャーへのリモート接続』
- 183 ページの『local_cons ローカル接続』
- 185 ページの『total_cons データベース活動化以降の接続』
- 186 ページの『appls_in_db2 データベースで現在実行中のアプリケーション』

appls_in_db2 データベースで現在実行中のアプリケーション

エレメント ID appls_in_db2

エレメント・タイプ ゲージ

表 117. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 現在データベースに接続されており、データベース・マネージャーが要求を処理中のアプリケーションの数を示します。

使用法 このエレメントを使用すると、このデータベースに接続されているアプリケーションで使用されているデータベース・マネージャーのエージェント・トークンの数がわかります。 *rem_cons_in_exec* および *local_cons_in_exec* の合計が *maxcagents* 構成パラメーターの値と同じ場合は、「*管理ガイド*」の説明に従って、そのパラメーターの値を増やす必要があります。

関連資料:

- 183 ページの『rem_cons_in_exec データベース・マネージャーで実行中のリモート接続』
- 184 ページの『local_cons_in_exec データベース・マネージャーで実行中のローカル接続』
- 185 ページの『total_cons データベース活動化以降の接続』
- 186 ページの『appls_cur_cons 現在接続されているアプリケーション』
- 312 ページの『locks_waiting ロックで待機中の現行エージェント』

agents_registered 登録済みエージェント

エレメント ID agents_registered

エレメント・タイプ ゲージ

表 118. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 モニター中のデータベース・マネージャー・インスタンスに登録されているエージェント (コーディネーター・エージェントとサブエージェント) の数。

使用法 このエレメントは、 *maxagents* 構成パラメーターの設定値を評価するときに利用できます。

関連資料:

- 188 ページの『agents_registered_top エージェント最大登録数』

agents_waiting_on_token トークン待ちエージェント

エレメント ID agents_waiting_on_token

エレメント・タイプ ゲージ

表 119. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 データベース・マネージャー内でトランザクションを実行するためにトークンを待機中のエージェントの数。

使用法 このエレメントを使用すると、*maxcagents* 構成パラメーターの設定値を評価するのに役立ちます。

各アプリケーションには、データベース・マネージャー内でデータベース要求を処理するための専用コーディネーター・エージェントが 1 つずつ組み込まれます。各エージェントは、トークンを取得してから、トランザクションを実行できます。データベース・マネージャーのトランザクションを実行

できるエージェントの最大数は、 *maxcagents* 構成パラメーターの値によって制限されます。このパラメーターについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 187 ページの『agents_registered 登録済みエージェント』

agents_registered_top エージェント最大登録数

エレメント ID agents_registered_top

エレメント・タイプ 水準点

表 120. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 データベース・マネージャーが開始されてからこれまでに同時に登録されていたエージェント (コーディネーター・エージェントとサブエージェント) の最大数。

使用法 このエレメントは、「*管理ガイド*」に記載されている *maxagents* 構成パラメーターの設定値を評価するときに利用できます。

スナップショットの実行時に登録されたエージェントの数は、agents_registered により記録されます。

関連資料:

- 187 ページの『agents_registered 登録済みエージェント』
- 188 ページの『agents_waiting_top エージェント最大待機数』

agents_waiting_top エージェント最大待機数

エレメント ID agents_waiting_top

エレメント・タイプ 水準点

表 121. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 データベース・マネージャーが開始されてから、同時にトークンを待機していたエージェントの最大数。

使用法 このエレメントは、「*管理ガイド*」に記載されている *maxcagents* 構成パラメーターの設定値を評価するときに利用できます。

スナップショットの実行時にトークンを待機していたエージェントの数は、*agents_waiting_on_token* により記録されます。

maxcagents パラメーターをデフォルト値 (-1) に設定すると、トークンを待つエージェントがなくなるため、このモニター・エレメントの値はゼロになります。

関連資料:

- 187 ページの『agents_waiting_on_token トークン待ちエージェント』
- 188 ページの『agents_registered_top エージェント最大登録数』

idle_agents アイドル・エージェント数

エレメント ID idle_agents

エレメント・タイプ ゲージ

表 122. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 アプリケーションにまだ割り当てられておらず、『アイドル』状態でエージェント・プール内に存在するエージェントの数。

使用法 このエレメントは、 *num_poolagents* 構成パラメーターを設定するときに利用できます。利用可能なエージェントを使用してエージェントの要求を処理できるので、パフォーマンスが向上します。詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 187 ページの『agents_registered 登録済みエージェント』
- 188 ページの『agents_registered_top エージェント最大登録数』
- 188 ページの『agents_waiting_top エージェント最大待機数』

agents_from_pool プールから割り当てられたエージェント

エレメント ID agents_from_pool

エレメント・タイプ カウンター

表 123. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 エージェント・プールから割り当てられたエージェントの数。

使用法 このエレメントと *agents_created_empty_pool* を組み合わせて使用すると、プールが空になってエージェントの作成が必要となる頻度を判別できます。

次の比率

エージェント・プールが空のために作成されたエージェント/プールから割り当てられたエージェント

が高い場合は、 *num_poolagents* 構成パラメーターの値を大きくする必要があることを示します。比率が低い場合は、*num_poolagents* の設定が高すぎて、プール内にほとんど使用されないエージェントがあり、システム・リソースがむだになっていることを示します。

比率が高い場合は、このノードの総合ワークロードが高すぎることを示します。ワークロードは、 *maxcagents* 構成パラメーターが指定するコーディネーター・エージェントの最大数を小さくするか、または各ノードにデータを再分散することで調整できます。

エージェント・プール・サイズ (*num_poolagents*) および同時コーディネーター・エージェントの最大数 (*maxcagents*) の各構成パラメーターについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 190 ページの『agents_created_empty_pool - Agents Created Due to Empty Agent Pool』
- 190 ページの『coord_agents_top コーディネーター・エージェント最大数』

agents_created_empty_pool - Agents Created Due to Empty Agent Pool

エレメント ID agents_created_empty_pool

エレメント・タイプ カウンター

表 124. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 エージェント・プールが空だったために作成されたエージェントの数。これには、 DB2 を始動したときに開始したエージェントの数が含まれます (*num_initagents)*。

使用法 agents_from_pool と組み合わせて使用すると、次の比率を計算できます。

エージェント・プールが空のために作成されたエージェント/プールから割り当てられたエージェント

このエレメントの使用法については、『agents_from_pool』を参照してください。

関連資料:

- 189 ページの『agents_from_pool プールから割り当てられたエージェント』
- 190 ページの『coord_agents_top コーディネーター・エージェント最大数』

coord_agents_top コーディネーター・エージェント最大数

エレメント ID coord_agents_top

エレメント・タイプ 水準点

表 125. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
データベース	dbase	基本

説明 同時に動作できるコーディネーター・エージェントの最大数。

使用法 コーディネーター・エージェントの最大値がこのノードのワークロードとして大きすぎる場合は、 *maxcagents* 構成パラメーターを変更することで、トランザクションを同時に実行する数を減らすことができます。

同時コーディネーター・エージェントの最大数 (*maxcagents*) 構成パラメーターについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 189 ページの『agents_from_pool プールから割り当てられたエージェント』
- 190 ページの『agents_created_empty_pool - Agents Created Due to Empty Agent Pool』

agents_stolen スチールされたエージェント

エレメント ID agents_stolen

エレメント・タイプ カウンター

表 126. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 エージェントがアプリケーションからスチールされた回数。アプリケーションに関連付けられているエージェントがほかのアプリケーションの処理に再割り当てされるとエージェントがスチールされます。

使用法 このエレメントと *associated_agents_top* を組み合わせて使用すると、このアプリケーションがシステムに与えている負荷状態を評価できます。

agents_stolen が高い場合は、 *num_poolagents* 構成パラメーターの値を大きくすることを考慮してください。

関連資料:

- 394 ページの『num_agents ステートメントで作動しているエージェントの数』

associated_agents_top 関連エージェント最大数

エレメント ID associated_agents_top

エレメント・タイプ 水準点

表 127. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

説明 このアプリケーションに関連付けられているサブエージェントの最大数。

使用法 サブエージェントの最大数が *num_poolagents* 構成パラメーターの値に近い場合は、その数がノードのワークロードには大きすぎることを示す場合があります。

エージェント・プール・サイズ (*num_poolagents*) 構成パラメーターについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 189 ページの『agents_from_pool プールから割り当てられたエージェント』
- 190 ページの『agents_created_empty_pool - Agents Created Due to Empty Agent Pool』

comm_private_mem コミット済み専用メモリー

エレメント ID comm_private_mem

エレメント・タイプ ゲージ

表 128. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 スナップショット時点で、データベース・マネージャーのインスタンスが現在コミットしている専用メモリーの量。

使用法 このエレメントを使用すると、 *min_priv_mem* 構成パラメーター (「*管理ガイド*」を参照) を設定するときに、利用可能な専用メモリーが十分にあるかどうかを確認できます。このエレメントはすべてのプラットフォームで戻されますが、調整を行うことができるのは DB2 がスレッドを使用するプラットフォーム (Windows 2000 など) に限られます。

total_sec_cons 2 次接続

エレメント ID total_sec_cons

エレメント・タイプ カウンター

表 129. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 サブエージェントがノードのデータベースに接続した数。

使用法 このエレメントと total_cons、db_conn_time、および db2start_time のモニター・エレメントを組み合わせて使用すると、各アプリケーションがデータベースに接続した頻度を計算できます。

関連資料:

- 140 ページの『db2start_time データベース・マネージャー開始タイム・スタンプ』
- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 185 ページの『total_cons データベース活動化以降の接続』

num_assoc_agents 関連したエージェント数

エレメント ID num_assoc_agents

エレメント・タイプ ゲージ

表 130. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl_info	基本

説明 これは、アプリケーション・レベルでは、 1 つのアプリケーションに関連付けられているサブエージェントの数です。データベース・レベルでは、すべてのアプリケーション用のサブエージェントの数です。

使用法 このエレメントは、エージェント構成パラメーターの設定を評価するのに役立ちます。

関連資料:

- 189 ページの『agents_from_pool プールから割り当てられたエージェント』
- 190 ページの『agents_created_empty_pool - Agents Created Due to Empty Agent Pool』
- 191 ページの『associated_agents_top 関連エージェント最大数』
- 193 ページの『max_agent_overflows 最大エージェント・オーバーフロー回数』

max_agent_overflows 最大エージェント・オーバーフロー回数

エレメント ID max_agent_overflows

エレメント・タイプ ゲージ

表 131. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 *maxagents* 構成パラメーターにすでに達しているときに、新しいエージェントを作成するための要求を受信した回数。

使用法 *maxagents* 構成パラメーターに達してもエージェント作成要求をまだ受け取る場合は、このノードのワークロードが高すぎることを示している可能性があります。

最大エージェント数 (*maxagents*) 構成パラメーターの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 189 ページの『agents_from_pool プールから割り当てられたエージェント』
- 190 ページの『agents_created_empty_pool - Agents Created Due to Empty Agent Pool』
- 191 ページの『associated_agents_top 関連エージェント最大数』
- 193 ページの『num_assoc_agents 関連したエージェント数』

num_gw_conn_switches - 接続切り替え回数

エレメント ID num_gw_conn_switches

エレメント・タイプ ゲージ

表 132. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 ある接続に対してエージェント・プールのエージェントがプライム状態にされ、別の DRDA データベースで使用するためにスチールされた回数。

使用法 このエレメントを使用して、エージェント・プールのサイズを大きくする必要があるかどうかを判別します。

関連資料:

- 187 ページの『agents_registered 登録済みエージェント』
- 188 ページの『agents_registered_top エージェント最大登録数』
- 192 ページの『total_sec_cons 2 次接続』
- 193 ページの『num_assoc_agents 関連したエージェント数』
- 193 ページの『max_agent_overflows 最大エージェント・オーバーフロー回数』

メモリー・プール

メモリー・プールに関するモニター・エレメント

次のエレメントにより、メモリー・プールについての情報が提供されます。

- pool_id メモリー・プール ID : モニター・エレメント
- pool_cur_size メモリー・プールの現行サイズ : モニター・エレメント
- pool_config_size メモリー・プールの構成済みサイズ : モニター・エレメント
- pool_watermark メモリー・プール水準点 : モニター・エレメント

memory_pool データ・エレメントの性質はプラットフォームによって異なります。その違いとして、 Windows システムではデータベース・システム・モニターはデータベース・スナップショットでメモリーを報告しませんが、 UNIX システムではデータベース・スナップショットでメモリーが報告されます。データベース・スナップショットでこのメモリーを報告する代わりに、 Windows システム用のシステム・モニターがデータベース・マネージャー・スナップショットでこのメモリーを報告します。報告のこの違いは、 Windows システムと UNIX システムでの基本的なメモリー・アーキテクチャーの違いによるものです。

pool_id メモリー・プール ID

エレメント ID pool_id

エレメント・タイプ 情報

表 133. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	memory_pool	基本
データベース	memory_pool	基本
アプリケーション	memory_pool	基本

表 134. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbmemuse	-
接続	event_connmemuse	-

説明 メモリー・プールのタイプ。

使用法 システム・メモリーの使用量を追跡するときに、この値と pool_max_size、pool_cur_size、および pool_watermark を組み合わせて使用します。

pool_id を使用すると、システム・モニター出力に示されているメモリー・プールを識別できます。 sqlmon.h に、さまざまなメモリー・プール ID があります。通常の操作条件では、次に示すプールの 1 つ以上が該当します。

API 定数	説明
SQLM_HEAP_APPLICATION	アプリケーション・ヒープ
SQLM_HEAP_DATABASE	データベース・ヒープ
SQLM_HEAP_APPL_CONTROL	アプリケーション・コントロール・ヒープ
SQLM_HEAP_LOCK_MGR	ロック・マネージャー・ヒープ
SQLM_HEAP_UTILITY	バックアップ/リストア/ユーティリティー・ヒープ
SQLM_HEAP_STATISTICS	統計ヒープ
SQLM_HEAP_PACKAGE_CACHE	パッケージ・キャッシュ・ヒープ
SQLM_HEAP_CAT_CACHE	カタログ・キャッシュ・ヒープ
SQLM_HEAP_DFM	DFM ヒープ
SQLM_HEAP_QUERY	照会ヒープ
SQLM_HEAP_MONITOR	データベース・モニター・ヒープ
SQLM_HEAP_STATEMENT	ステートメント・ヒープ
SQLM_HEAP_FCMBP	FCMBP ヒープ
SQLM_HEAP_IMPORT_POOL	インポート・プール
SQLM_HEAP_OTHER	その他のメモリー
SQLM_HEAP_BP	バッファー・プール・ヒープ

API 定数	説明
SQLM_HEAP_APP_GROUP	アプリケーション・グループ共有ヒープ
SQLM_HEAP_SHARED_SORT	ソート共有ヒープ

関連資料:

- 196 ページの『pool_cur_size メモリー・プールの現行サイズ』
- 197 ページの『pool_config_size メモリー・プールの構成済みサイズ』
- 197 ページの『pool_watermark メモリー・プール水準点』

pool_cur_size メモリー・プールの現行サイズ

エレメント ID pool_cur_size

エレメント・タイプ 情報

表 135. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	memory_pool	基本
データベース	memory_pool	基本
アプリケーション	memory_pool	基本

表 136. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbmemuse	-
接続	event_connmemuse	-

説明 メモリー・プールの現行サイズ。

使用法 システム・メモリーの使用量を追跡するときに、この値と *pool_config_size*、 *pool_id*、および *pool_watermark* を組み合わせて使用します。

メモリー・プールがほぼ満杯状態になっているかどうかを確認するには、*pool_config_size* と *pool_cur_size* を比較します。例えば、ユーティリティー・ヒープが小さすぎるとします。この問題は、一定間隔でスナップショットを取り、スナップショット出力のユーティリティー・ヒープ・セクションを調べることによって診断できます。 *pool_cur_size* の値が *pool_config_size* の値に近い場合は、ユーティリティー・ヒープのサイズを大きくすることを考慮してください。

関連資料:

- 195 ページの『pool_id メモリー・プール ID』
- 197 ページの『pool_config_size メモリー・プールの構成済みサイズ』
- 197 ページの『pool_watermark メモリー・プール水準点』

pool_config_size メモリー・プールの構成済みサイズ

エレメント ID pool_config_size

エレメント・タイプ 情報

表 137. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	memory_pool	基本
データベース	memory_pool	基本
アプリケーション	memory_pool	基本

表 138. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbmemuse	-
接続	event_connmemuse	-

説明 DB2 UDB 中のメモリー・プールの内部構成済みのサイズです。

使用法 システム・メモリーの使用量を追跡するときに、この値と *pool_cur_size*、*pool_id*、および *pool_watermark* を組み合わせて使用します。

メモリー・プールがほぼ満杯状態になっているかどうかを確認するには、*pool_config_size* と *pool_cur_size* を比較します。例えば、ユーティリティー・ヒープが小さすぎるとします。この問題は、一定間隔でスナップショットを取り、スナップショット出力のユーティリティー・ヒープ・セクションを調べることによって診断できます。必要な場合は、メモリー不足の障害が起きないように、 *pool_cur_size* を *pool_config_size* より大きくすることもできます。大きくなる頻度が非常に少ない場合は、おそらく追加のアクションは必要ありません。しかし、常に *pool_cur_size* の値が *pool_config_size* の値に近いか大きい場合は、ユーティリティー・ヒープのサイズを大きくすることを考慮してください。

関連資料:

- 195 ページの『pool_id メモリー・プール ID』
- 196 ページの『pool_cur_size メモリー・プールの現行サイズ』
- 197 ページの『pool_watermark メモリー・プール水準点』

pool_watermark メモリー・プール水準点

エレメント ID pool_watermark

エレメント・タイプ 情報

表 139. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	memory_pool	基本
データベース	memory_pool	基本

表 139. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	memory_pool	基本

表 140. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_dbmemuse	-
接続	event_connmemuse	-

説明 メモリー・プール作成後のその最大サイズ。

使用法 継続的に稼働しているシステムの場合は、 *pool_watermark* と *pool_config_size* のエレメントを組み合わせて使用すると、メモリーに関する潜在的な問題を予測できます。

例えば、一定間隔 (例えば 1 日に 1 回) でスナップショットを取り、*pool_watermark* と *pool_config_size* の値を調べます。 *pool_watermark* の値が *pool_config_size* の値に近づく場合は (メモリー関連のトラブルが起こる可能性を示しています)、メモリー・プールのサイズを大きくする必要があることを示します。

関連資料:

- 195 ページの『pool_id メモリー・プール ID』
- 196 ページの『pool_cur_size メモリー・プールの現行サイズ』
- 197 ページの『pool_config_size メモリー・プールの構成済みサイズ』

ソート

ソートに関するモニター・エレメント

次のエレメントにより、データベース・マネージャーで実行されたソート処理に関する情報が提供されます。

- sort_heap_allocated 割り振られたソート・ヒープの合計 : モニター・エレメント
- post_threshold_sorts ポストしきい値ソート : モニター・エレメント
- piped_sorts_requested 要求されたパイプ・ソート数 : モニター・エレメント
- piped_sorts_accepted 受け入れられたパイプ・ソート : モニター・エレメント
- total_sorts ソート合計 : モニター・エレメント
- total_sort_time ソート時間合計 : モニター・エレメント
- sort_overflows ソート・オーバーフロー : モニター・エレメント
- active_sorts アクティブ・ソート : モニター・エレメント
- sort_shrheap_allocated 現在割り振られているソート共有ヒープ : モニター・エレメント
- sort_shrheap_top ソート共有ヒープの最高水準点 : モニター・エレメント

sort_heap_allocated 割り振られたソート・ヒープの合計

エレメント ID sort_heap_allocated

エレメント・タイプ ゲージ

表 141. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
データベース	dbase	基本

説明 スナップショットが取られたときに、選択したレベルのすべてのソートに割り振られたソート・ヒープ・スペース用のページ数の合計。

使用法 各ソートに割り振られたメモリー量は、利用可能なソート・ヒープ・サイズの一部だけの場合とすべての場合があります。ソート・ヒープ・サイズは各ソートで利用できるメモリー量を示し、 *sortheap* データベース構成パラメーターに定義されている値です。

1 つのアプリケーションが同時に複数のソートをアクティブにすることができます。例えば、副照会付きの SELECT ステートメントを使用すると、同時に複数のソートが行われる場合があります。

情報は 2 つのレベルで収集できます。

- データベース・マネージャーのレベルでは、データベース・マネージャー内のアクティブなすべてのデータベースのすべてのソートを対象に、割り振られたソート・ヒープ・スペースの合計を示す。
- データベース・レベルでは、1 つのデータベース内のすべてのソートを対象に、割り振られたソート・ヒープ・スペースの合計を示す。

通常のメモリーの見積もりにはソート・ヒープ・スペースは含まれません。過剰なソートが発生している場合は、ソート・ヒープに使用される追加のメモリー量をデータベース・マネージャーを実行するのに必要な基本メモリー量に加える必要があります。一般に、ソート・ヒープが大きくなるほど、ソート効率は高くなります。索引を正しく使用すると、ソートに必要な量を少なくできます。

データベース・マネージャー・レベルに戻された情報は、 *sheapthres* 構成パラメーターの調整に利用できます。エレメントの値が *sheapthres* 以上になっている場合は、 *sortheap* パラメーターに定義されているソート・ヒープをソートで完全に得られていないことを示します。

関連資料:

- 201 ページの『total_sorts ソート合計』

post_threshold_sorts ポストしきい値ソート

エレメント ID post_threshold_sorts

エレメント・タイプ カウンター

表 142. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	ソート

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 ソート・ヒープしきい値に達した後でヒープを要求したソートの数。

使用法 通常の条件では、データベース・マネージャーは、 *sortheap* 構成パラメーターによって指定された値を使用して、ソート・ヒープを割り振ります。ソート・ヒープに割り振られたメモリー量がソート・ヒープのしきい値を超えると (*sheapthres* 構成パラメーター)、データベース・マネージャーは、*sortheap* 構成パラメーターが指定する値よりも低い値を使用してソート・ヒープを割り振ります。

システム上のアクティブなソートには、それぞれメモリーが割り振られるので、利用可能なシステム・メモリーからソートのためのメモリー量を多く取り過ぎることがあります。ソート・ヒープのしきい値を超えると、その後に開始したソートでは実行するための十分なメモリー量を得られないことがありますが、システム全体としてはそのほうが利点があります。ソート・ヒープしきい値およびソート・ヒープ・サイズの構成パラメーターを変更すると、ソート操作のパフォーマンスとシステム全体のパフォーマンスを改善できます。エレメントの値が高い場合は、次のいずれかを行うことができます。

- ソート・ヒープしきい値 (*sheapthres*) を上げる。
- SQL 照会で使用するソート数を少なくするか、小さくなるようにアプリケーションを調整する。

関連資料:

- 204 ページの『active_sorts アクティブ・ソート』
- 382 ページの『stmt_sorts ステートメント・ソート回数』

piped_sorts_requested 要求されたパイプ・ソート数

エレメント ID piped_sorts_requested

エレメント・タイプ カウンター

表 143. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 要求されたパイプ・ソートの数。

使用法 システム上のアクティブなソートには、それぞれメモリーが割り振られるため、使用可能なシステム・メモリーからソートのためのメモリー量を多く取り過ぎることがあります。

ソート・リスト・ヒープ (*sortheap*) およびソート・ヒープしきい値 (*sheapthres*) の構成パラメーターは、ソート操作に使用するメモリー量をコントロールするのに利用できます。また、これらのパラメーターを使用すると、ソートをパイプ化するかどうかを決定することもできます。

ソートをパイプ化するとディスク入出力が少なくなり、パイプ・ソートの数を増やせるので、ソート操作のパフォーマンスを改善し、さらにはシステム全体のパフォーマンスもよくなります。ただし、ソート・ヒープをソートに割り振る時にソート・ヒープしきい値を超えてしまうと、パイプ・ソートは

受け入れてもらえません。パイプ・ソートがリジェクトされる場合については、『*piped_sorts_accepted*』を参照してください。

SQL EXPLAIN 出力には、オプティマイザーがパイプ・ソートを要求するかどうかが示されます。パイプ・ソートおよび非パイプ・ソートについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 199 ページの『post_threshold_sorts ポストしきい値ソート』
- 201 ページの『piped_sorts_accepted 受け入れられたパイプ・ソート』

piped_sorts_accepted 受け入れられたパイプ・ソート

エレメント ID piped_sorts_accepted

エレメント・タイプ カウンター

表 144. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 受け入れられたパイプ・ソートの数。

使用法 システム上のアクティブなソートには、それぞれメモリーが割り振られるため、使用可能なシステム・メモリーからソートのためのメモリー量を多く取り過ぎることがあります。

パイプ・ソートを要求した数に対して、受け入れられたパイプ・ソートの数が少ない場合は、次の構成パラメーターのいずれかまたは両方を調整するとソート・パフォーマンスを改善できます。

- sortheap
- sheapthres

パイプ・ソートがリジェクトされる場合は、ソート・ヒープを少なくするか、またはソート・ヒープしきい値を大きくすることを考慮してください。ただし、これらのオプションが与えるそれぞれの影響を知っておく必要があります。ソート・ヒープしきい値を高くすると、ソート処理に割り振られるメモリーの量が多くなる可能性があります。その場合、メモリーがディスクにページングされることがあります。ソート・ヒープを少なくすると、マージ・フェーズが増えてソート処理が遅くなることがあります。

ソートの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 199 ページの『post_threshold_sorts ポストしきい値ソート』
- 200 ページの『piped_sorts_requested 要求されたパイプ・ソート数』

total_sorts ソート合計

エレメント ID total_sorts

エレメント・タイプ カウンター

表 145. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 146. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 実行されたソートの合計数。

使用法 データベースまたはアプリケーションのレベルでは、この値と *sort_overflows* を組み合わせて使用すると、ヒープ・スペースをさらに必要とするソートのパーセンテージを計算できます。さらに、*total_sort_time* と組み合わせると、平均ソート時間を計算できます。

ソート合計数に対してソートのオーバーフロー回数が少ない場合は、ソート・ヒープ・サイズを大きくしても、バッファー・サイズを極端に大きくしなければ、パフォーマンスに影響を与えることはほとんどありません。

ステートメント・レベルでこのエレメントを使用すると、多数のソート操作を実行するステートメントを識別できます。このようなステートメントは、さらに調整を行ってソート回数を少なくすると利点があります。また、SQL EXPLAIN ステートメントを使用すると、 1 つのステートメントが実行するソートの回数を識別できます。詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 202 ページの『total_sort_time ソート時間合計』
- 203 ページの『sort_overflows ソート・オーバーフロー』

total_sort_time ソート時間合計

エレメント ID total_sort_time

エレメント・タイプ カウンター

表 147. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	ソート
アプリケーション	appl	ソート
アプリケーション	stmt	ソート
動的 SQL	dynsql	ソート

スナップショット・モニターの場合、このカウンターはリセットできます。

表 148. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 実行されたすべてのソートの合計経過時間 (ミリ秒)。

使用法 データベースまたはアプリケーションのレベルでこのエレメントと *total_sorts* を組み合わせて使用すると、平均ソート時間を計算できます。平均ソート時間は、ソートがパフォーマンス上の問題となっているかどうかを表します。

ステートメント・レベルでこのエレメントを使用すると、ソート時間の多いステートメントを識別できます。このようなステートメントは、さらに調整を行ってソート時間を少なくすると効率が上がります。

このカウントには、関連操作のために作成される一時表のためのソート時間も含まれています。このカウントは、1 ステートメント、1 アプリケーション、または 1 つのデータベースにアクセスするすべてのアプリケーションについて情報を提供します。

経過時間を示すモニター・エレメントを使用するときは、次のことを考慮してください。

1. 経過時間は、システム負荷の影響を受けるので、実行する処理数が多くなると、この経過時間の値は大きくなります。
2. このモニター・エレメントをデータベース・レベルで計算する場合、データベース・システム・モニターはアプリケーション・レベルの時間を合計します。この場合、同時に複数のアプリケーション処理が実行されていることがあるので、データベース・レベルでは時間が二重に計算されます。

   データベース・レベルで意味のあるデータを得るためには、データを低いレベルに正規化する必要があります。例:

   ```
   total_sort_time / total_sorts
   ```

   これは、ソート当たりの平均経過時間に関する情報を示しています。

関連資料:

- 201 ページの『total_sorts ソート合計』
- 203 ページの『sort_overflows ソート・オーバーフロー』

sort_overflows ソート・オーバーフロー

エレメント ID sort_overflows

エレメント・タイプ カウンター

表 149. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本
アプリケーション	stmt	基本
動的 SQL	dynsql	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 150. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 ソート・ヒープを使い果たし、一時記憶用のディスク・スペースが必要になった可能性のあるソートの合計数。

使用法 データベース・レベルまたはアプリケーション・レベルでは、この値と *total_sorts* を組み合わせて使用すると、ディスクにオーバーフローしたソートのパーセンテージを計算できます。このパーセンテージが高い場合は、*sortheap* の値を大きくして、データベース構成を調整する必要があります。

ステートメント・レベルでこのエレメントを使用すると、大量のソートを必要とするステートメントを識別できます。このようなステートメントは、さらに調整を行って必要となるソート量を少なくすると効率が上がります。

ソートがオーバーフローすると、ソートにマージ・フェーズが必要となり、データをディスクに書き込む必要がある場合は入出力がさらに必要となるので、オーバーヘッドが増えます。

このエレメントは、1 ステートメント、1 アプリケーション、または 1 つのデータベースにアクセスするすべてのアプリケーションについて情報を提供します。

関連資料:

• 201 ページの『total_sorts ソート合計』

active_sorts アクティブ・ソート

エレメント ID active_sorts

エレメント・タイプ ゲージ

表 151. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 現在ソート・ヒープが割り振られているデータベース内のソート数。

使用法 この値と *sort_heap_allocated* を組み合わせて使用すると、各ソートで使用される平均ソート・ヒープ・スペースを判別できます。使用されている平均

ソート・ヒープと比較して、 *sortheap* 構成パラメーターが非常に大きい場合は、このパフォーマンス値を低くできます。 (詳しくは、「*管理ガイド*」を参照してください。)

この値には、関連操作で作成された一時表のソートのヒープが含まれます。

関連資料:

- 198 ページの『sort_heap_allocated 割り振られたソート・ヒープの合計』
- 201 ページの『total_sorts ソート合計』

sort_heap_top ソート専用ヒープの最高水準点

エレメント ID sort_heap_top

エレメント・タイプ 水準点

表 152. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

説明 データベース・マネージャーでの専用ソート・メモリーの最高水準点。

使用法 このエレメントを使用して、 SHEAPTHRES 構成パラメーターが最適な値に設定されているかどうかを判別できます。例えば、この水準点が SHEAPTHRES に近づいたり超えている場合は、おそらく SHEAPTHRES を大きくする必要があります。これは、SHEAPTHRES を超えると専用ソートに与えられるメモリーが常に少なくなり、その結果として逆にシステム・パフォーマンスに影響を与える場合があるためです。

sort_shrheap_allocated 現在割り振られているソート共有ヒープ

エレメント ID sort_shrheap_allocated

エレメント・タイプ 情報

表 153. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 データベースに割り振られている共有ソート・メモリーの合計量。

使用法 このエレメントを使用して、共有ソート・メモリーのしきい値を評価できます。この値が共有ソート・メモリーの現行しきい値より大幅に高いことや低いことが頻繁にある場合は、おそらく、しきい値を調整する必要があります。

注: 「共有ソート・メモリーしきい値」は、 SHEAPTHRES_SHR データベース構成パラメーターが 0 の場合は SHEAPTHRES データベース・マネージャー構成パラメーターの値で決まります。 0 でない場合は SHEAPTHRES_SHR の値で決まります。

sort_shrheap_top ソート共有ヒープの最高水準点

エレメント ID sort_shrheap_top

エレメント・タイプ 水準点

表 154. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 データベース全体の共有ソート・メモリーの最高水準点。

使用法 このエレメントを使用して、 SHEAPTHRES (または SHEAPTHRES_SHR) が最適な値に設定されているかどうかを評価できます。例えば、この最高水準点が常に共有ソート・メモリーしきい値よりも大幅に低い場合は、おそらくこのしきい値を小さくしてデータベースの他の機能にメモリーを解放する必要があります。逆にこの最高水準点が共有ソート・メモリーしきい値に近づき始めたら、そのしきい値を大きくする必要がある場合があります。共有ソート・メモリーしきい値は堅固な制限であるため、そのことは重要です。ソート・メモリーの合計量がそのしきい値に達したら、共有ソートは開始できなくなります。

このエレメントは、専用ソート・メモリーの最高水準点と組み合わせて使用すると、共有および専用ソートのしきい値をそれぞれ単独に設定する必要があるかどうかを判別することにも利用できます。 SHEAPTHRES_SHR データベース構成オプションの値が 0 の場合は通常、共有ソート・メモリーしきい値は SHEAPTHRES データベース・マネージャー構成オプションの値で決まります。ただし専用ソート・メモリーと共有ソート・メモリーの最高水準点に大きな違いがある場合は、 SHEAPTHRES をオーバーライドして、 SHEAPTHRES_SHR を共有ソート・メモリーの最高水準点を基にしたより適切な値に設定する必要がある場合があります。

ハッシュ結合

ハッシュ結合に関するモニター・エレメント

ハッシュ結合は、オプティマイザーの追加オプションです。ハッシュ結合は、まずハッシュ・コード を比較してから、結合に関与する表の述部を比較します。ハッシュ結合では、1 つの表 (オプティマイザーが選択した) をスキャンして、ソート・ヒープの割り振りによって引き出されたメモリー・バッファー内に行がコピーされます。メモリー・バッファーは、結合述部の列数から計算したハッシュ・コードに基づいて、パーティションに分割されます。結合に関係するその他の表の行は、ハッシュ・コードを比較して最初の表の行と突き合わせされます。ハッシュ・コードが一致すると、実際の結合述部の列が比較されます。

- total_hash_joins ハッシュ結合の合計 : モニター・エレメント
- post_threshold_hash_joins ハッシュ結合のしきい値 : モニター・エレメント
- total_hash_loops ハッシュ・ループの合計 : モニター・エレメント
- hash_join_overflows ハッシュ結合のオーバーフロー : モニター・エレメント
- hash_join_small_overflows ハッシュ結合の短精度オーバーフロー : モニター・エレメント

total_hash_joins ハッシュ結合の合計

エレメント ID total_hash_joins

エレメント・タイプ カウンター

表 155. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 156. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 実行されたハッシュ結合の合計数。

使用法 データベースまたはアプリケーション・レベルで、この値と hash_join_overflows および hash_join_small_overflows を組み合わせて使用すると、ソート・ヒープ・サイズを適度に大きくすることにより、ハッシュ結合のかなりの部分に良い影響を与えることができるかどうかを判別できます。

関連資料:

- 207 ページの『post_threshold_hash_joins ハッシュ結合のしきい値』
- 208 ページの『total_hash_loops ハッシュ・ループの合計』
- 208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
- 209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』

post_threshold_hash_joins ハッシュ結合のしきい値

エレメント ID post_threshold_hash_joins

エレメント・タイプ カウンター

表 157. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 共有または専用のソート・ヒープ・スペースが同時使用されていたためにハッシュ結合ヒープ要求が制限された合計回数。

使用法 この値が大きい (hash_join_overflows の 5% より大きい) 場合は、ソート・ヒープのしきい値を大きくしてください。

関連資料:

- 207 ページの『total_hash_joins ハッシュ結合の合計』

- 208 ページの『total_hash_loops ハッシュ・ループの合計』
- 208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
- 209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』

total_hash_loops ハッシュ・ループの合計

エレメント ID total_hash_loops

エレメント・タイプ カウンター

表 158. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 159. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 ハッシュ結合の単一パーティションが、使用可能なソート・ヒープ・スペースよりも大きかったときの合計回数。

使用法 このエレメントの値は、ハッシュ結合が効率的に実行されていないことを示します。ソート・ヒープ・サイズが小さすぎるか、またはソート・ヒープしきい値が小さすぎることを示します。この値とその他のハッシュ結合変数を組み合わせて使用すると、ソート・ヒープ・サイズ (*sortheap*) とソート・ヒープしきい値 (*sheapthres*) の構成パラメーターを調整できます。

関連資料:

- 207 ページの『total_hash_joins ハッシュ結合の合計』
- 207 ページの『post_threshold_hash_joins ハッシュ結合のしきい値』
- 208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』
- 209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』

hash_join_overflows ハッシュ結合のオーバーフロー

エレメント ID hash_join_overflows

エレメント・タイプ カウンター

表 160. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 161. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 ハッシュ結合データが、使用可能なソート・ヒープ・スペースを超えた回数。

使用法 データベース・レベルでは、 hash_join_small_overflows の値がこの hash_join_overflows の 10% を超える場合は、ソート・ヒープ・サイズを大きくすることを検討してください。アプリケーション・レベルの値は、個々のアプリケーションについてハッシュ結合のパフォーマンスを評価するときに使用できます。

関連資料:

- 207 ページの『total_hash_joins ハッシュ結合の合計』
- 207 ページの『post_threshold_hash_joins ハッシュ結合のしきい値』
- 208 ページの『total_hash_loops ハッシュ・ループの合計』
- 209 ページの『hash_join_small_overflows ハッシュ結合の短精度オーバーフロー』

hash_join_small_overflows ハッシュ結合の短精度オーバーフロー

エレメント ID hash_join_small_overflows

エレメント・タイプ カウンター

表 162. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 163. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 ハッシュ結合データが、使用可能なソート・ヒープ・スペースを 10% を超えない範囲で超えた回数。

使用法 この値と hash_join_overflows の値が大きい場合は、ソート・ヒープのしきい値を大きくすることを検討してください。この値が hash_join_overflows の 10% を超える場合は、ソート・ヒープ・サイズを大きくすることを検討してください。

関連資料:

- 207 ページの『total_hash_joins ハッシュ結合の合計』
- 207 ページの『post_threshold_hash_joins ハッシュ結合のしきい値』
- 208 ページの『total_hash_loops ハッシュ・ループの合計』

• 208 ページの『hash_join_overflows ハッシュ結合のオーバーフロー』

高速コミュニケーション・マネージャー

高速コミュニケーション・マネージャーに関するモニター・エレメント

次のデータベース・システム・モニター・エレメントにより、高速コミュニケーション・マネージャー (FCM) に関する情報が提供されます。

- buff_free 現在空いている FCM バッファー : モニター・エレメント
- buff_free_bottom 空き FCM バッファーの最小数 : モニター・エレメント
- ma_free 現在空いているメッセージ・アンカー : モニター・エレメント
- ma_free_bottom メッセージ・アンカーの最小数 : モニター・エレメント
- ce_free 現在空いている接続項目 : モニター・エレメント
- ce_free_bottom 接続項目の最小数 : モニター・エレメント
- rb_free 現在空いている要求ブロック : モニター・エレメント
- rb_free_bottom 要求ブロックの最小数 : モニター・エレメント
- connection_status 接続状況 : モニター・エレメント
- total_buffers_sent 送信された FCM バッファーの合計 : モニター・エレメント
- total_buffers_rcvd 受信された FCM バッファーの合計 : モニター・エレメント

buff_free 現在空いている FCM バッファー

エレメント ID buff_free

エレメント・タイプ ゲージ

表 164. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 このエレメントは、現在空いている FCM バッファーの数を示します。

使用法 現在空いている FCM バッファーの数を *fcm_num_buffers* 構成パラメーターとともに使用して、現在の FCM バッファー・プール使用率を判別できます。この情報を使用すると、*fcm_num_buffers* を調整できます。

関連資料:

• 210 ページの『buff_free_bottom 空き FCM バッファーの最小数』

buff_free_bottom 空き FCM バッファーの最小数

エレメント ID buff_free_bottom

エレメント・タイプ 水準点

表 165. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 処理中に到達した空き FCM バッファーの最小数。

使用法 このエレメントと *fcm_num_buffers* 構成パラメーターを組み合わせて使用すると、 FCM バッファー・プール使用率の最大値を判別できます。 buff_free_bottom が小さい場合は、*fcm_num_buffers* を大きくして、処理中に FCM バッファーが不足しないようにしてください。 buff_free_bottom が大きい場合は、*fcm_num_buffers* を小さくすると、システム・リソースを節約できます。

関連資料:

• 210 ページの『buff_free 現在空いている FCM バッファー』

ma_free 現在空いているメッセージ・アンカー

エレメント ID ma_free

エレメント・タイプ ゲージ

表 166. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 このエレメントは、現在空いているメッセージ・アンカーの数を示します。

使用法 現在のメッセージ・アンカーの空き数と *fcm_num_anchors* 構成パラメーターを組み合わせて使用すると、現在のメッセージ・アンカー使用率を判別できます。この情報を使用すると、*fcm_num_anchors* を調整できます。

関連資料:

• 211 ページの『ma_free_bottom メッセージ・アンカーの最小数』

ma_free_bottom メッセージ・アンカーの最小数

エレメント ID ma_free_bottom

エレメント・タイプ 水準点

表 167. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 処理中に到達した空きメッセージ・アンカーの最小数。

使用法 このエレメントと *fcm_num_anchors* 構成パラメーターを組み合わせて使用すると、メッセージ・アンカー使用率の最大値を判別できます。

関連資料:

• 211 ページの『ma_free 現在空いているメッセージ・アンカー』

ce_free 現在空いている接続項目

エレメント ID ce_free

エレメント・タイプ ゲージ

表 168. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 このエレメントは、現在空いている接続項目の数を示します。

使用法 接続項目の現在の空き数と *fcm_num_connect* 構成パラメーターを組み合わせて使用すると、現在の接続項目使用率を判別できます。この情報を使用すると、*fcm_num_connect* を調整できます。

関連資料:

• 212 ページの『ce_free_bottom 接続項目の最小数』

ce_free_bottom 接続項目の最小数

エレメント ID ce_free_bottom

エレメント・タイプ 水準点

表 169. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 処理中に到達した空き接続項目の最小数。

使用法 このエレメントと *fcm_num_connect* 構成パラメーターを組み合わせて使用すると、接続項目使用率の最大値を判別できます。

関連資料:

• 211 ページの『ce_free 現在空いている接続項目』

rb_free 現在空いている要求ブロック

エレメント ID rb_free

エレメント・タイプ ゲージ

表 170. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 このエレメントは、現在空いている要求ブロックの数を示します。

使用法 要求ブロックの現在の空き数と *fcm_num_rqb* 構成パラメーターを組み合わせて使用すると、現在の要求ブロック使用率を判別できます。この情報を使用すると、*fcm_num_rqb* を調整できます。

rb_free_bottom 要求ブロックの最小数

エレメント ID rb_free_bottom

エレメント・タイプ 水準点

表 171. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm	基本

説明 処理中に到達した空き要求ブロックの最小数。

使用法 このエレメントと *fcm_num_rqb* 構成パラメーターを組み合わせて使用すると、要求ブロック使用率の最大値を判別できます。 rb_free_bottom が小さいときは、*fcm_num_rqb* を大きくして、処理中に要求ブロックが不足しないようにしてください。 rb_free_bottom が大きい場合は、*fcm_num_rqb* を小さくすると、システム・リソースを節約できます。

関連資料:

- 212 ページの『rb_free 現在空いている要求ブロック』

connection_status 接続状況

エレメント ID connection_status

エレメント・タイプ 情報

表 172. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm_node	基本

説明 このエレメントは、 GET SNAPSHOT コマンドを発行するノードと *db2nodes.cfg* ファイルにリストされているその他のノードの間の通信接続状況を示します。

使用法 次の接続値があります。

SQLM_FCM_CONNECT_INACTIVE
現在、接続されていません。

SQLM_FCM_CONNECT_ACTIVE
接続はアクティブです。

SQLM_FCM_CONNECT_CONGESTED
接続が混雑しています。

2 つのノードがアクティブな状態になっても、これらのノード間に通信がなければその間の通信接続は非アクティブとなります。

関連資料:

- 213 ページの『total_buffers_sent 送信された FCM バッファーの合計』
- 214 ページの『total_buffers_rcvd 受信された FCM バッファーの合計』

total_buffers_sent 送信された FCM バッファーの合計

エレメント ID total_buffers_sent

エレメント・タイプ カウンター

表 173. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm_node	基本

説明 GET SNAPSHOT コマンドを発行するノードから *node_number* で識別されるノードに送信された FCM バッファーの合計数 (*db2nodes.cfg* ファイル参照)。

使用法 このエレメントを使用すると、現行ノードとリモート・ノードの間のトラフィック・レベルを測定できます。このノードに送信される FCM バッファーの数が多い場合は、データベースを再分散するか、または表を移動してノード間のトラフィックを少なくしてください。

関連資料:

- 213 ページの『connection_status 接続状況』
- 214 ページの『total_buffers_rcvd 受信された FCM バッファーの合計』

total_buffers_rcvd 受信された FCM バッファーの合計

エレメント ID total_buffers_rcvd

エレメント・タイプ カウンター

表 174. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	fcm_node	基本

説明 *node_number* で識別されるノードが送信して、 GET SNAPSHOT コマンドを発行するノードで受信された FCM バッファーの合計数 (*db2nodes.cfg* ファイル参照)。

使用法 このエレメントを使用すると、現行ノードとリモート・ノードの間のトラフィック・レベルを測定できます。このノードから受信した FCM バッファーの数が多い場合は、データベースを再分散するか、または表を移動してノード間のトラフィックを少なくしてください。

関連資料:

- 213 ページの『connection_status 接続状況』
- 213 ページの『total_buffers_sent 送信された FCM バッファーの合計』

ユーティリティー

ユーティリティーに関するモニター・エレメント

次のエレメントにより、ユーティリティーに関する情報が提供されます。

- utility_dbname ユーティリティーで操作されるデータベース : モニター・エレメント
- utility_id ユーティリティー ID : モニター・エレメント
- utility_type ユーティリティー・タイプ : モニター・エレメント

- utility_priority ユーティリティー優先度 : モニター・エレメント
- utility_start_time ユーティリティー開始時刻 : モニター・エレメント
- utility_description ユーティリティー記述 : モニター・エレメント
- progress_list_current_seq_num 現行の進行リストのシーケンス番号 : モニター・エレメント
- progress_seq_num 進行シーケンス番号 : モニター・エレメント
- progress_description 進行の記述 : モニター・エレメント
- progress_start_time 進行開始時刻 : モニター・エレメント
- progress_work_metric 進行作業メトリック : モニター・エレメント
- progress_total_units 合計進行作業単位 : モニター・エレメント
- progress_completed_units 完了した進行作業単位 : モニター・エレメント

utility_dbname ユーティリティーで操作されるデータベース

エレメント ID utility_dbname

エレメント・タイプ 情報

表 175. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本

説明 ユーティリティーで操作されているデータベース。

utility_id ユーティリティー ID

エレメント ID utility_id

エレメント・タイプ 情報

表 176. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本

説明 ユーティリティー呼び出しに対応するユニーク ID。

utility_type ユーティリティー・タイプ

エレメント ID utility_type

エレメント・タイプ 情報

表 177. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本

説明 ユーティリティーのクラス。

使用法

以下のリストにある、このフィールドの値は、sqlmon.h で定義されています。

API 定数	ユーティリティー
SQLM_UTILITY_REBALANCE	再バランス
SQLM_UTILITY_BACKUP	バックアップ
SQLM_UTILITY_RUNSTATS	統計の実行
SQLM_UTILITY_REORG	REORG
SQLM_UTILITY_RESTORE	リストア
SQLM_UTILITY_CRASH_RECOVERY	クラッシュ・リカバリー
SQLM_UTILITY_ROLLFORWARD_RECOVERY	ロールフォワード・リカバリー
SQLM_UTILITY_LOAD	ロード
SQLM_UTILITY_RESTART_RECREATE_INDEX	索引の再作成の再始動

utility_priority ユーティリティー優先度

エレメント ID utility_priority

エレメント・タイプ 情報

表 178. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本

説明 ユーティリティー優先度は、スロットルされたピアに関連したスロットル・ユーティリティーの相対的な重要度を指定します。優先度 0 は、ユーティリティーがスロットルされずに実行されることを意味します。非ゼロの優先度は 1 から 100 の範囲にする必要があります。 100 は最高の優先度、1 は最低の優先度です。

utility_start_time ユーティリティー開始時刻

エレメント ID utility_start_time

エレメント・タイプ timestamp

表 179. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本

説明 現在のユーティリティーがもともと呼び出された日付と時刻。

utility_description ユーティリティー記述

エレメント ID utility_description

エレメント・タイプ 情報

表 180. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	utility_info	基本

説明 ユーティリティーが実行している作業を簡潔に示す記述。例えば、rebalance 呼び出しに「Tablespace ID: 2」が含まれている場合、この再平衡プログラムが ID 2 の表スペースに対して機能していることを示します。このフィールドのフォーマットはユーティリティー・クラスに依存し、リリースごとに変更される可能性があります。

progress_list_current_seq_num 現行の進行リストのシーケンス番号

エレメント ID progress_list_current_seq_num

エレメント・タイプ 情報

表 181. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress_list	基本

説明 ユーティリティーに複数の順次フェーズが含まれている場合、このエレメントは現行フェーズの番号を表示します。

使用法 このエレメントを使用して、複数フェーズ・ユーティリティーの現行フェーズを判別できます。『*progress_seq_num*』の説明を参照してください。

関連資料:

- 217 ページの『progress_seq_num 進行シーケンス番号』

progress_seq_num 進行シーケンス番号

エレメント ID progress_seq_num

エレメント・タイプ 情報

表 182. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress	基本

説明 フェーズ番号。

注: 複数の実行フェーズから成るユーティリティーの場合のみ、フェーズ番号が表示されます。

使用法 このエレメントを使用して、複数フェーズ・ユーティリティー中のフェーズの順序を判別できます。このユーティリティーは、進行シーケンス番号の昇

順でフェーズを実行します。複数フェーズ・ユーティリティーの現行フェーズを見つけるには、 *progress_seq_num* と、 *progress_list_current_seq_num* の値を突き合わせます。

関連資料:

- 217 ページの『progress_list_current_seq_num 現行の進行リストのシーケンス番号』

progress_description 進行の記述

エレメント ID progress_description

エレメント・タイプ 情報

表 183. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress	基本

説明 作業のフェーズの説明。ロード・ユーティリティーの値の例には、以下が含まれます。

- DELETE
- LOAD
- REDO

使用法 このエレメントを使用して、フェーズの一般説明を取得します。

progress_start_time 進行開始時刻

エレメント ID progress_start_time

エレメント・タイプ 情報

表 184. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress	基本

説明 フェーズの開始を示すタイム・スタンプ。

使用法 このエレメントを使用すると、フェーズが開始した時点を判別できます。フェーズがまだ開始されていない場合は、このエレメントは省略されます。

progress_work_metric 進行作業メトリック

エレメント ID progress_work_metric

エレメント・タイプ 情報

表 185. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress	基本

説明 *progress_total_units* エレメントと *progress_completed_units* エレメントを解釈するメトリック。値の例には、以下のものがあります。

- SQLM_WORK_METRIC_BYTES
- SQLM_WORK_METRIC_EXTENTS

注:

1. このエレメントが組み込まれていないユーティリティーが存在する可能性があります。
2. このエレメントの値は sqlmon.h 中にあります。

使用法 このエレメントを使用して、 *progress_total_units* と *progress_completed_units* が報告メトリックとして使用しているものを判別します。

関連資料:

- 219 ページの『progress_total_units 合計進行作業単位』
- 219 ページの『progress_completed_units 完了した進行作業単位』

progress_total_units 合計進行作業単位

エレメント ID progress_total_units

エレメント・タイプ 情報

表 186. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress	基本

説明 フェーズを完了するために実行する作業の合計量。合計作業の量を定量化できないので、このエレメントが継続的に更新されるユーティリティーもあれば、合計作業を見積もれないので、このエレメントが完全に省略されるユーティリティーもあります。

このエレメントは、 *progress_work_metric* モニター・エレメントで表示される単位で表されます。

使用法 このエレメントを使用して、フェーズ中の作業の合計量を判別します。フェーズ中の完了した作業のパーセンテージを計算するには、このエレメントと *progress_completed_units* を併用してください。

```
完了した作業のパーセンテージ = progress_completed_units / progress_total_units* 100
```

関連資料:

- 218 ページの『progress_work_metric 進行作業メトリック』
- 219 ページの『progress_completed_units 完了した進行作業単位』

progress_completed_units 完了した進行作業単位

エレメント ID progress_completed_units

エレメント・タイプ 情報

表 187. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	progress	基本

説明 現行フェーズの、完了した作業単位の数。通常このエレメントの値は、ユーティリティーが作動するにつれて大きくなります。このエレメントは、常に *progress_total_units* 以下になります (両方のエレメントとも定義されている場合)。

注:

1. このエレメントが組み込まれていないユーティリティーが存在する可能性があります。
2. このエレメントは、 *progress_work_metric* モニター・エレメントで表示される単位で表されます。

使用法 このエレメントを使用して、フェーズ中の完了した作業の量を判別できます。このエレメントを単独で使用すると、実行中のユーティリティーのアクティビティーをモニターできます。このエレメントは、ユーティリティーの実行につれて連続的に大きくなるはずです。 *progress_completed_units* が長期間大きくならない場合は、ユーティリティーが停止している可能性があります。

progress_total_units を定義している場合は、このエレメントを使用して、完了した作業のパーセンテージを計算できます。

```
完了した作業のパーセンテージ = progress_completed_units / progress_total_units* 100
```

関連資料:

- 218 ページの『progress_work_metric 進行作業メトリック』
- 219 ページの『progress_total_units 合計進行作業単位』

データベース構成

データベース構成に関するモニター・エレメント

次のエレメントにより、データベース・パフォーマンスのチューニングにとって特に便利な情報が提供されます。

- バッファー・プール・アクティビティーに関するモニター・エレメント
- バッファーを使用しない入出力アクティビティーに関するモニター・エレメント
- カタログ・キャッシュに関するモニター・エレメント
- パッケージ・キャッシュに関するモニター・エレメント
- SQL ワークスペースに関するモニター・エレメント
- データベース・ヒープに関するモニター・エレメント
- ロギングに関するモニター・エレメント

バッファー・プール・アクティビティー

バッファー・プール・アクティビティーに関するモニター・エレメント

データベース・サーバーは、バッファー・プールのすべてのデータについて読み取りと更新を行います。アプリケーションの要求に応じて、データはディスクからバッファー・プールにコピーされます。

ページは、次のように 1 つのバッファー・プール内に置かれます。

- エージェントによる。同期入出力です。
- 入出力サーバー (プリフェッチャー) による。非同期入出力です。

ページは、次のようにバッファー・プールからディスクに書き込まれます。

- エージェントにより、同期で。
- ページ・クリーナーにより、非同期で。

サーバーが 1 つのデータ・ページを読み取る必要があり、そのページがすでにバッファー・プール内にある場合は、ページをディスクから読み取るときよりも高速にそのページにアクセスできます。バッファー・プール内でできるだけたくさんのページを**ヒット**することが必要になります。サーバーのパフォーマンスを改善するには、ディスク入出力を避けることが主な課題となるので、パフォーマンスのチューニングとしては、バッファー・プールを適切に構成することが最も重要な考慮事項となります。

バッファー・プールのヒット率は、データベース・マネージャーがページ要求を実行するときに、ディスクからページをロードする必要がなかった場合のパーセンテージを示します。つまり、ページがすでにバッファー・プール内にある状態です。バッファー・プール・ヒット率が高いほど、ディスク入出力の頻度は低くなります。

バッファー・プール・ヒット率は、次のように計算できます。

```
(1 - ((pool_data_p_reads + pool_index_p_reads) /
  (pool_data_l_reads + pool_index_l_reads))) * 100%
```

この計算には、バッファー・プールによってキャッシュされているすべてのページ (索引とデータ) が含まれます。

大規模なデータベースでは、バッファー・プールを大きくしても、バッファー・プール・ヒット率の効果が得られないことがあります。データ・ページ数が大きすぎるために、ヒットの統計的な確率が高くならず、数値がよくならないことが考えられます。しかし、索引のバッファー・プール・ヒット率を調整すると、よい結果を得られることがあります。これには 2 つの方法があります。

1. データと索引を 2 つの異なるバッファー・プールに分割し、それぞれを個別に調整します。
2. 1 つのバッファー・プールを使用し、索引のヒット率がこれ以上高くならないというところまでそのバッファー・プールのサイズを大きくします。索引のバッファー・プール・ヒット率は、次のように計算できます。

```
(1 - ((pool_index_p_reads) / (pool_index_l_reads))) * 100%
```

最初の方法のほうが効果が上がることが多いのですが、索引とデータを別の表スペースに置く必要があるので、既存データベースには利用できないことがあります。さらに、1 つではなく、2 つのバッファー・プールを調整する必要があるので、特にメモリーに制約があるときなどは困難な作業となります。

さらに、プリフェッチャーのヒット率に対する影響を考慮する必要があります。プリフェッチャーは、アプリケーションの必要性を予想して、データ・ページをバッファー・プール内に読み取ります (非同期)。多くの場合、これらのページは必要になる直前に読み込まれます (理想的な場合)。ただし、プリフェッチャーは、使用されないページをバッファー・プール内に読み込むことで不要な入出力を行うこともあります。例えば、あるアプリケーションが表の読み取りを開始するとします。このことが検出されると、プリフェッチが開始しますが、アプリケーションはアプリケーション・バッファーを充てんして読み取りを停止します。この間に、その他の多数のページがプリフェッチされます。結局使用されることのないページについて入出力が行われ、バッファー・プールの一部がこうしたページで占められることになってしまいます。

ページ・クリーナーは、バッファー・プールをモニターし、ディスクにページを非同期で書き込みます。これには次の目的があります。

- エージェントがバッファー・プール内でフリー・ページを必ず見つけられるようにする。エージェントがバッファー・プール内でフリー・ページを見つけられない場合は、エージェント自身がページをクリーニングしなければならないため、関連アプリケーションに対してよい応答を返せないことになります。
- システムがクラッシュした場合に、データベースのリカバリーを迅速に行う。ディスクに書き込まれたページ数が多いほど、データベースのリカバリーで処理が必要となるログ・ファイル・レコードの数は少なくなります。

ダーティー・ページはディスクに書き出されますが、バッファー・プールに新しいページを読み取るためのスペースが必要な場合を除いて、このページはバッファー・プールからすぐには除去されません。

注: バッファー・プールに関する情報は、通常は表スペース・レベルで収集されますが、データベース・システム・モニターの機能により、この情報はバッファー・プールおよびデータベースのレベルにまで丸めることができます。実行する分析の種類にもよりますが、いずれかのレベルまたはすべてのレベルでこのデータを調査する必要があります。

次のエレメントにより、バッファー・プール・アクティビティーに関する情報が提供されます。

- pool_data_l_reads バッファー・プール・データの論理読み取り : モニター・エレメント
- pool_temp_data_l_reads バッファー・プール一時データの論理読み取り : モニター・エレメント
- pool_data_p_reads バッファー・プール・データの物理読み取り : モニター・エレメント
- pool_temp_data_p_reads バッファー・プール一時データの物理読み取り : モニター・エレメント
- pool_data_writes バッファー・プールへのデータの書き込み : モニター・エレメント

- pool_index_l_reads バッファー・プール索引の論理読み取り : モニター・エレメント
- pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り : モニター・エレメント
- pool_index_p_reads バッファー・プール索引の物理読み取り : モニター・エレメント
- pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り : モニター・エレメント
- pool_index_writes バッファー・プール索引の書き込み : モニター・エレメント
- pool_read_time バッファー・プール物理読み取り時間の合計 : モニター・エレメント
- pool_write_time バッファー・プール物理書き込み時間の合計 : モニター・エレメント
- files_closed 閉じられたデータベース・ファイル : モニター・エレメント
- pool_async_data_reads バッファー・プール非同期データ読み取り : モニター・エレメント
- pool_async_data_writes バッファー・プール非同期データ書き込み : モニター・エレメント
- pool_async_index_writes バッファー・プール非同期索引書き込み : モニター・エレメント
- pool_async_index_reads バッファー・プール非同期索引読み取り : モニター・エレメント
- pool_async_read_time バッファー・プール非同期読み取り時間 : モニター・エレメント
- pool_async_write_time バッファー・プール非同期書き込み時間 : モニター・エレメント
- pool_async_data_read_reqs バッファー・プール非同期読み取り要求 : モニター・エレメント
- pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求 : モニター・エレメント
- pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー : モニター・エレメント
- pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー : モニター・エレメント
- pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数 : モニター・エレメント
- pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー : モニター・エレメント
- bp_name バッファー・プール名 : モニター・エレメント
- prefetch_wait_time プリフェッチ待ち時間 : モニター・エレメント
- unread_prefetch_pages 読み取り不能プリフェッチ・ページ : モニター・エレメント
- 拡張ストレージに関するモニター・エレメント
- pages_from_vectored_ios ベクトル化入出力によって読み取られたページ数の合計 : モニター・エレメント
- block_ios ブロック入出力要求数 : モニター・エレメント
- vectored_ios ベクトル化入出力要求数 : モニター・エレメント

- pages_from_block_ios ブロック入出力によって読み取られたページ数の合計 : モニター・エレメント
- physical_page_maps 物理ページ・マップ数 : モニター・エレメント

pool_data_l_reads バッファー・プール・データの論理読み取り

エレメント ID pool_data_l_reads

エレメント・タイプ カウンター

表 188. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 189. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 バッファー・プールを通ったデータ・ページの論理読み取り要求の数を示します。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このカウントには、次のデータへのアクセスが含まれます。

- データベース・マネージャーがページの処理を必要としたときにバッファー・プールにすでにあるデータ。
- データベース・マネージャーがページを処理する前にバッファー・プールに読み取られたデータ。

pool_data_p_reads と組み合わせて使用すると、次の公式でバッファー・プール当たりのデータ・ページ・ヒット率を計算できます。

```
1 - (pool_data_p_reads / pool_data_l_reads)
```

pool_data_p_reads、pool_index_p_reads、および pool_index_l_reads と組み合わせて使用すると、次の公式で総合バッファー・プール・ヒット率を計算できます。

```
1 - ((pool_data_p_reads + pool_index_p_reads)
    / (pool_data_l_reads + pool_index_l_reads))
```

バッファー・プール・サイズを大きくすると、一般的にヒット率は高くなりますが、ある点を超えると逆に低くなります。理想的には、データベース全体を保管できるような大きなバッファー・プールを割り振ることができれば、システムが稼働中のヒット率は 100% になります。しかし、現実的にはそうしたことは起こりません。実際には、使用するデータのサイズとそのデータへのアクセス方法によってヒット率の意味は異なります。非常に大きなデータベースでアクセスが均等な場合は、ヒット率が低くなります。表が非常に大きな場合は、対応する方法はほとんどありません。この場合、より小さく、頻繁にアクセスがあるような表、および索引に焦点を当ててください。そして、ヒット率を高くしたいバッファー・プールにこれらを個別に割り当ててください。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
- 229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
- 225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』

pool_temp_data_l_reads バッファー・プール一時データの論理読み取り

エレメント ID pool_temp_data_l_reads

エレメント・タイプ カウンター

表 190. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 191. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 データ・ページを一時表スペースに入れるための入出力を必要とした論理読み取り要求の数を示します。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 *pool_temp_data_p_reads* エレメントと組み合わせて使用すると、次の公式で一時表スペースにあるバッファー・プールのデータ・ページ・ヒット率を計算できます。

```
1 - (pool_temp_data_p_reads / pool_temp_data_l_reads)
```

別の使用法として、次の公式でバッファー・プール全体のヒット率を計算して決定できます。

```
1 - ((pool_temp_data_p_reads +  pool_temp_index_p_reads)
  /  (pool_temp_data_l_reads + pool_temp_index_l_reads))
```

関連資料:

- 224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
- 227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
- 232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
- 231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』

pool_data_p_reads バッファー・プール・データの物理読み取り

エレメント ID pool_data_p_reads

エレメント・タイプ カウンター

表 192. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 193. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 データ・ページをバッファー・プールに入れるための入出力を必要とした読み取り要求の数。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このエレメントの使用法については、 『pool_data_l_reads』および『pool_async_data_reads』を参照してください。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
- 228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
- 229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
- 237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』
- 227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』

pool_temp_data_p_reads バッファー・プール一時データの物理読み取り

エレメント ID pool_temp_data_p_reads

エレメント・タイプ カウンター

表 194. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 195. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 データ・ページを一時表スペースに入れるための入出力を必要とした物理読み取り要求の数。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このエレメントの使用法については、『*pool_temp_data_l_reads*』を参照してください。

関連資料:

- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
- 225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
- 231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』

pool_data_writes バッファー・プールへのデータの書き込み

エレメント ID pool_data_writes

エレメント・タイプ カウンター

表 196. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 197. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-

説明 バッファー・プール・データ・ページがディスクに物理的に書き込まれた回数。

使用法 pool_data_p_reads のパーセンテージが高いためにバッファー・プール・データ・ページがディスクへ書き込まれる場合は、データベースで利用可能なバッファー・プール・ページ数を増やすとパフォーマンスを改善できます。

バッファー・プール・データ・ページをディスクに書き込む理由は次のとおりです。

- バッファー・プール内のページを解放して、次のページを読み取れるようにする。
- バッファー・プールを空にする。

システムがあるページを書き込むときに、新しいページのためのスペースを用意するとは限りません。そのページが更新されていなければ、単純に置換されます。このエレメントでは、このような置換はカウントされません。

データ・ページは、バッファー・プール・スペースが必要になる前に、非同期ページ・クリーナー・エージェントにより書き込まれます。非同期ページの書き込みは、同期ページの書き込みと合わせて、このエレメントの値に含まれます (pool_async_data_writes 参照)。

このパーセンテージを計算するときは、バッファー・プールを最初に埋めるために必要となる物理読み取り数は無視してください。書き込みページ数は、次のように求めます。

1. アプリケーションを実行します (バッファーをロードする)。
2. このエレメントの値を書き取ります。
3. アプリケーションを再び実行します。
4. このエレメントの新しい値からステップ 2 で記録した値を引きます。

アプリケーションを終了してから次に実行するまでの間にバッファー・プールの割り振りが解除されるのを防止するには、次のいずれかを行います。

- ACTIVATE DATABASE コマンドを使用してデータベースを活動化する。
- アイドル状態のアプリケーションをデータベースに接続する。

すべてのアプリケーションがデータベースを更新するような場合は、ほとんどのバッファー・プール・ページが更新されたデータを含んでおり、これをディスクに書き込む必要があるので、バッファー・プールのサイズを大きくしてもパフォーマンスはあまり改善されません。ただし、更新されたページを書き出す前に、ほかの作業単位がこのページを使用できる場合は、バッファー・プールが書き込み操作と読み取り操作を節約できるので、パフォーマンスが向上する場合があります。

バッファー・プール・サイズの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
- 238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』

pool_index_l_reads バッファー・プール索引の論理読み取り

エレメント ID pool_index_l_reads

エレメント・タイプ カウンター

表 198. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 199. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 バッファー・プールを通った索引ページの論理読み取り要求の数を示します。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このカウントには、次の索引ページへのアクセスが含まれます。

- データベース・マネージャーがページの処理を必要としたときにバッファー・プールにすでにあるデータ。
- データベース・マネージャーがページを処理する前にバッファー・プールに読み取られたデータ。

pool_index_p_reads と組み合わせて使用すると、次の公式でバッファー・プールの索引ページ・ヒット率を計算できます。

```
1 - (pool_index_p_reads / pool_index_l_reads)
```

バッファー・プールの総合ヒット率を計算する方法については、『pool_data_l_reads』を参照してください。

ヒット率が低い場合は、バッファー・プール・ページ数を増やすと、パフォーマンスが向上する場合があります。バッファー・プール・サイズの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』

- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
- 233 ページの『pool_index_writes バッファー・プール索引の書き込み』
- 232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』

pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り

エレメント ID pool_temp_index_l_reads

エレメント・タイプ カウンター

表 200. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 201. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 索引ページを一時表スペースに入れるための入出力を必要とした論理読み取り要求の数を示します。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このエレメントの使用法については、『*pool_temp_data_l_reads*』を参照してください。

関連資料:

- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
- 227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
- 232 ページの『pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り』
- 225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』

pool_index_p_reads バッファー・プール索引の物理読み取り

エレメント ID pool_index_p_reads

エレメント・タイプ カウンター

表 202. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 203. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 索引ページをバッファー・プールに入れるための物理読み取り要求の数を示します。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このエレメントの使用法については、『pool_index_l_reads』を参照してください。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 224 ページの『pool_data_l_reads バッファー・プール・データの論理読み取り』
- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
- 233 ページの『pool_index_writes バッファー・プール索引の書き込み』
- 231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』

pool_temp_index_p_reads バッファー・プール一時索引の物理読み取り

エレメント ID pool_temp_index_p_reads

エレメント・タイプ カウンター

表 204. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール
アプリケーション	stmt	バッファー・プール
動的 SQL	dynsql	バッファー・プール、ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 205. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 索引ページを一時表スペースに入れるための入出力を必要とした物理読み取り要求の数を示します。

ステートメント・レベルでバッファー・プール情報を記録する機能は、 API および CLP スナップショット要求用にサポートされています。

使用法 このエレメントの使用法については、『*pool_temp_data_l_reads*』を参照してください。

関連資料:

- 229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
- 227 ページの『pool_temp_data_p_reads バッファー・プール一時データの物理読み取り』
- 225 ページの『pool_temp_data_l_reads バッファー・プール一時データの論理読み取り』
- 231 ページの『pool_temp_index_l_reads バッファー・プール一時索引の論理読み取り』

pool_index_writes バッファー・プール索引の書き込み

エレメント ID pool_index_writes

エレメント・タイプ カウンター

表 206. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 207. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-

説明 バッファー・プール索引ページがディスクに物理的に書き込まれた回数。

使用法 データ・ページと同様に、バッファー・プール索引ページは以下の理由でディスクに書き込まれます。

- バッファー・プール内のページを解放して、次のページを読み取れるようにする。
- バッファー・プールを空にする。

システムがあるページを書き込むときに、新しいページのためのスペースを用意するとは限りません。そのページが更新されていなければ、単純に置換されます。このエレメントでは、このような置換はカウントされません。

索引ページは、バッファー・プール・スペースが必要になる前に、非同期ページ・クリーナー・エージェントにより書き込まれます。非同期索引ページの書き込みは、同期索引ページの書き込みとあわせて、このエレメントの値に含まれます (pool_async_index_writes 参照)。

pool_index_p_reads のパーセンテージが高いためにバッファー・プール索引ページがディスクに書き込まれる場合は、データベースで利用可能なバッファー・プール・ページ数を増やすとパフォーマンスを改善できます。

このパーセンテージを計算するときは、バッファー・プールを最初に埋めるために必要となる物理読み取り数は無視してください。書き込みページ数は、次のように求めます。

1. アプリケーションを実行します (バッファーをロードする)。
2. このエレメントの値を書き取ります。
3. アプリケーションを再び実行します。
4. このエレメントの新しい値からステップ 2 で記録した値を引きます。

アプリケーションを終了してから次に実行するまでのあいだにバッファー・プールの割り振りが解除されるのを防止するには、次のいずれかを行います。

- ACTIVATE DATABASE コマンドを使用してデータベースを活動化する。
- アイドル状態のアプリケーションをデータベースに接続する。

すべてのアプリケーションがデータベースを更新するような場合は、ほとんどのページが更新されたデータを含んでおり、これをディスクに書き込む必要があるので、バッファー・プールのサイズを大きくしてもパフォーマンスはあまり改善されません。

バッファー・プール・サイズの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 229 ページの『pool_index_l_reads バッファー・プール索引の論理読み取り』
- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
- 239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』

pool_read_time バッファー・プール物理読み取り時間の合計

エレメント ID pool_read_time

エレメント・タイプ カウンター

表 208. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 209. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-

説明 ディスクまたはバッファー・プールからデータ・ページまたは索引ページを物理的に読み取る必要があった読み取り要求の処理に要した合計時間を示します。経過時間はマイクロ秒単位で示されます。

使用法 このエレメントと *pool_data_p_reads* および *pool_index_p_reads* を組み合わせて使用すると、ページ読み取りの平均時間を計算できます。この平均値は、入出力待ちがあるかどうかを示すので重要です。これにより、データをほかの装置に移動すべきかどうかが示されます。

データベースおよび表スペースのレベルでは、このエレメントには *pool_async_read_time* の値が含まれます。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』
- 241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』

pool_write_time バッファー・プール物理書き込み時間の合計

エレメント ID pool_write_time

エレメント・タイプ カウンター

表 210. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 211. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-

説明 データ・ページまたは索引ページをバッファー・プールからディスクに物理的に書き込むのに要した合計時間を示します。経過時間はマイクロ秒単位で示されます。

使用法 このエレメントと *buffer_pool_data_writes* および *pool_index_writes* を組み合わせて使用すると、ページ書き込みの平均時間を計算できます。この平均値は、入出力待ちがあるかどうかを示すので重要です。これにより、データをほかの装置に移動すべきかどうかが示されます。

データベースおよび表スペースのレベルでは、このエレメントには *pool_async_write_time* の値が含まれます。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』
- 233 ページの『pool_index_writes バッファー・プール索引の書き込み』

files_closed 閉じられたデータベース・ファイル

エレメント ID files_closed

エレメント・タイプ カウンター

表 212. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 213. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 閉じられたデータベース・ファイルの合計数。

使用法 データベース・マネージャーは、バッファー・プールへの書き込みおよびバッファー・プールからの読み取りを行うためにファイルを開きます。アプリケーションが同時に開けるデータベース・ファイルの最大数は、 *maxfilop* 構成パラメーターによりコントロールされています。最大値に達すると、新しいファイルを開く前に、ファイルが 1 つ閉じられます。実際に開かれたファイルの数と閉じられたファイルの数は等しくならないことに注意してください。

このエレメントは、 *maxfilop* 構成パラメーターの最適な値を判別するときに利用できます (詳しくは「*管理ガイド*」を参照してください)。

pool_async_data_reads バッファー・プール非同期データ読み取り

エレメント ID pool_async_data_reads

エレメント・タイプ カウンター

表 214. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 215. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 バッファー・プールに非同期で読み取られるページ数。

使用法 このエレメントと pool_data_p_reads を組み合わせて使用すると、同期で実行された物理読み取り数を計算できます (つまり、データベース・マネージャーのエージェントが実行したデータ・ページ物理読み取り数)。次の公式を使用します。

```
pool_data_p_reads - pool_async_data_reads
```

非同期読み取り数と同期読み取り数を比較すると、プリフェッチャーの動作状態がわかります。このエレメントは、 *num_ioservers* 構成パラメーターを調整するときに役に立ちます (「*管理ガイド*」参照)。

非同期読み取りは、データベース・マネージャーのプリフェッチャーが実行します。これらのプリフェッチャーについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 226 ページの『pool_data_p_reads バッファー・プール・データの物理読み取り』
- 241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
- 243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
- 258 ページの『direct_reads データベースからの直接読み取り』

pool_async_data_writes バッファー・プール非同期データ書き込み

エレメント ID pool_async_data_writes

エレメント・タイプ カウンター

表 216. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 217. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 非同期ページ・クリーナーまたはプリフェッチャーによりバッファー・プール・データ・ページがディスクに物理的に書き込まれた回数。プリフェッチャーは、プリフェッチするページの場所を作るために、ダーティー・ページをディスクに書き込む場合があります。

使用法 このエレメントと buffer_pool_data_writes を組み合わせて使用すると、同期で実行された物理書き込み要求の数を計算できます (つまり、データベース・マネージャーのエージェントが実行したデータ・ページ物理書き込み数)。次の公式を使用します。

```
pool_data_writes - pool_async_data_writes
```

非同期読み取り数と同期読み取り数を比較すると、バッファー・プール・ページ・クリーナーの動作状態がわかります。この比率は、 *num_iocleaners* 構成パラメーターを調整するときに役に立ちます。

非同期ページ・クリーナーの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 228 ページの『pool_data_writes バッファー・プールへのデータの書き込み』

- 239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
- 242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
- 259 ページの『direct_writes データベースへの直接書き込み』

pool_async_index_writes バッファー・プール非同期索引書き込み

エレメント ID pool_async_index_writes

エレメント・タイプ カウンター

表 218. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 219. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 非同期ページ・クリーナーまたはプリフェッチャーによりバッファー・プール索引ページがディスクに物理的に書き込まれた回数。プリフェッチャーは、プリフェッチするページの場所を作るために、ダーティー・ページをディスクに書き込む場合があります。

使用法 このエレメントと pool_index_writes を組み合わせて使用すると、同期で実行された物理索引書き込み要求数を計算できます。つまり、データベース・マネージャーのエージェントが実行した物理索引ページ書き込み数です。次の公式を使用します。

```
pool_index_writes - pool_async_index_writes
```

非同期読み取り数と同期読み取り数を比較すると、バッファー・プール・ページ・クリーナーの動作状態がわかります。この比率は、 *num_iocleaners* 構成パラメーターを調整するときに役に立ちます。

非同期ページ・クリーナーの詳細については、「*管理ガイド*」を参照してください。

関連資料:

- 233 ページの『pool_index_writes バッファー・プール索引の書き込み』

- 238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
- 240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』
- 242 ページの『pool_async_write_time バッファー・プール非同期書き込み時間』
- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
- 259 ページの『direct_writes データベースへの直接書き込み』

pool_async_index_reads バッファー・プール非同期索引読み取り

エレメント ID pool_async_index_reads

エレメント・タイプ カウンター

表 220. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 221. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 プリフェッチャーが非同期でバッファー・プールに読み取る索引ページ数。

使用法 このエレメントと pool_index_p_reads を組み合わせて使用すると、同期で実行された物理読み取り数を計算できます (つまり、データベース・マネージャーのエージェントが実行した索引ページ物理読み取り数)。次の公式を使用します。

```
pool_index_p_reads - pool_async_index_reads
```

非同期読み取り数と同期読み取り数を比較すると、プリフェッチャーの動作状態がわかります。このエレメントは、 *num_ioservers* 構成パラメーターを調整するときに役に立ちます (「*管理ガイド*」参照)。

非同期読み取りは、データベース・マネージャーのプリフェッチャーが実行します。これらのプリフェッチャーについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 232 ページの『pool_index_p_reads バッファー・プール索引の物理読み取り』

- 238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
- 239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
- 241 ページの『pool_async_read_time バッファー・プール非同期読み取り時間』
- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
- 258 ページの『direct_reads データベースからの直接読み取り』

pool_async_read_time バッファー・プール非同期読み取り時間

エレメント ID pool_async_read_time

エレメント・タイプ カウンター

表 222. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 223. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 データベース・マネージャーのプリフェッチャーが読み取りに要した合計経過時間。

使用法 このエレメントを使用すると同期読み取りの経過時間を計算できます。次の公式を使用します。

```
pool_read_time - pool_async_read_time
```

このエレメントを使用すると、平均非同期読み取り時間も計算できます。次の公式を使用します。

```
pool_async_read_time / pool_async_data_reads
```

これらの計算は、実行中の入出力操作を把握するときに使用します。

関連資料:

- 235 ページの『pool_read_time バッファー・プール物理読み取り時間の合計』
- 237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』

- 243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
- 261 ページの『direct_read_time 直接読み取り時間』

pool_async_write_time バッファー・プール非同期書き込み時間

エレメント ID pool_async_write_time

エレメント・タイプ カウンター

表 224. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 225. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 データベース・マネージャーのページ・クリーナーがデータ・ページまたは索引ページをバッファー・プールからディスクに書き込むのに要した合計経過時間。

使用法 同期によるページ書き込みでの経過時間を計算するには、次の公式を使用します。

```
pool_write_time - pool_async_write_time
```

このエレメントを使用すると、平均非同期読み取り時間も計算できます。次の公式を使用します。

```
pool_async_write_time
 / (pool_async_data_writes
   + pool_async_index_writes)
```

これらの計算は、実行中の入出力操作を把握するときに使用します。

関連資料:

- 236 ページの『pool_write_time バッファー・プール物理書き込み時間の合計』
- 238 ページの『pool_async_data_writes バッファー・プール非同期データ書き込み』
- 239 ページの『pool_async_index_writes バッファー・プール非同期索引書き込み』
- 243 ページの『pool_async_data_read_reqs バッファー・プール非同期読み取り要求』
- 262 ページの『direct_write_time 直接書き込み時間』

pool_async_data_read_reqs バッファー・プール非同期読み取り要求

エレメント ID pool_async_data_read_reqs

エレメント・タイプ カウンター

表 226. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 227. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 非同期読み取り要求の数。

使用法 非同期要求当たりのデータ・ページ読み取りの平均数を計算するには、次の公式を使用します。

```
pool_async_data_reads / pool_async_data_read_reqs
```

この平均値は、プリフェッチャーが動作するときの非同期入出力量を判別するのに利用します。

関連資料:

- 237 ページの『pool_async_data_reads バッファー・プール非同期データ読み取り』

pool_async_index_read_reqs バッファー・プール非同期索引読み取り要求

エレメント ID pool_async_index_read_reqs

エレメント・タイプ カウンター

表 228. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 229. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

表 229. イベント・モニター情報 (続き)

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表スペース	event_tablespace	-

説明 索引ページの非同期読み取り要求の数。

使用法 非同期要求当たりの索引ページ読み取りの数を計算するには、次の公式を使用します。

```
pool_async_index_reads / pool_async_index_read_reqs
```

この平均値は、プリフェッチャーとのそれぞれの対話で索引ページに関して行われる非同期入出力量を判別するのに役立ちます。

関連資料:

- 240 ページの『pool_async_index_reads バッファー・プール非同期索引読み取り』

pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー

エレメント ID pool_lsn_gap_clns

エレメント・タイプ カウンター

表 230. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 231. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 使用されているロギング・スペースがデータベースの定義済み基準に達したためにページ・クリーナーが呼び出された回数。

使用法 このエレメントは、ロギングに十分なスペースがあるかどうか、またログ・ファイルやさらに大きなログ・ファイルの追加が必要かどうかを判別するときに利用できます。

ページ・クリーニングの基準は、 *softmax* 構成パラメーターの設定値により決定されます。バッファー・プール内の最も古いページに含まれている更新内容が現行ログ位置と比較して基準よりも古いログ・レコードにより記述されている場合に、ページ・クリーナーが起動されます。詳細については、「*管理ガイド*」を参照してください。

DB2_USE_ALTERNATE_PAGE_CLEANING レジストリー変数が OFF の場合は、以下のようになります。

- *pool_lsn_gap_clns* モニター・エレメントがモニター・ストリーム中に挿入される。

- バッファー・プール内の最も古いページに含まれている更新内容が現行ログ位置と比較して基準よりも古いログ・レコードにより記述されている場合に、ページ・クリーナーが起動される。

DB2_USE_ALTERNATE_PAGE_CLEANING レジストリー変数が ON の場合は、以下のようになります。

- *pool_lsn_gap_clns* モニター・エレメントがモニター・ストリーム中に 0 を挿入する。
- ページ・クリーナーが、基準値によって起動されるのを待たずに、先行してページを書き込む。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『chngpgs_thresh - 「変更済みページ数しきい値」構成パラメーター』
- 245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
- 「*管理ガイド: パフォーマンス*」の『パフォーマンス変数』

pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー

エレメント ID pool_drty_pg_steal_clns

エレメント・タイプ カウンター

表 232. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 233. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 データベースのビクティム・バッファー置換の際に同期書き込みが必要になりページ・クリーナーが呼び出された回数。

使用法 次の公式を使用すると、クリーナー呼び出しの合計に占めるこのエレメントのパーセンテージを計算できます。

```
   pool_drty_pg_steal_clns
/ (pool_drty_pg_steal_clns
+  pool_drty_pg_thrsh_clns
+  pool_lsn_gap_clns)
```

この比率が低い場合は、定義したページ・クリーナー数が多すぎることを示します。 *chngpgs_thresh* をあまり低く設定すると、ダーティー・ページに

なるページを書き出す可能性が多くなります。クリーニングをあまり積極的にすると、バッファー・プールの 1 つの目的である、書き込みをできるだけ遅らせることができなくなります。

この比率が高い場合は、定義したページ・クリーナー数が少なすぎることを示します。ページ・クリーナー数が少なすぎると、障害が発生したときのリカバリー時間が長くなります (「*管理ガイド*」を参照)。

DB2_USE_ALTERNATE_PAGE_CLEANING レジストリー変数が OFF の場合は、以下のようになります。

- *pool_drty_pg_steal_clns* モニター・エレメントがモニター・ストリーム中に挿入される。
- *pool_drty_pg_steal_clns* モニター・エレメントが、データベースのビクティム・バッファー置換の際に同期書き込みが必要になり、ページ・クリーナーが呼び出された回数をカウントする。

DB2_USE_ALTERNATE_PAGE_CLEANING レジストリー変数が ON の場合は、以下のようになります。

- *pool_drty_pg_steal_clns* モニター・エレメントが、モニター・ストリーム中に 0 を挿入する。
- ビクティム・バッファー置換の際に同期書き込みが必要でも、ページ・クリーナーは明示的に起動されない。データベースまたは特定のバッファー・プール用に構成されているページ・クリーナーの数が正しいかどうかを判別するには、 『pool_no_victim_buffer』モニター・エレメントを参照してください。

注: ダーティー・ページはディスクに書き出されますが、バッファー・プールに新しいページを読み取るためのスペースが必要な場合を除いて、このページはバッファー・プールからすぐには除去されません。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『chngpgs_thresh - 「変更済みページ数しきい値」構成パラメーター』
- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
- 「*管理ガイド: パフォーマンス*」の『パフォーマンス変数』
- 246 ページの『pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数』

pool_no_victim_buffer バッファー・プールの非ビクティム・バッファー数

エレメント ID pool_no_victim_buffer

エレメント・タイプ カウンター

表 234. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 235. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-

説明 エージェントに、事前選択された使用可能なビクティム・バッファーがなかった回数。

使用法 このエレメントは、特定のバッファー・プールに十分なページ・クリーナーがあるかどうかを判別するときに利用できます。この比率が高い場合は、このバッファー・プールには定義したページ・クリーナー数が少なすぎることを示します。

関連資料:

- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』

pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー

エレメント ID pool_drty_pg_thrsh_clns

エレメント・タイプ カウンター

表 236. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 237. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 バッファー・プールがデータベースのダーティー・ページしきい値基準に達したためにページ・クリーナーが呼び出された回数。

使用法 しきい値は *chngpgs_thresh* 構成パラメーターによって設定されます。この

値は、バッファー・プール・サイズに適用されるパーセンテージです。プール内のダーティー・ページ数がこの値を超えると、クリーナーを起動します。

この設定値が低すぎると、ページの書き出しが早すぎて、再読み取りが必要になります。設定値が高すぎると、累積されるページ数が多くなり、ページを同期で書き出す必要が生じます。詳細については、「*管理ガイド*」を参照してください。

DB2_USE_ALTERNATE_PAGE_CLEANING レジストリー変数が OFF の場合は、以下のようになります。

- *pool_drty_pg_thrsh_clns* モニター・エレメントがモニター・ストリーム中に挿入される。
- *pool_drty_pg_thrsh_clns* モニター・エレメントが、バッファー・プールがデータベースのダーティー・ページしきい値基準に達したためにページ・クリーナーが呼び出された回数をカウントする。

DB2_USE_ALTERNATE_PAGE_CLEANING レジストリー変数が ON の場合は、以下のようになります。

- *pool_drty_pg_thrsh_clns* モニター・エレメントがモニター・ストリーム中に 0 を挿入する。
- ページ・クリーナーは、基準値によって起動されるのを待たず、常時アクティブで、使用可能なビクティム用の空きバッファーが十分あるか確認する。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『chngpgs_thresh - 「変更済みページ数しきい値」構成パラメーター』
- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 「*管理ガイド: パフォーマンス*」の『パフォーマンス変数』

bp_name バッファー・プール名

エレメント ID bp_name

エレメント・タイプ 情報

表 238. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	基本

説明 バッファー・プールの名前。

使用法 新規データベースには、IBMDEFAULTBP と呼ぶデフォルトのバッファー・プールがあり、このサイズはプラットフォームによって決まります。各データベースには少なくとも 1 つのバッファー・プールが必要です。ただし、必要に応じて、単一のデータベースのためにそれぞれサイズの違ういくつかのバッファー・プールを作成できます。 CREATE、ALTER、および DROP BUFFERPOOL ステートメントを使用して、バッファー・プールを作成、変更、ドロップすることが可能です。

prefetch_wait_time プリフェッチ待ち時間

エレメント ID prefetch_wait_time

エレメント・タイプ カウンター

表 239. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
アプリケーション	appl	バッファー・プール

表 240. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 アプリケーションが入出力サーバー (プリフェッチャー) によるバッファー・プールへのページのロードの終了を待機していた時間。

使用法 このエレメントを使用すると、入出力サーバーの数と入出力サーバーのサイズの変更を試すことができます。

unread_prefetch_pages 読み取り不能プリフェッチ・ページ

エレメント ID unread_prefetch_pages

エレメント・タイプ カウンター

表 241. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 242. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表スペース	event_tablespace	-
接続	event_conn	-

説明 プリフェッチャー読み取りが使用されなかったページ数を示します。

使用法 ページ数が多い場合、プリフェッチャーは、使用されないページをバッファー・プール内に読み込むことで不必要な入出力を行うことがあります。プリフェッチの詳細については、「*管理ガイド*」を参照してください。

vectored_ios ベクトル化入出力要求数

エレメント ID vectored_ios

エレメント・タイプ ゲージ

表 243. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	バッファー・プール

説明 ベクトル化入出力要求の数です。より厳密には、 DB2 がバッファー・プールのページ域に対してページの順次プリフェッチを実行する回数です。

使用法 このエレメントを使用すると、ベクトル化入出力の実行頻度を判別できます。ベクトル化入出力要求の数がモニターされるのは、順次プリフェッチの実行中だけです。

関連資料:

- 250 ページの『pages_from_vectored_ios ベクトル化入出力によって読み取られたページ数の合計』
- 250 ページの『block_ios ブロック入出力要求数』
- 252 ページの『pages_from_block_ios ブロック入出力によって読み取られたページ数の合計』

pages_from_vectored_ios ベクトル化入出力によって読み取られたページ数の合計

エレメント ID pages_from_vectored_ios

エレメント・タイプ ゲージ

表 244. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	バッファー・プール

説明 ベクトル化入出力でバッファー・プールのページ・エリアに読み取られたページ数の合計。

関連資料:

- 249 ページの『vectored_ios ベクトル化入出力要求数』
- 250 ページの『block_ios ブロック入出力要求数』
- 252 ページの『pages_from_block_ios ブロック入出力によって読み取られたページ数の合計』

block_ios ブロック入出力要求数

エレメント ID block_ios

エレメント・タイプ カウンター

表 245. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	バッファー・プール

説明 ブロック入出力要求の数。より厳密には、 DB2 がバッファー・プールのブロック域に対してページの順次プリフェッチを実行する回数。

使用法 ブロック・ベースのバッファー・プールを使用可能にすると、このモニター・エレメントがブロック入出力の実行頻度を報告します。それ以外の場合、このモニター・エレメントは 0 を戻します。ブロック入出力要求の数がモニターされるのは、ブロック・ベースのバッファー・プールの使用中に順次プリフェッチが実行される間だけです。

ブロック・ベースのバッファー・プールを使用可能にしてあり、この数が非常に小さい場合、またはベクトル化入出力の数 (「ベクトル化入出力要求の数」モニター・エレメントの値) に近い場合は、ブロック・サイズを変更することを考慮してください。このようなときは、以下の状態を示している場合があります。

- バッファー・プールにバインドされている 1 つ以上の表スペースのエクステント・サイズが、バッファー・プールに指定されているブロック・サイズよりも小さい。
- プリフェッチ要求されたページのうち、いくつかのページがバッファー・プールのページ・エリアにすでに存在している。

プリフェッチャーはそれぞれのバッファー・プール・ブロックに無駄なページが多少あっても見過ごしますが、無駄なページの数が非常に多くなると、プリフェッチャーはバッファー・プールのページ・エリアにベクトル化入出力を実行することにします。

ブロック・ベースのバッファー・プールを使用することで順次プリフェッチのパフォーマンスが改善される点を活用するには、ブロック・サイズに適切な値を選ぶことが非常に重要です。しかし、エクステント・サイズの異なる複数の表スペースが同じブロック・ベースのバッファー・プールにバインドされていることがあるので、値の選択が難しいこともあります。最適なパフォーマンスを得るためには、同じエクステント・サイズの表スペースを、ブロック・サイズがエクステント・サイズと等しいブロック・ベースのバッファー・プールにバインドすることをお勧めします。表スペースのエクステント・サイズがブロック・サイズより大きい場合にも良好なパフォーマンスが得られますが、ブロック・サイズよりも小さい場合にはパフォーマンスが低下します。

例えば、エクステント・サイズが 2 でブロック・サイズが 8 の場合は、ブロック入出力の代わりにベクトル化入出力が使用されます (ブロック入出力では 6 ページの無駄が発生します)。ブロック・サイズを 2 に下げると、この問題は解決できます。

関連資料:

- 249 ページの『vectored_ios ベクトル化入出力要求数』
- 250 ページの『pages_from_vectored_ios ベクトル化入出力によって読み取られたページ数の合計』
- 252 ページの『pages_from_block_ios ブロック入出力によって読み取られたページ数の合計』

pages_from_block_ios ブロック入出力によって読み取られたページ数の合計

エレメント ID pages_from_block_ios

エレメント・タイプ カウンター

表 246. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	バッファー・プール

説明 ブロック入出力でバッファー・プールのページ・エリアに読み取られたページ数の合計。

使用法 ブロック・ベースのバッファー・プールを使用可能にすると、このエレメントにブロック入出力が読み取ったページ数の合計が含まれます。使用可能になっていない場合は 0 が戻されます。

pages_from_block_ios を *block_ios* エレメントで除算すると、ブロック・ベース入出力当たりの順次プリフェッチされる平均ページ数を求められます。*pages_from_block_ios* を *block_ios* で除算した値が、ブロック・ベースのバッファー・プールの BLOCKSIZE の定義値より大きい場合は、ブロック・ベース入出力の利点を完全に活用できません。考えられる原因の 1 つは、順次プリフェッチされる表スペースのエクステント・サイズと、ブロック・ベースのバッファー・プールのブロック・サイズが一致していないことです。

関連資料:

- 249 ページの『vectored_ios ベクトル化入出力要求数』
- 250 ページの『pages_from_vectored_ios ベクトル化入出力によって読み取られたページ数の合計』
- 250 ページの『block_ios ブロック入出力要求数』

physical_page_maps 物理ページ・マップ数

エレメント ID physical_page_maps

エレメント・タイプ ゲージ

表 247. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	バッファー・プール

説明 物理ページ・マップの数。

拡張ストレージ

拡張ストレージに関するモニター・エレメント: 拡張ストレージは、バッファー・プール用の 2 次レベルのストレージを備えています。これにより、ユーザーは、各処理で使用可能な最大限度を超えたメモリーにアクセスできるようになります。拡張ストレージは、バッファー・プールに加えて割り振られるセグメントで構成されます。拡張ストレージはセグメントにページを割り当て、必要に応じてセグメント

のアタッチまたは切り離しを行います。セグメントのサイズは構成可能です。アタッチできるセグメントは特定の一時点で 1 つだけです。

すべてのバッファー・プールに対して拡張ストレージは 1 つであり、それぞれのバッファー・プールごとに使用するかしないかを構成できます。

拡張ストレージは、大容量の実メモリーを持つシステムでのみ使用してください。つまり、単一の処理でアタッチ可能なメモリー容量を超えるメモリーを持つシステムです。

バッファー・プールに拡張ストレージを設定している場合、バッファー・プールから除去されたページがすべて拡張ストレージに書き込まれます。書き込みには、毎回コストが発生します。ページの中にはまったく使用されないページも含まれます。あるいは、バッファー・プールに再び読み取られる前に、拡張ストレージから強制的に排除されるページもあります。

拡張ストレージの読み取りと書き込みの比率は次のように計算できます。

```
(data + index copied from extended storage)
/ (data + index copied to extended storage)
```

この式の分子は拡張ストレージからバッファー・プールにコピーされたページ数、分母はバッファー・プールから拡張ストレージにコピーされたページ数です。

この式の分子は、パフォーマンスが節約された部分を表しています。拡張ストレージからバッファー・プールにページが転送されると、システム入出力呼び出しが節約できます。ただし、拡張メモリー・セグメントへのアタッチ、ページのコピー、およびセグメントの切り離しなどのコストは発生します。分母は、拡張ストレージのページの転送コストを表し、これにはセグメントへのアタッチ、ページのコピー、および切り離しなどのコストが含まれます。

この比率が高いほど、拡張ストレージが効果的に使用されていることを示します。一般的には、システム上の入出力アクティビティーが非常に多い場合に拡張ストレージが特に役に立ちます。

バッファー・プールから削除するページを拡張ストレージにコピーするコストとディスクから読み取る代わりに拡張ストレージからページを読み取る節約分が交差する点があります。この交差点は、次の条件によって異なります。

- システム上の入出力のコスト
- データをメモリー内にコピーして共有メモリー・セグメントにアクセスするときのコスト

正確な交差点の設定は困難です。ベースラインを設定するには、さまざまなバッファー・プールに対して拡張ストレージの使用を実験し、データベースのパフォーマンスが総合的に改善されるかどうかを判断する必要があります。これにはアプリケーション・ベンチマークを使用して測定できます。例えば、トランザクション速度と実行時間をモニターします。

一部のバッファー・プールに対して拡張ストレージの効果があることがわかったら、次のようにします。ベースラインを取得するために、読み取りと書き込みの比率を測定します。この比率は、データベースの作成および初期設定時に最も重要に

なります。その後は、この比率をモニターして、そのベースラインが最初に設定したベースラインからはずれていないかどうかを確認します。

次のエレメントにより、バッファー・プールと拡張ストレージに関する情報が提供されます。

- pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ : モニター・エレメント
- pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ : モニター・エレメント
- pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ : モニター・エレメント
- pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ : モニター・エレメント

pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ:

エレメント ID pool_data_to_estore

エレメント・タイプ カウンター

表 248. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 249. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 拡張ストレージにコピーされたバッファー・プールのデータ・ページの数。

使用法 ビクティム・ページとして選択されたページは、バッファー・プールから拡張ストレージにコピーされます。このコピー処理が必要になるのは、バッファー・プール内に新しいページのためのスペースが必要な場合です。

関連資料:

- 255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
- 255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
- 256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』

pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ:

エレメント ID pool_index_to_estore

エレメント・タイプ カウンター

表 250. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 251. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 拡張ストレージにコピーされたバッファー・プール索引ページの数。

使用法 ビクティム・ページとして選択されたページは、バッファー・プールから拡張ストレージにコピーされます。このコピー処理が必要になるのは、バッファー・プール内に新しいページのためのスペースが必要な場合です。

関連資料:

- 254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
- 255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』
- 256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』

pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ:

エレメント ID pool_data_from_estore

エレメント・タイプ カウンター

表 252. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 253. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 拡張ストレージからコピーされたバッファー・プールのデータ・ページの数。

使用法 必要なページがバッファー・プールに存在せず拡張ストレージにある場合は、そのページは拡張ストレージからバッファー・プールにコピーされます。このコピー処理により共有メモリー・セグメントへの接続コストが必要になりますが、ディスク読み取りコストを節約できます。

関連資料:

- 254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
- 255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
- 256 ページの『pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ』

pool_index_from_estore 拡張ストレージからコピーされたバッファー・プール索引ページ:

エレメント ID pool_index_from_estore

エレメント・タイプ カウンター

表 254. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 255. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 拡張ストレージからコピーされたバッファー・プール索引ページの数。

使用法 必要な索引ページがバッファー・プールに存在せず拡張ストレージにある場合は、そのページは拡張ストレージからバッファー・プールにコピーされます。このコピー処理により共有メモリー・セグメントへの接続コストが必要になりますが、ディスク読み取りコストを節約できます。

関連資料:

- 254 ページの『pool_data_to_estore 拡張ストレージにコピーされたバッファー・プール・データ・ページ』
- 255 ページの『pool_index_to_estore 拡張ストレージにコピーされたバッファー・プール索引ページ』
- 255 ページの『pool_data_from_estore 拡張ストレージからコピーされたバッファー・プール・データ・ページ』

動的バッファー・プール

bp_cur_buffsz バッファー・プールの現行サイズ:

エレメント ID bp_cur_buffsz

エレメント・タイプ ゲージ

表 256. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool_nodeinfo	バッファー・プール

説明 現行のバッファー・プール・サイズ。

bp_new_buffsz 新規バッファー・プール・サイズ:

エレメント ID bp_new_buffsz

エレメント・タイプ 情報

表 257. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool_nodeinfo	バッファー・プール

説明 データベースが再始動されるとバッファー・プールが変更されるサイズ。ALTER BUFFERPOOL ステートメントが DEFERRED として実行されると、データベースを停止するか再始動するまでバッファー・プール・サイズは変更されません。

bp_pages_left_to_remove 除去残ページ数:

エレメント ID bp_pages_left_to_remove

エレメント・タイプ ゲージ

表 258. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool_nodeinfo	バッファー・プール

説明 バッファー・プールのサイズ変更が完了する前に、バッファー・プールからの除去が残っているページ数。これは IMMEDIATE として実行される ALTER BUFFERPOOL ステートメントによって呼び出されるバッファー・プール・サイズ変更操作にのみ適用されます。

bp_tbsp_use_count バッファー・プールにマップされている表スペースの数:

エレメント ID bp_tbsp_use_count

エレメント・タイプ ゲージ

表 259. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool_nodeinfo	バッファー・プール

説明 このバッファー・プールを使用している表スペースの数。

バッファーを使用しない入出力アクティビティー

バッファーを使用しない入出力アクティビティーに関するモニター・エレメント

次のエレメントにより、バッファー・プールを使用しない入出力アクティビティーに関する情報が提供されます。

- direct_reads データベースからの直接読み取り : モニター・エレメント
- direct_writes データベースへの直接書き込み : モニター・エレメント
- direct_read_reqs 直接読み取り要求 : モニター・エレメント
- direct_write_reqs 直接書き込み要求 : モニター・エレメント
- direct_read_time 直接読み取り時間 : モニター・エレメント
- direct_write_time 直接書き込み時間 : モニター・エレメント

direct_reads データベースからの直接読み取り

エレメント ID direct_reads

エレメント・タイプ カウンター

表 260. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 261. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 バッファー・プールを使用しない読み取り操作の数。

使用法 次の公式を使用すると、直接読み取りにより読み取られるセクターの平均数を計算できます。

```
direct_reads / direct_read_reqs
```

システム・モニターを使用して入出力を追跡するときにこのエレメントを利用すると、装置上のデータベース入出力とそれ以外の入出力を区別できます。

直接読み取りは、最小 512 バイト・セクター単位で処理されます。次の目的に使用します。
- LONG VARCHAR 列の読み取り
- LOB (ラージ・オブジェクト) 列の読み取り
- バックアップの実行

関連資料:

- 259 ページの『direct_writes データベースへの直接書き込み』
- 260 ページの『direct_read_reqs 直接読み取り要求』
- 261 ページの『direct_read_time 直接読み取り時間』

direct_writes データベースへの直接書き込み

エレメント ID direct_writes

エレメント・タイプ カウンター

表 262. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 263. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 バッファー・プールを使用しない書き込み操作の数。

使用法 次の公式を使用すると、直接書き込みにより書き込まれるセクターの平均数を計算できます。

```
direct_writes / direct_write_reqs
```

システム・モニターを使用して入出力を追跡するときにこのエレメントを利用すると、装置上のデータベース入出力とそれ以外の入出力を区別できます。

直接書き込みは、最小 512 バイト・セクター単位で処理されます。次の目的に使用します。
- LONG VARCHAR 列の書き込み
- LOB (ラージ・オブジェクト) 列の書き込み

- リストアの実行
- ロードの実行

関連資料:

- 258 ページの『direct_reads データベースからの直接読み取り』
- 260 ページの『direct_write_reqs 直接書き込み要求』
- 262 ページの『direct_write_time 直接書き込み時間』

direct_read_reqs 直接読み取り要求

エレメント ID direct_read_reqs

エレメント・タイプ カウンター

表 264. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 265. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 1 つ以上のデータ・セクターの直接読み取り実行要求の数。

使用法 次の公式を使用すると、直接読み取りにより読み取られるセクターの平均数を計算できます。

```
direct_reads / direct_read_reqs
```

関連資料:

- 258 ページの『direct_reads データベースからの直接読み取り』
- 260 ページの『direct_write_reqs 直接書き込み要求』
- 261 ページの『direct_read_time 直接読み取り時間』

direct_write_reqs 直接書き込み要求

エレメント ID direct_write_reqs

エレメント・タイプ カウンター

表 266. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール

表 266. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 267. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 1 つ以上のデータ・セクターの直接書き込み実行要求の数。

使用法 次の公式を使用すると、直接書き込みにより書き込まれるセクターの平均数を計算できます。

```
direct_writes / direct_write_reqs
```

関連資料:

- 259 ページの『direct_writes データベースへの直接書き込み』
- 260 ページの『direct_read_reqs 直接読み取り要求』
- 262 ページの『direct_write_time 直接書き込み時間』

direct_read_time 直接読み取り時間

エレメント ID direct_read_time

エレメント・タイプ カウンター

表 268. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 269. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 直接読み取りの実行に要した経過時間 (ミリ秒)。

使用法 次の公式を使用して、セクター当たりの直接読み取り平均時間を計算します。

```
direct_read_time / direct_reads
```

平均時間が長い場合には、入出力が競合していることがあります。

関連資料:

- 258 ページの『direct_reads データベースからの直接読み取り』
- 260 ページの『direct_read_reqs 直接読み取り要求』
- 262 ページの『direct_write_time 直接書き込み時間』

direct_write_time 直接書き込み時間

エレメント ID direct_write_time

エレメント・タイプ カウンター

表 270. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	バッファー・プール
表スペース	tablespace	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
アプリケーション	appl	バッファー・プール

スナップショット・モニターの場合、このカウンターはリセットできます。

表 271. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
表スペース	event_tablespace	-

説明 直接書き込みの実行に要した経過時間 (ミリ秒)。

使用法 次の公式を使用して、セクター当たりの直接書き込み平均時間を計算します。

```
direct_write_time / direct_writes
```

平均時間が長い場合には、入出力が競合していることがあります。

関連資料:

- 259 ページの『direct_writes データベースへの直接書き込み』
- 260 ページの『direct_write_reqs 直接書き込み要求』
- 261 ページの『direct_read_time 直接読み取り時間』

カタログ・キャッシュ

カタログ・キャッシュに関するモニター・エレメント

カタログ・キャッシュは、次の内容を保管します。

- 表、ビュー、および別名の表記述子。記述子は、表、ビュー、または別名に関する情報をコンデンス内部フォーマットで保管します。 SQL ステートメントが表を参照すると、表記述子がキャッシュに挿入されます。そのため、同じ表を参照する後続の SQL ステートメントはその記述子を使用でき、ディスクからの読み取りが不要になります。 (トランザクションは、 SQL ステートメントのコンパイル時に表記述子を参照します。)
- データベース許可情報。 BIND、CONNECT、CREATE および LOAD などのステートメントを処理すると、データベース許可情報へのアクセスが行われます。ステートメントがデータベース許可情報を参照すると、その後の操作で同じユーザーやグループについてデータベース許可情報を参照するときには、ディスクからではなく、カタログ・キャッシュからアクセスできます。
- ユーザー定義関数やストアード・プロシージャーなどのルーチンのための実行特権。特定のルーチンについてトランザクションが実行特権を参照すると、その後の操作で同じルーチンを参照するときには、情報をディスクからではなく、カタログ・キャッシュから取り出すことができます。

カタログ・キャッシュには、次のデータベース・システム・モニター・エレメントが使用されます。

- cat_cache_lookups カタログ・キャッシュ参照数 : モニター・エレメント
- cat_cache_inserts カタログ・キャッシュ挿入数 : モニター・エレメント
- cat_cache_overflows カタログ・キャッシュ・オーバーフロー数 : モニター・エレメント
- cat_cache_size_top カタログ・キャッシュ最高水準点 : モニター・エレメント

cat_cache_lookups カタログ・キャッシュ参照数

エレメント ID cat_cache_lookups

エレメント・タイプ カウンター

表 272. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 273. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 表の記述子情報または許可情報を取得するためにカタログ・キャッシュが参照された回数。

使用法 このエレメントには、カタログ・キャッシュへの正常に行われたアクセスと失敗したアクセスの両方が含まれます。カタログ・キャッシュは、次の場合に参照されます。

- SQL ステートメントのコンパイル中に、表、ビュー、または別名を処理したとき。

- データベース許可情報にアクセスがあったとき。
- SQL ステートメントのコンパイル中にルーチンを処理したとき。

カタログ・キャッシュ・ヒット率の計算には次の公式を使用します。

```
(1 - (cat_cache_inserts / cat_cache_lookups))
```

この値は、カタログ・キャッシュがどの程度カタログ・アクセスを回避しているかを示します。この比率が高い場合 (0.8 を超える値)、キャッシュは効果的に動作しています。比率が低い場合は、 *catalogcache_sz* を大きくする必要があることを示します。データベースへの最初の接続の直後は、この比率は高くなります。

表、ビュー、別名などに関係するデータ定義言語 (DDL) SQL ステートメントは、そのようなオブジェクトに関する表記述子情報を取り除くため、それらのオブジェクトは次の参照で再挿入されることになります。さらに、データベース許可およびルーチンの実行特権のための GRANT および REVOKE のステートメントによって、該当する許可情報がカタログ・キャッシュから取り除かれます。したがって、DDL ステートメントと GRANT/REVOKE ステートメントを多用した場合も、この比率は大きくなります。

「カタログ・キャッシュ・サイズ」構成パラメーターについて詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 264 ページの『cat_cache_inserts カタログ・キャッシュ挿入数』
- 265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』
- 367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』
- 266 ページの『cat_cache_size_top カタログ・キャッシュ最高水準点』

cat_cache_inserts カタログ・キャッシュ挿入数

エレメント ID cat_cache_inserts

エレメント・タイプ カウンター

表 274. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 275. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 システムが表記述子または許可情報をカタログ・キャッシュに挿入しようとした回数。

使用法 カタログ・キャッシュ参照数と組み合わせると、次の公式を使用してカタログ・キャッシュ・ヒット率を計算できます。

```
1 - (カタログ・キャッシュ挿入数/カタログ・キャッシュ参照数)
```

このエレメントの使用法については、『cat_cache_lookups』を参照してください。

関連資料:

- 263 ページの『cat_cache_lookups カタログ・キャッシュ参照数』
- 265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』
- 266 ページの『cat_cache_size_top カタログ・キャッシュ最高水準点』

cat_cache_overflows カタログ・キャッシュ・オーバーフロー数

エレメント ID cat_cache_overflows

エレメント・タイプ カウンター

表 276. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 277. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 割り振られたメモリーの境界からカタログ・キャッシュがオーバーフローした回数。

使用法 このエレメントと *cat_cache_size_top* を組み合わせて使用すると、オーバーフローを防止するのにカタログ・キャッシュのサイズを大きくする必要があるかどうかを判別できます。

カタログ・キャッシュのスペースは、どのトランザクションでも現在使用されていない、表、ビュー、または別名に関する表記述子情報や、許可情報を除去することで取り戻します。

cat_cache_overflows が大きい場合は、ワークロードに対してカタログ・キャッシュが小さすぎることが考えられます。カタログ・キャッシュを大きくすると、パフォーマンスが改善されることがあります。多数の表、ビュー、別名、ユーザー定義関数またはストアード・プロシージャーを参照する多数の SQL ステートメントを、 1 つの作業単位にコンパイルするトランザクションがワークロードに含まれている場合は、 1 つのトランザクションにコンパイルする SQL ステートメントの数を少なくすると、カタログ・キャッシュのパフォーマンスが改善されることがあります。あるいは多数の表、ビュー、別名、ユーザー定義関数またはストアード・プロシージャーを参照する多数の SQL ステートメントが入ったパッケージのバインドがワークロード

に含まれる場合は、パッケージを分割してその中に含まれる SQL ステートメントの数を少なくすると、パフォーマンスが改善されることがあります。

関連資料:

- 263 ページの『cat_cache_lookups カタログ・キャッシュ参照数』
- 264 ページの『cat_cache_inserts カタログ・キャッシュ挿入数』
- 266 ページの『cat_cache_size_top カタログ・キャッシュ最高水準点』

cat_cache_size_top カタログ・キャッシュ最高水準点

エレメント ID cat_cache_size_top

エレメント・タイプ 水準点

表 278. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 279. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 カタログ・キャッシュが到達した最大サイズ。

使用法 このエレメントは、データベースが活動化されて以降、データベースでワークロードを実行したときに必要となったカタログ・キャッシュの最大バイト数を示します。

カタログ・キャッシュがオーバーフローした場合、このエレメントは、オーバーフロー時にカタログ・キャッシュが到達した最大サイズになります。このような状態が発生したかどうかを確認するには、「カタログ・キャッシュ・オーバーフロー」をチェックしてください。

ワークロードに必要なカタログ・キャッシュの最小サイズは次のように決定できます。

```
maximum catalog cache size / 4096
```

この結果を切り上げた整数が、オーバーフローを避けるためにカタログ・キャッシュが必要とする 4K ページの最小数になります。

関連資料:

- 265 ページの『cat_cache_overflows カタログ・キャッシュ・オーバーフロー数』

パッケージ・キャッシュ

パッケージ・キャッシュに関するモニター・エレメント

動的および静的 SQL ステートメントの実行に必要なパッケージとセクションの情報は、必要に応じてパッケージ・キャッシュに置かれます。この情報は、動的または静的ステートメントを実行する際に必ず必要になります。パッケージ・キャッシュはデータベース・レベルです。これは、同じような環境のエージェントが別のエージェントの作業の利点を共有できることを意味します。静的 SQL ステートメン

トの場合は、カタログ・アクセスしなくて済みます。動的 SQL ステートメントの場合は、コンパイルのコストをかからなくすることができます。

パッケージ・キャッシュには、次のデータベース・システム・モニター・エレメントが使用されます。

- pkg_cache_lookups パッケージ・キャッシュ参照 : モニター・エレメント
- pkg_cache_inserts パッケージ・キャッシュ挿入 : モニター・エレメント
- pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー : モニター・エレメント
- pkg_cache_size_top パッケージ・キャッシュの最高水準点 : モニター・エレメント
- appl_section_lookups セクションの参照回数 : モニター・エレメント
- appl_section_inserts セクション挿入数 : モニター・エレメント

pkg_cache_lookups パッケージ・キャッシュ参照

エレメント ID pkg_cache_lookups

エレメント・タイプ カウンター

表 280. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 281. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 アプリケーションがパッケージ・キャッシュ内のセクションやパッケージを検索した回数。データベース・レベルでは、データベースの開始以降、またはモニター・データのリセット以降の参照の合計数を示します。

注: このカウンターには、セクションをキャッシュにすでにロードしてある場合と、セクションをキャッシュにロードする必要がある場合が含まれます。

エージェントがさまざまなアプリケーションと関連付けられているようなコンセントレーター環境では、新しいエージェントに必要なセクションやパッケージがローカル・ストレージ内にない場合に、パッケージ・キャッシュの検索がさらに必要になります。

使用法 パッケージ・キャッシュ・ヒット率の計算には次の公式を使用します。

```
1 - (パッケージ・キャッシュ参照/パッケージ・キャッシュ挿入)
```

パッケージ・キャッシュ・ヒット率は、パッケージ・キャッシュが効果的に利用されているかどうかを示します。この比率が高い場合 (0.8 を超える

値)、キャッシュは効果的に動作しています。この比率が低い場合は、パッケージ・キャッシュを大きくする必要があることを示します。

パッケージ・キャッシュのサイズを変えて試すことにより、 *pckcachesz* 構成パラメーターに最適な値を見つける必要があります。例えば、キャッシュのサイズを小さくしても *pkg_cache_inserts* エレメントが増えない場合は、パッケージ・キャッシュのサイズをさらに小さくできます。パッケージ・キャッシュのサイズを小さくすれば、その分のシステム・リソースを他の作業のために使えるようになります。または、pkg_cache_inserts の数を少なくして、パッケージ・キャッシュのサイズを大きくすると、システム全体のパフォーマンスを向上させることができます。この実験は、フル・ワークロードの条件で行うのが最善です。

このエレメントと *ddl_sql_stmts* を組み合わせて使用すると、 DDL ステートメントを実行したときにパッケージ・キャッシュのパフォーマンスに影響を与えるかどうかを判別できます。 DDL ステートメントを実行すると、動的 SQL ステートメントの一部のセクションが無効になる場合があります。無効なセクションは、次に使用されるときにシステムが暗黙的に準備します。 DDL ステートメントを実行すると、多数のセクションが無効になり、こうしたセクションを準備するときに余分に必要になるオーバーヘッドのためにパフォーマンスが大きく低下することがあります。この場合のパッケージ・キャッシュ・ヒット率は、無効なセクションの暗黙的な再コンパイルを反映しますが、キャッシュに挿入される新しいセクションは反映しないので、パッケージ・キャッシュのサイズを大きくしても総合的なパフォーマンスは改善できません。フル環境を対象に作業する前に、アプリケーション自体のキャッシュを調整すれば、混乱を避けることができます。

実行すべきアクションを考える前に、パッケージ・キャッシュ・ヒット率の値に DDL ステートメントがどのような役割を果たしているのかを明確にする必要があります。 DDL ステートメントがあまり発生しない場合は、キャッシュのサイズを大きくするとキャッシュのパフォーマンスを改善できる場合があります。 DDL ステートメントが頻繁に使用される場合は、 DDL ステートメントを制限する (時間を限定するなど) と改善できる場合があります。

static_sql_stmts および *dynamic_sql_stmts* のカウントは、キャッシュに入れるセクションの数量とタイプに関する情報を提供するときに利用できます。

「パッケージ・キャッシュ・サイズ (*pckcachesz*)」構成パラメーターについて詳しくは、「*管理ガイド*」を参照してください。

注: この情報をデータベース・レベルで使用すると、すべてのアプリケーションについて個別の平均パッケージ・キャッシュ・ヒット率を計算できます。特定のアプリケーションのパッケージ・キャッシュ・ヒット率が知りたいときには、この情報をアプリケーション・レベルで調べてください。実行頻度の少ないアプリケーションのキャッシュ要件を満たすためにパッケージ・キャッシュのサイズを大きくしてもあまり意味がありません。

関連資料:

- 269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
- 361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
- 362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』
- 367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』

pkg_cache_inserts パッケージ・キャッシュ挿入

エレメント ID pkg_cache_inserts

エレメント・タイプ カウンター

表 282. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 283. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 要求したセクションが使用できなかったためにパッケージ・キャッシュへのロードが必要になった回数の合計。このカウントには、システムが暗黙に準備した数が含まれます。

使用法 「パッケージ・キャッシュ参照」と組み合わせると、次の公式を使用してパッケージ・キャッシュ・ヒット率を計算できます。

```
1 - (パッケージ・キャッシュ参照/パッケージ・キャッシュ挿入)
```

このエレメントの使用法については、『pkg_cache_lookups』を参照してください。

関連資料:

- 267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』

pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー

エレメント ID pkg_cache_num_overflows

エレメント・タイプ カウンター

表 284. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 285. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 割り振られたメモリーの境界からパッケージ・キャッシュがオーバーフローした回数。

使用法 このエレメントと pkg_cache_size_top を組み合わせて使用すると、オーバーフローを回避するのにパッケージ・キャッシュのサイズを大きくする必要があるかどうかを判別できます。

関連資料:

- 269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
- 361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
- 362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』
- 367 ページの『ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント』

pkg_cache_size_top パッケージ・キャッシュの最高水準点

エレメント ID pkg_cache_size_top

エレメント・タイプ 水準点

表 286. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 287. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 パッケージ・キャッシュが到達した最大サイズ。

使用法 このエレメントは、データベースが活動化されて以降、データベースでのワークロードの実行にパッケージ・キャッシュが必要とした最大バイト数を示します。

パッケージ・キャッシュがオーバーフローした場合、このエレメントは、オーバーフロー時にパッケージ・キャッシュが到達した最大サイズになります。このような状態が発生したかどうかを判別するには、パッケージ・キャッシュ・オーバーフローをチェックしてください。

ワークロードに必要なパッケージ・キャッシュの最小サイズは次のように決定できます。

```
パッケージ・キャッシュの最大サイズ / 4096
```

この結果を切り上げた整数が、オーバーフローを避けるためにパッケージ・キャッシュが必要とする 4K ページの最小数になります。

関連資料:

• 269 ページの『pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー』

appl_section_lookups セクションの参照回数 : モニター・エレメント

エレメント ID appl_section_lookups

エレメント・タイプ カウンター

表 288. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 289. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 SQL 作業域からのアプリケーションによる SQL セクション参照数。

使用法 個々のエージェントには、実行可能セクションの作業用コピーが保持されるユニークな SQL 作業域へのアクセス権があります。パーティション・データベースでは、この作業域はすべての非 SMP エージェントに共有されます。 SMP エージェントのあるその他の環境では、個々のエージェントに独自かつユニークな SQL 作業域があります。

このカウンターは、アプリケーションのエージェントにより SQL 作業域がアクセスされた回数を示します。このカウンターは、このアプリケーションに関して作動しているエージェント用の SQL 作業ヒープすべてに対する、参照回数すべての累計です。

このエレメントと *appl_section_inserts* を組み合わせて使用すると、 SQL 作業域に使用されるヒープのサイズを調整できます。パーティション・データベースでは、このサイズを制御しているのは、 *app_ctl_heap_sz* 構成パラメーターです。その他のデータベース環境では、 SQL 作業域のサイズには *applheapsz* 構成パラメーターが使用されます。すべての環境で、SMP エージェント用の SQL 作業域のサイズは、 *applheapsz* によって制御されます。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『app_ctl_heap_sz - 「アプリケーション・コントロール・ヒープ・サイズ」構成パラメーター』
- 「*管理ガイド: パフォーマンス*」の『applheapsz - 「アプリケーション・ヒープ・サイズ」構成パラメーター』
- 267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』
- 269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
- 272 ページの『appl_section_inserts セクション挿入数 : モニター・エレメント』

appl_section_inserts セクション挿入数 : モニター・エレメント

エレメント ID appl_section_inserts

エレメント・タイプ カウンター

表 290. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

表 291. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 SQL 作業域からのアプリケーションによる SQL セクション挿入数。

使用法 実行可能セクションの作業用コピーは、ユニークな SQL 作業域に保管されます。このカウンターは、コピーが使用できなかったために挿入が必要だった場合のカウントです。

セクションの使用法について詳しくは、『*appl_section_lookups*』を参照してください。

関連資料:

- 267 ページの『pkg_cache_lookups パッケージ・キャッシュ参照』
- 269 ページの『pkg_cache_inserts パッケージ・キャッシュ挿入』
- 271 ページの『appl_section_lookups セクションの参照回数 : モニター・エレメント』

SQL ワークスペース

SQL ワークスペースに関するモニター・エレメント

動的または静的 SQL ステートメントを実行するためにアプリケーションでセクションが必要になると、セクションは、必要に応じて共有ワークスペースまたは専用ワークスペース内に置かれます。共有ワークスペースは、アプリケーション・レベルに存在し、 1 つ以上のアプリケーションが共有しています。専用ワークスペースはエージェント・レベルに存在し、それぞれのエージェントに 1 つの専用ワークスペースが関連付けられています。

共有ワークスペースは多数のアプリケーションが共有しているので、同じような環境のアプリケーションが別のエージェントの作業の利点を共有できます。利点としては、セットアップおよび初期設定のコストなどがあります。

以下のデータベース・システム・モニター・エレメントは、SQL ワークスペースに使用されます。

- shr_workspace_size_top 最大共有ワークスペース・サイズ : モニター・エレメント
- shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数 : モニター・エレメント

- shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数 : モニター・エレメント
- shr_workspace_section_inserts 共有ワークスペース・セクション挿入数 : モニター・エレメント
- priv_workspace_size_top 専用ワークスペースの最大サイズ : モニター・エレメント
- priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数 : モニター・エレメント
- priv_workspace_section_lookups 専用ワークスペース・セクションの参照 : モニター・エレメント
- priv_workspace_section_inserts 専用ワークスペース・セクション挿入 : モニター・エレメント

shr_workspace_size_top 共有ワークスペースの最大サイズ

エレメント ID shr_workspace_size_top

エレメント・タイプ 水準点

表 292. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

表 293. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 共有ワークスペースが到達した最大サイズ。

使用法 このエレメントは、データベースが活動化されて以降、データベースでワークロードを実行したときに必要となった共有ワークスペースの最大バイト数を示します。データベース・レベルでは、すべての共有ワークスペースが到達した最大サイズを示します。アプリケーション・レベルでは、現行アプリケーションが使用する共有ワークスペースの最大サイズです。

共有ワークスペースがオーバーフローした場合、このエレメントは、オーバーフロー時に共有ワークスペースが到達した最大サイズになります。このような状態が発生したかどうかを確認するには、「共有ワークスペースのオーバーフロー回数」をチェックしてください。

共有ワークスペースがオーバーフローすると、アプリケーションの共有メモリーにあるほかのエンティティーからメモリーを一時的に借用します。この結果、これらのエンティティーでメモリー不足エラーが発生したり、パフォーマンスが低下することがあります。 APPL_CTL_HEAP_SZ を大きくすると、オーバーフローの確率を低くすることができます。

関連資料:

- 274 ページの『shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数』

shr_workspace_num_overflows 共有ワークスペースのオーバーフロー回数

エレメント ID shr_workspace_num_overflows

エレメント・タイプ カウンター

表 294. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 295. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 割り振られたメモリーの境界から共有ワークスペースがオーバーフローした回数。

使用法 このエレメントと shr_workspace_size_top を組み合わせて使用すると、オーバーフローを防止するのに共有ワークスペースのサイズを大きくする必要があるかどうかを判別できます。共有ワークスペースがオーバーフローすると、パフォーマンスが低下するだけではなく、アプリケーションの共有メモリーから割り振られたほかのヒープでメモリー不足エラーが発生することがあります。

データベース・レベルでは、「共有ワークスペースの最大サイズ」のある共有ワークスペースとして報告された共有ワークスペースがこのエレメントの報告の対象となります。アプリケーション・レベルでは、現行アプリケーションが使用するワークスペースのオーバーフロー回数を示します。

関連資料:

• 273 ページの『shr_workspace_size_top 共有ワークスペースの最大サイズ』

shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数

エレメント ID shr_workspace_section_lookups

エレメント・タイプ カウンター

表 296. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 297. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 共有ワークスペースでの、アプリケーションによる SQL セクション参照数。

使用法 各アプリケーションは、実行可能セクションの作業用コピーがある共有ワークスペースにアクセスできます。

このカウンターは、アプリケーションの特定のセクションを見つけるために共有ワークスペースがアクセスされた回数を示します。データベース・レベルでは、データベース内のすべての共有ワークスペースを対象に、すべてのアプリケーションでの累計参照数を示します。アプリケーション・レベルでは、このアプリケーションの共有ワークスペース内にあるすべてのセクションを対象とした累計参照数を示します。

このエレメントと「共有ワークスペース・セクション挿入数」を組み合わせて使用すると、共有ワークスペースのサイズを調整できます。共有ワークスペースのサイズをコントロールしているのは、 app_ctl_heap_sz 構成パラメーターです。

関連資料:

- 275 ページの『shr_workspace_section_inserts 共有ワークスペース・セクション挿入数』

shr_workspace_section_inserts 共有ワークスペース・セクション挿入数

エレメント ID shr_workspace_section_inserts

エレメント・タイプ カウンター

表 298. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 299. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 共有ワークスペースへの、アプリケーションによる SQL セクション挿入数。

使用法 実行可能セクションの作業用コピーは、共有ワークスペース内に保管されます。このカウンターは、コピーが使用できなかったために挿入が必要だった場合を示します。

データベース・レベルでは、データベース内のすべての共有ワークスペースを対象に、すべてのアプリケーションでの累計挿入数を示します。アプリケーション・レベルでは、このアプリケーションの共有ワークスペース内にあるすべてのセクションを対象とした累計挿入数を示します。

関連資料:

- 274 ページの『shr_workspace_section_lookups 共有ワークスペース・セクションの参照回数』

priv_workspace_size_top 専用ワークスペースの最大サイズ

エレメント ID priv_workspace_size_top

エレメント・タイプ 水準点

表 300. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

表 301. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 専用ワークスペースが到達した最大サイズ。

使用法 各エージェントには 1 つの専用ワークスペースがあり、エージェントがサービスを提供するアプリケーションはこれにアクセスをします。このエレメントは、アプリケーションにサービスを提供するエージェントが必要とする専用ワークスペースの最大バイト数を示します。データベース・レベルでは、現行データベースにアタッチされているすべてのエージェントが必要とする、すべての専用ワークスペースの最大バイト数を示します。アプリケーション・レベルでは、現行アプリケーションにサービスを提供したエージェントのすべての専用ワークスペースの中での最大サイズを示します。

専用ワークスペースがオーバーフローすると、エージェント専用メモリーにあるほかのエンティティーからメモリーを一時的に借用します。この結果、これらのエンティティーでメモリー不足エラーが発生したり、パフォーマンスが低下することがあります。 APPLHEAPSZ を大きくすると、オーバーフローの確率を低くすることができます。

関連資料:

- 276 ページの『priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数』

priv_workspace_num_overflows 専用ワークスペースのオーバーフロー回数

エレメント ID priv_workspace_num_overflows

エレメント・タイプ カウンター

表 302. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 303. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 割り振られたメモリーの境界から専用ワークスペースがオーバーフローした回数。

使用法 このエレメントと priv_workspace_size_top を組み合わせて使用すると、オーバーフローを防止するのに専用ワークスペースのサイズを大きくする必要があるかどうかを判別できます。専用ワークスペースがオーバーフローすると、パフォーマンスが低下するだけではなく、エージェントの専用メモリーから割り振られたほかのヒープでメモリー不足エラーが発生することがあります。

データベース・レベルでは、「専用ワークスペースの最大サイズ」のある専用ワークスペースとして報告された専用ワークスペースがこのエレメントの報告の対象となります。アプリケーション・レベルでは、現行アプリケーションにサービスを提供した各エージェントのワークスペースがオーバーフローした回数となります。

関連資料:

• 276 ページの『priv_workspace_size_top 専用ワークスペースの最大サイズ』

priv_workspace_section_lookups 専用ワークスペース・セクションの参照

エレメント ID priv_workspace_section_lookups

エレメント・タイプ カウンター

表 304. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 305. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 エージェントの専用ワークスペースでの、アプリケーションによる SQL セクション参照数。

使用法 各アプリケーションは、自分に代わって作業するエージェントの専用ワークスペースにアクセスできます。

このカウンターは、アプリケーション用の特定セクションを見つけるために専用ワークスペースがアクセスされた回数を示します。データベース・レベルでは、データベース内のすべての専用ワークスペースを対象に、すべてのアプリケーションでの累計参照数を示します。アプリケーション・レベルでは、このアプリケーションの専用ワークスペース内にあるすべてのセクションを対象とした累計参照数を示します。

このエレメントと「専用ワークスペース・セクション挿入」を組み合わせて使用すると、専用ワークスペースのサイズを調整できます。専用ワークスペースのサイズをコントロールしているのは、applheapsz 構成パラメーターです。

関連資料:

- 278 ページの『priv_workspace_section_inserts 専用ワークスペース・セクション挿入』

priv_workspace_section_inserts 専用ワークスペース・セクション挿入

エレメント ID priv_workspace_section_inserts

エレメント・タイプ カウンター

表 306. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 307. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 専用ワークスペースへの、アプリケーションによる SQL セクション挿入数。

使用法 実行可能セクションの作業用コピーは、専用ワークスペース内に保管されます。

このカウンターは、コピーが使用できなかったために挿入が必要だった場合を示します。データベース・レベルでは、データベース内のすべての専用ワークスペースを対象に、すべてアプリケーションでの累計挿入数を示します。アプリケーション・レベルでは、このアプリケーションの専用ワークスペース内にあるすべてのセクションを対象とした累計挿入数を示します。

エージェントが異なるアプリケーションに関連付けられているようなコンセントレーター環境では、新しいエージェントに必要な使用可能なセクションが専用ワークスペース内にない場合に、専用ワークスペースの追加挿入が必要になります。

関連資料:

- 277 ページの『priv_workspace_section_lookups 専用ワークスペース・セクションの参照』

データベース・ヒープ

データベース・ヒープに関するモニター・エレメント

データベース・ヒープには、次のデータベース・システム・モニター・エレメントが使用されます。

- db_heap_top 割り振られた最大データベース・ヒープ : モニター・エレメント

db_heap_top 割り振られた最大データベース・ヒープ

エレメント ID db_heap_top

エレメント・タイプ 水準点

表 308. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 309. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 このエレメントは、DB2 のバージョン間での互換性を確保するために維持されています。現在は、メモリーの使用量を計算しますが、データベース・ヒープの使用量だけが対象ではありません。

ロギング

ロギングに関するモニター・エレメント

ロギングには、次のデータベース・システム・モニター・エレメントが使用されます。

- sec_log_used_top 使用された最大 2 次ログ・スペース : モニター・エレメント
- tot_log_used_top 使用された最大合計ログ・スペース : モニター・エレメント
- sec_logs_allocated 現在割り振られている 2 次ログ : モニター・エレメント
- log_reads 読み取られたログ・ページの数 : モニター・エレメント
- log_writes 書き込まれたログ・ページの数 : モニター・エレメント
- uow_log_space_used 作業単位ログ・スペース : モニター・エレメント
- total_log_used 使用されているログ・スペースの合計 : モニター・エレメント
- total_log_available 使用可能なログの合計 : モニター・エレメント

- log_held_by_dirty_pages ダーティー・ページ別に計算されるログ・スペースの量 : モニター・エレメント
- log_to_redo_for_recovery リカバリーの場合に再実行されるログの量 : モニター・エレメント
- log_write_time ログ書き込み時間 : モニター・エレメント
- log_read_time ログ読み取り時間 : モニター・エレメント
- num_log_write_io ログ書き込み数 : モニター・エレメント
- num_log_read_io ログ読み取り数 : モニター・エレメント
- num_log_part_page_io 部分ログ・ページ書き込み数 : モニター・エレメント
- num_log_buffer_full フル・ログ・バッファーの回数 : モニター・エレメント
- num_log_data_found_in_buffer ログ・データがバッファーにある回数 : モニター・エレメント
- first_active_log 先頭アクティブ・ログ・ファイル番号 : モニター・エレメント
- last_active_log 最終アクティブ・ログ・ファイル番号 : モニター・エレメント
- current_active_log 現行アクティブ・ログ・ファイル番号 : モニター・エレメント
- current_archive_log 現行アーカイブ・ログ・ファイル番号 : モニター・エレメント

sec_log_used_top 使用された最大 2 次ログ・スペース

エレメント ID sec_log_used_top

エレメント・タイプ 水準点

表 310. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 311. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 使用された 2 次ログ・スペースの最大量 (バイト単位)。

使用法 このエレメントと *sec_logs_allocated* および *tot_log_used_top* を組み合わせて使用すると、 2 次ログへの現在の依存度が示されます。この値が高い場合は、より大きなログ・ファイル、またはより多くの 1 次ログ・ファイル、あるいはアプリケーション内でより頻度の高い COMMIT ステートメントが必要になります。

結果として、次の構成パラメーターの調整が必要になります。
- logfilsiz
- logprimary
- logsecond
- logretain

データベースに 2 次ログ・ファイルがまったくない場合は、この値はゼロになります。定義されていない場合もゼロになります。

詳しくは、「*管理ガイド*」を参照してください。

注: データベース・システム・モニター情報はバイト単位で示されますが、構成パラメーターは各 4K バイトのページ単位で設定されます。

関連資料:

- 281 ページの『tot_log_used_top 使用された最大合計ログ・スペース』
- 282 ページの『sec_logs_allocated 現在割り振られている 2 次ログ』
- 283 ページの『uow_log_space_used 作業単位ログ・スペース』

tot_log_used_top 使用された最大合計ログ・スペース

エレメント ID tot_log_used_top

エレメント・タイプ 水準点

表 312. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 313. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 使用された全ログ・スペースの最大量 (バイト単位)。

使用法 このエレメントは、ユーザーが割り振った 1 次ログ・スペースの量を評価するときに利用できます。このエレメントの値と割り振った 1 次ログ・スペースの量を比較すると、構成パラメーターの設定値を評価するときに利用できます。割り振った 1 次ログ・スペースは、次の公式で計算できます。

```
logprimary x logfilsiz x 4096 (下記の注を参照)
```

このエレメントと *sec_log_used_top* および *sec_logs_allocated* を組み合わせて使用すると、 2 次ログの依存度を示すことができます。

この値には、1 次と 2 次の両方のログ・ファイルに使用されるスペースが含まれます。

次の構成パラメーターの調整が必要になります。

- logfilsiz
- logprimary
- logsecond

詳しくは、「*管理ガイド*」を参照してください。

注: データベース・システム・モニター情報はバイト単位で示されますが、構成パラメーターは各 4K バイトのページ単位で設定されます。

関連資料:

- 280 ページの『sec_log_used_top 使用された最大 2 次ログ・スペース』

- 282 ページの『sec_logs_allocated 現在割り振られている 2 次ログ』
- 283 ページの『uow_log_space_used 作業単位ログ・スペース』

sec_logs_allocated 現在割り振られている 2 次ログ

エレメント ID sec_logs_allocated

エレメント・タイプ ゲージ

表 314. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 データベースで現在使用されている 2 次ログ・ファイルの合計数。

使用法 このエレメントと *sec_log_used_top* および *tot_log_used_top* を組み合わせて使用すると、 2 次ログへの現在の依存度が分かります。この値が常に高い場合は、より大きなログ・ファイル、またはより多くの 1 次ログ・ファイル、あるいはアプリケーション内でより頻度の高い COMMIT ステートメントが必要になります。

結果として、次の構成パラメーターの調整が必要になります。
- logfilsiz
- logprimary
- logsecond
- logretain

詳しくは、「*管理ガイド*」を参照してください。

関連資料:

- 280 ページの『sec_log_used_top 使用された最大 2 次ログ・スペース』
- 281 ページの『tot_log_used_top 使用された最大合計ログ・スペース』
- 283 ページの『uow_log_space_used 作業単位ログ・スペース』

log_reads 読み取られたログ・ページの数

エレメント ID log_reads

エレメント・タイプ カウンター

表 315. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 316. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーがディスクから読み取ったログ・ページの数。

使用法 このエレメントとオペレーティング・システム・モニターを組み合わせて使用すると、データベース・アクティビティーによって発生した装置上の入出力の量がわかります。

関連資料:

• 283 ページの『log_writes 書き込まれたログ・ページの数』

log_writes 書き込まれたログ・ページの数

エレメント ID log_writes

エレメント・タイプ カウンター

表 317. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 318. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーがディスクに書き込んだログ・ページの数。

使用法 このエレメントとオペレーティング・システム・モニターを組み合わせて使用すると、データベース・アクティビティーによって発生した装置上の入出力の量がわかります。

注: ログ・ページがディスクに書き込まれる際、最後のページがいっぱいになっていない場合があります。その場合、部分的なログ・ページがログ・バッファー内に残り、さらにログ・レコードがページに書き込まれます。その結果、ログ・ページは、ロガーによってディスクに複数回書き込まれることがあります。このエレメントからは、DB2 が生成したページ数は測定できません。

関連資料:

• 282 ページの『log_reads 読み取られたログ・ページの数』

uow_log_space_used 作業単位ログ・スペース

エレメント ID uow_log_space_used

エレメント・タイプ ゲージ

表 319. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位

表 320. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
トランザクション	event_xact	-

説明 モニター対象のアプリケーションで現行作業単位に使用されているログ・スペースの量 (バイト単位)。

使用法 このエレメントを使用すると、作業単位レベルでのロギングの所要量を把握することができます。

関連資料:

- 280 ページの『sec_log_used_top 使用された最大 2 次ログ・スペース』
- 281 ページの『tot_log_used_top 使用された最大合計ログ・スペース』
- 282 ページの『sec_logs_allocated 現在割り振られている 2 次ログ』

total_log_used 使用されているログ・スペースの合計

エレメント ID total_log_used

エレメント・タイプ ゲージ

表 321. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 データベースで現在使用されているアクティブ・ログ・スペースの合計量 (バイト単位)。

使用法 このエレメントを total_log_available とともに使用して、ログ・スペースを使い果たすことを避けるために以下の構成パラメーターを調整する必要があるかどうかを判別します。

- logfilsiz
- logprimary
- logsecond

詳しくは、「*管理ガイド*」を参照してください。

注: データベース・システム・モニター情報はバイト単位で示されますが、構成パラメーターは各 4K バイトのページ単位で設定されます。

関連資料:

- 281 ページの『tot_log_used_top 使用された最大合計ログ・スペース』
- 282 ページの『sec_logs_allocated 現在割り振られている 2 次ログ』
- 283 ページの『uow_log_space_used 作業単位ログ・スペース』
- 157 ページの『appl_id_oldest_xact 最も古いトランザクションを持つアプリケーション』

total_log_available 使用可能なログの合計

エレメント ID total_log_available

エレメント・タイプ ゲージ

表 322. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 非コミット・トランザクションによって使用されていない、データベース内のアクティブ・ログ・スペースの量 (バイト単位)。

使用法 このエレメントを total_log_used とともに使用して、ログ・スペースを使い果たすことを避けるために以下の構成パラメーターを調整する必要があるかどうかを判別します。

- logfilsiz
- logprimary
- logsecond

total_log_available の値が 0 まで下がった場合、SQL0964N が返されます。上記の構成パラメーターの値を大きくするか、あるいは COMMIT、ROLLBACK または FORCE APPLICATION によって最も古いトランザクションを終了する必要があります。

logsecond が -1 に設定されていると、このエレメントには SQLM_LOGSPACE_INFINITE が含まれます。

注: データベース・システム・モニター情報はバイト単位で示されますが、構成パラメーターは各 4K バイトのページ単位で設定されます。

関連資料:

- 282 ページの『sec_logs_allocated 現在割り振られている 2 次ログ』
- 283 ページの『uow_log_space_used 作業単位ログ・スペース』
- 284 ページの『total_log_used 使用されているログ・スペースの合計』
- 157 ページの『appl_id_oldest_xact 最も古いトランザクションを持つアプリケーション』

log_held_by_dirty_pages ダーティー・ページ別に計算されるログ・スペースの量

エレメント ID log_held_by_dirty_pages

エレメント・タイプ 水準点

表 323. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 324. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 データベース中の最も古いダーティー・ページと、アクティブ・ログの先頭との間の差に対応するログの量 (バイト単位)。

使用法 この値は、スナップショットをとる際に、そのスナップショットの時点の条件に基づいて計算されます。

このエレメントは、バッファー・プール中の最も古いページに関するページ・クリーニングの有効性を評価するのに使用してください。

バッファー・プール中の古いページのクリーニングは、 *softmax* データベース構成パラメーターによって管理されます。ページ・クリーニングが有効な場合は、 *log_held_by_dirty_pages* がほぼ次の値以下である必要があります。

```
(softmax / 100)  *  logfilsiz * 4096
```

このステートメントが真でない場合は、ページ・クリーナー数 (*num_iocleaners*) 構成パラメーターを大きくしてください。

この条件が真で、ダーティー・ページに保持されるログを少なくしたい場合は、 *softmax* 構成パラメーターを小さくしてください。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『logfilsiz - 「ログ・ファイルのサイズ」構成パラメーター』
- 「*管理ガイド: パフォーマンス*」の『softmax - 「リカバリー範囲およびソフト・チェックポイント・インターバル」構成パラメーター』
- 「*管理ガイド: パフォーマンス*」の『num_iocleaners - 「非同期ページ・クリーナーの数」構成パラメーター』
- 244 ページの『pool_lsn_gap_clns 起動されたバッファー・プール・ログ・スペース・クリーナー』
- 245 ページの『pool_drty_pg_steal_clns 起動されたバッファー・プール・ビクティム・ページ・クリーナー』
- 247 ページの『pool_drty_pg_thrsh_clns 起動されたバッファー・プールしきい値クリーナー』
- 286 ページの『log_to_redo_for_recovery リカバリーの場合に再実行されるログの量』

log_to_redo_for_recovery リカバリーの場合に再実行されるログの量

エレメント ID log_to_redo_for_recovery

エレメント・タイプ 水準点

表 325. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 326. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 クラッシュ・リカバリーの場合に再実行する必要のあるログの量 (バイト単位)。

使用法 この値は、スナップショットをとる際に、そのスナップショットの時点の条

件に基づいて計算されます。値が大きいほど、システムのクラッシュ後のリカバリー時間が長くなることを示します。値が大きすぎるように思える場合は、 *log_held_by_dirty_pages* モニター・エレメントを検査して、ページ・クリーニングを調整する必要があるかどうか調べてください。また、終了する必要のある長期実行トランザクションがあるかどうかも検査してください。

関連資料:

- 285 ページの『log_held_by_dirty_pages ダーティー・ページ別に計算されるログ・スペースの量』

log_write_time ログ書き込み時間

エレメント ID log_write_time

エレメント・タイプ 時間

表 327. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 328. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーがログ・データをディスクに書き込むのに要した合計経過時間。

使用法 このエレメントと *log_writes* および *num_log_write_io* エレメントを組み合わせて使用すると、現行ディスクがロギングに適しているかどうかを判別できます。

関連資料:

- 283 ページの『log_writes 書き込まれたログ・ページの数』
- 288 ページの『num_log_write_io ログ書き込み数』

log_read_time ログ読み取り時間

エレメント ID log_read_time

エレメント・タイプ 時間

表 329. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 330. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーがログ・データをディスクから読み取るのに要した合計経過時間。

使用法 このエレメントと *log_reads*、 *num_log_read_io*、および *num_log_data_found_in_buffer* エレメントを組み合わせて使用すると、次のことを判別できます。

- 現行ディスクがロギングに適しているかどうか。
- ログ・バッファー・サイズが適切かどうか。

関連資料:

- 282 ページの『log_reads 読み取られたログ・ページの数』
- 288 ページの『num_log_read_io ログ読み取り数』
- 290 ページの『num_log_data_found_in_buffer ログ・データがバッファーにある回数』

num_log_write_io ログ書き込み数

エレメント ID num_log_write_io

エレメント・タイプ カウンター

表 331. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 332. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーがログ・データをディスクに書き込むのに発行した入出力要求の数。

使用法 このエレメントと *log_writes* および *log_write_time* エレメントを組み合わせて使用すると、現行ディスクがロギングに適しているかどうかを判別できます。

関連資料:

- 283 ページの『log_writes 書き込まれたログ・ページの数』
- 287 ページの『log_write_time ログ書き込み時間』

num_log_read_io ログ読み取り数

エレメント ID num_log_read_io

エレメント・タイプ カウンター

表 333. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 334. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーがログ・データをディスクから読み取るのに発行した入出力要求の数。

使用法 このエレメントと *log_reads* および *log_read_time* エレメントを組み合わせて使用すると、現行ディスクがロギングに適しているかどうかを判別できます。

関連資料:

- 282 ページの『log_reads 読み取られたログ・ページの数』
- 287 ページの『log_read_time ログ読み取り時間』

num_log_part_page_io 部分ログ・ページ書き込み数

エレメント ID num_log_part_page_io

エレメント・タイプ カウンター

表 335. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 336. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 ロガーが部分的なログ・データをディスクに書き込むのに発行した入出力要求の数。

使用法 このエレメントと *log_writes*、 *log_write_time*、および *num_log_write_io* エレメントを組み合わせて使用すると、現行ディスクがロギングに適しているかどうかを判別できます。

関連資料:

- 283 ページの『log_writes 書き込まれたログ・ページの数』
- 287 ページの『log_write_time ログ書き込み時間』
- 288 ページの『num_log_write_io ログ書き込み数』

num_log_buffer_full フル・ログ・バッファーの回数

エレメント ID num_log_buffer_full

エレメント・タイプ カウンター

表 337. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 エージェントが、ログ・レコードをログ・バッファーにコピーする際に、ログ・データをディスクに書き込むのを待つ回数。この値は、問題が生じるたびにエージェント数単位で大きくなります。例えば、バッファーがいっぱいの場合に 2 つのエージェントがログ・データをコピーしようとすると、この値は 2 単位ずつ大きくなります。

使用法 このエレメントを使用して、 LOGBUFSZ データベース構成パラメーターの値を大きくする必要があるかどうかを判別します。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『logbufsz - 「ログ・バッファー・サイズ」構成パラメーター』

num_log_data_found_in_buffer ログ・データがバッファーにある回数

エレメント ID num_log_data_found_in_buffer

エレメント・タイプ カウンター

表 338. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 339. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 エージェントがバッファーからログ・データを読み取る回数。

バッファーからログ・データを読み取る方が、ディスクから読み取るより速いので望ましいといえます。

使用法 このエレメントと *num_log_read_io* エレメントを組み合わせて使用すると、LOGBUFSZ データベース構成パラメーターの値を大きくする必要があるかどうかを判別します。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『logbufsz - 「ログ・バッファー・サイズ」構成パラメーター』
- 288 ページの『num_log_read_io ログ読み取り数』

first_active_log 先頭アクティブ・ログ・ファイル番号

エレメント ID first_active_log

エレメント・タイプ 情報

表 340. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	detail_log	基本

表 341. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 最初のアクティブ・ログ・ファイルのファイル番号。

使用法 このエレメントと *last_active_log* および *current_active_log* エレメントを組み合わせて使用すると、アクティブ・ログ・ファイルの範囲を判別できます。アクティブ・ログ・ファイルの範囲を知っていると、ログ・ファイルに必要なディスク・スペースを判別するのに役立ちます。

また、このエレメントを使用すると、どのログ・ファイルにデータがあるか判別でき、分割ミラー・サポートが必要なログ・ファイルを識別するのに役立ちます。

関連資料:

- 291 ページの『last_active_log 最終アクティブ・ログ・ファイル番号』
- 292 ページの『current_active_log 現行アクティブ・ログ・ファイル番号』

last_active_log 最終アクティブ・ログ・ファイル番号

エレメント ID last_active_log

エレメント・タイプ 情報

表 342. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	detail_log	基本

表 343. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 最後のアクティブ・ログ・ファイルのファイル番号。

使用法 このエレメントと *first_active_log* および *current_active_log* エレメントを組み合わせて使用すると、アクティブ・ログ・ファイルの範囲を判別できます。アクティブ・ログ・ファイルの範囲を知っていると、ログ・ファイルに必要なディスク・スペースを判別するのに役立ちます。

また、このエレメントを使用すると、どのログ・ファイルにデータがあるか判別でき、分割ミラー・サポートが必要なログ・ファイルを識別するのに役立ちます。

関連資料:

- 290 ページの『first_active_log 先頭アクティブ・ログ・ファイル番号』

• 292 ページの『current_active_log 現行アクティブ・ログ・ファイル番号』

current_active_log 現行アクティブ・ログ・ファイル番号

エレメント ID current_active_log

エレメント・タイプ 情報

表 344. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	detail_log	基本

表 345. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 現在 DB2 UDB が書き込んでいるアクティブ・ログ・ファイルのファイル番号。

使用法 このエレメントと *first_active_log* および *last_active_log* エレメントを組み合わせて使用すると、アクティブ・ログ・ファイルの範囲を判別できます。アクティブ・ログ・ファイルの範囲を知っていると、ログ・ファイルに必要なディスク・スペースを判別するのに役立ちます。

また、このエレメントを使用すると、どのログ・ファイルにデータがあるか判別でき、分割ミラー・サポートが必要なログ・ファイルを識別するのに役立ちます。

関連資料:

• 290 ページの『first_active_log 先頭アクティブ・ログ・ファイル番号』
• 291 ページの『last_active_log 最終アクティブ・ログ・ファイル番号』

current_archive_log 現行アーカイブ・ログ・ファイル番号

エレメント ID current_archive_log

エレメント・タイプ 情報

表 346. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	detail_log	基本

表 347. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

説明 現在 DB2 UDB がアーカイブしているログ・ファイルのファイル番号。

DB2 UDB がログ・ファイルをアーカイブしていない場合は、このエレメントの値は SQLM_LOGFILE_NUM_UNKNOWN になります。

使用法 このエレメントを使用して、ログ・ファイルのアーカイブに問題があるかどうかを判別します。この種の問題には、以下のものがあります。

- 低速のアーカイブ・メディア
- 使用不可であるアーカイブ・メディア

データベースおよびアプリケーション・アクティビティー

データベースおよびアプリケーション・アクティビティーに関するモニター・エレメント

次のセクションに、データベースおよびアプリケーションのアクティビティーについての情報が記載されています。

- ロックおよびデッドロックに関するモニター・エレメント
- ロック待機情報に関するモニター・エレメント
- ロールフォワード・モニターに関するモニター・エレメント
- 表スペース・アクティビティーに関するモニター・エレメント
- 表アクティビティーに関するモニター・エレメント
- 表再編成モニター・エレメント
- SQL カーソルに関するモニター・エレメント
- SQL ステートメント・アクティビティーに関するモニター・エレメント
- SQL ステートメント詳細に関するモニター・エレメント
- サブセクション詳細に関するモニター・エレメント
- 動的 SQL に関するモニター・エレメント
- 照会内並列処理に関するモニター・エレメント
- CPU 使用量に関するモニター・エレメント
- スナップショット・モニターに関するモニター・エレメント
- イベント・モニターに関するモニター・エレメント

ロックおよびデッドロック

ロックおよびデッドロックに関するモニター・エレメント

次のエレメントにより、ロックおよびデッドロックに関する情報が提供されます。

- locks_held ロック保持数 : モニター・エレメント
- lock_list_in_use 使用中のロック・リスト・メモリーの合計 : モニター・エレメント
- deadlocks デッドロック検出数 : モニター・エレメント
- lock_escals ロック・エスカレーション数 : モニター・エレメント
- x_lock_escals 排他ロック・エスカレーション数 : モニター・エレメント
- lock_mode ロック・モード : モニター・エレメント
- lock_status ロック状況 : モニター・エレメント
- lock_object_type 待機中のロック対象タイプ : モニター・エレメント
- lock_object_name ロック対象名 : モニター・エレメント

- lock_node ロック・ノード : モニター・エレメント
- lock_timeouts ロック・タイムアウト数 : モニター・エレメント
- locks_held_top ロック保持最大数 : モニター・エレメント
- dl_conns デッドロックに関係している接続 : モニター・エレメント
- lock_escalation ロック・エスカレーション : モニター・エレメント
- lock_mode_requested 要求されているロック・モード : モニター・エレメント
- deadlock_id デッドロック・イベント ID : モニター・エレメント
- deadlock_node デッドロック発生場所のパーティション番号 : モニター・エレメント
- participant_no デッドロック内の参加者 : モニター・エレメント
- participant_no_holding_lk アプリケーションが必要とするオブジェクトのロックを保留する参加者 : モニター・エレメント
- rolled_back_participant_no ロールバック参加アプリケーション : モニター・エレメント
- locks_in_list 報告されたロックの回数 : モニター・エレメント
- lock_name ロック名 : モニター・エレメント
- lock_attributes ロック属性 : モニター・エレメント
- lock_release_flags ロック保留解除フラグ : モニター・エレメント
- lock_count ロック・カウント : モニター・エレメント
- lock_hold_count ロック保留カウント : モニター・エレメント
- lock_current_mode 移行前の元のロック・モード : モニター・エレメント
- num_indoubt_trans 未確定トランザクション数 : モニター・エレメント

locks_held ロック保持数

エレメント ID locks_held

エレメント・タイプ ゲージ

表 348. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本
ロック	db_lock_list	基本
ロック	appl_lock_list	基本

表 349. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-

説明 現在保持されているロックの数。

使用法 モニター情報がデータベース・レベルの場合は、データベース内のすべてのアプリケーションが現在保持しているロックの合計数を示します。

モニター情報がアプリケーション・レベルの場合は、アプリケーションのすべてのエージェントが現在保持しているロックの合計数を示します。

関連資料:

- 296 ページの『lock_escals ロック・エスカレーション数』
- 297 ページの『x_lock_escals 排他ロック・エスカレーション数』
- 303 ページの『locks_held_top ロック保持最大数』

lock_list_in_use 使用中のロック・リスト・メモリーの合計

エレメント ID lock_list_in_use

エレメント・タイプ *ゲージ*

表 350. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 使用中のロック・リスト・メモリーの合計量 (バイト単位)。

使用法 このエレメントと *locklist* 構成パラメーターを組み合わせて使用すると、ロック・リスト使用率を計算できます。ロック・リスト使用率が高い場合は、そのパラメーターのサイズを増やすことを考慮してください。詳細については、「*管理ガイド*」を参照してください。

注: 使用率を計算する場合、 *locklist* 構成パラメーターが各 4K バイトのページ単位で割り振られるのに対し、モニター・エレメントの結果はバイト数で表されることに注意してください。

deadlocks デッドロック検出数

エレメント ID deadlocks

エレメント・タイプ *カウンター*

表 351. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	ロック

スナップショット・モニターの場合、このカウンターはリセットできます。

表 352. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 発生したデッドロックの合計数。

使用法 このエレメントは、アプリケーション間で競合の問題が起きていることを示す場合があります。問題の原因としては、次の状態が考えられます。

- データベースでロック・エスカレーションが発生している場合。

- システムが生成した行のロッキング数が十分なときに、アプリケーションが表を明示的にロッキングしている場合。
- アプリケーションがバインディングのときに不適切な分離レベルを使用している場合。
- カタログ表が反復可能読み取りのためにロックされている場合。
- 複数のアプリケーションが同じロックを異なる順序で獲得しているために、デッドロックになっている場合。

この問題は、デッドロックが発生しているアプリケーション (またはアプリケーション処理) が判別できれば解決できます。この場合、アプリケーションが並行して実行できるようにアプリケーションを変更できます。ただし、一部のアプリケーションでは並行して実行できない場合があります。

接続タイム・スタンプ・モニター・エレメント (*last_reset*、*db_conn_time*、および *appl_con_time)* を使用すると、デッドロックの重大度を判別できます。例えば、デッドロックが 5 時間に 10 回起こるよりも、5 分間に 10 回起こるほうが重大です。

上記の関連エレメントについての記述部分では、調整に関するその他の推奨事項が示されています。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 296 ページの『lock_escals ロック・エスカレーション数』
- 297 ページの『x_lock_escals 排他ロック・エスカレーション数』
- 315 ページの『appl_id_holding_lk ロックを保持しているアプリケーション ID』

lock_escals ロック・エスカレーション数

エレメント ID lock_escals

エレメント・タイプ カウンター

表 353. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 354. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
トランザクション	event_xact	-

説明 ロックが複数の行ロックから 1 つの表ロックにエスカレートされた回数。

使用法 アプリケーションが保留するロックの合計数がそのアプリケーションで使用可能なロック・リスト・スペースの最大量に達した場合、またはすべてのア

プリケーションが使用するロック・リスト・スペースが合計ロック・リスト・スペースに近くなると、ロックはエスカレートされます。使用可能なロック・リスト・スペースの量は、 *maxlocks* および *locklist* 構成パラメーターによって決まります。

1 つのアプリケーションが使用可能な最大ロック数に達して、エスカレートするロックがほかにない場合は、ほかのアプリケーションに割り振られているロック・リストのスペースが使用されます。ロック・リスト全体が満杯になるとエラーが起こります。

このデータ項目には、排他ロック・エスカレーションも含めて、すべてのロック・エスカレーションのカウントが含まれます。

過剰なロック・エスカレーションが起こる場合は、いくつかの原因が考えられます。

- 同時アプリケーションの数に対してロック・リスト・サイズ (*locklist*) が小さい場合。
- 各アプリケーションが使用できるロック・リストのパーセント値 (*maxlocks*) が小さい場合。
- 1 つ以上のアプリケーションが使用しているロックの数が多すぎる場合。

これらの問題を解決するには、次のようにしてください。

- *locklist* 構成パラメーター値を大きくする。この構成パラメーターの記述については、「*管理ガイド*」を参照してください。
- *maxlocks* 構成パラメーター値を大きくする。この構成パラメーターの記述については、「*管理ガイド*」を参照してください。
- 次の公式を使用して、ロック数の多いアプリケーション (『*locks_held_top*』参照)、または大量のロック・リストを保留しているアプリケーションを識別する。

```
(((locks held * 36) / (locklist * 4096)) * 100)
```

ここで、maxlocks の値を比較します。これらのアプリケーションがロック・リストの多くを使用すると、ほかのアプリケーションでロック・エスカレーションを起こします。これらのアプリケーションは行ロックではなく表ロックを使用して解決しようとしますが、表ロックを使用すると lock_waits および lock_wait_time の増加の原因となることがあります。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 297 ページの『x_lock_escals 排他ロック・エスカレーション数』
- 303 ページの『locks_held_top ロック保持最大数』

x_lock_escals 排他ロック・エスカレーション数

エレメント ID x_lock_escals

エレメント・タイプ カウンター

表 355. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

表 355. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 356. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
トランザクション	event_xact	-

説明 ロックが複数の行ロックから 1 つの排他表ロックにエスカレートされた回数。または行に対する排他ロックにより表ロックが排他ロックになった回数。

使用法 他のアプリケーションは排他ロックによって保留されているデータにアクセスすることができません。そのため、排他ロックはデータの並行性に影響を与える可能性があるため、それを追跡することは重要です。

アプリケーションが保留するロックの合計数がそのアプリケーションで使用可能なロック・リスト・スペースの最大量に達すると、ロックはエスカレートされます。使用可能なロック・リスト・スペースの量は、 *locklist* および *maxlocks* 構成パラメーターによって決まります。

1 つのアプリケーションが使用可能な最大ロック数に達して、エスカレートするロックがほかにない場合は、ほかのアプリケーションに割り振られているロック・リストのスペースが使用されます。ロック・リスト全体が満杯になるとエラーが起こります。

過度の排他ロック・エスカレーションが起こる場合の考えられる原因と対策については、『*lock_escals*』を参照してください。

共有ロックが十分にあるのに、アプリケーションは排他ロックを使用することがあります。共有ロックによってロック・エスカレーションの合計数を減らすことはできませんが、排他ロックのエスカレーションよりも共有ロックのエスカレーションのほうが望ましいと考えられます。

関連資料:

- 148 ページの『db_conn_time データベース活動化タイム・スタンプ』
- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 296 ページの『lock_escals ロック・エスカレーション数』
- 303 ページの『locks_held_top ロック保持最大数』

lock_mode ロック・モード

エレメント ID lock_mode

エレメント・タイプ 情報

表 357. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック
ロック	lock	ロック
ロック	lock_wait	ロック

表 358. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 保持されているロックのタイプ。

使用法 このモードは、リソースの競合の原因を判別するときに利用できます。

このエレメントは、調査するモニター情報のタイプにより、次の内容を示します。

- 1 つのアプリケーションがロック待ちをしているオブジェクトに対して、別のアプリケーションが保留しているロックのタイプ (アプリケーション・モニターおよびデッドロック・モニターのレベル)。
- このアプリケーションが保持しているオブジェクトのロック・タイプ (オブジェクト・ロック・レベル)。

このフィールドの値は以下のとおりです。

モード	ロックのタイプ	API 定数
	ロックなし	SQLM_LNON
IS	意図的共有ロック	SQLM_LOIS
IX	意図的排他ロック	SQLM_LOIX
S	共有ロック	SQLM_LOOS
SIX	意図的排他ロックで共有	SQLM_LSIX
X	排他ロック	SQLM_LOOX
IN	意図なし	SQLM_LOIN
Z	超排他ロック	SQLM_LOOZ
U	更新ロック	SQLM_LOOU
NS	次キー共有ロック	SQLM_LONS
NX	次キー排他ロック	SQLM_LONX
W	弱排他ロック	SQLM_LOOW
NW	次キー弱排他ロック	SQLM_LONW

lock_status ロック状況

エレメント ID lock_status

エレメント・タイプ 情報

表 359. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本

表 360. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-

説明 ロックの内部状況を示します。

使用法 このエレメントは、アプリケーションがオブジェクトに対するロックを取得するために待機しているときに起きていることを明らかにするのに役立ちます。アプリケーションがすでに必要なオブジェクトのロックを取得しているように見える場合でも、同じオブジェクトについて異なるタイプのロックの取得を待たなければならないことがあります。

ロックの状況は、次のいずれかになります。

付与済み状態 アプリケーションは、lock_mode が指定する状態のロックを保有していることを示します。

変換中状態 アプリケーションが保持ロックのタイプを変更しようとしていることを示します。例えば、共有ロックを排他ロックに変更するなど。

注: API ユーザーは、データベース・システム・モニターの定数の定義が含まれているヘッダー・ファイル *sqlmon.h* を参照してください。

関連資料:

- 298 ページの『lock_mode ロック・モード』
- 300 ページの『lock_object_type 待機中のロック対象タイプ』
- 301 ページの『lock_object_name ロック対象名』
- 349 ページの『table_file_id 表ファイル ID』

lock_object_type 待機中のロック対象タイプ

エレメント ID lock_object_type

エレメント・タイプ 情報

表 361. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック
ロック	appl_lock_list	ロック
ロック	lock	基本
ロック	lock_wait	ロック

表 362. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 アプリケーションがロックを保持しているオブジェクトのタイプ (オブジェクト・ロック・レベルの情報)、またはアプリケーションがロックの取得を待機しているオブジェクトのタイプ (アプリケーション・レベルおよびデッドロック・レベルの情報)。

使用法 このエレメントは、リソースの競合の原因を判別するときに役立ちます。

オブジェクト・タイプ ID は sqlmon.h で定義されます。オブジェクトのタイプは、次のいずれかになります。

- 表スペース (sqlmon.h の SQLM_TABLESPACE_LOCK)
- 表
- バッファー・プール
- ブロック
- レコード (または行)
- 内部 (データベース・マネージャーが内部で保持するほかのタイプのロック)。

lock_object_name ロック対象名

エレメント ID lock_object_name

エレメント・タイプ 情報

表 363. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック
ロック	appl_lock_list	ロック
ロック	lock	基本

表 364. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 このエレメントは情報提供のみを目的としています。アプリケーションがロックを保留するオブジェクトの名前 (オブジェクト・ロック・レベルの情報)、またはアプリケーションがロックの取得を待機しているオブジェクトの名前 (アプリケーション・レベルおよびデッドロック・レベルの情報)。

使用法 表レベル・ロックの場合のオブジェクト名は、SMS および DMS 表スペースのファイル ID (FID)。行レベルのロックの場合のオブジェクト名は行 ID (RID)。表スペース・ロックの場合のオブジェクト名はブランク。バッファー・プール・ロックの場合のオブジェクト名は、バッファー・プールの名前。

ロックを保留する表を判別するときは、ファイル ID はユニークなものとは限らないため、ファイル ID ではなく、*table_name* および *table_schema* を使用します。

ロックを保留している表スペースを判別するときは、*tablespace_name* を使用します。

関連資料:

- 300 ページの『lock_object_type 待機中のロック対象タイプ』
- 320 ページの『tablespace_name 表スペース名』
- 340 ページの『table_name 表名』
- 341 ページの『table_schema 表スキーマ名』

lock_node ロック・ノード

エレメント ID lock_node

エレメント・タイプ 情報

表 365. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント
デッドロック	event_dlconn	ステートメント
デッドロックの詳細	event_detailed_dlconn	ステートメント

説明 ロックに関係しているノード。

使用法 これはトラブルシューティングに使用できます。

lock_timeouts ロック・タイムアウト数

エレメント ID lock_timeouts

エレメント・タイプ カウンター

表 366. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 367. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 オブジェクトをロックするための要求が許可されずにタイムアウトになった回数。

使用法 このエレメントは、 *locktimeout* データベース構成パラメーターの設定値を調整するときに利用できます。通常の操作レベルと比較して、ロックのタイムアウト回数が多くなった場合は、ロックを長期にわたって保有しているアプリケーションがある可能性があります。この場合このエレメントは、ロックおよびデッドロックに関する他のいくつかのモニター・エレメントを分析して、アプリケーションに問題があるかどうかを判別する必要があることを示している場合があります。

locktimeout データベース構成パラメーターの設定値が高すぎると、ロックのタイムアウト回数が極端に少なくなります。この場合は、アプリケーションがロックを取得するための待機時間が長くなります。詳細については、「*管理ガイド*」を参照してください。

locks_held_top ロック保持最大数

エレメント ID locks_held_top

エレメント・タイプ カウンター

表 368. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
トランザクション	event_xact	-

説明 このトランザクション中に保持されたロックの最大数。

使用法 このエレメントを使用すると、 *maxlocks* 構成パラメーターで定義されている、アプリケーションが使用可能な最大ロック数に近づいているかどうかを判別できます。このパラメーターは、ロック・エスカレーションを起こさずに各アプリケーションが使用できるロック・リストのパーセンテージを示します。ロック・エスカレーションが起こると、データベースに接続されている各アプリケーション間の並行性が低下します。 (このパラメーターについての詳細は、「*管理ガイド*」を参照してください。)

maxlocks パラメーターがパーセンテージで指定され、このエレメントはカウンターとなっているので、次の公式を使用すると、このエレメントが提供するカウントと 1 つのアプリケーションが保持できる合計ロック数を比較できます。

```
(locklist * 4096 / 36 ) * (maxlocks / 100)
```

ロック数が多い場合は、アプリケーション内で実行するコミットの数を多くして、一部のロックを解放する必要があります。

関連資料:

- 294 ページの『locks_held ロック保持数』
- 296 ページの『lock_escals ロック・エスカレーション数』
- 297 ページの『x_lock_escals 排他ロック・エスカレーション数』

dl_conns デッドロックに関係している接続

エレメント ID dl_conns

エレメント・タイプ *ゲージ*

表 369. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-

説明 デッドロックに関係している接続の数。

使用法 モニター・アプリケーションでこのエレメントを使用すると、イベント・モニター・データ・ストリーム内で処理されるデッドロック接続イベント・レコードの数を確認できます。

lock_escalation ロック・エスカレーション

エレメント ID lock_escalation

エレメント・タイプ 情報

表 370. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	ロック
ロック	lock_wait	ロック

表 371. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 ロック要求がロック・エスカレーションの一部として行われたかどうかを示します。

使用法 このエレメントを使用すると、デッドロックの原因が分かりやすくなります。アプリケーションがロック・エスカレーションを起こすようなデッドロックが発生した場合は、ロック・メモリーの量を増やすか、または任意のアプリケーションが要求できるロックのパーセンテージを変更してください。

lock_mode_requested 要求されているロック・モード

エレメント ID lock_mode_requested

エレメント・タイプ 情報

表 372. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock_wait	ロック

表 373. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 アプリケーションが要求しているロック・モード。

使用法 アプリケーションが要求したロックのモード。この値は、リソース競合の原因を判別するのに役立ちます。

deadlock_id デッドロック・イベント ID

エレメント ID deadlock_id

エレメント・タイプ 情報

表 374. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 デッドロックのデッドロック ID。

使用法 モニター・アプリケーションでこのエレメントを使用すると、デッドロック接続イベント・レコードとデッドロック・イベント・レコードを関連付けることができます。

deadlock_node デッドロック発生場所のパーティション番号

エレメント ID deadlock_node

エレメント・タイプ 情報

表 375. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 デッドロックが発生した場所のパーティション番号。

使用法 このエレメントが該当するのは、パーティション・データベースだけです。モニター・アプリケーションでこのエレメントを使用すると、デッドロック接続イベント・レコードとデッドロック・イベント・レコードを関連付けることができます。

participant_no デッドロック内の参加者

エレメント ID participant_no

エレメント・タイプ 情報

表 376. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 このデッドロック内のこの参加者を一意的に識別するシーケンス番号。

使用法 モニター・アプリケーションでこのエレメントを使用すると、デッドロック接続イベント・レコードとデッドロック・イベント・レコードを関連付けることができます。

participant_no_holding_lk アプリケーションが必要とするオブジェクトのロックを保留する参加者

エレメント ID participant_no_holding_lk

エレメント・タイプ 情報

表 377. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 このアプリケーションがロックの取得を待機しているオブジェクトのロックを保留しているアプリケーションの参加者番号。

使用法 このエレメントは、リソースの競合状態にあるアプリケーションの判別に利用できます。

関連資料:

- 315 ページの『appl_id_holding_lk ロックを保持しているアプリケーション ID』

rolled_back_participant_no ロールバック参加アプリケーション

エレメント ID rolled_back_participant_no

エレメント・タイプ 情報

表 378. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-

説明 ロールバックされたアプリケーションを識別する参加者の番号。

使用法 システム管理者はこの情報を基にして、更新を完了できなかったアプリケーションや始動する必要があるアプリケーションを判別できます。

関連資料:

- 316 ページの『rolled_back_appl_id ロールバック・アプリケーション』

locks_in_list 報告されたロックの回数

エレメント ID locks_in_list

エレメント・タイプ 情報

表 379. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-

説明 イベント・モニターの報告対象の特定のアプリケーションが保留するロック数。

lock_name ロック名

エレメント ID lock_name

エレメント・タイプ 情報

表 380. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本
ロック	lock_wait	lock_wait

表 381. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-

説明 内部バイナリー・ロック名。このエレメントはロックのユニーク ID を示します。

lock_attributes ロック属性

エレメント ID lock_attributes

エレメント・タイプ 情報

表 382. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本
ロック	lock_wait	基本

表 383. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-

説明 ロックの属性。

使用法 ロック属性の設定として次のものがあります。ロック属性の各設定は、sqlmon.h で定義されているビット・フラグ値に基づいています。

API 定数	説明
SQLM_LOCKATTR_WAIT_FOR_AVAIL	使用可能を待機
SQLM_LOCKATTR_RR_IN_BLOCK	ブロック内の RR ロック
SQLM_LOCKATTR_DELETE_IN_BLOCK	ブロック内の削除対象行
SQLM_LOCKATTR_RR	RR スキャンによるロック
SQLM_LOCKATTR__UPDATE_DELETE	行ロックの更新/削除
SQLM_LOCKATTR_ALLOW_NEW	新規ロック要求を許可
SQLM_LOCKATTR_NEW_REQUEST	新規ロック・リクエスター

lock_release_flags ロック保留解除フラグ

エレメント ID lock_release_flags

エレメント・タイプ 情報

表 384. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本
ロック	lock_wait	基本

表 385. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-

説明 ロック保留解除フラグ。

使用法 保留解除フラグの設定として次のものがあります。各保留解除フラグは sqlmon.h で定義されているビット・フラグ値に基づいています。

API 定数	説明
SQLM_LOCKRELFLAGS_SQLCOMPILER	SQL コンパイラーによるロック
SQLM_LOCKRELFLAGS_UNTRACKED	非ユニーク、非追跡のロック

注: 割り当てられていないビットはすべてアプリケーション・カーソルに使用されます。

lock_count ロック・カウント

エレメント ID lock_count

エレメント・タイプ ゲージ

表 386. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本

表 387. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-

説明 保持されているロックに関するロックの数。

使用法 この値の範囲は 1 から 255 です。新規ロックが獲得されると増分し、ロックが解放されると減分されます。

lock_count の値が 255 のときは、*トランザクション期間ロック* が保持されていることを示します。この時点でロックが獲得または解放されても

lock_count は増分または減分されなくなります。 lock_count エレメントは次の 2 つのいずれかの場合に値が 255 に設定されます。

1. 獲得されている新規ロックによって lock_count が 255 回増分された場合。
2. トランザクション期間ロックが明示的に獲得された場合。例えば LOCK TABLE ステートメントや INSERT が使用された場合です。

lock_hold_count ロック保留カウント

エレメント ID lock_hold_count

エレメント・タイプ *ゲージ*

表 388. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本

表 389. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-

説明 ロックに置かれている保留の数。保留は WITH HOLD 文節に登録されたカーソルといくつかの DB2 ユーティリティーによってロック上に置かれます。保留があるロックはトランザクションがコミットされても保留解除されません。

lock_current_mode 移行前の元のロック・モード

エレメント ID lock_current_mode

エレメント・タイプ 情報

表 390. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	lock	基本
ロック	lock_wait	基本

表 391. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-

説明 ロック変換操作のとき、保持されているロックのタイプは変換前に完了になります。ロック変換のシナリオの例を挙げます。更新または削除操作のときにターゲット行の X ロックを待機するとします。トランザクションがその行で S または V ロックを保持している場合、変換が必要になります。この時点で lock_current_mode エレメントには値として S または V が割り当てられますが、ロック待機は X ロックに変換されます。

num_indoubt_trans 未確定トランザクション数

エレメント ID num_indoubt_trans

エレメント・タイプ ゲージ

表 392. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本

説明 データベース内に残っている未確定トランザクションの数。

使用法 未確定トランザクションは非コミット・トランザクションのログ・スペースを保留するため、ログがいっぱいになる可能性があります。ログがいっぱいになると、追加のトランザクションは完了できません。この問題を解決するには、手動による試行錯誤によって未確定トランザクションを解決する必要があります。このモニター・エレメントは、試行錯誤的に解決すべき未確定トランザクションの現在の残存数を提供します。

ロック待機情報

ロック待機情報に関するモニター・エレメント

次のエレメントにより、 DB2 エージェントがアプリケーションに代わってロックの取得を待機しているときに戻される情報が提供されます。

- lock_waits ロック待機数 : モニター・エレメント
- lock_wait_time ロック待機中の時間 : モニター・エレメント
- locks_waiting ロックで待機中の現行エージェント : モニター・エレメント
- uow_lock_wait_time ロック待機中の作業単位の合計時間 : モニター・エレメント
- lock_wait_start_time ロック待機開始タイム・スタンプ : モニター・エレメント
- lock_timeout_val ロック・タイムアウト : モニター・エレメント
- agent_id_holding_lock ロックを保持しているエージェント ID : モニター・エレメント
- appl_id_holding_lk ロックを保持しているアプリケーション ID : モニター・エレメント
- sequence_no_holding_lk ロックを保持しているシーケンス番号 : モニター・エレメント
- rolled_back_appl_id ロールバック・アプリケーション : モニター・エレメント
- rolled_back_agent_id ロールバックされたエージェント : モニター・エレメント
- rolled_back_sequence_no ロールバックされたシーケンス番号 : モニター・エレメント

lock_waits ロック待機数

エレメント ID lock_waits

エレメント・タイプ カウンター

表 393. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 394. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 アプリケーションまたは接続がロックを待機した合計回数。

使用法 データベース・レベルでは、アプリケーションがデータベース内でロックを待機した合計回数を示します。

アプリケーション接続レベルでは、この接続がロックを要求し、ほかの接続がデータ上でロックを保留していたために待機した合計回数を示します。

このエレメントと *lock_wait_time* を組み合わせて使用すると、データベース・レベルの場合は平均ロック待機時間を計算できます。この計算は、データベース・レベルとアプリケーション接続レベルのいずれでも行えます。

平均ロック待機時間が長い場合は、多数のロックを保留するアプリケーションまたはロック・エスカレーションを起こしているアプリケーションを探します。これにより、必要に応じてアプリケーションをエスカレーションして並行性を改善します。エスカレーションが原因で平均ロック待機時間が長くなっている場合は、 *locklist* および *maxlocks* 構成パラメーターのどちらか、または両方の設定値が低すぎることが原因と考えられます。

関連概念:

- 「*管理ガイド: パフォーマンス*」の『ロックおよび並行性の制御』

関連資料:

- 173 ページの『appl_con_time 接続要求開始タイム・スタンプ』
- 311 ページの『lock_wait_time ロック待機中の時間』

lock_wait_time ロック待機中の時間

エレメント ID lock_wait_time

エレメント・タイプ カウンター

表 395. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	ロック
アプリケーション	appl	ロック
ロック	appl_lock_list	appl_lock_list

スナップショット・モニターの場合、このカウンターはリセットできます。

表 396. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
トランザクション	event_xact	-

説明 ロック待機の合計経過時間。経過時間はミリ秒単位で示されます。

使用法 データベース・レベルでは、このデータベース内ですべてのアプリケーションが 1 つのロックを待機した合計経過時間を示します。

アプリケーション接続およびトランザクションのレベルでは、この接続またはトランザクションがロックの付与を待機した合計経過時間を示します。

このエレメントと *lock_waits* モニター・エレメントを組み合わせて使用すると、平均ロック待機時間を計算できます。この計算は、データベース・レベルとアプリケーション接続レベルのいずれでも行えます。

経過時間を示すモニター・エレメントを使用するときは、次のことを考慮してください。

- 経過時間は、システム負荷の影響を受けるので、実行する処理数が多くなると、この経過時間の値は大きくなる。
- このエレメントをデータベース・レベルで計算する場合、データベース・システム・モニターはアプリケーション・レベルの時間を合計する。この場合、同時に複数のアプリケーション処理が実行されていることがあるので、データベース・レベルでは時間が二重に計算されます。

  意味のあるデータを提供するためには、上記の説明に従って平均ロック待機時間を計算してください。

関連資料:

- 310 ページの『lock_waits ロック待機数』
- 312 ページの『locks_waiting ロックで待機中の現行エージェント』

locks_waiting ロックで待機中の現行エージェント

エレメント ID locks_waiting

エレメント・タイプ ゲージ

表 397. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本
ロック	db_lock_list	基本

説明 ロック待機中のエージェントの数を示します。

使用法 このエレメントと *appls_cur_cons* と組み合わせて使用すると、ロックを待機中のアプリケーションのパーセンテージが分かります。この値が大きい場

合は、アプリケーションに並行性の問題がある可能性があるため、ロックや排他ロックを長時間にわたって保留しているアプリケーションを確認する必要があります。

関連資料:

• 186 ページの『appls_cur_cons 現在接続されているアプリケーション』

uow_lock_wait_time ロック待機中の作業単位の合計時間

エレメント ID uow_lock_wait_time

エレメント・タイプ カウンター

表 398. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	作業単位

説明 この作業単位がロックの待機に要した合計経過時間。

使用法 このエレメントは、リソース競合問題の重大度を判別するときに利用できます。

lock_wait_start_time ロック待機開始タイム・スタンプ

エレメント ID lock_wait_start_time

エレメント・タイプ タイム・スタンプ

表 399. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック、タイム・スタンプ
ロック	lock_wait	ロック、タイム・スタンプ

表 400. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	タイム・スタンプ
デッドロックの詳細	event_detailed_dlconn	タイム・スタンプ

説明 現在、別のアプリケーションによってロックされているオブジェクトに対するロックを取得するために、このアプリケーションが待機を開始した日時。

使用法 このエレメントは、リソース競合の重大度を判別するときに役立ちます。

関連資料:

• 314 ページの『agent_id_holding_lock ロックを保持しているエージェント ID』

lock_timeout_val ロック・タイムアウト

エレメント ID lock_timeout_val

エレメント・タイプ 情報

表 401. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本
アプリケーション	agent	基本

説明 アプリケーションが SET CURRENT LOCK TIMEOUT ステートメントを発行した時点のタイムアウト値 (秒単位) を示します。ステートメントが実行されていない場合は、データベース・レベルのロック・タイムアウトが示されます。

使用法 SET CURRENT LOCK TIMEOUT ステートメントを使用して、アプリケーション・エージェントが表または索引のロックを待機する最大期間を指定できます。

アプリケーションがロックを待つ時間が長すぎる場合は、アプリケーション中で *lock_timeout_val* 値を検査して、設定値が大きすぎるかどうか調べることができます。アプリケーション・ロジックにとって適切な場合は、アプリケーションに変更を加え、 *lock_timeout_val* の値を小さくして、アプリケーションをタイムアウトさせることができます。 SET CURRENT LOCK TIMEOUT ステートメントを使用して、この変更を行えます。

アプリケーションが頻繁にタイムアウトする場合は、 *lock_timeout_val* の設定値が小さすぎるかどうかを検査し、該当する場合は大きくすることができます。

agent_id_holding_lock ロックを保持しているエージェント ID

エレメント ID agent_id_holding_lock

エレメント・タイプ 情報

表 402. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック
ロック	appl_lock_list	ロック
ロック	lock_wait	ロック

説明 このアプリケーションが待機しているロックを保持してしているエージェントのアプリケーション・ハンドル。この情報を取得するには、ロック・モニター・グループをオンにする必要があります。

使用法 このエレメントは、リソースの競合状態にあるアプリケーションの判別に利用できます。

このエレメントが 0 (ゼロ) で、アプリケーションがロックを待機中の場合は、ロックが未確定トランザクションによって保持されていることを示します。 appl_id_holding_lk または コマンド行プロセッサー LIST INDOUBT TRANSACTIONS コマンドのいずれかを使用して (未確定となったときにトランザクションを処理していた CICS エージェントのアプリケーション ID を表示します)、未確定トランザクションを識別してから、コミットまたはロールバックを行います。

このアプリケーションが待機している 1 つのオブジェクトに対して、複数のアプリケーションが共有ロックを保有していることがあるので注意してください。アプリケーションが保留しているロックのタイプについては、『lock_mode』を参照してください。アプリケーションのスナップショットをとる場合は、オブジェクトのロックを保留しているエージェント ID のうち戻されるのは 1 つだけです。ロックのスナップショットを取れば、オブジェクトのロックを保留しているすべてのエージェント ID を識別できます。

関連資料:

- 313 ページの『lock_wait_start_time ロック待機開始タイム・スタンプ』
- 315 ページの『appl_id_holding_lk ロックを保持しているアプリケーション ID』

appl_id_holding_lk ロックを保持しているアプリケーション ID

エレメント ID appl_id_holding_lk

エレメント・タイプ 情報

表 403. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック
ロック	appl_lock_list	ロック
ロック	lock_wait	ロック

表 404. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 このアプリケーションが取得のために待機しているオブジェクトのロックを保留しているアプリケーションのアプリケーション ID。

使用法 このエレメントは、リソースの競合状態にあるアプリケーションの判別に利用できます。具体的には、ロックを保留しているアプリケーション・ハンドル (エージェント ID) と表 ID を識別するときに利用します。 LIST APPLICATIONS コマンドを使用してアプリケーション ID をエージェントに関連付ける情報を取得することもできます。ただし、このタイプの情報はスナップショットをとる際に収集する方が賢明です。 LIST APPLICATIONS コマンドを実行する前にアプリケーションが終了してしまうと情報を得られなくなることがあるからです。

このアプリケーションがロックの取得を待機している 1 つのオブジェクトに対して、複数のアプリケーションが共有ロックを保留していることがあるので注意してください。アプリケーションが保留しているロックのタイプについては、『lock_mode』を参照してください。アプリケーションのスナップショットをとる場合は、オブジェクトに対してロックを保留しているアプリケーション ID のうち戻されるのは 1 つだけです。ロックのスナップショットをとる場合は、オブジェクトに対してロックを保留しているすべてのアプリケーション ID が戻されます。

関連資料:

- 295 ページの『deadlocks デッドロック検出数』
- 314 ページの『agent_id_holding_lock ロックを保持しているエージェント ID』

sequence_no_holding_lk ロックを保持しているシーケンス番号

エレメント ID sequence_no_holding_lk

エレメント・タイプ 情報

表 405. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本
ロック	appl_lock_list	基本

表 406. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 このアプリケーションが取得のために待機しているオブジェクトのロックを保留しているアプリケーションのシーケンス番号。

使用法 この ID と appl_id を組み合わせて使用すると、このアプリケーションが取得しようと待機しているオブジェクトのロックを保留しているトランザクションを一意的に識別できます。

rolled_back_appl_id ロールバック・アプリケーション

エレメント ID rolled_back_appl_id

エレメント・タイプ 情報

表 407. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-

説明 デッドロックが発生したときにロールバックされたアプリケーション ID。

使用法 システム管理者はこの情報を基にして、更新を完了できなかったアプリケーションや再始動する必要があるアプリケーションを判別できます。

関連資料:

- 190 ページの『coord_agents_top コーディネーター・エージェント最大数』

rolled_back_agent_id ロールバックされたエージェント

エレメント ID rolled_back_agent_id

エレメント・タイプ 情報

表 408. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-

説明 デッドロックが発生したときにロールバックされたエージェント。

使用法 システム管理者はこの情報を基にして、更新を完了できなかったアプリケーションや再始動する必要があるアプリケーションを判別できます。

関連資料:

- 190 ページの『coord_agents_top コーディネーター・エージェント最大数』

rolled_back_sequence_no ロールバックされたシーケンス番号

エレメント ID rolled_back_sequence_no

エレメント・タイプ 情報

表 409. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_deadlock	-

説明 デッドロックが発生したときにロールバックされたアプリケーションのシーケンス番号。

使用法 システム管理者はこの情報を基にして、更新を完了できなかったアプリケーションや再始動する必要があるアプリケーションを判別できます。

ロールフォワードのモニター

ロールフォワード・モニターに関するモニター・エレメント

データベースの変更内容をリカバリーする処理には時間がかかります。データベース・システム・モニターを使用すると、リカバリーの進行状況をモニターできます。次のエレメントにより、ロールフォワード状況に関する情報が提供されます。

- rf_timestamp ロールフォワード・タイム・スタンプ : モニター・エレメント
- ts_name ロールフォワードされている表スペース : モニター・エレメント
- rf_type ロールフォワード・タイプ : モニター・エレメント
- rf_log_num ロールフォワードされているログ : モニター・エレメント
- rf_status ログ・フェーズ : モニター・エレメント

rf_timestamp ロールフォワード・タイム・スタンプ

エレメント ID rf_timestamp

エレメント・タイプ タイム・スタンプ

表 410. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	rollforward	タイム・スタンプ

説明 処理中のログのタイム・スタンプ。

使用法 ロールフォワードが進行中の場合、これは処理中のログ・レコードのタイム・スタンプです。これは、リカバリーされるデータ変更の標識です。

関連資料:

• 318 ページの『ts_name ロールフォワードされている表スペース』

ts_name ロールフォワードされている表スペース

エレメント ID ts_name

エレメント・タイプ 情報

表 411. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	rollforward	基本

説明 現在ロールフォワードされている表スペースの名前。

使用法 ロールフォワードが進行中の場合は、このエレメントは関係する表スペースを示します。

関連資料:

• 317 ページの『rf_timestamp ロールフォワード・タイム・スタンプ』

rf_type ロールフォワード・タイプ

エレメント ID rf_type

エレメント・タイプ 情報

表 412. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	rollforward	基本

説明 進行中のロールフォワードのタイプ。

使用法 データベース・レベルと表スペース・レベルのどちらでリカバリーが起きているかを示す標識です。

rf_log_num ロールフォワードされているログ

エレメント ID rf_log_num

エレメント・タイプ 情報

表 413. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	rollforward	基本

説明 処理中のログ。

使用法 ロールフォワードが進行中の場合、このエレメントは関係するログを示します。

rf_status ログ・フェーズ

エレメント ID rf_status

エレメント・タイプ 情報

表 414. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	rollforward	基本

説明 リカバリーの状況。

使用法 このエレメントはリカバリーの進行状況を示します。リカバリーが取り消し (ロールバック) 段階にあるのか、あるいは再実行 (ロールフォワード) フェーズにあるのかを示します。

表スペース・アクティビティー

表スペース・アクティビティーに関するモニター・エレメント

次のエレメントにより、表スペースに関する情報が提供されます。

- tablespace_id 表スペース ID : モニター・エレメント
- tablespace_name 表スペース名 : モニター・エレメント
- tablespace_type 表スペース・タイプ : モニター・エレメント
- tablespace_content_type 表スペースのコンテンツ・タイプ : モニター・エレメント
- tablespace_state 表スペースの状態 : モニター・エレメント
- tablespace_page_size 表スペースのページ・サイズ : モニター・エレメント
- tablespace_extent_size 表スペースのエクステント・サイズ : モニター・エレメント
- tablespace_prefetch_size 表スペースのプリフェッチ・サイズ : モニター・エレメント
- tablespace_cur_pool_id 現在使用中のバッファー・プール : モニター・エレメント
- tablespace_next_pool_id 次の始動時に使用されるバッファー・プール : モニター・エレメント
- tablespace_total_pages 表スペース内の合計ページ数 : モニター・エレメント
- tablespace_usable_pages 表スペース内の使用可能ページ数 : モニター・エレメント
- tablespace_used_pages 表スペース内の使用されているページ数 : モニター・エレメント
- tablespace_free_pages 表スペース内のフリー・ページ数 : モニター・エレメント
- tablespace_pending_free_pages 表スペース内のペンディング・フリー・ページ数 : モニター・エレメント
- tablespace_page_top 表スペース最高水準点 : モニター・エレメント
- tablespace_rebalancer_mode 再平衡化モード : モニター・エレメント
- tablespace_rebalancer_start_time 再平衡化開始時刻 : モニター・エレメント
- tablespace_rebalancer_restart_time 再平衡化再始動時刻 : モニター・エレメント

- tablespace_rebalancer_extents_remaining 再平衡化で処理されるエクステントの合計数
- tablespace_rebalancer_extents_processed 再平衡化で処理されたエクステントの数 : モニター・エレメント
- tablespace_rebalancer_last_extent_moved 再平衡化によって最後に移動されたエクステント : モニター・エレメント
- tablespace_rebalancer_priority 現行の再平衡化優先順位 : モニター・エレメント
- tablespace_num_quiescers - 静止プログラム数 : モニター・エレメント
- fs_caching ファイル・システム・キャッシング : モニター・エレメント
- 表スペース静止プログラム・アクティビティーに関するモニター・エレメント
- tablespace_state_change_object_id 状態変更オブジェクト ID : モニター・エレメント
- tablespace_state_change_ts_id 状態変更表スペース ID : モニター・エレメント
- tablespace_min_recovery_time ロールフォワードの最小リカバリー時間 : モニター・エレメント
- tablespace_num_containers 表スペース内のコンテナー数 : モニター・エレメント
- コンテナー状況に関するモニター・エレメント
- tablespace_num_ranges 表スペース・マップ内の範囲数 : モニター・エレメント
- 表スペース範囲状況に関するモニター・エレメント

tablespace_id 表スペース ID

エレメント ID tablespace_id

エレメント・タイプ 情報

表 415. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

表 416. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 現行データベースが使用する表スペースを一意的に示す整数。

使用法 このエレメントの値は、SYSCAT.TABLESPACES のビューの TBSPACEID 列の値と一致します。

tablespace_name 表スペース名

エレメント ID tablespace_name

エレメント・タイプ 情報

表 417. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

表 417. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
ロック	appl_lock_list	基本
ロック	lock	ロック
ロック	lock_wait	ロック

表 418. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-
表スペース	tablespace_list	-

説明 表スペースの名前。

使用法 このエレメントは、リソースの競合の原因を判別するときに役立ちます。

これはデータベース・カタログ表の SYSCAT.TABLESPACES にある TBSPACE 列と同じです。アプリケーション・レベル、アプリケーション・ロック・レベル、およびデッドロック・モニター・レベルでは、アプリケーションがロックを待機している表スペースの名前です。ほかのアプリケーションがこの表スペースのロックを保留しています。

ロック・レベルでは、アプリケーションがロックを保留している表スペースの名前です。

表スペース・レベルでは (バッファー・プール・モニター・グループが ON の場合)、情報が戻される表スペースの名前です。

関連資料:

• 300 ページの『lock_object_type 待機中のロック対象タイプ』

tablespace_type 表スペース・タイプ

エレメント ID tablespace_type

エレメント・タイプ 情報

表 419. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

説明 表スペースのタイプ。

使用法 このエレメントは、この表スペースがデータベース管理の表スペース (DMS) か、システム管理の表スペース (SMS) かを示します。

tablespace_type の値 (sqlmon.h 内に定義) は次のとおりです。

- DMS の場合: SQLM_TABLESPACE_TYP_DMS
- SMS の場合: SQLM_TABLESPACE_TYP_SMS

tablespace_content_type 表スペースのコンテンツ・タイプ

エレメント ID tablespace_content_type

エレメント・タイプ 情報

表 420. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

説明 表スペース内のコンテンツのタイプ。

使用法 表スペース内のコンテンツのタイプ (sqlmon.h 内に定義) は、次のいずれかになります。

- 任意のデータ: SQLM_TABLESPACE_CONTENT_ANY
- 長いデータ: SQLM_TABLESPACE_CONTENT_LONG
- システム一時データ: SQLM_TABLESPACE_CONTENT_SYSTEMP
- ユーザー一時データ: SQLM_TABLESPACE_CONTENT_USRTEMP

tablespace_state 表スペースの状態

エレメント ID tablespace_state

エレメント・タイプ 情報

表 421. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 このエレメントは、表スペースの現在の状態を示します。

使用法 このエレメントには、表スペースの現在の状態を示す 16 進値が含まれています。外部から見ることができる表スペースの状態は、特定の状態値が 16 進数の合計値で構成されています。例えば、状態が「静止: EXCLUSIVE」および「ロード・ペンディング」の場合、値は 0x0004 + 0x0008 で 0x000c になります。「db2tbst - Get Tablespace State」を使用すると、特定の 16 進値に関連付けられた表スペースを取得できます。

表 422. sqlutil.h 中にリストされているビット定義

16 進値	10 進値	状態
0x0	0	標準 (sqlutil.h 内の SQLB_NORMAL の定義を参照)
0x1	1	静止モードでの共有
0x2	2	静止モードでの更新
0x4	4	静止モードでの排他
0x8	8	ロード・ペンディング
0x10	16	削除ペンディング
0x20	32	バックアップ・ペンディング
0x40	64	ロールフォワード進行中
0x80	128	ロールフォワード・ペンディング

表 422. sqlutil.h 中にリストされているビット定義 (続き)

16 進値	10 進値	状態
0x100	256	リストア・ペンディング
0x100	256	リカバリー・ペンディング (未使用)
0x200	512	使用不可ペンディング
0x400	1024	REORG 進行中
0x800	2048	バックアップ進行中
0x1000	4096	ストレージを定義する必要がある
0x2000	8192	リストア進行中
0x4000	16384	オフラインのためアクセス不可
0x8000	32768	ドロップ・ペンディング
0x2000000	33554432	ストレージを定義可能
0x4000000	67108864	ストレージ定義は最終状態
0x8000000	134217728	ストレージ定義はロールフォワードの前に変更された
0x10000000	268435456	DMS 再平衡化進行中
0x20000000	536870912	表スペース削除進行中
0x40000000	1073741824	表スペース作成進行中

tablespace_page_size 表スペースのページ・サイズ

エレメント ID tablespace_page_size

エレメント・タイプ 情報

表 423. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

説明 表スペースが使用するページ・サイズ (バイト単位)。

tablespace_extent_size 表スペースのエクステント・サイズ

エレメント ID tablespace_extent_size

エレメント・タイプ 情報

表 424. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

説明 表スペースが使用するエクステント・サイズ。

tablespace_prefetch_size 表スペースのプリフェッチ・サイズ

エレメント ID tablespace_prefetch_size

エレメント・タイプ 情報

表 425. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本
表スペース	tablespace_nodeinfo	基本

説明 プリフェッチャーがディスクから 1 回に取得できる最大ページ数。

自動プリフェッチ・サイズが使用可能な場合は、このエレメントは値「-1」を *tablespace* 論理データ・グループ中に報告し、実際の値を *tablespace_nodeinfo* 論理データ・グループ中に報告します。

自動プリフェッチ・サイズが使用可能でない場合は、このエレメントは実際の値を *tablespace* 論理データ・グループ中に報告し、 *tablespace_nodeinfo* 論理データ・グループ中には表示されません。

tablespace_cur_pool_id 現在使用中のバッファー・プール

エレメント ID tablespace_cur_pool_id

エレメント・タイプ 情報

表 426. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

説明 表スペースが現在使用中のバッファー・プールのバッファー・プール ID。

使用法 バッファー・プールは、それぞれユニークな整数で識別できます。このエレメントの値は、SYSCAT.BUFFERPOOLS ビューの BUFFERPOOLID 列の値と一致します。

tablespace_next_pool_id 次の始動時に使用されるバッファー・プール

エレメント ID tablespace_next_pool_id

エレメント・タイプ 情報

表 427. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

説明 データベースを次に始動したときに表スペースが使用するバッファー・プールのバッファー・プール ID。

使用法 バッファー・プールは、それぞれユニークな整数で識別できます。このエレメントの値は、SYSCAT.BUFFERPOOLS ビューの BUFFERPOOLID 列の値と一致します。

tablespace_total_pages 表スペース内の合計ページ数

エレメント ID tablespace_total_pages

エレメント・タイプ 情報

表 428. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本 (DMS 表スペース) バッファー・プール (SMS 表スペース)

説明 1 つの表スペース内の合計ページ数。

使用法 1 つの表スペースが使用するオペレーティング・システムの合計スペースです。 DMS の場合は、コンテナー・サイズの合計となります (オーバーヘッドを含む)。 SMS の場合は、この表スペースに保管される表に使用されるすべてのファイル・スペースの合計です (この情報は、バッファー・プール・スイッチが ON の場合にのみ収集されます)。

tablespace_usable_pages 表スペース内の使用可能ページ数

エレメント ID tablespace_usable_pages

エレメント・タイプ 情報

表 429. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本 (DMS 表スペース) バッファー・プール (SMS 表スペース)

説明 表スペース内の合計ページ数からオーバーヘッド・ページ数を引いた数値。

使用法 このエレメントが適用されるのは DMS 表スペースだけです。 SMS の場合は、このエレメントには tablespace_total_pages と同じ値が含まれます。

表スペースの再平衡化中に、使用可能ページ数に新たに追加されたコンテナーが加えられますが、これらの新しいページ数は、再平衡化完了までの間フリー・ページ数には反映されません。表スペースの再平衡化が実行されていない場合は、使用済みページ数、フリー・ページ数、およびペンディング・フリー・ページ数の合計が使用可能ページ数に等しくなります。

tablespace_used_pages 表スペース内の使用されているページ数

エレメント ID tablespace_used_pages

エレメント・タイプ 情報

表 430. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本 (DMS 表スペース) バッファー・プール (SMS 表スペース)

説明 表スペース内で現在使用中 (フリーではない) の合計ページ数。

使用法 DMS 表スペースで使用中の合計ページ数です。 SMS 表スペースの場合は、tablespace_total_pages と同じ値になります。

tablespace_free_pages 表スペース内のフリー・ページ数

エレメント ID tablespace_free_pages

エレメント・タイプ 情報

表 431. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 1 つの表スペース内で現在フリーの合計ページ数。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

tablespace_pending_free_pages 表スペース内のペンディング・フリー・ページ数

エレメント ID tablespace_pending_free_pages

エレメント・タイプ 情報

表 432. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 すべてのペンディング・トランザクションがコミットまたはロールバックされ、オブジェクトのための新しいスペースが要求されたときにフリーになる表スペースのページ数。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

tablespace_page_top 表スペース最高水準点

エレメント ID tablespace_page_top

エレメント・タイプ 情報

表 433. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 最高水準点を保持している表スペース内のページ。

使用法 DMS の場合、このエレメントは、表スペースで最後に割り振られたエクステントの次にある最初のフリー・エクステントのページ番号を示します。この値は減少するので、実際の「最高水準点」ではなく「現在の水準点」であることに注意してください。この情報は SMS には適用できません。

tablespace_rebalancer_mode 再平衡化モード

エレメント ID tablespace_rebalancer_mode

エレメント・タイプ 情報

表 434. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 フォワードまたはリバースの再平衡化が行われているかどうかを示す整数。次の値があります (sqlmon.h 内に定義)。

- 再平衡化は行われていない: SQLM_TABLESPACE_NO_REBAL
- フォワード: SQLM_TABLESPACE_FWD_REBAL
- リバース: SQLM_TABLESPACE_REV_REBAL

使用法 現在の再平衡化処理で表スペースからスペースを除去しているのかまたは追加しているのかを示す標識として使用できます。これは、DMS 表スペースにのみ適用できます。

tablespace_rebalancer_start_time 再平衡化開始時刻

エレメント ID tablespace_rebalancer_start_time

エレメント・タイプ 情報

表 435. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 再平衡化を最初に開始した時刻を示すタイム・スタンプ。

使用法 再平衡化を最初に開始した時刻を知るのに使用します。これは、再平衡化の処理速度を測定したり、再平衡化の終了時刻を推定するときに使用できます。これは、DMS 表スペースにのみ適用できます。

tablespace_rebalancer_restart_time 再平衡化再始動時刻

エレメント ID tablespace_rebalancer_restart_time

エレメント・タイプ 情報

表 436. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 再平衡化が休止または停止後に再始動されたときを示すタイム・スタンプ。

使用法 再平衡化の完了レベルを示す標識として使用できます。再平衡化が再始動されたときを示し、再平衡化の速度を導出できるので、推定完了時刻を得られます。これは、DMS 表スペースにのみ適用できます。

tablespace_rebalancer_extents_remaining 再平衡化で処理されるエクステントの合計数

エレメント ID tablespace_rebalancer_extents_remaining

エレメント・タイプ 情報

表 437. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 移動するエクステントの数。この値は、再平衡化の開始時刻または再始動時刻 (どちらか後に実行された方) の時点で計算されます。

使用法 再平衡化の完了レベルを示す標識として使用できます。このエレメントの内容が時間とともに変化する様子を追跡すると、再平衡化の進行状況をモニターできます。再平衡化が終了したかどうかを確認するには、tablespace_state を使用します。これは、DMS 表スペースにのみ適用できます。

tablespace_rebalancer_extents_processed 再平衡化で処理されたエクステントの数

エレメント ID tablespace_rebalancer_extents_processed

エレメント・タイプ 情報

表 438. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 再平衡化の開始後または再始動後 (どちらか後で実行された方) に、再平衡化ですでに移動されたエクステントの数。

使用法 再平衡化の完了レベルを示す標識として使用できます。このエレメントの内容が時間とともに変化する様子を追跡すると、再平衡化の進行状況をモニターできます。 tablespace_state と rebalance_mode を使用すると、再平衡化が完了したかどうかチェックできます。これは、DMS 表スペースにのみ適用できます。

tablespace_rebalancer_last_extent_moved 再平衡化によって最後に移動されたエクステント

エレメント ID tablespace_rebalancer_last_extent_moved

エレメント・タイプ 情報

表 439. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 再平衡化によって最後に移動されたエクステント。

使用法 再平衡化の完了レベルを示す標識として使用できます。このエレメントの内容が時間とともに変化する様子を追跡すると、再平衡化の進行状況をモニターできます。 tablespace_state と rebalance_mode を使用すると、再平衡化が完了したかどうかチェックできます。これは、DMS 表スペースにのみ適用できます。

tablespace_rebalancer_priority 現行の再平衡化優先順位

エレメント ID tablespace_rebalancer_priority

エレメント・タイプ 情報

表 440. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 データベース内で実行されている再平衡化の優先順位。

使用法 これは、DMS 表スペースにのみ適用できます。

tablespace_num_quiescers - 静止プログラム数

エレメント ID tablespace_num_quiescers

エレメント・タイプ 情報

表 441. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 表スペースを静止させているユーザーの数 (0 から 5 の範囲)。

使用法 この値は表スペースを静止させたエージェントの数を示します (「共有」、「更新」、または「排他」モード)。それぞれの静止プログラムについて、次の情報が tablespace_quiescer 論理データ・グループに戻されます。

- 静止プログラムのユーザー許可 ID
- 静止プログラムのエージェント ID
- この表スペースが静止することになった、静止されたオブジェクトの表スペース ID
- この表スペースが静止することになった、静止されたオブジェクトのオブジェクト ID
- 静止状態

表スペース静止プログラム・アクティビティーに関するモニター・エレメント

表スペース静止プログラム・アクティビティーに関するモニター・エレメント: 次のエレメントにより、表スペースの静止プログラム・アクティビティーに関する情報が提供されます。

- quiescer_auth_id 静止プログラム・ユーザー許可 ID : モニター・エレメント
- quiescer_agent_id 静止プログラム・エージェント ID : モニター・エレメント
- quiescer_ts_id 静止プログラム表スペース ID : モニター・エレメント
- quiescer_obj_id 静止プログラム・オブジェクト ID : モニター・エレメント
- quiescer_state 静止プログラムの状態 : モニター・エレメント

quiescer_auth_id 静止プログラム・ユーザー許可 ID:

エレメント ID quiescer_auth_id

エレメント・タイプ 情報

表 442. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_quiescer	基本

説明 静止状態を保持するユーザーの許可 ID。

使用法 このエレメントを使用すると、表スペースを静止させたユーザーを判別できます。

quiescer_agent_id 静止プログラム・エージェント ID:

エレメント ID quiescer_agent_id

エレメント・タイプ 情報

表 443. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_quiescer	基本

説明 静止状態を保持するエージェントのエージェント ID。

使用法 このエレメントと quiescer_auth_id を組み合わせて使用すると、表スペースを静止させたユーザーを判別できます。

quiescer_ts_id 静止プログラム表スペース ID:

エレメント ID quiescer_ts_id

エレメント・タイプ 情報

表 444. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_quiescer	基本

説明 表スペースを静止させたオブジェクトの表スペース ID。

使用法 このエレメントと quiescer_obj_id および quiescer_auth_id を組み合わせて使用すると、表スペースを静止させたユーザーを判別できます。このエレメントの値は、SYSCAT.TABLES ビューの TBSPACEID 列の値と一致します。

quiescer_obj_id 静止プログラム・オブジェクト ID:

エレメント ID quiescer_obj_id

エレメント・タイプ 情報

表 445. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_quiescer	基本

説明 表スペースを静止させたオブジェクトのオブジェクト ID。

使用法 このエレメントと quiescer_ts_id および quiescer_auth_id を組み合わせて使用すると、表スペースを静止させたオブジェクトを判別できます。このエレメントの値は、SYSCAT.TABLES ビューの TABLEID 列の値と一致します。

quiescer_state 静止プログラムの状態:

エレメント ID quiescer_state

エレメント・タイプ 情報

表 446. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_quiescer	基本

説明 行われている静止タイプ (例えば、「共有」、「更新する」、または「排他」)。

使用法 このエレメントの値は、sqlutil.h の SQLB_QUIESCED_SHARE、SQLB_QUIESCED_UPDATE、または SQLB_QUIESCED_EXCLUSIVE の定数の値と一致します。

tablespace_state_change_object_id 状態変更オブジェクト ID

エレメント ID tablespace_state_change_object_id

エレメント・タイプ 情報

表 447. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 表スペースの状態を「ロード・ペンディング」または「削除ペンディング」に設定したオブジェクト。

使用法 このエレメントに意味があるのは、表スペースの状態が「ロード・ペンディング」または「削除ペンディング」の場合だけです。このエレメントの値がゼロ以外の場合は、SYSCAT.TABLES ビューの TABLEID 列の値と一致します。

tablespace_state_change_ts_id 状態変更表スペース ID

エレメント ID tablespace_state_change_ts_id

エレメント・タイプ 情報

表 448. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 表スペースの状態が「ロード・ペンディング」または「削除ペンディング」の場合は、表スペースの状態の設定を引き起こしたオブジェクトの表スペース ID が示されます。

使用法 このエレメントに意味があるのは、表スペースの状態が「ロード・ペンディ

ング」または「削除ペンディング」の場合だけです。このエレメントの値がゼロ以外の場合は、 SYSCAT.TABLES ビューの TABLESPACEID 列の値と一致します。

tablespace_min_recovery_time ロールフォワードの最小リカバリー時間

エレメント ID tablespace_min_recovery_time

エレメント・タイプ 情報

表 449. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 表スペースをロールフォワードできる最も早い時点を示すタイム・スタンプ。

使用法 ゼロ以外のときだけ表示されます。

tablespace_num_containers 表スペース内のコンテナー数

エレメント ID tablespace_num_containers

エレメント・タイプ 情報

表 450. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 表スペース内のコンテナーの合計数。

コンテナー状況

コンテナー状況に関するモニター・エレメント: 次のエレメントにより、コンテナー状況に関する情報が提供されます。

- container_id コンテナー ID : モニター・エレメント
- container_name コンテナー名 : モニター・エレメント
- container_type コンテナー・タイプ : モニター・エレメント
- container_total_pages コンテナー内の合計ページ数 : モニター・エレメント
- container_usable_pages コンテナー内の使用可能なページ数 : モニター・エレメント
- container_stripe_set ストライプ・セット : モニター・エレメント
- container_accessible コンテナーのアクセス可能性 : モニター・エレメント

container_id コンテナー ID:

エレメント ID container_id

エレメント・タイプ 情報

表 451. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本

説明 表スペース内のコンテナーを一意的に定義する整数。

使用法 このエレメントと container_name、 container_type、 container_total_pages、 container_usable_pages、 container_stripe_set、および container_accessible の各エレメントを組み合わせて使用すると、コンテナーを記述できます。

container_name コンテナー名:

エレメント ID container_name

エレメント・タイプ 情報

表 452. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本

説明 コンテナーの名前。

使用法 このエレメントと container_id、 container_type、 container_total_pages、 container_usable_pages、 container_stripe_set、および container_accessible の各エレメントを組み合わせて使用すると、コンテナーを記述できます。

container_type コンテナー・タイプ:

エレメント ID container_type

エレメント・タイプ 情報

表 453. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本

説明 コンテナーのタイプ。

使用法 このエレメントはコンテナーのタイプを戻します。コンテナーのタイプは、ディレクトリー・パス (SMS の場合のみ)、ファイル (DMS の場合)、ロー・デバイス (DMS の場合) のいずれかです。このエレメントと container_id、 container_name、 container_total_pages、 container_usable_pages、 container_stripe_set、および container_accessible の各エレメントを組み合わせて使用すると、コンテナーを記述できます。

sqlutil.h には次の値が定義されています。

- ディレクトリー・パス (SMS): SQLB_CONT_PATH
- ロー・デバイス (DMS): SQLB_CONT_DISK
- ファイル (DMS): SQLB_CONT_FILE
- ストライピングされたディスク (DMS): SQLB_CONT_STRIPED_DISK
- ストライピングされたファイル (DMS): SQLB_CONT_STRIPED_FILE

container_total_pages コンテナー内の合計ページ数:

エレメント ID container_total_pages

エレメント・タイプ 情報

表 454. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本 (DMS 表スペース) バッファー・プール (SMS 表スペース)

説明 コンテナーが占有するページ数の合計。

使用法 このエレメントと container_id、 container_name、 container_type、 container_usable_pages、 container_stripe_set、および container_accessible の各エレメントを組み合わせて使用すると、コンテナーを記述できます。

container_usable_pages コンテナー内の使用可能なページ数:

エレメント ID container_usable_pages

エレメント・タイプ 情報

表 455. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本 (DMS 表スペース) バッファー・プール (SMS 表スペース)

説明 コンテナー内の使用可能なページ数の合計。

使用法 このエレメントと container_id、 container_name、 container_type、 container_total_pages、 container_stripe_set、および container_accessible の各エレメントを組み合わせて使用すると、コンテナーを記述できます。 SMS 表スペースの場合、この値は container_total_pages と同じになります。

container_stripe_set ストライプ・セット:

エレメント ID container_stripe_set

エレメント・タイプ 情報

表 456. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本

説明 コンテナーが属するストライプ・セット。

使用法 このエレメントと container_id、 container_name、 container_type、 container_total_pages、 container_usable_pages、および container_accessible の各エレメントを組み合わせて使用すると、コンテナーを記述できます。これは、DMS 表スペースにのみ適用できます。

container_accessible コンテナーのアクセス可能性:

エレメント ID container_accessible

エレメント・タイプ 情報

表 457. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_container	基本

説明 このエレメントは、コンテナーがアクセス可能かどうかを示します (1 が可能、 0 が不可を意味します)。

使用法 このエレメントと container_id、 container_name、 container_type、 container_total_pages、 container_usable_pages、および container_stripe_set の各エレメントを組み合わせて使用すると、コンテナーを記述できます。

tablespace_num_ranges 表スペース・マップ内の範囲数

エレメント ID tablespace_num_ranges

エレメント・タイプ 情報

表 458. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_nodeinfo	基本

説明 表スペース・マップ内の範囲 (項目) の数。この範囲は 1 から 100 です (ただし通常は 12 未満)。表スペース・マップがあるのは、DMS 表スペースの場合だけです。

表スペース範囲状況

表スペース範囲状況に関するモニター・エレメント: 表スペース・マップは、論理表スペースのページ番号を物理ディスクのロケーションにマップするときに使用します。このマップは、一連の範囲で構成されます。

例えば、次のような範囲があります。

ストライプ	範囲	MaxPage	MaxExtent	StartStripe	EndStripe	Adj	# Conts	Containers
0	[0]	249	124	0	124	0	1	(0)
1	[1]	999	499	125	249	0	3	(0,1,2)
2	[2]	1499	749	250	374	0	1	(1,2)

それぞれの範囲について、スナップショットで次の情報が戻されます (テンプレート参照)。

- range_stripe_set_number ストライプ・セット番号 : モニター・エレメント
- range_number 範囲番号 : モニター・エレメント
- range_max_page_number 範囲内の最大ページ : モニター・エレメント
- range_max_extent 範囲内の最大エクステント : モニター・エレメント
- range_start_stripe 開始ストライプ : モニター・エレメント
- range_end_stripe 終了ストライプ : モニター・エレメント

- range_adjustment 範囲調整 : モニター・エレメント
- range_num_containers 範囲内コンテナーの数 : モニター・エレメント
- コンテナー配列 (範囲に属するコンテナーをリストします。この配列のサイズは、表スペース内のコンテナー合計数によって決まります)。
  - range_container_id 範囲コンテナー : モニター・エレメント
  - range_offset 範囲オフセット : モニター・エレメント

range_stripe_set_number ストライプ・セット番号:

エレメント ID range_stripe_set_number

エレメント・タイプ 情報

表 459. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、中に 1 つ範囲が含まれているストライプ・セットを示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_number 範囲番号:

エレメント ID range_number

エレメント・タイプ 情報

表 460. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、表スペース・マップ内にある 1 つの範囲の番号を示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_max_page_number 範囲内の最大ページ:

エレメント ID range_max_page_number

エレメント・タイプ 情報

表 461. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、1 つの範囲がマップする最大ページ番号を示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_max_extent 範囲内の最大エクステント:

エレメント ID range_max_extent

エレメント・タイプ 情報

表 462. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、1 つの範囲がマップする最大エクステント番号を示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_start_stripe 開始ストライプ:

エレメント ID range_start_stripe

エレメント・タイプ 情報

表 463. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、1 つの範囲内の最初のストライプの番号を示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_end_stripe 終了ストライプ:

エレメント ID range_end_stripe

エレメント・タイプ 情報

表 464. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、1 つの範囲内の最後のストライプの番号を示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_adjustment 範囲調整:

エレメント ID range_adjustment

エレメント・タイプ 情報

表 465. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、1 つの範囲が実際に開始するコンテナー配列のオフセットを示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_num_containers 範囲内コンテナーの数:

エレメント ID range_num_containers

エレメント・タイプ 情報

表 466. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 この値は、現行範囲のコンテナー数を示します。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_container_id 範囲コンテナー:

エレメント ID range_container_id

エレメント・タイプ 情報

表 467. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 範囲内でコンテナーを一意的に定義する整数。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

range_offset 範囲オフセット:

エレメント ID range_offset

エレメント・タイプ 情報

表 468. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace_range	基本

説明 1 つの範囲が属するストライプ・セット開始点のストライプ 0 からのオフセット。

使用法 このエレメントは、DMS 表スペースにのみ適用できます。

fs_caching ファイル・システム・キャッシング

エレメント ID fs_caching

エレメント・タイプ 情報

表 469. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表スペース	tablespace	基本

表 470. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表スペース	event_tablespace	-

説明 特定の表スペースがファイル・システム・キャッシングを使用するかどうかを示します。

表アクティビティー

表アクティビティーに関するモニター・エレメント

次のエレメントにより、表に関する情報が提供されます。

- table_type 表タイプ : モニター・エレメント
- table_name 表名 : モニター・エレメント
- table_schema 表スキーマ名 : モニター・エレメント
- rows_deleted 削除行数 : モニター・エレメント
- rows_inserted 挿入行数 : モニター・エレメント
- rows_updated 更新行数 : モニター・エレメント
- rows_selected 選択行数 : モニター・エレメント
- rows_written 書き込み行数 : モニター・エレメント
- rows_read 読み取り行数 : モニター・エレメント
- overflow_accesses オーバーフロー・レコードへのアクセス : モニター・エレメント
- int_rows_deleted 削除された内部行 : モニター・エレメント
- int_rows_updated 更新された内部行 : モニター・エレメント
- int_rows_inserted 挿入された内部行 : モニター・エレメント
- table_file_id 表ファイル ID : モニター・エレメント
- page_reorgs ページ再編成 : モニター・エレメント
- data_object_pages データ・オブジェクト・ページ数 : モニター・エレメント
- index_object_pages 索引オブジェクト・ページ数 : モニター・エレメント
- lob_object_pages LOB オブジェクト・ページ数 : モニター・エレメント
- long_object_pages 長いオブジェクト・ページ数 : モニター・エレメント

table_type 表タイプ

エレメント ID table_type

エレメント・タイプ 情報

表 471. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

表 472. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 情報が戻される表のタイプ。

使用法 このエレメントは、情報が戻される表の識別に利用できます。表がユーザー表またはシステム・カタログ表の場合は、 *table_name* および *table_schema* を使用して表を識別できます。

表のタイプは、次のいずれかになります。

- ユーザー表。
- ドロップされたユーザー表。

- 一時表。表が使用された後に表がデータベース内に保持されない場合も、一時表に関する情報が戻されます。その場合でも、このタイプの表に関する情報は役に立ちます。
- システム・カタログ表。

関連資料:

- 349 ページの『table_file_id 表ファイル ID』

table_name 表名

エレメント ID table_name

エレメント・タイプ 情報

表 473. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本
アプリケーション	appl	ロック
ロック	appl_lock_list	ロック
ロック	lock	ロック
ロック	lock_wait	ロック

表 474. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 表の名前。

使用法 このエレメントと *table_schema* を組み合わせて使用すると、リソースの競合の原因を判別できます。

アプリケーション・レベル、アプリケーション・ロック・レベル、およびデッドロック・モニター・レベルでは、現在別のアプリケーションにロックされているためにロック待ちになっているアプリケーションの表を示します。スナップショット・モニターの場合、この項目が有効なのは、「lock」モニター・グループ情報がオンに設定され、さらにアプリケーションが表ロックの取得待ちであることを *lock_object_type* が示している場合に限ります。

オブジェクト・レベルのスナップショット・モニターの場合は、表レベルおよび行レベルのロックについてこの項目が戻されます。このレベルで報告される表は、このアプリケーションがこれらのロックを保留している表です。

表レベルのスナップショット・モニターおよびイベント・モニターの場合は、情報が収集された表を示します。一時表の場合、 *table_name* の形式は『TEMP (*n*, *m*)』 です。

- *n* は表スペース ID
- *m* は *table_file_id* エレメント

関連資料:

- 300 ページの『lock_object_type 待機中のロック対象タイプ』
- 301 ページの『lock_object_name ロック対象名』
- 341 ページの『table_schema 表スキーマ名』

table_schema 表スキーマ名

エレメント ID table_schema

エレメント・タイプ 情報

表 475. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本
アプリケーション	appl	ロック
ロック	appl_lock_list	ロック
ロック	lock	ロック
ロック	lock_wait	ロック

表 476. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-
デッドロック	lock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 表のスキーマ。

使用法 このエレメントと *table_name* を組み合わせて使用すると、リソースの競合の原因を判別できます。

アプリケーション・レベル、アプリケーション・ロック・レベル、およびデッドロック・モニター・レベルでは、現在別のアプリケーションにロックされているためにロック待ちになっているアプリケーションの表のスキーマを示します。このエレメントは、アプリケーションが表ロックの取得待ちであることを *lock_object_type* が示している場合にのみ設定できます。アプリケーション・レベルおよびアプリケーション・ロック・レベルのスナップショット・モニターでは、 “lock”モニター・グループ情報がオンの場合にのみこの項目が有効になります。

オブジェクト・レベルのスナップショット・モニターの場合は、表レベルおよび行レベルのロックについてこの項目が戻されます。このレベルで報告される表は、このアプリケーションがこれらのロックを保留している表です。

表レベルのスナップショット・モニターおよびイベント・モニターの場合は、このエレメントは情報が収集された表スキーマを示します。一時表の場合、 *table_schema* の形式は「*<agent_id><auth_id>*」です。

- *agent_id* は、一時表を作成するアプリケーションのアプリケーション・ハンドル

- *auth_id* は、アプリケーションがデータベースに接続するときに使用する許可 ID

関連資料:

- 300 ページの『lock_object_type 待機中のロック対象タイプ』
- 301 ページの『lock_object_name ロック対象名』
- 340 ページの『table_name 表名』

rows_deleted 削除行数

エレメント ID rows_deleted

エレメント・タイプ カウンター

表 477. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 478. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 これは、試行された行の削除の数です。

使用法 このエレメントを使用すると、データベース内の現在のアクティビティーのレベルがわかります。

このカウントには、 *int_rows_deleted* でカウントされる試行回数は含まれません。

関連資料:

- 347 ページの『int_rows_deleted 削除された内部行数』

rows_inserted 挿入行数

エレメント ID rows_inserted

エレメント・タイプ カウンター

表 479. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 480. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 これは、試行された行の挿入の数です。

使用法 このエレメントを使用すると、データベース内の現在のアクティビティーのレベルがわかります。

フェデレーテッド・システムの場合は、状況によってはフェデレーテッド・サーバーが INSERT FROM SUBSELECT をデータ・ソースにプッシュできるので、 INSERT ステートメントごとに複数の行を挿入できます。

このカウントには、 *int_rows_inserted* でカウントされる試行回数は含まれません。

rows_updated 更新行数

エレメント ID rows_updated

エレメント・タイプ カウンター

表 481. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 482. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 これは、試行された行の更新の数です。

使用法 このエレメントを使用すると、データベース内の現在のアクティビティーのレベルがわかります。

この値には、*int_rows_updated* でカウントされる更新は含まれません。ただし、複数の UPDATE ステートメントにより更新された行は、更新ごとにカウントされます。

関連資料:

- 347 ページの『int_rows_updated 更新された内部行数』

rows_selected 選択行数

エレメント ID rows_selected

エレメント・タイプ カウンター

表 483. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 484. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 これは、選択されてアプリケーションに戻された行の数です。

使用法 このエレメントを使用すると、データベース内の現在のアクティビティーのレベルがわかります。

このエレメントには、 COUNT(*) や結合などのアクションで読み込まれた行のカウントは含まれません。

フェデレーテッド・システムの場合は、データ・ソースからフェデレーテッド・サーバーに行を戻すのに要する平均時間を計算できます。

```
平均時間 = 戻された行数/照会応答合計時間
```

この結果を使用すると、SYSCAT.SERVERS 内の CPU 速度または通信速度などのパラメーターを変更できます。これらのパラメーターを変更すると、オプティマイザーが要求をデータ・ソースに送信するかしないかに影響を与えます。

注: モニター対象のゲートウェイが DB2 UDB バージョン 7.2 以前のものである場合は、このエレメントはスナップショット・モニターの dcs_dbase および dcs_appl 論理データ・グループで収集されます。

関連資料:

- 365 ページの『select_sql_stmts 実行された選択 SQL ステートメント』

rows_written 書き込み行数

エレメント ID rows_written

エレメント・タイプ カウンター

表 485. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本
アプリケーション	appl	基本
アプリケーション	stmt	基本

表 485. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 486. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
表	event_table	-
ステートメント	event_stmt	-
トランザクション	event_xact	-

説明 これは、表内で変更 (挿入、削除、または更新) された行の数です。

使用法 表レベルの情報で数値が高い場合は表の使用率が高いことを示しているため、 Run Statistics (RUNSTATS) ユーティリティーを使用して、この表に使用されるパッケージの効率を維持する必要があります。

アプリケーション接続およびステートメントでは、一時表で挿入、更新および削除があった行数がこのエレメントに含まれます。

アプリケーション、トランザクション、およびステートメントのレベルでこのエレメントを使用すると、相対的アクティビティー・レベルを解析したり、調整できる項目を見つけるのに便利です。

関連資料:

- 345 ページの『rows_read 読み取り行数』
- 347 ページの『int_rows_deleted 削除された内部行数』
- 347 ページの『int_rows_updated 更新された内部行数』
- 348 ページの『int_rows_inserted 挿入された内部行数』

rows_read 読み取り行数

エレメント ID rows_read

エレメント・タイプ カウンター

表 487. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
表	table	表
アプリケーション	appl	基本
アプリケーション	stmt	基本
アプリケーション	subsection	ステートメント
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 488. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
表	event_table	-
ステートメント	event_stmt	-
トランザクション	event_xact	-

説明 これは、表から読み取られた行の数です。

使用法 このエレメントを使用すると、使用率が高く、新たな索引を追加する必要があるような表を識別できます。不要な索引の保守作業を回避するには、「*管理ガイド*」に記載されている SQL EXPLAIN ステートメントを使用して、パッケージが索引を使用しているかどうかを判別します。

このカウントは、呼び出し側のアプリケーションに戻された行数では**ありません**。結果セットを戻すために、読み取りが必要になった行数です。例えば、次のステートメントを使用すると、アプリケーションに戻されるのは 1 行ですが、平均給与を判別するために多数の行が読み取られます。

```
SELECT AVG(SALARY) FROM USERID.EMPLOYEE
```

このカウントには、*overflow_accesses* の値が含まれます。ただし、索引アクセスは含まれません。つまり、アクセス・プランが索引アクセスだけを使用して、実際の行を見るために表の読み取りを行わなければ、 rows_read は増えません。

関連資料:

- 344 ページの『rows_written 書き込み行数』
- 346 ページの『overflow_accesses オーバーフロー・レコードへのアクセス』

overflow_accesses オーバーフロー・レコードへのアクセス

エレメント ID overflow_accesses

エレメント・タイプ カウンター

表 489. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 490. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 この表のオーバーフローした行へのアクセス (読み取りおよび書き込み) 数。

使用法 オーバーフローした行は、データのフラグメント化が生じたことを示しま

す。この数値が大きい場合は、 REORG ユーティリティーを使用してこのフラグメント化をクリーンアップし、表を再編成すると、表のパフォーマンスを改善できます。

行が更新されて、以前に書き込まれていた表に収まらなくなった場合に、行はオーバーフローします。 VARCHAR または ALTER TABLE ステートメントを更新すると、こうしたことが起こります。

関連資料:

- 344 ページの『rows_written 書き込み行数』
- 345 ページの『rows_read 読み取り行数』

int_rows_deleted 削除された内部行数

エレメント ID int_rows_deleted

エレメント・タイプ カウンター

表 491. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本
アプリケーション	stmt	基本
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 492. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 これは、内部アクティビティーの結果としてデータベースから削除された行の数です。

使用法 このエレメントを使用すると、ユーザーが気付かないようなデータベース・マネージャー内の内部アクティビティーを把握できます。このアクティビティーが高い場合は、表の設計内容を検討して、データベースに定義した参照制約やトリガーが必要かどうかを判別してください。

内部の削除アクティビティーは、次の原因により起こります。

- カスケード削除により ON CASCADE DELETE 参照制約が強制された場合。
- トリガーが起動された場合。

関連資料:

- 342 ページの『rows_deleted 削除行数』

int_rows_updated 更新された内部行数

エレメント ID int_rows_updated

エレメント・タイプ カウンター

表 493. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本
アプリケーション	stmt	基本
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 494. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-
ステートメント	event_stmt	-

説明 これは、内部アクティビティーの結果としてデータベースから更新された行の数です。

使用法 このエレメントを使用すると、ユーザーが気付かないようなデータベース・マネージャー内の内部アクティビティーを把握できます。このアクティビティーが高い場合は、表の設計内容を検討して、データベースに定義した参照制約が必要かどうかを判別してください。

内部の更新アクティビティーは、次の原因により起こります。

- *set null* 行更新により ON DELETE SET NULL ルールに定義した参照制約が強制された場合。
- トリガーが起動された場合。

関連資料:

- 343 ページの『rows_updated 更新行数』

int_rows_inserted 挿入された内部行数

エレメント ID int_rows_inserted

エレメント・タイプ カウンター

表 495. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本
アプリケーション	stmt	基本
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

表 496. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

表 496. イベント・モニター情報 (続き)

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
ステートメント	event_stmt	-

説明 トリガーによって行われた内部アクティビティーの結果としてデータベースに挿入された行の数。

使用法 このエレメントを使用すると、データベース・マネージャー内の内部アクティビティーを把握できます。このアクティビティーが高い場合は、このアクティビティーを低減するように設計を変更できるかどうか、検討してください。

関連資料:

- 342 ページの『rows_inserted 挿入行数』

table_file_id 表ファイル ID

エレメント ID table_file_id

エレメント・タイプ 情報

表 497. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ロック
表	table	基本
ロック	appl_lock_list	ロック
ロック	lock	ロック

表 498. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	lock	-

説明 これは表のファイル ID (FID) です。

使用法 このエレメントは情報提供のみを目的としています。データベース・システム・モニターの以前のバージョンとの互換性について情報を戻します。表を個別に識別することは**できません**。 *table_name* および *table_schema* を使用すると表を識別できます。

関連資料:

- 339 ページの『table_type 表タイプ』
- 340 ページの『table_name 表名』
- 341 ページの『table_schema 表スキーマ名』

page_reorgs ページ再編成

エレメント ID page_reorgs

エレメント・タイプ カウンター

表 499. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 500. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 表のページ再編成が実行された回数。

使用法

以下の状態では、ページに十分なスペースがあってもフラグメント化されることがあります。

- 新しい行が挿入される場合
- 既存の行が更新され、その結果レコード・サイズが大きくなる場合

ページがフラグメント化されている場合は、ページの再編成が必要になることがあります。再編成すると、フラグメント化されたすべてのスペースを連続したエリアに移動して、そこに新しいレコードが書き込まれます。このようなページの再編成 (page reorg) の実行には、数千の指示が必要になることがあります。また、操作のログ・レコードも生成されます。

ページ再編成の回数が多すぎると、最適な挿入パフォーマンスを達成できないことがあります。 REORG TABLE ユーティリティーを使用すると、表を再編成してフラグメント化をなくすことができます。さらに、ALTER TABLE ステートメントで APPEND パラメーターを使用すると、表の末尾にすべての挿入を付加するように指定できるので、ページの再編成を回避できます。

行の更新をすると行の長さが増えるような場合は、そのページに新しい行を挿入するだけの場所があっても、そのスペースのフラグメント化を解消するためにページ REORG が必要になる場合があります。あるいは、ページに新しい大きな行を挿入するだけの場所がない場合は、オーバーフロー・レコードが作成されて、読み取りのときに *overflow_accesses* の原因となります。どちらの状態も、可変長列の代わりに固定長列を使用することで回避できます。

関連資料:

- 342 ページの『rows_inserted 挿入行数』
- 343 ページの『rows_updated 更新行数』

data_object_pages データ・オブジェクト・ページ数

エレメント ID data_object_pages

エレメント・タイプ 情報

表 501. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

表 502. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 表が使用するディスク・ページの数。このサイズは、基本表のサイズのみを表します。索引オブジェクト、LOB データ、および長いデータが使用するスペースは、それぞれ *index_object_pages*、 *lob_object_pages*、および *long_object_pages* によって報告されます。

使用法 このエレメントを使用すると、特定の表が使用する実際の量のスペースを表示できます。このエレメントと表イベント・モニターを組み合わせて使用すると、時間とともに表が大きくなる比率を追跡できます。

関連資料:

- 351 ページの『index_object_pages 索引オブジェクト・ページ数』
- 351 ページの『lob_object_pages LOB オブジェクト・ページ数』
- 352 ページの『long_object_pages 長いオブジェクト・ページ数』

index_object_pages 索引オブジェクト・ページ数

エレメント ID index_object_pages

エレメント・タイプ 情報

表 503. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

表 504. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 表に対して定義されたすべての索引が使用するディスク・ページの数。

使用法 このエレメントを使用すると、特定の表に対して定義された索引が使用する実際の量のスペースを表示できます。このエレメントと表イベント・モニターを組み合わせて使用すると、時間とともに索引が大きくなる比率を追跡できます。

lob_object_pages LOB オブジェクト・ページ数

エレメント ID lob_object_pages

エレメント・タイプ 情報

表 505. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

表 506. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 LOB データが使用するディスク・ページの数。

使用法 このエレメントを使用すると、特定の表中の LOB データが使用する実際の量のスペースを表示できます。このエレメントと表イベント・モニターを組み合わせて使用すると、時間とともに LOB データが大きくなる比率を追跡できます。

long_object_pages 長いオブジェクト・ページ数

エレメント ID long_object_pages

エレメント・タイプ 情報

表 507. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table	基本

表 508. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-

説明 表中の長いデータが使用するディスク・ページの数。

使用法 このエレメントを使用すると、特定の表中の長いデータが使用する実際の量のスペースを表示できます。このエレメントと表イベント・モニターを組み合わせて使用すると、時間とともに長いデータが大きくなる比率を追跡できます。

表の再編成

表再編成モニター・エレメント

次のエレメントにより、表再編成についての情報が提供されます。

- reorg_type 表再編成の属性 : モニター・エレメント
- reorg_status 表再編成の状況 : モニター・エレメント
- reorg_phase 表再編成のフェーズ数 : モニター・エレメント
- reorg_phase_start 表再編成フェーズ開始時刻 : モニター・エレメント
- reorg_max_phase 表再編成の最大フェーズ数 : モニター・エレメント
- reorg_current_counter 表再編成の進行状況 : モニター・エレメント
- reorg_max_counter 表再編成の合計量 : モニター・エレメント
- reorg_completion 表再編成完了フラグ : モニター・エレメント
- reorg_start 表再編成開始時刻 : モニター・エレメント

- reorg_end 表再編成終了時刻 : モニター・エレメント
- reorg_index_id 表の再編成に使用される索引 : モニター・エレメント
- reorg_tbspc_id 表が再編成される表スペース : モニター・エレメント

reorg_type 表再編成の属性

エレメント ID reorg_type

エレメント・タイプ 情報

表 509. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成の属性設定値。

使用法 属性設定値として次のものがあります。各属性設定は、 db2ApiDf.h で定義されるビット・フラグ値に基づいています。

- 書き込みアクセスの許可: DB2REORG_ALLOW_WRITE
- 読み取りアクセスの許可: DB2REORG_ALLOW_READ
- アクセスを許可しない: DB2REORG_ALLOW_NONE
- 索引スキャンを介した再クラスタリング: DB2REORG_INDEXSCAN
- 長いフィールドの LOB データの REORG: DB2REORG_LONGLOB
- 表の切り捨てなし: DB2REORG_NOTRUNCATE_ONLINE

前記の属性設定値に加えて、 GET SNAPSHOT FOR TABLES コマンドの CLP 出力に以下の属性が示されます。これらの属性設定値は、他の属性設定値や表再編成に関するモニター・エレメントの値に基づいています。

- 再クラスタリング: reorg_index_id モニター・エレメントの値がゼロ以外の場合は、表再編成処理はこの属性を持ちます。
- レクラメーション処理: reorg_index_id モニター・エレメントの値がゼロの場合は、表再編成処理はこの属性を持ちます。
- インプレース表再編成: reorg_status モニター・エレメントの値が非 NULL の場合は、インプレース (オンライン) 再編成方式が使用されています。
- 表の再編成: reorg_phase モニター・エレメントの値が非 NULL の場合は、クラシック (オフライン) 再編成方式が使用されています。
- 表スキャンを介した再クラスタリング: DB2REORG_INDEXSCAN フラグが設定されていない場合は、表再編成処理はこの属性を持ちます。
- データのみの再編成: DB2REORG_LONGLOB フラグが設定されていない場合は、表再編成処理はこの属性を持ちます。

reorg_status 表再編成の状況

エレメント ID reorg_status

エレメント・タイプ 情報

表 510. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 インプレース (オンライン) 表再編成の状況。これはクラシック (オフライン) 表再編成には適用できません。

使用法 インプレース表再編成は、次の状態のいずれかになります (各状態は sqlmon.h での対応する定義とともに示しています)。

- 開始/再開: SQLM_REORG_STARTED
- 休止: SQLM_REORG_PAUSED
- 停止: SQLM_REORG_STOPPED
- 完了: SQLM_REORG_COMPLETED
- 切り捨て: SQLM_REORG_TRUNCATE

reorg_phase 表再編成のフェーズ数

エレメント ID reorg_phase

エレメント・タイプ 情報

表 511. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成のフェーズ。これはクラシック (オフライン) 表再編成のみに適用されます。

使用法 クラシック表再編成の場合は、次のフェーズがあります。

- ソート: SQLM_REORG_SORT
- ビルド: SQLM_REORG_BUILD
- 置換: SQLM_REORG_REPLACE
- 索引の再作成: SQLM_REORG_INDEX_RECREATE

reorg_phase_start 表再編成フェーズ開始時刻

エレメント ID reorg_phase_start

エレメント・タイプ タイム・スタンプ

表 512. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成のフェーズの開始時刻です。

reorg_max_phase 表再編成の最大フェーズ数

エレメント ID reorg_max_phase

エレメント・タイプ 情報

表 513. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 再編成処理のときに発生する再編成フェーズの最大数。これはクラシック (オフライン) 再編成にのみ適用されます。

reorg_current_counter 表再編成の進行状況

エレメント ID reorg_current_counter

エレメント・タイプ 情報

表 514. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 完了した表再編成の量を示す進行状況の単位。この値が示す進行の量は、行われる表再編成の合計量を示す reorg_max_counter の値に関連しています。次の公式を使用して、完了した表再編成のパーセンテージを判別することができます。

```
表再編成の進行状況 = reorg_current_counter / reorg_max_counter * 100
```

reorg_max_counter 表再編成の合計量

エレメント ID reorg_max_counter

エレメント・タイプ 情報

表 515. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成において行われる作業の合計量を示す値です。この値を、完了した作業の量を示す reorg_current_counter とともに使用して、表再編成の進行状況を判別することができます。

reorg_completion 表再編成完了フラグ

エレメント ID reorg_completion

エレメント・タイプ 情報

表 516. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成の成功の標識。

使用法 表再編成処理が正常に終了すると、このエレメントの値は 0 になります。表再編成処理が失敗すると、このエレメントの値は -1 になります。成功および失敗の値は、sqlmon.h で次のように定義されています。

- 成功: SQLM_REORG_SUCCESS
- 失敗: SQLM_REORG_FAIL

表再編成が異常終了した場合は、履歴ファイルを使用して、警告やエラーなどの診断情報を参照してください。このデータにアクセスするには、LIST HISTORY コマンドを使用します。診断情報について詳しくは、管理上の通知を参照してください。

reorg_start 表再編成開始時刻

エレメント ID reorg_start

エレメント・タイプ タイム・スタンプ

表 517. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成の開始時刻です。

reorg_end 表再編成終了時刻

エレメント ID reorg_end

エレメント・タイプ タイム・スタンプ

表 518. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表再編成の終了時刻です。

reorg_index_id 表の再編成に使用される索引

エレメント ID reorg_index_id

エレメント・タイプ 情報

表 519. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表の再編成に使用されている索引。

reorg_tbspc_id 表が再編成される表スペース

エレメント ID reorg_tbspc_id

エレメント・タイプ 情報

表 520. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
表	table_reorg	基本

説明 表が再編成される表スペース。

SQL カーソル

SQL カーソルに関するモニター・エレメント

次のエレメントにより、SQL カーソルに関する情報が提供されます。

- open_rem_curs 開かれているリモート・カーソル : モニター・エレメント
- open_rem_curs_blk 開かれているリモート・ブロック・カーソル : モニター・エレメント
- rej_curs_blk リジェクトされたブロック・カーソル要求 : モニター・エレメント
- acc_curs_blk 受け入れられたブロック・カーソル要求 : モニター・エレメント
- open_loc_curs 開かれているローカル・カーソル : モニター・エレメント
- open_loc_curs_blk 開かれているローカル・ブロック・カーソル : モニター・エレメント

open_rem_curs 開かれているリモート・カーソル

エレメント ID open_rem_curs

エレメント・タイプ ゲージ

表 521. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

説明 このアプリケーション用に現在開かれているリモート・カーソルの数。*open_rem_curs_blk* がカウントするカーソルを含みます。

使用法 このエレメントと *open_rem_curs_blk* を組み合わせて使用すると、リモート・ブロック・カーソルのパーセンテージを計算できます。パーセンテージが低い場合は、アプリケーション内の行ブロッキングを改善するとパフォーマンスが向上します。詳しくは、『*open_rem_curs_blk*』を参照してください。

ローカル・データベースに接続されているアプリケーションが使用するオープン・カーソルの数については、『*open_loc_curs*』を参照してください。

関連資料:

- 357 ページの『open_rem_curs_blk 開かれているリモート・ブロック・カーソル』
- 359 ページの『open_loc_curs 開かれているローカル・カーソル』

open_rem_curs_blk 開かれているリモート・ブロック・カーソル

エレメント ID open_rem_curs_blk

エレメント・タイプ ゲージ

表 522. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

説明 このアプリケーション用に現在開かれているリモート・ブロック・カーソルの数。

使用法 このエレメントと *open_rem_curs* を組み合わせて使用すると、リモート・ブロック・カーソルのパーセンテージを計算できます。パーセンテージが低い場合は、アプリケーション内の行ブロッキングを改善するとパフォーマンスが向上します。

- 未確定カーソル処理のレコード・ブロッキングについて、プリコンパイル・オプションをチェックする。
- カーソルを再定義してブロッキングを許可する (例えば、可能な場合は、カーソルに FOR FETCH ONLY を指定する)。

rej_curs_blk および *acc_curs_blk* が提供する情報を使用すると、構成パラメーターを調整して、アプリケーション内の行ブロッキングを改善できます。

ローカル・データベースに接続されているアプリケーションが使用するオープン・ブロック・カーソルの数については、『*open_loc_curs_blk*』を参照してください。

関連資料:

- 357 ページの『open_rem_curs 開かれているリモート・カーソル』
- 358 ページの『rej_curs_blk リジェクトされたブロック・カーソル要求』
- 359 ページの『acc_curs_blk 受け入れられたブロック・カーソル要求』
- 359 ページの『open_loc_curs 開かれているローカル・カーソル』
- 360 ページの『open_loc_curs_blk 開かれているローカル・ブロック・カーソル』

rej_curs_blk リジェクトされたブロック・カーソル要求

エレメント ID rej_curs_blk

エレメント・タイプ カウンター

表 523. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

表 524. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 サーバーでの入出力ブロック要求がリジェクトされて、要求が非ブロック化入出力に変換された回数。

使用法 多数のカーソル・ブロッキング・データがあると、通信ヒープが満杯になることがあります。このヒープが満杯になると、エラーは戻されません。その代わりに、カーソルをブロックするための入出力ブロックが割り振られなくなります。カーソルがデータをブロックできない場合は、パフォーマンスに影響が現れます。

多数のカーソルがデータ・ブロックを実行できない場合は、次のようにしてパフォーマンスを改善できます。

- *query_heap* データベース・マネージャー構成パラメーターのサイズを大きくする。

関連資料:

- 357 ページの『open_rem_curs 開かれているリモート・カーソル』
- 357 ページの『open_rem_curs_blk 開かれているリモート・ブロック・カーソル』
- 359 ページの『acc_curs_blk 受け入れられたブロック・カーソル要求』
- 359 ページの『open_loc_curs 開かれているローカル・カーソル』
- 360 ページの『open_loc_curs_blk 開かれているローカル・ブロック・カーソル』

acc_curs_blk 受け入れられたブロック・カーソル要求

エレメント ID acc_curs_blk

エレメント・タイプ カウンター

表 525. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

表 526. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 入出力ブロック要求が受け入れられた回数。

使用法 このエレメントと *rej_curs_blk* を組み合わせて使用すると、受け入れられたブロッキング要求、リジェクトされたブロッキング要求、またはその両方のパーセンテージを計算できます。

この情報を使用して構成パラメーターを調整する方法については、『*rej_curs_blk*』を参照してください。

関連資料:

- 357 ページの『open_rem_curs 開かれているリモート・カーソル』
- 357 ページの『open_rem_curs_blk 開かれているリモート・ブロック・カーソル』
- 358 ページの『rej_curs_blk リジェクトされたブロック・カーソル要求』
- 359 ページの『open_loc_curs 開かれているローカル・カーソル』
- 360 ページの『open_loc_curs_blk 開かれているローカル・ブロック・カーソル』

open_loc_curs 開かれているローカル・カーソル

エレメント ID open_loc_curs

エレメント・タイプ ゲージ

表 527. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

説明 このアプリケーション用に現在開かれているローカル・カーソルの数。*open_loc_curs_blk* がカウントするカーソルを含みます。

使用法 このエレメントと *open_loc_curs_blk* を組み合わせて使用すると、ローカル・ブロック・カーソルのパーセンテージを計算できます。パーセンテージが低い場合は、アプリケーション内の行ブロッキングを改善するとパフォーマンスが向上します。

リモート・アプリケーションが使用するカーソルについては、『*open_rem_curs*』を参照してください。

関連資料:

- 357 ページの『open_rem_curs 開かれているリモート・カーソル』
- 357 ページの『open_rem_curs_blk 開かれているリモート・ブロック・カーソル』
- 358 ページの『rej_curs_blk リジェクトされたブロック・カーソル要求』
- 359 ページの『acc_curs_blk 受け入れられたブロック・カーソル要求』
- 360 ページの『open_loc_curs_blk 開かれているローカル・ブロック・カーソル』

open_loc_curs_blk 開かれているローカル・ブロック・カーソル

エレメント ID open_loc_curs_blk

エレメント・タイプ ゲージ

表 528. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

説明 このアプリケーション用に現在開かれているローカル・ブロック・カーソルの数。

使用法 このエレメントと *open_loc_curs* を組み合わせて使用すると、ローカル・ブロック・カーソルのパーセンテージを計算できます。パーセンテージが低い場合は、アプリケーション内の行ブロッキングを改善するとパフォーマンスが向上します。

- 未確定カーソル処理のレコード・ブロッキングについて、プリコンパイル・オプションをチェックする。
- カーソルを再定義してブロッキングを許可する (例えば、可能な場合は、カーソルに FOR FETCH ONLY を指定する)。

rej_curs_blk および acc_curs_blk が提供する情報を使用すると、構成パラメーターを調整して、アプリケーション内の行ブロッキングを改善できます。

リモート・アプリケーションが使用するブロック・カーソルについては、『*open_rem_curs_blk*』を参照してください。

関連資料:

- 357 ページの『open_rem_curs 開かれているリモート・カーソル』
- 357 ページの『open_rem_curs_blk 開かれているリモート・ブロック・カーソル』
- 358 ページの『rej_curs_blk リジェクトされたブロック・カーソル要求』
- 359 ページの『acc_curs_blk 受け入れられたブロック・カーソル要求』

• 359 ページの『open_loc_curs 開かれているローカル・カーソル』

SQL ステートメント・アクティビティー

SQL ステートメント・アクティビティーに関するモニター・エレメント

次のエレメントにより、SQL ステートメントに関する情報が提供されます。

- static_sql_stmts 試行された静的 SQL ステートメント : モニター・エレメント
- dynamic_sql_stmts 試行された動的 SQL ステートメント : モニター・エレメント
- failed_sql_stmts 失敗したステートメント操作 : モニター・エレメント
- commit_sql_stmts 試行されたコミット・ステートメント : モニター・エレメント
- rollback_sql_stmts 試行されたロールバック・ステートメント : モニター・エレメント
- select_sql_stmts 実行された選択 SQL ステートメント : モニター・エレメント
- uid_sql_stmts 実行された更新/挿入/削除 SQL ステートメント : モニター・エレメント
- ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント : モニター・エレメント
- int_auto_rebinds 内部自動再バインド : モニター・エレメント
- int_commits 内部コミット数 : モニター・エレメント
- int_rollbacks 内部ロールバック数 : モニター・エレメント
- int_deadlock_rollbacks デッドロックによる内部ロールバック : モニター・エレメント
- sql_reqs_since_commit 最終コミット後の SQL 要求 : モニター・エレメント
- stmt_node_number ステートメント・ノード : モニター・エレメント
- binds_precompiles 試行されたバインド/プリコンパイル : モニター・エレメント

static_sql_stmts 試行された静的 SQL ステートメント

エレメント ID static_sql_stmts

エレメント・タイプ カウンター

表 529. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 530. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試行された静的 SQL ステートメントの数。

使用法 このエレメントを使用すると、データベース・レベルまたはアプリケーション・レベルで成功した SQL ステートメントの合計数を計算できます。

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= モニター期間中のスループット
```

関連資料:

- 362 ページの『failed_sql_stmts 失敗したステートメント操作』

dynamic_sql_stmts 試行された動的 SQL ステートメント

エレメント ID dynamic_sql_stmts

エレメント・タイプ カウンター

表 531. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 532. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試行された動的 SQL ステートメントの数。

使用法 このエレメントを使用すると、データベース・レベルまたはアプリケーション・レベルで成功した SQL ステートメントの合計数を計算できます。

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= モニター期間中のスループット
```

関連資料:

- 362 ページの『failed_sql_stmts 失敗したステートメント操作』

failed_sql_stmts 失敗したステートメント操作

エレメント ID failed_sql_stmts

エレメント・タイプ カウンター

表 533. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本

表 533. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_remote	基本
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 534. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試行されたが失敗した SQL ステートメントの数。

使用法 このエレメントを使用すると、データベース・レベルまたはアプリケーション・レベルで成功した SQL ステートメントの合計数を計算できます。

```
  dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= モニター期間中のスループット
```

このカウントには、負の SQLCODE を受信したすべての SQL ステートメントを含みます。

このエレメントは、パフォーマンスが低い場合の原因の判別にも役に立ちます。これは、失敗したステートメントがあると、データベース・マネージャーで余分な時間がかかり、その結果としてデータベースのスループットが落ちるからです。

関連資料:

- 361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
- 362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』

commit_sql_stmts 試行されたコミット・ステートメント

エレメント ID commit_sql_stmts

エレメント・タイプ カウンター

表 535. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 536. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試行された SQL COMMIT ステートメントの合計数。

使用法 モニター期間中にこのカウンターの変化量が少ない場合は、各アプリケーションのコミット頻度が少ないことを示し、ロギングとデータの並行性について問題となる場合があります。

このエレメントを使用すると、次の項目を合計して合計作業単位数も計算できます。

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 計算した作業単位に含まれるのは、次の時点以降の作業単位だけです。
- データベースへの接続 (データベース・レベルの情報の場合は、最初の接続時刻)
- データベース・モニター・カウンターの最後のリセット

この計算は、データベース・レベルとアプリケーション・レベルのいずれでも行えます。

関連資料:

- 364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
- 368 ページの『int_commits 内部コミット数』
- 369 ページの『int_rollbacks 内部ロールバック数』
- 371 ページの『int_deadlock_rollbacks デッドロックによる内部ロールバック』

rollback_sql_stmts 試行されたロールバック・ステートメント

エレメント ID rollback_sql_stmts

エレメント・タイプ カウンター

表 537. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 538. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試行された SQL ROLLBACK ステートメントの合計数。

使用法 ロールバックは、アプリケーション要求、デッドロック、またはエラー状態の結果として起こります。このエレメントでは、アプリケーションが発行したロールバック・ステートメント**のみ**カウントされます。

アプリケーション・レベルでは、このエレメントはアプリケーションのデータベース・アクティビティー・レベルとその他のアプリケーションとの競合の量を判別するのに役立ちます。データベース・レベルでは、データベース内のアクティビティーの量とデータベース上でのアプリケーション間の競合の量を判別できます。

注: ロールバック・アクティビティーが多くなるとデータベースのスループットが低下するので、ロールバックの回数を最小限にとどめてください。

次の項目を合計すると、作業単位の合計数も計算できます。

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

関連資料:

- 363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
- 368 ページの『int_commits 内部コミット数』
- 369 ページの『int_rollbacks 内部ロールバック数』
- 371 ページの『int_deadlock_rollbacks デッドロックによる内部ロールバック』
- 373 ページの『stmt_type ステートメント・タイプ』

select_sql_stmts 実行された選択 SQL ステートメント

エレメント ID select_sql_stmts

エレメント・タイプ カウンター

表 539. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
データベース	dbase_remote	基本
表スペース	tablespace	基本
アプリケーション	appl	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 540. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 実行された SQL SELECT ステートメントの数。

使用法 このエレメントを使用すると、アプリケーション・レベルまたはデータベース・レベルのデータベース・アクティビティーのレベルを判別できます。

次の公式を使用すると、すべてのステートメントに対する SELECT ステートメントの比率を計算できます。

```
  select_sql_stmts
/ ( static_sql_stmts
  + dynamic_sql_stmts )
```

この情報は、アプリケーションのアクティビティーおよびスループットの分析に役立ちます。

関連資料:

- 361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
- 362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』

uid_sql_stmts 実行された更新/挿入/削除 SQL ステートメント

エレメント ID uid_sql_stmts

エレメント・タイプ カウンター

表 541. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 542. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 実行された SQL UPDATE、INSERT、および DELETE ステートメントの数。

使用法 このエレメントを使用すると、アプリケーション・レベルまたはデータベース・レベルのデータベース・アクティビティーのレベルを判別できます。

次の公式を使用すると、すべてのステートメントに対する UPDATE、INSERT および DELETE ステートメントの比率を計算できます。

```
  uid_sql_stmts
/ (static_sql_stmts + dynamic_sql_stmts )
```

この情報は、アプリケーションのアクティビティーおよびスループットの分析に役立ちます。

関連資料:

- 361 ページの『static_sql_stmts 試行された静的 SQL ステートメント』
- 362 ページの『dynamic_sql_stmts 試行された動的 SQL ステートメント』

ddl_sql_stmts データ定義言語 (DDL) SQL ステートメント

エレメント ID ddl_sql_stmts

エレメント・タイプ カウンター

表 543. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 544. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 このエレメントは、実行された SQL データ定義言語 (DDL) ステートメントの数を示します。

使用法 このエレメントを使用すると、アプリケーション・レベルまたはデータベース・レベルのデータベース・アクティビティーのレベルを判別できます。DDL ステートメントは、システム・カタログ表への影響のために実行にコストがかかります。そのため、このエレメントの値が大きい場合は、その原因を突き止めて、このアクティビティーが実行されないように制約すべきです。

このエレメントを使用すると、次の公式を使用して、 DDL アクティビティーのパーセンテージも計算できます。

```
ddl_sql_stmts / total number of statements
```

この情報は、アプリケーションのアクティビティーおよびスループットの分析に役立ちます。 DDL ステートメントも次の項目に影響を与えます。

- カタログ・キャッシュ。保管されている表記述子情報と許可情報が無効になるので、システム・カタログから情報を取り出すためのシステム・オーバーヘッドが増加します。
- パッケージ・キャッシュ。保管されているセクションが無効になるので、セクションの再コンパイルのためのシステム・オーバーヘッドが増加します。

DDL ステートメントの例としては、CREATE TABLE、CREATE VIEW、ALTER TABLE、および DROP INDEX があります。

int_auto_rebinds 内部自動再バインド

エレメント ID int_auto_rebinds

エレメント・タイプ カウンター

表 545. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 546. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試行された自動再バインド (または再コンパイル) の数。

使用法 自動再バインドは、パッケージが無効にされている場合にシステムが実行する内部バインドです。再バインドは、データベース・マネージャーが初めてパッケージから SQL ステートメントを実行する必要が生じたときに行われます。パッケージは、例えば次の場合に無効にされます。

- プランが従属している表、ビュー、または索引などのオブジェクトのドロップ。
- 外部キーの追加またはドロップ。
- プランが従属しているオブジェクト特権の取り消し。

このエレメントを使用すると、アプリケーション・レベルまたはデータベース・レベルのデータベース・アクティビティーのレベルを判別できます。int_auto_rebinds はパフォーマンスに大きな影響を与えるので、できるだけ最小限に抑える必要があります。

このエレメントを使用すると、次の公式を使用して、再バインド・アクティビティーのパーセンテージも計算できます。

```
int_auto_rebinds / total number of statements
```

この情報は、アプリケーションのアクティビティーおよびスループットの分析に役立ちます。

関連資料:

• 372 ページの『binds_precompiles 試行されたバインド/プリコンパイル』

int_commits 内部コミット数

エレメント ID int_commits

エレメント・タイプ カウンター

表 547. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 548. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 データベース・マネージャーによって内部で開始されたコミットの合計数。

使用法 内部コミットは、以下のいずれかの間に発生する場合があります。

- 再編成
- インポート
- バインドまたはプリコンパイル
- 明示的な SQL COMMIT ステートメントを実行せずにアプリケーションが終了した場合 (UNIX の場合)

この値には明示的な SQL COMMIT ステートメントは含まれず、次の時点以降のこれらの内部コミットの数を表します。

- データベースへの接続 (データベース・レベルの情報の場合は、最初の接続時刻)
- データベース・モニター・カウンターの最後のリセット

このエレメントを使用すると、次の項目を合計して合計作業単位数を計算できます。

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 計算した作業単位に含まれるのは、次の時点以降の作業単位だけです。

- データベースへの接続 (データベース・レベルの情報の場合は、最初の接続時刻)
- データベース・モニター・カウンターの最後のリセット

この計算は、アプリケーションまたはデータベース・レベルで実行できます。

関連資料:

- 363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』
- 364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
- 369 ページの『int_rollbacks 内部ロールバック数』

int_rollbacks 内部ロールバック数

エレメント ID int_rollbacks

エレメント・タイプ カウンター

表 549. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 550. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 データベース・マネージャーによって内部で開始されたロールバックの合計数。

使用法 内部ロールバックは、以下のいずれかが正常に完了**できない**ときに起こります。

- 再編成
- インポート
- バインドまたはプリコンパイル
- デッドロック状態またはロック・タイムアウト状態によりアプリケーションが終了した場合
- 明示的なコミットまたはロールバック・ステートメントを実行せずにアプリケーションが終了した場合 (Windows の場合)

この値は、次の時点以降のこれらの内部ロールバックの数を表します。

- データベースへの接続 (データベース・レベルの情報の場合は、最初の接続時刻)
- データベース・モニター・カウンターの最後のリセット

この値には明示的な SQL ROLLBACK ステートメントは含まれませんが、int_deadlock_rollbacks のカウントは含まれます。

このエレメントを使用すると、次の項目を合計して合計作業単位数を計算できます。

```
  commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 計算した作業単位には、次の時点以降の作業単位が含まれます。

- データベースへの接続 (データベース・レベルの情報の場合は、最初の接続時刻)
- データベース・モニター・カウンターの最後のリセット

この計算は、アプリケーションまたはデータベース・レベルで実行できます。

関連資料:

- 363 ページの『commit_sql_stmts 試行されたコミット・ステートメント』

- 364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
- 368 ページの『int_commits 内部コミット数』
- 371 ページの『int_deadlock_rollbacks デッドロックによる内部ロールバック』

int_deadlock_rollbacks デッドロックによる内部ロールバック

エレメント ID int_deadlock_rollbacks

エレメント・タイプ カウンター

表 551. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 552. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-

説明 デッドロックのためにデータベース・マネージャーによって開始された強制ロールバックの合計数。ロールバックは、デッドロックを解決するためにデータベース・マネージャーが選択したアプリケーション内の現在の作業単位上で実行されます。

使用法 このエレメントは中断されたデッドロックの数を示しており、並行性の問題の標識として使用できます。 int_deadlock_rollbackss は、データベースのスループットが低下するため、この問題は重要です。

この値は、int_rollbacks が示す値に含まれています。

関連資料:

- 295 ページの『deadlocks デッドロック検出数』
- 364 ページの『rollback_sql_stmts 試行されたロールバック・ステートメント』
- 369 ページの『int_rollbacks 内部ロールバック数』

sql_reqs_since_commit 最終コミット後の SQL 要求

エレメント ID sql_reqs_since_commit

エレメント・タイプ 情報

表 553. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	基本

説明 最後のコミット以降にサブミットされた SQL 要求の数。

使用法 このエレメントを使用すると、トランザクションの進行状況をモニターできます。

stmt_node_number ステートメント・ノード

エレメント ID stmt_node_number

エレメント・タイプ 情報

表 554. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント

説明 ステートメントが実行されたノード。

使用法 各ステートメントをそのステートメントが実行されたノードと関連付けるときに使用します。

binds_precompiles 試行されたバインド/プリコンパイル

エレメント ID binds_precompiles

エレメント・タイプ カウンター

表 555. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

表 556. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
接続	event_conn	-

説明 試みられたバインドおよびプリコンパイルの数。

使用法 このエレメントを使用すると、データベース・マネージャー内の現在のアクティビティーのレベルがわかります。

この値には *int_auto_rebinds* のカウントは含まれませんが、 REBIND PACKAGE コマンドの結果として起こるバインド数は含まれます。

関連資料:

• 368 ページの『int_auto_rebinds 内部自動再バインド』

SQL ステートメントの詳細

SQL ステートメント詳細に関するモニター・エレメント

注: ステートメント・イベント・モニターは、取り出しのログを取りません。
次のエレメントにより、SQL ステートメントに関する詳細情報が提供されます。

- stmt_type ステートメント・タイプ : モニター・エレメント
- stmt_operation/operation ステートメント操作 : モニター・エレメント
- package_name パッケージ名 : モニター・エレメント

- package_version_id パッケージ・バージョン : モニター・エレメント
- consistency_token パッケージ整合性トークン : モニター・エレメント
- section_number セクション番号 : モニター・エレメント
- cursor_name カーソル名 : モニター・エレメント
- creator アプリケーション作成者 : モニター・エレメント
- stmt_start ステートメント操作開始タイム・スタンプ : モニター・エレメント
- stmt_stop ステートメント操作停止タイム・スタンプ : モニター・エレメント
- stop_time イベント停止時刻 : モニター・エレメント
- start_time イベント開始時刻 : モニター・エレメント
- stmt_elapsed_time 最新のステートメント経過時間 : モニター・エレメント
- stmt_text SQL 動的ステートメント・テキスト : モニター・エレメント
- stmt_sorts ステートメント・ソート回数 : モニター・エレメント
- fetch_count 成功した取り出しの数 : モニター・エレメント
- sqlca SQL 連絡域 (SQLCA) : モニター・エレメント
- query_card_estimate 行数見積もりの照会 : モニター・エレメント
- query_cost_estimate 照会コストの見積もり : モニター・エレメント

stmt_type ステートメント・タイプ

エレメント ID stmt_type

エレメント・タイプ 情報

表 557. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント

表 558. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 処理されるステートメントのタイプ。

使用法 このエレメントを使用すると、実行中のステートメントのタイプを判別できます。タイプは次のいずれかになります。

- 静的 SQL ステートメント。
- 動的 SQL ステートメント。
- SQL ステートメント以外の操作。例えば、バインドやプリコンパイルなどの操作。

スナップショット・モニターの場合は、このエレメントにより、現在処理中または最後に処理されたステートメントがわかります。

注: API ユーザーは、データベース・システム・モニターの定数の定義が含まれているヘッダー・ファイル *sqlmon.h* を参照してください。

関連資料:

- 375 ページの『package_name パッケージ名』
- 377 ページの『section_number セクション番号』
- 378 ページの『creator アプリケーション作成者』
- 381 ページの『stmt_text SQL 動的ステートメント・テキスト』
- 376 ページの『package_version_id パッケージ・バージョン』

stmt_operation/operation ステートメント操作

エレメント ID stmt_operation (スナップショット・モニター)
operation (イベント・モニター)

エレメント・タイプ 情報

表 559. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

表 560. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 現在処理中または (現在実行中のものがない場合は) 最後に処理されたステートメント操作。

使用法 このエレメントを使用すると、実行中の操作または最後に終了した操作を判別できます。

次のいずれかになります。

SQL 操作の場合:
- SELECT
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- OPEN
- FETCH
- CLOSE
- DESCRIBE
- STATIC COMMIT
- STATIC ROLLBACK
- FREE LOCATOR
- PREP_COMMIT
- CALL
- PREP_OPEN
- PREP_EXEC
- COMPILE

非 SQL 操作の場合:

- RUN STATISTICS
- REORG
- REBIND
- REDISTRIBUTE
- GET TABLE AUTHORIZATION
- GET ADMINISTRATIVE AUTHORIZATION

注: API ユーザーは、データベース・システム・モニターの定数の定義が含まれているヘッダー・ファイル *sqlmon.h* を参照してください。

関連資料:

- 373 ページの『stmt_type ステートメント・タイプ』
- 375 ページの『package_name パッケージ名』
- 377 ページの『section_number セクション番号』
- 378 ページの『creator アプリケーション作成者』
- 381 ページの『stmt_text SQL 動的ステートメント・テキスト』
- 383 ページの『fetch_count 成功した取り出しの数』
- 376 ページの『package_version_id パッケージ・バージョン』

package_name パッケージ名

エレメント ID package_name

エレメント・タイプ 情報

表 561. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

表 562. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 現在実行中の SQL ステートメントが含まれているパッケージの名前。

使用法 このエレメントを使用すると、実行中のアプリケーション・プログラムや SQL ステートメントを識別できます。

関連資料:

- 377 ページの『section_number セクション番号』
- 378 ページの『creator アプリケーション作成者』
- 381 ページの『stmt_text SQL 動的ステートメント・テキスト』
- 376 ページの『package_version_id パッケージ・バージョン』

consistency_token パッケージ整合性トークン

エレメント ID consistency_token

エレメント・タイプ 情報

表 563. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント

表 564. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_stmt	-

説明 特定のパッケージ名および作成者について、複数のバージョンが存在する場合があります (DB2 バージョン 8 以降)。パッケージ整合性トークンを使用すると、現在実行中の SQL を含むパッケージのバージョンを識別できます。

使用法 このエレメントを、パッケージおよび実行中の SQL ステートメントの識別に利用できます。

package_version_id パッケージ・バージョン

エレメント ID package_version_id

エレメント・タイプ 情報

表 565. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント

表 566. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_stmt	-

説明 特定のパッケージ名および作成者について、複数のバージョンが存在する場合があります (DB2 バージョン 8 以降)。パッケージ・バージョンは、現在実行中の SQL を含むパッケージのバージョン ID を示します。パッケージのバージョンは、組み込み SQL プログラムをプリコンパイル (PREP) するときに VERSION キーワードを使用して決めます。プリコンパイル時に指定がない場合は、パッケージ・バージョンが "" (空ストリング) となります。

使用法 このエレメントを、パッケージおよび実行中の SQL ステートメントの識別に利用できます。

関連資料:

- 375 ページの『package_name パッケージ名』
- 377 ページの『section_number セクション番号』
- 378 ページの『creator アプリケーション作成者』
- 381 ページの『stmt_text SQL 動的ステートメント・テキスト』

section_number セクション番号

エレメント ID section_number

エレメント・タイプ 情報

表 567. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

表 568. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 現在処理中または最後に処理された SQL ステートメントのパッケージにある内部セクション番号。

使用法 静的 SQL の場合は、このエレメントと creator、package_version_id、および package_name を組み合わせて使用すると、次の照会例を利用して、SYSCAT.STATEMENTS システム・カタログ表を照会し、静的 SQL ステートメント・テキストを取得できます。

```
SELECT SEQNO, SUBSTR(TEXT,1,120)
       FROM SYSCAT.STATEMENTS
       WHERE PKGNAME   = 'package_name' AND
             PKGSCHEMA = 'creator'      AND
             VERSION = 'package_version_id' AND
             SECTNO    = section_number
       ORDER BY SEQNO
```

注: 静的ステートメント・テキストを取得するときは、システム・カタログ表にこの照会をするとロックの競合を起こすことがあるので注意してください。この照会を使用するのは、できるだけデータベースに対するその他のアクティビティーが少ないときだけにしてください。

関連資料:

- 375 ページの『package_name パッケージ名』
- 378 ページの『creator アプリケーション作成者』
- 381 ページの『stmt_text SQL 動的ステートメント・テキスト』
- 376 ページの『package_version_id パッケージ・バージョン』

cursor_name カーソル名

エレメント ID cursor_name

エレメント・タイプ 情報

表 569. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント

表 570. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 この SQL ステートメントに対応するカーソルの名前。

使用法 このエレメントを使用すると、処理中の SQL ステートメントを識別できます。この名前は、SQL SELECT ステートメントの OPEN、 FETCH、CLOSE、および PREPARE に使用されます。カーソルを使用しない場合は、このフィールドはブランクになります。

関連資料:

- 373 ページの『stmt_type ステートメント・タイプ』
- 381 ページの『stmt_text SQL 動的ステートメント・テキスト』
- 383 ページの『fetch_count 成功した取り出しの数』

creator アプリケーション作成者

エレメント ID creator

エレメント・タイプ 情報

表 571. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

表 572. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロック	event_dlconn	-
ステートメント	event_stmt	-

説明 アプリケーションをプリコンパイルしたユーザーの許可 ID。

使用法 このエレメントとカタログ内のパッケージ・セクション情報の CREATOR 列を組み合わせて使用し、処理中の SQL ステートメントを識別してください。

CURRENT PACKAGE PATH 特殊レジスターを設定した場合には、 SQL ステートメントの存続期間中に *creator* 値はさまざまな値を反映します。PACKAGE PATH の解決の前にスナップショットまたはイベント・モニター・レコードが取られる場合は、 *creator* 値はクライアント要求から流れる値を反映します。 PACKAGE PATH の解決の後にスナップショットまたはイベント・モニター・レコードが取られる場合は、 *creator* 値は解決されたパッケージの作成者を反映します。解決されたパッケージの *creator* 値は CURRENT PACKAGE PATH SPECIAL REGISTER 中に最初に表示される値で、パッケージ名およびユニーク ID はクライアント要求の名前および ID と一致します。

関連資料:

- 375 ページの『package_name パッケージ名』
- 377 ページの『section_number セクション番号』
- 376 ページの『package_version_id パッケージ・バージョン』

stmt_start ステートメント操作開始タイム・スタンプ

エレメント ID stmt_start

エレメント・タイプ タイム・スタンプ

表 573. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント、タイム・スタンプ
DCS ステートメント	dcs_stmt	ステートメント、タイム・スタンプ

説明 stmt_operation の実行開始日時。

使用法 このエレメントと stmt_stop を組み合わせて使用すると、ステートメント操作の実行経過時間を計算できます。

関連資料:

- 374 ページの『stmt_operation/operation ステートメント操作』
- 379 ページの『stmt_stop ステートメント操作停止タイム・スタンプ』

stmt_stop ステートメント操作停止タイム・スタンプ

エレメント ID stmt_stop

エレメント・タイプ タイム・スタンプ

表 574. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント、タイム・スタンプ
DCS ステートメント	dcs_stmt	ステートメント、タイム・スタンプ

説明 stmt_operation の実行停止日時。

使用法 このエレメントと stmt_start を組み合わせて使用すると、ステートメント操作の実行経過時間を計算できます。

関連資料:

- 374 ページの『stmt_operation/operation ステートメント操作』
- 379 ページの『stmt_start ステートメント操作開始タイム・スタンプ』
- 380 ページの『stop_time イベント停止時刻』

stop_time イベント停止時刻

エレメント ID stop_time

エレメント・タイプ タイム・スタンプ

表 575. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_stmt	タイム・スタンプ

説明 ステートメントが実行を停止した日時。

使用法 このエレメントと *start_time* を組み合わせて使用すると、ステートメントの実行経過時間を計算できます。

FETCH ステートメント・イベントの場合は、最後に正常な取り出しが行われた時刻です。

注: 「タイム・スタンプ」スイッチが OFF のときは、このエレメントは「0」を報告します。

関連資料:

- 379 ページの『stmt_stop ステートメント操作停止タイム・スタンプ』

start_time イベント開始時刻

エレメント ID start_time

エレメント・タイプ タイム・スタンプ

表 576. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_start	タイム・スタンプ
トランザクション	event_xact	タイム・スタンプ
ステートメント	event_stmt	タイム・スタンプ
デッドロック	event_deadlock	タイム・スタンプ
デッドロック	event_dlconn	タイム・スタンプ
デッドロックの詳細	event_detailed_dlconn	タイム・スタンプ

説明 作業単位開始、ステートメント開始、またはデッドロック検出の日時。

このエレメントは、event_start API 構造内ではイベント・モニターの開始を示します。

使用法 このエレメントを使用すると、デッドロック接続レコードとデッドロック・イベント・レコードを関連付けることができます。 *stop_time* と組み合わせて使用すると、ステートメントの経過時間またはトランザクション実行時間を計算できます。

注: 「タイム・スタンプ」スイッチが OFF のときは、このエレメントは「0」を報告します。

関連資料:

- 374 ページの『stmt_operation/operation ステートメント操作』

stmt_elapsed_time 最新のステートメント経過時間

エレメント ID stmt_elapsed_time

エレメント・タイプ 時間

表 577. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント、タイム・スタンプ
DCS ステートメント	dcs_stmt	ステートメント、タイム・スタンプ

説明 最後に完了したステートメントの実行経過時間。

使用法 ステートメントの完了にかかる時間の標識として、このエレメントを使用します。

関連資料:

- 459 ページの『gw_comm_errors 通信エラー』
- 460 ページの『gw_comm_error_time 通信エラー時刻』

stmt_text SQL 動的ステートメント・テキスト

エレメント ID stmt_text

エレメント・タイプ 情報

表 578. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
動的 SQL	dynsql	基本
DCS ステートメント	dcs_stmt	ステートメント

表 579. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 これは動的 SQL ステートメントのテキストです。

使用法 アプリケーション・スナップショットの場合は、このステートメント・テキストに基づいて、スナップショットを取った時点でアプリケーションが何を実行していたかを識別できます。またスナップショットを取った時点でステートメントが処理されていなかった場合は、最後に処理されたものを識別できます。

このエレメントが戻す情報は、SQL ステートメント・キャッシュから取り出されるので、キャッシュがオーバーフローした場合は情報は得られませ

ん。ステートメントの SQL テキストを必ずキャプチャーするには、ステートメントのイベント・モニターを使用してください。

動的 SQL ステートメントの場合は、このエレメントを使用してパッケージに関連付けられた SQL テキストを識別します。

イベント・モニターの場合は、このエレメントは動的ステートメントの場合に限り戻されます。イベント・モニター・レコードがイベント・モニターの BUFFERSIZE に適合しない場合は、レコードが適合するように *stmt_text* が切り捨てられることがあります。

パフォーマンスを考慮したために静的 SQL ステートメントが提供されない場合、これを取得するためにシステム・カタログ表を照会する方法については、『*section_number*』を参照してください。

関連資料:

- 374 ページの『stmt_operation/operation ステートメント操作』
- 375 ページの『package_name パッケージ名』
- 377 ページの『section_number セクション番号』
- 377 ページの『cursor_name カーソル名』
- 378 ページの『creator アプリケーション作成者』
- 404 ページの『input_db_alias 入力データベース別名』
- 376 ページの『package_version_id パッケージ・バージョン』

stmt_sorts ステートメント・ソート回数

エレメント ID stmt_sorts

エレメント・タイプ カウンター

表 580. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ステートメント
アプリケーション	stmt	ステートメント
動的 SQL	dynsql	ステートメント

説明 stmt_operation を処理するためにデータ集合がソートされた合計回数。

使用法 このエレメントを使用すると、索引が必要かどうかを識別できます。索引があればデータをソートする必要性を少なくできるからです。上記の表の関連エレメントを使用すると、このエレメントがソート情報を提供している SQL ステートメントを識別できます。次にこのステートメントを分析し、ソート対象の列を見ると索引候補を判別できます (例えば、 ORDER BY および GROUP BY 文節に使用されている列、および結合列)。ソート効率を最適化するために索引が使用されているかどうかを確認する方法については、「*管理ガイド*」の『**EXPLAIN**』を参照してください。

このカウントには、ステートメントを実行するためにデータベース・マネージャーが内部的に生成する一時表のソートが含まれます。ソート数は、SQL ステートメントの最初の FETCH 操作と関連しています。この情報は、ステートメントの操作が最初の FETCH の場合にユーザーに戻されます。ブロッ

ク・カーソルの場合は、カーソルが開いたときに複数の取り出しが行われるので注意してください。このような場合、 DB2 が最初の FETCH を内部で発行している間にスナップショットをとる必要があるため、スナップショット・モニターを使用してソート回数を取得するのは困難になります。

ブロック・カーソルを使用して実行されたソートの数を確認するより確実な方法としては、ステートメントに宣言されたイベント・モニターを使用する方法があります。 CLOSE カーソルのステートメント・イベントにある total_sorts カウンターには、カーソルが定義されたステートメントを実行したときに実行されるソートの合計回数が含まれています。

関連資料:

- 201 ページの『total_sorts ソート合計』

fetch_count 成功した取り出しの数

エレメント ID fetch_count

エレメント・タイプ カウンター

表 581. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

表 582. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_stmt	-

説明 stmt スナップショット・モニター・レベルでありイベント・タイプがステートメントの場合は、特定のカーソル上で実行されて成功した取り出しの数。

dcs_stmt スナップショット・モニター・レベルの場合は、 (アプリケーションによって取り出された行数にかかわらず) ステートメントの実行時に試みられた物理取り出しの数。つまり fetch_count は、ステートメントの処理中にサーバーがゲートウェイに対して応答データを送り返す必要があった回数を表します。

使用法 このエレメントを使用すると、データベース・マネージャー内の現在のアクティビティーのレベルがわかります。

ステートメント・イベント・モニターは、パフォーマンス上の理由から、すべての FETCH ステートメントを対象にステートメント・イベント・レコードを生成するわけではありません。レコード・イベントが生成されるのは、FETCH がゼロ以外の SQLCODE を戻した場合だけです。

関連資料:

- 373 ページの『stmt_type ステートメント・タイプ』
- 374 ページの『stmt_operation/operation ステートメント操作』
- 377 ページの『cursor_name カーソル名』

- 379 ページの『stmt_start ステートメント操作開始タイム・スタンプ』
- 379 ページの『stmt_stop ステートメント操作停止タイム・スタンプ』

sqlca SQL 連絡域 (SQLCA)

エレメント ID sqlca

エレメント・タイプ 情報

表 583. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_stmt	-

説明 ステートメントの完了時にアプリケーションに戻された SQLCA データ構造体。

使用法 SQLCA データ構造体を使用すると、ステートメントが正常に終了したかどうかを判別できます。 SQLCA の内容については、「*SQL リファレンス*」または「*管理 API リファレンス*」を参照してください。

関連資料:

- 374 ページの『stmt_operation/operation ステートメント操作』

query_card_estimate 行数見積もりの照会

エレメント ID query_card_estimate

エレメント・タイプ 情報

表 584. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

説明 照会によって戻される行数の見積もり。

使用法 SQL コンパイラーによるこの見積もりは、ランタイムの実際のものと比較できます。

このエレメントは、DB2 Connect をモニター中は次の SQL ステートメントに関する情報も戻します。

- INSERT、UPDATE、および DELETE

  影響を受ける行数を示します。

- PREPARE

  戻される行数の見積もり。 DRDA サーバーが DB2 Universal Database、DB2 for VM/VSE、または DB2 for OS/400 の場合にのみ収集します。

- FETCH

  取り出された行数に設定されます。 DRDA サーバーが DB2 for OS/400 の場合にのみ収集します。

DRDA サーバーで情報が収集されないと、エレメントはゼロに設定されます。

関連資料:

• 385 ページの『query_cost_estimate 照会コストの見積もり』

query_cost_estimate 照会コストの見積もり

エレメント ID query_cost_estimate

エレメント・タイプ 情報

表 585. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

説明 SQL コンパイラーによって判別された照会コストの見積もり (単位は timeron)。

使用法 これにより、ランタイムの値とコンパイル時の見積もりの相関をとることができます。

このエレメントは、DB2 Connect をモニター中は次の SQL ステートメントに関する情報も戻します。

- PREPARE

  準備済み SQL ステートメントの相対コストを示します。

- FETCH

  取り出した行の長さが含まれています。 DRDA サーバーが DB2 for OS/400 の場合にのみ収集します。

DRDA サーバーで情報が収集されないと、エレメントはゼロに設定されます。

注: DRDA サーバーが DB2 for OS/390 and z/OS の場合は、この見積もりが 2**32 - 1 よりも大きい値になることがあります (符号なしの長形式変数を使用して表現できる最大整数)。この場合は、モニターがこのエレメントに戻す値は 2**32 - 1 になります。

サブセクションの詳細

サブセクション詳細に関するモニター・エレメント

パーティション・データベースに対してステートメントを実行すると、ステートメントがサブセクションに分割されて、異なるパーティションで実行されます。 1 つのアプリケーションに含まれる複数のサブセクションが 1 つのパーティションで同時に実行される場合もあります。

問題判別のために、問題のあるサブセクションの場所を突き止める必要があります。例えば、表キューのライターの 1 つがほかのノード上でロック待機をしているために、サブセクションがその表キュー上で待機している場合があります。 1 つの

アプリケーションについて全体像を得るためには、そのアプリケーションを実行している各ノードでアプリケーションのスナップショットをとる必要があります。

次のデータベース・システム・モニター・エレメントにより、サブセクションに関する情報が提供されます。
- ss_number サブセクション番号 : モニター・エレメント
- ss_node_number サブセクション・ノード番号 : モニター・エレメント
- ss_status サブセクションの状況 : モニター・エレメント
- ss_exec_time サブセクション実行経過時間 : モニター・エレメント
- tq_wait_for_any 表キュー上のノード送信待機 : モニター・エレメント
- tq_node_waited_for 表キュー上のノード待機 : モニター・エレメント
- tq_tot_send_spills オーバーフローした表キュー・バッファーの合計数 : モニター・エレメント
- tq_cur_send_spills オーバーフローした表キュー・バッファーの現在数 : モニター・エレメント
- tq_rows_read 表キューから読み取られた行数 : モニター・エレメント
- tq_rows_written 表キューに書き込まれた行数 : モニター・エレメント
- tq_max_send_spills 表キュー・バッファー・オーバーフローの最大数 : モニター・エレメント
- tq_id_waiting_on ノード上の表キュー待機 : モニター・エレメント

ss_number サブセクション番号

エレメント ID ss_number

エレメント・タイプ 情報

表 586. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 587. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 戻された情報に関連したサブセクションを示します。

使用法 この数値は、db2expln を使用して取得可能なアクセス・プラン内のサブセクションに関連しています。

ss_node_number サブセクション・ノード番号

エレメント ID ss_node_number

エレメント・タイプ 情報

表 588. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 589. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 サブセクションが実行されたノード。

使用法 各サブセクションとそれが実行されたデータベース・パーティションを関連付けるために使用します。

ss_status サブセクションの状況

エレメント ID ss_status

エレメント・タイプ 情報

表 590. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

説明 実行中のサブセクションの現在の状況。

使用法 現在の状況の値として、次のものがあります。

- 実行中 (sqlmon.h の SQLM_SSEXEC)
- ロック待ち
- 表キュー (tablequeue) でデータの受信待ち
- 表キュー (tablequeue) でデータの送信待ち

関連資料:

- 387 ページの『tq_wait_for_any 表キュー上のノード送信待機』
- 388 ページの『tq_node_waited_for 表キュー上のノード待機』

ss_exec_time サブセクション実行経過時間

エレメント ID ss_exec_time

エレメント・タイプ カウンター

表 591. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 592. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 サブセクションの実行に要した時間 (秒数)。

使用法 サブセクションの進行状況を追跡することができます。

tq_wait_for_any 表キュー上のノード送信待機

エレメント ID tq_wait_for_any

エレメント・タイプ 情報

表 593. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

説明 このフラグは、サブセクションがノードからの行の受信を待っているためにサブセクションがブロックされていることを示すのに使用されます。

使用法 ss_status が「*表キュー上でデータの受信待ち*」を示し、このフラグが TRUE の場合は、サブセクションはノードからの行の受信を待機中です。この場合は通常、 SQL ステートメントが待機中のエージェントにデータを渡すところまでまだ処理が進んでいないことを示します。例えば、書き込みエージェントがソートを実行中で、ソートが終了するまでは行を書き込まない場合があります。 db2expln の出力を使用して、エージェントが行の受信を待機している表キューに関連付けられたサブセクション番号を判別します。次に、サブセクションを実行している各ノードでスナップショットをとると、サブセクションの状況を確認できます。

関連資料:

- 387 ページの『ss_status サブセクションの状況』
- 388 ページの『tq_node_waited_for 表キュー上のノード待機』

tq_node_waited_for 表キュー上のノード待機

エレメント ID tq_node_waited_for

エレメント・タイプ 情報

表 594. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

説明 サブセクション状況の ss_status が「*受信待ち*」または「*送信待ち*」で、tq_wait_for_any が FALSE の場合は、エージェントが待機中のノード番号を示します。

使用法 これはトラブルシューティングに使用できます。サブセクションが待機しているノード上でアプリケーションのスナップショットが必要になる場合があります。例えば、アプリケーションがそのノード上でロック待機になっている場合です。

関連資料:

- 387 ページの『ss_status サブセクションの状況』
- 387 ページの『tq_wait_for_any 表キュー上のノード送信待機』

tq_tot_send_spills オーバーフローした表キュー・バッファーの合計数

エレメント ID tq_tot_send_spills

エレメント・タイプ カウンター

表 595. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 596. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 一時表にオーバーフローした表キュー・バッファーの合計数。

使用法 一時表に書き込まれた表キュー・バッファーの合計数を示します。詳しくは、『tq_cur_send_spills』を参照してください。

関連資料:

- 387 ページの『ss_status サブセクションの状況』
- 389 ページの『tq_cur_send_spills オーバーフローした表キュー・バッファーの現在数』

tq_cur_send_spills オーバーフローした表キュー・バッファーの現在数

エレメント ID tq_cur_send_spills

エレメント・タイプ ゲージ

表 597. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

説明 一時表内にある表キュー・バッファーの現在の数。

使用法 表キューへの書き込みを行っているエージェントは、複数のリーダーに行を送信している可能性があります。書き込みエージェントが行を送信したときに、相手のエージェントが行を受け付けず、別のエージェントが処理のために行を必要とすると、書き込みエージェントはバッファーを一時表にオーバーフローします。一時表にオーバーフローすると、ライターとほかのリーダーがともに処理を続行できるようになります。

オーバーフローした行は、読み取りエージェントが行を受け付ける準備ができると読み取りエージェントに送信されます。

この数値が大きく、照会が sqlcode -968 で失敗し、さらに *db2diad.log* 内のメッセージにより TEMP 表スペースの一時スペースがなくなったことを示す場合は、表キューのオーバーフローが原因の可能性があります。この場合、ほかのノードで問題が発生していることを示します (ロッキングなど)。この照会のすべてのパーティション上でスナップショットを取って、調査してください。

データをパーティション化する方法が原因で、照会によって多数のバッファーをオーバーフローすることが必要になる場合もあります。このような原因がある場合は、 TEMPORARY 表スペースにさらにディスクを追加する必要があります。

関連資料:

- 387 ページの『ss_status サブセクションの状況』
- 388 ページの『tq_tot_send_spills オーバーフローした表キュー・バッファーの合計数』

tq_rows_read 表キューから読み取られた行数

エレメント ID tq_rows_read

エレメント・タイプ カウンター

表 598. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 599. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 表キューから読み取られた合計行数。

使用法 この数が増加していることをモニターが示していない場合は、処理は進行していません。

ノードごとにこの数値が大きく異なる場合は、使用率が過剰なノードと過少なノードがあります。

この数値が大きい場合は、ノード間で転送されるデータ量が多く、最適化によりアクセス・プランを改善できることを示します。

tq_rows_written 表キューに書き込まれた行数

エレメント ID tq_rows_written

エレメント・タイプ カウンター

表 600. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 601. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 表キューに書き込まれた合計行数。

使用法 この数が増加していることをモニターが示していない場合は、処理は進行していません。

ノードごとにこの数値が大きく異なる場合は、使用率が過剰なノードと過少なノードがあります。

この数値が大きい場合は、ノード間で転送されるデータ量が多く、最適化によりアクセス・プランを改善できることを示します。

tq_max_send_spills 表キュー・バッファー・オーバーフローの最大数

エレメント ID tq_max_send_spills

エレメント・タイプ 水準点

表 602. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

表 603. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	-

説明 一時表にオーバーフローした表キュー・バッファーの最大数。

使用法 一時表に書き込まれた表キュー・バッファーの最大数を示します。

関連資料:

- 388 ページの『tq_tot_send_spills オーバーフローした表キュー・バッファーの合計数』
- 389 ページの『tq_cur_send_spills オーバーフローした表キュー・バッファーの現在数』

tq_id_waiting_on ノード上の表キュー待機

エレメント ID tq_id_waiting_on

エレメント・タイプ 情報

表 604. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	ステートメント

説明 待機中のエージェント。

使用法 これはトラブルシューティングに使用できます。

関連資料:

- 387 ページの『ss_status サブセクションの状況』
- 388 ページの『tq_node_waited_for 表キュー上のノード待機』

動的 SQL

動的 SQL に関するモニター・エレメント

DB2 ステートメント・キャッシュは、使用頻度の高い SQL ステートメントについてパッケージと統計値を保管します。このキャッシュの内容を調べることにより、使用頻度の高い動的 SQL ステートメントとリソースを大量に消費する照会を識別できます。この情報を使用すると、実行頻度が高くコストがかかっている SQL 操作を調べ、 SQL のチューニングによってデータベースのパフォーマンスを改善できるかどうか判別できます。

- num_executions ステートメント実行回数 : モニター・エレメント
- num_compilations ステートメント・コンパイル数 : モニター・エレメント
- prep_time_worst ステートメント最長準備時間 : モニター・エレメント
- prep_time_best ステートメント最短準備時間 : モニター・エレメント
- total_exec_time ステートメント実行の経過時間 : モニター・エレメント

num_executions ステートメント実行回数

エレメント ID num_executions

エレメント・タイプ カウンター

表 605. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 SQL ステートメントが実行された回数。

使用法 このエレメントを使用して、システムで最も頻繁に実行された SQL ステートメントを識別できます。

関連資料:

- 392 ページの『num_compilations ステートメント・コンパイル数』

num_compilations ステートメント・コンパイル数

エレメント ID num_compilations

エレメント・タイプ カウンター

表 606. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 特定の SQL ステートメントに対する異なるコンパイルの数。

使用法 異なるスキーマで発行された SQL ステートメント (″select t1 from foo″ など) の中には、それらが異なるアクセス・プランを参照する場合でも DB2 キャッシュ内では同じステートメントと見なされるものがあります。この値

と num_executions を組み合わせて使用すると、コンパイル環境に問題があるために動的 SQL スナップショット統計の結果に狂いが生じていないかどうかを判別できます。

関連資料:

- 392 ページの『num_executions ステートメント実行回数』

prep_time_worst ステートメント最長準備時間

エレメント ID prep_time_worst

エレメント・タイプ 水準点

表 607. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	基本

説明 特定の SQL ステートメントの準備に要した最長時間 (マイクロ秒単位)。

使用法 この値を prep_time_best とともに使用して、コンパイルに長い時間がかかる SQL ステートメントを識別します。

関連資料:

- 393 ページの『prep_time_best ステートメント最短準備時間』

prep_time_best ステートメント最短準備時間

エレメント ID prep_time_best

エレメント・タイプ 水準点

表 608. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	基本

説明 特定の SQL ステートメントの準備に要した最短時間。

使用法 この値を prep_time_worst とともに使用して、コンパイルに長い時間がかかる SQL ステートメントを識別します。

関連資料:

- 393 ページの『prep_time_worst ステートメント最長準備時間』

total_exec_time ステートメント実行の経過時間

エレメント ID total_exec_time

エレメント・タイプ 時間

表 609. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 SQL キャッシュ内の特定のステートメントの実行に要した合計時間 (秒およびマイクロ秒単位)。

使用法 このエレメントを num_executions とともに使用して、ステートメントの平均経過時間を判別し、 SQL の調整によって最も望ましい影響を受ける SQL ステートメントを識別します。このエレメントの内容を評価する際は、num_compilation も考慮する必要があります。

関連資料:

- 392 ページの『num_executions ステートメント実行回数』
- 392 ページの『num_compilations ステートメント・コンパイル数』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

照会内並列処理

照会内並列処理に関するモニター・エレメント

次のデータベース・システム・モニター・エレメントにより、並列処理の度合いが 1 を超える照会についての情報が提供されます。

- num_agents ステートメントで作動しているエージェントの数 : モニター・エレメント
- agents_top 作成されたエージェントの数 : モニター・エレメント
- degree_parallelism 並列処理の度合い : モニター・エレメント

num_agents ステートメントで作動しているエージェントの数

エレメント ID num_agents

エレメント・タイプ ゲージ

表 610. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
アプリケーション	subsection	ステートメント

説明 現在ステートメントまたはサブセクションを実行している並行エージェントの数。

使用法 照会の並列処理の度合いを示します。スナップショットを連続的にとることによって、照会の実行進行状況を追跡するときに役に立ちます。

関連資料:

- 394 ページの『agents_top 作成されたエージェントの数』
- 395 ページの『degree_parallelism 並列処理の度合い』

agents_top 作成されたエージェントの数

エレメント ID agents_top

エレメント・タイプ 水準点

表 611. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	ステートメント
アプリケーション	stmt	ステートメント

説明 アプリケーション・レベルでは、ステートメントの実行時に使用されたエージェントの最大数です。データベース・レベルでは、これはすべてのアプリケーション用のエージェントの最大数です。

使用法 照会内並列処理の実現の度合いを示します。

関連資料:

- 394 ページの『num_agents ステートメントで作動しているエージェントの数』
- 395 ページの『degree_parallelism 並列処理の度合い』

degree_parallelism 並列処理の度合い

エレメント ID degree_parallelism

エレメント・タイプ 情報

表 612. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント

説明 照会がバインドされたときに要求された並列処理の度合い。

使用法 agents_top と組み合わせて使用すると、照会で最大レベルの並列処理ができたかどうかを判別できます。

関連資料:

- 394 ページの『num_agents ステートメントで作動しているエージェントの数』
- 394 ページの『agents_top 作成されたエージェントの数』

CPU 使用状況

CPU 使用量に関するモニター・エレメント

1 つのアプリケーションに関する CPU 使用量は、**ユーザー CPU** (アプリケーション・コードを実行するときの CPU 使用量) と**システム** CPU (システム呼び出しを実行するときの CPU 使用量) に分けることができます。

CPU 使用量は、アプリケーション、トランザクション、ステートメント、およびサブセクションの各レベルで利用できます。

- agent_usr_cpu_time エージェントが使用したユーザー CPU 時間 : モニター・エレメント
- agent_sys_cpu_time エージェントが使用したシステム CPU 時間 : モニター・エレメント
- stmt_usr_cpu_time ステートメントに使用されたユーザー CPU 時間 : モニター・エレメント

- stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間 : モニター・エレメント
- user_cpu_time ユーザー CPU 時間 : モニター・エレメント
- system_cpu_time システム CPU 時間 : モニター・エレメント
- ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間 : モニター・エレメント
- ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間 : モニター・エレメント
- total_sys_cpu_time ステートメントのシステム CPU の合計 : モニター・エレメント
- total_usr_cpu_time ステートメントのユーザー CPU の合計 : モニター・エレメント

agent_usr_cpu_time エージェントが使用したユーザー CPU 時間

エレメント ID agent_usr_cpu_time

エレメント・タイプ 時間

表 613. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 データベース・マネージャーのエージェント・プロセスで使用された CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントとその他の CPU 時間関連のエレメントを組み合わせて使用すると、 CPU を大量に使用しているアプリケーションや照会を識別できます。

このカウンターには、SQL および 非 SQL のステートメントに要した時間のほか、アプリケーションが実行した unfenced ユーザー定義関数 (UDF) およびストアード・プロシージャーも含まれます。

システム CPU は、システム呼び出しに要した時間を示します。ユーザー CPU は、データベース・マネージャーのコードを実行するのに要した時間を示します。

注: 使用しているオペレーティング・システムでこの情報を得られない場合は、このエレメントはゼロ (0) を戻します。

関連資料:

- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』
- 400 ページの『system_cpu_time システム CPU 時間』

- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

agent_sys_cpu_time エージェントが使用したシステム CPU 時間

エレメント ID agent_sys_cpu_time

エレメント・タイプ 時間

表 614. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	タイム・スタンプ

アプリケーション・レベルでのスナップショット・モニターの場合、このカウンターはリセットできます。その他のレベルではこのカウンターはリセットできません。

説明 データベース・マネージャーのエージェント・プロセスで使用されたシステム CPU 時間の合計 (秒単およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

このエレメントには、SQL および 非 SQL のステートメントの両者の CPU 時間、およびその他の unfenced ユーザー定義関数 (UDF) の CPU 時間が含まれます。

システム CPU は、システム呼び出しに要した時間を示します。ユーザー CPU は、データベース・マネージャーのコードを実行するのに要した時間を示します。

注: ユーザーのオペレーティング・システムでこの情報が利用できないとき、このエレメントは 0 に設定されます。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』
- 400 ページの『system_cpu_time システム CPU 時間』
- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』

- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間

エレメント ID stmt_usr_cpu_time

エレメント・タイプ 時間

表 615. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ステートメント、タイム・スタンプ
アプリケーション	stmt	ステートメント、タイム・スタンプ

説明 現在実行中のステートメントによって使用されたユーザー CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

このカウンターには、SQL および 非 SQL のステートメントに要した時間のほか、アプリケーションが実行した unfenced ユーザー定義関数 (UDF) およびストアード・プロシージャーも含まれます。

システム CPU は、システム呼び出しに要した時間を示します。ユーザー CPU は、データベース・マネージャーのコードを実行するのに要した時間を示します。

注: ユーザーのオペレーティング・システムでこの情報が利用できないとき、このエレメントは 0 に設定されます。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』
- 400 ページの『system_cpu_time システム CPU 時間』
- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間

エレメント ID stmt_sys_cpu_time

エレメント・タイプ 時間

表 616. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl	ステートメント、タイム・スタンプ
アプリケーション	stmt	ステートメント、タイム・スタンプ

説明 現在実行中のステートメントによって使用されたシステム CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

このカウンターには、SQL および 非 SQL のステートメントに要した時間のほか、アプリケーションが実行した unfenced ユーザー定義関数 (UDF) およびストアード・プロシージャーも含まれます。

システム CPU は、システム呼び出しに要した時間を示します。ユーザー CPU は、データベース・マネージャーのコードを実行するのに要した時間を示します。

注: ユーザーのオペレーティング・システムでこの情報が利用できないとき、このエレメントは 0 に設定されます。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』
- 400 ページの『system_cpu_time システム CPU 時間』
- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

user_cpu_time ユーザー CPU 時間

エレメント ID user_cpu_time

エレメント・タイプ 時間

表 617. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
トランザクション	event_xact	-
ステートメント	event_stmt	-

説明 データベース・マネージャーのエージェント・プロセス、作業単位、またはステートメントで使用されたユーザー CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

注: ユーザーのオペレーティング・システムでこの情報が利用できないとき、このエレメントは 0 に設定されます。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 400 ページの『system_cpu_time システム CPU 時間』
- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

system_cpu_time システム CPU 時間

エレメント ID system_cpu_time

エレメント・タイプ 時間

表 618. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
接続	event_conn	-
トランザクション	event_xact	-
ステートメント	event_stmt	-

説明 データベース・マネージャーのエージェント・プロセス、作業単位、またはステートメントで使用されたシステム CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

注: ユーザーのオペレーティング・システムでこの情報が利用できないとき、このエレメントは 0 に設定されます。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』
- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間

エレメント ID ss_usr_cpu_time

エレメント・タイプ 時間

表 619. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	タイム・スタンプ

表 620. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	タイム・スタンプ

説明 現在実行中のステートメント・サブセクションによって使用されたユーザー CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

システム CPU は、システム呼び出しに要した時間を示します。ユーザー CPU は、データベース・マネージャーのコードを実行するのに要した時間を示します。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』
- 400 ページの『system_cpu_time システム CPU 時間』
- 402 ページの『ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

ss_sys_cpu_time サブセクションに使用されたシステム CPU 時間

エレメント ID ss_sys_cpu_time

エレメント・タイプ 時間

表 621. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	subsection	タイム・スタンプ

表 622. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_subsection	タイム・スタンプ

説明 現在実行中のステートメント・サブセクションによって使用されたシステム CPU 時間の合計 (秒およびマイクロ秒単位)。

使用法 このエレメントと CPU 時間に関連する他のエレメントを組み合わせて使用すると、アプリケーション内のアクティビティーのレベルがわかります。また、さらに調整するとその効果が得られる可能性があるアプリケーションを識別できます。

システム CPU は、システム呼び出しに要した時間を示します。ユーザー CPU は、データベース・マネージャーのコードを実行するのに要した時間を示します。

関連資料:

- 396 ページの『agent_usr_cpu_time エージェントが使用したユーザー CPU 時間』
- 397 ページの『agent_sys_cpu_time エージェントが使用したシステム CPU 時間』
- 398 ページの『stmt_usr_cpu_time ステートメントが使用したユーザー CPU 時間』
- 399 ページの『stmt_sys_cpu_time ステートメントが使用したシステム CPU 時間』
- 399 ページの『user_cpu_time ユーザー CPU 時間』

- 401 ページの『ss_usr_cpu_time サブセクションに使用されたユーザー CPU 時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

total_sys_cpu_time ステートメントのシステム CPU の合計

エレメント ID total_sys_cpu_time

エレメント・タイプ 時間

表 623. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 SQL ステートメントに使われたシステム CPU 時間の合計。

使用法 このエレメントをステートメント実行の経過時間およびステートメントのユーザー CPU の合計とともに使用して、最もコストがかかるステートメントを評価します。

関連資料:

- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 403 ページの『total_usr_cpu_time ステートメントのユーザー CPU の合計』

total_usr_cpu_time ステートメントのユーザー CPU の合計

エレメント ID total_usr_cpu_time

エレメント・タイプ 時間

表 624. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
動的 SQL	dynsql	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 SQL ステートメントに使われたユーザー CPU 時間の合計。

使用法 このエレメントをステートメント実行の経過時間とともに使用して、最も実行時間が長いステートメントを評価します。

関連資料:

- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 403 ページの『total_sys_cpu_time ステートメントのシステム CPU の合計』

スナップショット・モニター

スナップショット・モニターに関するモニター・エレメント

次のエレメントにより、アプリケーションのモニターに関する情報が提供されます。これらエレメントは、各スナップショットの出力として戻されます。

- last_reset 最後のリセット・タイム・スタンプ : モニター・エレメント

- input_db_alias 入力データベース別名 : モニター・エレメント
- time_stamp スナップショット時刻 : モニター・エレメント
- num_nodes_in_db2_instance パーティション内のノード数 : モニター・エレメント

last_reset 最後のリセット・タイム・スタンプ

エレメント ID last_reset

エレメント・タイプ タイム・スタンプ

表 625. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	タイム・スタンプ
データベース	dbase	タイム・スタンプ
アプリケーション	appl	タイム・スタンプ
表スペース	tablespace_list	バッファー・プール、タイム・スタンプ
表	table_list	タイム・スタンプ
DCS データベース	dcs_dbase	タイム・スタンプ
DCS アプリケーション	dcs_appl	タイム・スタンプ

説明 GET SNAPSHOT を発行するアプリケーションのモニター・カウンターがリセットされた日時を示します。

使用法 このエレメントを使用すると、データベース・システム・モニターによって戻された情報の有効範囲を判別できます。

カウンターがこれまでにリセットされたことがない場合は、このエレメントはゼロになります。

データベース・マネージャーのカウンターがリセットされるのは、ユーザーがすべてのアクティブ・データベースをリセットしたときだけです。

関連資料:

- 404 ページの『input_db_alias 入力データベース別名』

input_db_alias 入力データベース別名

エレメント ID input_db_alias

エレメント・タイプ 情報

表 626. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	基本
アプリケーション	appl_id_info	基本
表スペース	tablespace_list	バッファー・プール
バッファー・プール	bufferpool	バッファー・プール
表	table_list	表
ロック	db_lock_list	基本

説明 スナップショット関数を呼び出すときに指定するデータベースの別名。

使用法 このエレメントを使用すると、モニター・データが適用される特定のデータベースを識別できます。特定のデータベースに関連するモニター情報を要求したとき以外は、このエレメントはブランクとなります。

1 つのデータベースが複数の別名を持つことがあるので、このフィールドの値と *client_db_alias* モニター・エレメントの値とが異なる場合があります。複数のアプリケーションやユーザーが異なる別名を使用して、同じデータベースに接続することができます。

関連資料:

- 164 ページの『client_db_alias アプリケーションで使用するデータベース別名』
- 404 ページの『last_reset 最後のリセット・タイム・スタンプ』

time_stamp スナップショット時刻

エレメント ID time_stamp

エレメント・タイプ タイム・スタンプ

表 627. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	collected	基本

説明 データベース・システム・モニター情報が収集された日時。

使用法 このエレメントを使用すると、継続的な分析作業でその結果をファイルやデータベースに保管してある場合は、データを時系列的に関連付けることができます。

num_nodes_in_db2_instance パーティション内のノード数

エレメント ID num_nodes_in_db2_instance

エレメント・タイプ 情報

表 628. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本

表 629. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-

説明 スナップショットが取られたインスタンス上のノード数。

使用法 このエレメントを使用して、インスタンスにおけるノード数を判別します。非パーティション・システムのデータベースの場合、この値は 1 になります。

イベント・モニター

イベント・モニターに関するモニター・エレメント

次のエレメントにより、アプリケーションのモニターに関する情報が提供されます。これらエレメントは、イベントの出力として戻されます。

- count イベント・モニター・オーバーフロー数 : モニター・エレメント
- first_overflow_time 最初のイベント・オーバーフロー時刻 : モニター・エレメント
- last_overflow_time 最後のイベント・オーバーフロー時刻 : モニター・エレメント
- byte_order イベント・データのバイト・オーダー : モニター・エレメント
- version モニター・データのバージョン : モニター・エレメント
- event_monitor_name イベント・モニター名 : モニター・エレメント
- partial_record 部分レコード : モニター・エレメント
- event_time イベント時刻 : モニター・エレメント
- evmon_flushes イベント・モニター・フラッシュ回数 : モニター・エレメント
- evmon_activates イベント・モニター活動化回数 : モニター・エレメント
- sql_req_id SQL ステートメントの要求 ID : モニター・エレメント
- message コントロール表メッセージ : モニター・エレメント
- message_time タイム・スタンプ・コントロール表メッセージ : モニター・エレメント
- partition_number パーティション番号 : モニター・エレメント

count イベント・モニター・オーバーフロー数

エレメント ID count

エレメント・タイプ カウンター

表 630. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
オーバーフロー・レコード	event_overflow	-

説明 連続して発生したオーバーフローの数。

使用法 このエレメントを使用すると、失われたモニター・データの量がわかります。

イベント・モニターは、一連のオーバーフローについて 1 つのオーバーフロー・レコードを送信します。

first_overflow_time 最初のイベント・オーバーフロー時刻

エレメント ID first_overflow_time

エレメント・タイプ タイム・スタンプ

表 631. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
オーバーフロー・レコード	event_overflow	-

説明 このオーバーフロー・レコードに記録されている最初のオーバーフローの日時。

使用法 このエレメントと *last_over_flow time* を組み合わせて使用すると、オーバーフロー・レコードを生成するのに要した経過時間を計算できます。

関連資料:

• 406 ページの『count イベント・モニター・オーバーフロー数』

last_overflow_time 最後のイベント・オーバーフロー時刻

エレメント ID last_overflow_time

エレメント・タイプ タイム・スタンプ

表 632. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
オーバーフロー・レコード	event_overflow	-

説明 このオーバーフロー・レコードに記録されている最後のオーバーフローの日時。

使用法 このエレメントと *first_overflow_time* を組み合わせて使用すると、オーバーフロー・レコードを生成するのに要した経過時間を計算できます。

関連資料:

• 406 ページの『count イベント・モニター・オーバーフロー数』

byte_order イベント・データのバイト・オーダー

エレメント ID byte_order

エレメント・タイプ 情報

表 633. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-

説明 数値データのバイト配列。具体的には、イベント・データ・ストリームが「ビッグ・エンディアン」サーバー (RISC System/6000 など) で生成されたか、あるいは『リトル・エンディアン』サーバー (Windows 2000 が稼働する Intel 系 PC) で生成されたかを示します。

使用法 「ビッグ・エンディアン」サーバーの整数のバイト・オーダーは、「リトル・エンディアン」サーバーのバイト・オーダーとは逆になるため、データ・ストリームの数値データを解釈するときにこの情報が必要になります。

データを処理するアプリケーションが一方のタイプのコンピューター・ハードウェア上で実行していることを認識し (例えば、ビッグ・エンディアン・コンピューター)、イベント・データがもう一方のタイプのコンピューター・ハードウェア上で生成された場合 (例えば、リトル・エンディアン・コンピューター)、モニター・アプリケーションはこれらのデータを解釈する

前に数値データ・フィールドのバイトを逆転する必要があります。タイプが同じ場合は、バイトの再配列は必要ありません。

このエレメントは、次のいずれかの API 定数に設定できます。

- SQLM_BIG_ENDIAN
- SQLM_LITTLE_ENDIAN

version モニター・データのバージョン

エレメント ID version

エレメント・タイプ 情報

表 634. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-

説明 イベント・モニター・データ・ストリームを作成したデータベース・マネージャーのバージョン。

使用法 イベント・モニターが使用するデータ構造は、データベース・マネージャーのリリースによって異なっている場合があります。そのため、モニター・アプリケーションは、受信するデータを処理できるかどうかを判別するために、データ・ストリームのバージョンを確認する必要があります。

このリリースでは、このエレメントは API 定数の SQLM_DBMON_VERSION8 に設定されています。

event_monitor_name イベント・モニター名

エレメント ID event_monitor_name

エレメント・タイプ 情報

表 635. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
イベント・ログ・ヘッダー	event_log_header	-

説明 イベント・データ・ストリームを作成したイベント・モニターの名前。

使用法 このエレメントを使用すると、分析中のデータとシステム・カタログ表内の特定のイベント・モニターを関連付けることができます。この名前は、SYSCAT.EVENTMONITORS カタログ表の NAME 列にある名前 (CREATE EVENT MONITOR および SET EVENT MONITOR のステートメントに指定されている) と同じものです。

partial_record 部分レコード

エレメント ID partial_record

エレメント・タイプ 情報

表 636. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-

表 636. イベント・モニター情報 (続き)

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表	event_table	-
表スペース	event_tablespace	-
バッファー・プール	event_bufferpool	-
接続	event_conn	-
ステートメント	event_stmt	-
ステートメント	event_subsection	-
トランザクション	event_xact	-

説明 イベント・モニター・レコードが部分レコードでしかないことを示します。

使用法 ほとんどのイベント・モニターは、データベースが非活動状態になるまでそれぞれの結果を出力しません。 FLUSH EVENT MONITORS ステートメントを使用すると、モニター値をイベント・モニター出力ライターに強制的に渡すことができます。これにより、イベント・モニターを停止して再始動する必要なしに、イベント・モニター・レコードをライターに強制的に渡すことができます。このエレメントは、イベント・モニター・レコードがフラッシュ操作の結果であり、したがってそれが部分レコードであるかどうかを示します。

イベント・モニターのフラッシュが原因で値がリセットされることはありません。これは、イベント・モニターが起動するときに、完全イベント・モニター・レコードが生成されることを意味します。

event_time イベント時刻

エレメント ID event_time

エレメント・タイプ 情報

表 637. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
表スペース	event_tablespace	-
表	event_table	-

説明 イベントが発生した日時。

使用法 このエレメントは、イベントを時系列順に関連付けるのに利用できます。

evmon_flushes イベント・モニター・フラッシュ回数

エレメント ID evmon_flushes

エレメント・タイプ 情報

表 638. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表	event_table	-
表スペース	event_tablespace	-

表 638. イベント・モニター情報 (続き)

イベント・タイプ	論理データ・グループ	モニター・スイッチ
バッファー・プール	event_bufferpool	-

説明 FLUSH EVENT MONITOR SQL ステートメントが発行された回数。

使用法 アプリケーションがデータベースに接続した後、データベース・マネージャーが FLUSH EVENT MONITOR SQL 要求を処理するごとに、この ID が増分します。このエレメントを使用すると、データベース、表、表スペース、およびバッファー・プール・データを特定できます。

evmon_activates イベント・モニター活動化回数

エレメント ID evmon_activates

エレメント・タイプ カウンター

表 639. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
データベース	event_db	-
表	event_table	-
表スペース	event_tablespace	-
バッファー・プール	event_bufferpool	-
デッドロック	event_deadlock	-
デッドロック	event_dlconn	-
デッドロックの詳細	event_detailed_dlconn	-

説明 イベント・モニターが活動化された回数。

使用法 このエレメントを使用して、上記のイベント・タイプで戻された情報を関連付けます。このエレメントは、write-to-table イベント・モニターにのみ適用できます。このモニター・エレメントは、ファイルまたはパイプに書き込むイベント・モニター用には保持されていません。

一部のタイプの write-to-table イベント・モニターのみが evmon_activates モニター・エレメントを使用します (このエレメントを使用するイベント・モニター・タイプは、前述の「イベント・モニター情報」の表にリストされています)。それらのイベント・モニターは、活動化時に SYSCAT.EVENTMONITORS カタログ表の evmon_activates 列を更新します。この変更はログに記録されるため、DATABASE CONFIGURATION によって次が表示されます。

```
Database is consistent                                  = NO
```

イベント・モニターが AUTOSTART オプションを使用して作成されている場合、最初のユーザーがデータベースに接続してデータベースを非活動化するために即時に切断すると、ログ・ファイルが作成されます。

sql_req_id SQL ステートメントの要求 ID

エレメント ID sql_req_id

エレメント・タイプ 情報

表 640. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
ステートメント	event_stmt	-

説明 SQL ステートメントでの操作の要求 ID。

使用法 最初のアプリケーションがデータベースに接続した後、データベース・マネージャーが SQL 操作を処理するごとに、この ID が増分します。この値はデータベース全体でユニークであり、ステートメント操作を一意的に識別します。

message コントロール表メッセージ

エレメント ID message

エレメント・タイプ 情報

表 641. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
-	-	-

説明 MESSAGE_TIME 列内のタイム・スタンプの性質。このエレメントは、表書き込みイベント・モニターによってコントロール表でのみ使用されます。

使用法 次の値が使用されます。

FIRST_CONNECT
活動化後のデータベースへの最初の接続の時刻。

EVMON_START
EVMONNAME 列にリストされているイベント・モニターが開始された時刻。

OVERFLOWS(*n*)
バッファー・オーバーフローのために *n* 個のレコードが廃棄されたことを示します。

message_time タイム・スタンプ・コントロール表メッセージ

エレメント ID message_time

エレメント・タイプ タイム・スタンプ

表 642. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
-	-	-

説明 MESSAGE 列に記述されているイベントに対応したタイム・スタンプ。このエレメントは、表書き込みイベント・モニターによってコントロール表でのみ使用されます。

partition_number パーティション番号

エレメント ID partition_number

エレメント・タイプ 情報

表 643. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
-	-	-

説明 このエレメントは、パーティション・データベース環境で、表書き込みイベント・モニターによってコントロール表でのみ使用されます。この値はイベント・モニターのデータが挿入されたパーティションの番号を示します。

高可用性災害時リカバリー

高可用性災害時リカバリーに関するモニター・エレメント

DB2 UDB 高可用性災害時リカバリー (HADR) は、部分的なサイト障害と完全なサイト障害の両方に関する可用性の高い解決方法を備えたデータベース・レプリケーション機能です。

HADR は、1 次データベースというソース・データベースから、スタンバイ・データベースというターゲット・データベースにデータの変更内容を複製して、データ損失に対する保護を行います。 1 次データベースに障害が発生すると、スタンバイ・データベースにフェイルオーバーできます。フェイルオーバーすると、スタンバイ・データベースが新しい 1 次データベースになります。スタンバイ・データベース・サーバーはすでにオンラインになっているので、フェイルオーバーは非常に高速に行うことができ、その結果ダウン時間は最小限に抑えられます。

次のモニター・エレメントを使用すると、 HADR サブシステムの現行の構成と状態を調べることができます。

- hadr_role HADR の役割 : モニター・エレメント
- hadr_state HADR の状態 : モニター・エレメント
- hadr_syncmode HADR 同期モード : モニター・エレメント
- hadr_connect_status HADR 接続状況 : モニター・エレメント
- hadr_connect_time HADR 接続時刻 : モニター・エレメント
- hadr_heartbeat HADR ハートビート : モニター・エレメント
- hadr_local_host HADR ローカル・ホスト : モニター・エレメント
- hadr_local_service HADR ローカル・サービス : モニター・エレメント
- hadr_remote_host HADR リモート・ホスト : モニター・エレメント
- hadr_remote_service HADR リモート・サービス : モニター・エレメント
- hadr_remote_instance HADR リモート・インスタンス : モニター・エレメント
- hadr_timeout HADR タイムアウト : モニター・エレメント
- hadr_primary_log_file HADR 1 次ログ・ファイル : モニター・エレメント
- hadr_primary_log_page HADR 1 次ログ・ページ : モニター・エレメント
- hadr_primary_log_lsn HADR 1 次ログ LSN : モニター・エレメント
- hadr_standby_log_file HADR スタンバイ・ログ・ファイル : モニター・エレメント

- hadr_standby_log_page HADR スタンバイ・ログ・ページ : モニター・エレメント
- hadr_standby_log_lsn HADR スタンバイ・ログ LSN : モニター・エレメント
- hadr_log_gap HADR ログ・ギャップ : モニター・エレメント

関連概念:

- 「*データ・リカバリーと高可用性 ガイドおよびリファレンス*」の『高可用性災害時リカバリーの概要』

hadr_role HADR の役割

エレメント ID hadr_role

エレメント・タイプ 情報

表 644. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 データベースの現在のHADR の役割。このエレメントのデータ・タイプは整数です。このエレメントの値は、以下のいずれかの定数です。

- SQLM_HADR_ROLE_STANDARD: データベースは HADR データベースではありません。
- SQLM_HADR_ROLE_PRIMARY: データベースは1 次 HADR データベースです。
- SQLM_HADR_ROLE_STANDBY: データベースはスタンバイ HADR データベースです。

使用法 このエレメントを使用して、データベースの HADR の役割を判別できます。

hadr_state HADR の状態 : モニター・エレメント

エレメント ID hadr_state

エレメント・タイプ 情報

表 645. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 データベースの現在のHADR の状態。このエレメントのデータ・タイプは整数です。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。データベースが HADR 1 次またはスタンバイの役割の場合は、このエレメントの値は、以下のいずれかの定数です。

- SQLM_HADR_STATE_DISCONNECTED: データベースはそのパートナー・ノードに接続していません。
- SQLM_HADR_STATE_LOC_CATCHUP: データベースはローカル・キャッチアップを行っています。
- SQLM_HADR_STATE_REM_CATCH_PEND: データベースはそのパートナーに接続してリモート・キャッチアップを行うのを待っています。

- SQLM_HADR_STATE_REM_CATCHUP: データベースはリモート・キャッチアップを行っています。
- SQLM_HADR_STATE_PEER: 1 次データベースとスタンバイ・データベースが接続され、対等な状態になっています。

使用法 このエレメントを使用して、データベースの HADR の状態を判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_syncmode HADR 同期モード : モニター・エレメント

エレメント ID hadr_syncmode

エレメント・タイプ 情報

表 646. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 データベースの現在のHADR 同期モード。このエレメントのデータ・タイプは整数です。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。データベースが HADR 1 次またはスタンバイの役割の場合は、このエレメントの値は、以下のいずれかの定数です。

- SQLM_HADR_SYNCMODE_SYNC: 同期モード。
- SQLM_HADR_SYNCMODE_NEARSYNC: 近似同期モード。
- SQLM_HADR_SYNCMODE_ASYNC: 非同期モード。

使用法 このエレメントを使用して、データベースの HADR 同期モードを判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、HADR システムが実際に使用している値を報告します。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_connect_status HADR 接続状況 : モニター・エレメント

エレメント ID hadr_connect_status

エレメント・タイプ 情報

表 647. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 データベースの現在のHADR 接続状況。このエレメントのデータ・タイプは整数です。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。データベースが HADR 1 次またはスタンバイの役割の場合は、このエレメントの値は、以下のいずれかの定数です。

- SQLM_HADR_CONN_CONNECTED: データベースはそのパートナー・ノードに接続しています。
- SQLM_HADR_CONN_DISCONNECTED: データベースはそのパートナー・ノードに接続していません。
- SQLM_HADR_CONN_CONGESTED: データベースはそのパートナー・ノードに接続していますが、接続が混雑しています。 1 次とスタンバイのペアの間に TCP/IP ソケット接続が依然として存在しているものの、一方の終端が他方の終端に送信できない場合に、接続は混雑します。例えば、受信側の終端がソケット接続から受信していないと、 TCP/IP 送信スペースがいっぱいになります。ネットワーク接続が混雑する理由には、次のものがあります。
  - ネットワークを共有するリソースが多すぎるか、ネットワークが 1 次 HADR ノードのトランザクション・ボリュームにとって低速である。
  - スタンバイ HADR ノードのあるサーバーが強力でないので、必要な速度で通信サブシステムから情報を取り出せない。

使用法 このエレメントを使用して、データベースの HADR 接続状況を判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_connect_time HADR 接続時刻 : モニター・エレメント

エレメント ID hadr_connect_time

エレメント・タイプ タイム・スタンプ

表 648. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 以下のいずれかを示します。

- HADR 接続時刻
- HADR 輻輳 (ふくそう) 時刻
- HADR 切断時刻

データベースの HADR の役割が標準の場合は、このエレメントは無視されます。データベースが HADR 1 次またはスタンバイの役割の場合は、このエレメントの意味は、 *hadr_connect_status* エレメントの値に従属します。

- *hadr_connect_status* エレメントの値が SQLM_HADR_CONN_CONNECTED の場合は、このエレメントは接続時刻を示す。

- *hadr_connect_status* エレメントの値が SQLM_HADR_CONN_CONGESTED の場合は、このエレメントは輻輳 (ふくそう) の開始時刻を示す。
- *hadr_connect_status* エレメントの値が SQLM_HADR_CONN_DISCONNECTED の場合は、このエレメントは切断時刻を示す。

HADR エンジン・ディスパッチ可能単位 (EDU) の開始以降に接続が行われていない場合、接続状況は「切断」として報告され、切断時刻に HADR EDU 起動時刻が使用されます。 HADR 接続イベントや切断イベントは比較的頻度が低いので、 DFT_MON_TIMESTAMP スイッチが OFF の場合でも時刻は収集されて報告されます。

使用法 このエレメントを使用すると、現在の HADR 接続状況が始まった時刻を判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 「*管理ガイド: パフォーマンス*」の『dft_monswitches - 「デフォルトのデータベース・システム・モニター・スイッチ」構成パラメーター』
- 413 ページの『hadr_role HADR の役割』
- 414 ページの『hadr_connect_status HADR 接続状況 : モニター・エレメント』

hadr_heartbeat HADR ハートビート : モニター・エレメント

エレメント ID hadr_heartbeat

エレメント・タイプ カウンター

表 649. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

スナップショット・モニターの場合、このカウンターはリセットできません。

説明 HADR 接続上で欠落しているハートビートの数。データベースが HADR 1 次またはスタンバイの役割の場合は、このエレメントは HADR 接続の正常性を示します。ハートビートとは、他の HADR ノードから定期的に送信されるメッセージのことです。このエレメントの値がゼロの場合は、ハートビートは欠落しておらず、接続は健全です。値が大きくなるほど、接続の状態は悪くなります。

HADR ノードは、 HADR_TIMEOUT データベース構成パラメーターで定義されている時間間隔の 4 分の 1 か、または 30 秒のうちの短い方の時間内に、他のノードからの 1 つ以上のハートビート・メッセージを予期します。例えば、HADR_TIMEOUT 値が 80 (秒) の場合は、 HADR ノードは 20 秒ごとに他のノードからの 1 つ以上のハートビート・メッセージを予期します。

注:

1. このエレメントのデータ・タイプは整数です。

2. データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、HADR 接続の正常性を判別できます。*hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_local_host HADR ローカル・ホスト : モニター・エレメント

エレメント ID hadr_local_host

エレメント・タイプ 情報

表 650. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 ローカル HADR ホスト名。値は、ホスト名のストリングか、「1.2.3.4」などの IP アドレスのストリングとして表示されます。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、有効な HADR ローカル・ホスト名を判別できます。 HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、 HADR システムが実際に使用している値を報告します。

hadr_role モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_local_service HADR ローカル・サービス : モニター・エレメント

エレメント ID hadr_local_service

エレメント・タイプ 情報

表 651. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 ローカル HADR TCP サービス。この値は、サービス名のストリングか、ポート番号のストリングとして表示されます。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、有効な HADR ローカル・サービス名を判別で

きます。 HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、 HADR システムが実際に使用している値を報告します。

hadr_role モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_remote_host HADR リモート・ホスト : モニター・エレメント

エレメント ID hadr_remote_host

エレメント・タイプ 情報

表 652. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 リモート HADR ホスト名。値は、ホスト名のストリングか、「1.2.3.4」などの IP アドレスのストリングとして表示されます。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、有効な HADR リモート・ホスト名を判別できます。 HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、 HADR システムが実際に使用している値を報告します。

hadr_role モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_remote_service HADR リモート・サービス : モニター・エレメント

エレメント ID hadr_remote_service

エレメント・タイプ 情報

表 653. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 リモート HADR TCP サービス。この値は、サービス名のストリングか、ポート番号のストリングとして表示されます。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、有効な HADR リモート・サービス名を判別できます。 HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、 HADR システムが実際に使用している値を報告します。

hadr_role モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_remote_instance HADR リモート・インスタンス : モニター・エレメント

エレメント ID hadr_remote_instance

エレメント・タイプ 情報

表 654. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 リモート HADR インスタンス名。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、有効な HADR リモート・インスタンス名を判別できます。 HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、 HADR システムが実際に使用している値を報告します。

hadr_role モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_timeout HADR タイムアウト : モニター・エレメント

エレメント ID hadr_timeout

エレメント・タイプ 情報

表 655. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 HADR データベース・サーバーが通信の試行に失敗したと見なすまでに要する秒数。試行に失敗した場合、HADR データベース・サーバーは、この

エレメントによってリストされた秒数以内にパートナーから応答メッセージを受信していないはずです。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、有効な HADR タイムアウト値を判別できます。 HADR データベース構成パラメーターは静的です。パラメーターに対する変更内容は、データベースが停止して再始動するまでは有効になりません。このモニター・エレメントは、データベース構成ファイル中の値ではなく、 HADR システムが実際に使用している値を報告します。

hadr_role モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_primary_log_file HADR 1 次ログ・ファイル : モニター・エレメント

エレメント ID hadr_primary_log_file

エレメント・タイプ 情報

表 656. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 1 次 HADR データベース上の現行ログ・ファイルの名前。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、1 次 HADR データベース上の現行ログ・ファイルを判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_primary_log_page HADR 1 次ログ・ページ : モニター・エレメント

エレメント ID hadr_primary_log_page

エレメント・タイプ 情報

表 657. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 1 次 HADR データベース上の現在のログ位置を示す現行ログ・ファイルのページ番号。ページ番号はログ・ファイルに関連しています。例えば、ページ・ゼロはファイルの先頭です。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、1 次 HADR データベース上の現行ログ・ページを判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_primary_log_lsn HADR 1 次ログ LSN : モニター・エレメント

エレメント ID hadr_primary_log_lsn

エレメント・タイプ 情報

表 658. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 1 次 HADR データベースの現在のログの位置。ログ・シーケンス番号 (LSN) とは、データベースのログ・ストリーム中のバイト・オフセットのことです。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、1 次 HADR データベース上の現在のログの位置を判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_standby_log_file HADR スタンバイ・ログ・ファイル : モニター・エレメント

エレメント ID hadr_standby_log_file

エレメント・タイプ 情報

表 659. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 スタンバイ HADR データベース上の現行ログ・ファイルの名前。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、スタンバイ HADR データベース上の現行ログ・ファイルを判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

hadr_standby_log_page HADR スタンバイ・ログ・ページ : モニター・エレメント

エレメント ID hadr_standby_log_page

エレメント・タイプ 情報

表 660. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 スタンバイ HADR データベース上の現在のログ位置を示す現行ログ・ファイルのページ番号。ページ番号はログ・ファイルに関連しています。例えば、ページ・ゼロはファイルの先頭です。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、スタンバイ HADR データベース上の現行ログ・ページを判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_standby_log_lsn HADR スタンバイ・ログ LSN : モニター・エレメント

エレメント ID hadr_standby_log_lsn

エレメント・タイプ 情報

表 661. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 スタンバイ HADR データベースの現在のログの位置。ログ・シーケンス番号 (LSN) とは、データベースのログ・ストリーム中のバイト・オフセットのことです。データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、スタンバイ HADR データベース上の現在のログの位置を判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

• 413 ページの『hadr_role HADR の役割』

hadr_log_gap HADR ログ・ギャップ

エレメント ID hadr_log_gap

エレメント・タイプ 情報

表 662. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	hadr	基本

説明 このエレメントは、 1 次ログ・シーケンス番号 (LSN) とスタンバイ・ログ LSN の間のギャップの現行の平均を示します。ギャップはバイト数で測定されます。

ログ・ファイルが切り捨てられると、次のログ・ファイルの LSN は、直前のファイルが切り捨てられていないかのように始まります。この LSN の穴には、ログ・データは含まれません。この種の穴により、ログ・ギャップが 1 次とスタンバイの間の実際のログの差を反映しない可能性があります。

データベースの HADR の役割が標準の場合は、このエレメントは無視されます。

使用法 このエレメントを使用して、 1 次 HADR データベース・ログとスタンバイ HADR データベース・ログの間のギャップを判別できます。 *hadr_role* モニター・エレメントを使用して、データベースの HADR の役割を判別できます。

関連資料:

- 413 ページの『hadr_role HADR の役割』

DB2 Connect

DB2 Connect モニター・エレメント

以下のエレメントにより、データベース、アプリケーション、トランザクション、およびステートメントの各レベルでの DB2 接続情報が提供されます。

- dcs_db_name DCS データベース名 : モニター・エレメント
- host_db_name ホスト・データベース名 : モニター・エレメント
- gw_db_alias ゲートウェイでのデータベース別名 : モニター・エレメント
- gw_con_time DB2 Connect ゲートウェイの最初の接続開始 : モニター・エレメント
- gw_connections_top ホスト・データベースへの同時接続の最大数 : モニター・エレメント
- gw_total_cons DB2 Connect の接続試行合計回数 : モニター・エレメント
- gw_cur_cons DB2 Connect の現在の接続数 : モニター・エレメント
- gw_cons_wait_host ホストの応答を待機している接続の数 : モニター・エレメント
- gw_cons_wait_client クライアントの要求送信を待機している接続の数 : モニター・エレメント
- gw_exec_time DB2 Connect ゲートウェイ処理の経過時間 : モニター・エレメント
- sql_stmts 試行された SQL ステートメントの数 : モニター・エレメント
- sql_chains 試行された SQL チェーンの数 : モニター・エレメント

- open_cursors オープン・カーソル数 : モニター・エレメント
- dcs_appl_status DCS アプリケーション状況 : モニター・エレメント
- agent_status DCS アプリケーション・エージェント : モニター・エレメント
- host_ccsid ホスト・コード化文字セット ID : モニター・エレメント
- outbound_comm_protocol アウトバウンド通信プロトコル : モニター・エレメント
- outbound_comm_address アウトバウンド通信アドレス : モニター・エレメント
- inbound_comm_address インバウンド通信アドレス : モニター・エレメント
- inbound_bytes_received 受信されたインバウンド・バイト数 : モニター・エレメント
- outbound_bytes_sent 送信されたアウトバウンド・バイト数 : モニター・エレメント
- outbound_bytes_received 受信されたアウトバウンド・バイト数 : モニター・エレメント
- inbound_bytes_sent 送信されたインバウンド・バイト数 : モニター・エレメント
- outbound_bytes_sent_top 送信された最大アウトバウンド・バイト数 : モニター・エレメント
- outbound_bytes_received_top 受信された最大アウトバウンド・バイト数 : モニター・エレメント
- outbound_bytes_sent_bottom 送信された最小アウトバウンド・バイト数 : モニター・エレメント
- outbound_bytes_received_bottom 受信された最小アウトバウンド・バイト数 : モニター・エレメント
- max_data_sent_128 送信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数 : モニター・エレメント
- max_data_received_128 受信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数 : モニター・エレメント
- max_data_sent_256 送信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数 : モニター・エレメント
- max_data_received_256 受信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数 : モニター・エレメント
- max_data_sent_512 送信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数 : モニター・エレメント
- max_data_received_512 受信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数 : モニター・エレメント
- max_data_sent_1024 送信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数 : モニター・エレメント
- max_data_received_1024 受信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数 : モニター・エレメント
- max_data_sent_2048 送信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数 : モニター・エレメント
- max_data_received_2048 受信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数 : モニター・エレメント

- max_data_sent_4096 送信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数 : モニター・エレメント
- max_data_received_4096 受信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数 : モニター・エレメント
- max_data_sent_8192 送信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数 : モニター・エレメント
- max_data_received_8192 受信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数 : モニター・エレメント
- max_data_sent_16384 送信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数 : モニター・エレメント
- max_data_received_16384 受信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数 : モニター・エレメント
- max_data_sent_31999 送信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数 : モニター・エレメント
- max_data_received_31999 受信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数 : モニター・エレメント
- max_data_sent_64000 送信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数 : モニター・エレメント
- max_data_received_64000 受信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数 : モニター・エレメント
- max_data_sent_gt64000 送信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数 : モニター・エレメント
- max_data_received_gt64000 受信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数 : モニター・エレメント
- max_network_time_1_ms ネットワーク時間が 1 ミリ秒以下のステートメント数 : モニター・エレメント
- max_network_time_4_ms ネットワーク時間が 1 から 4 ミリ秒のステートメント数 : モニター・エレメント
- max_network_time_16_ms ネットワーク時間が 4 から 16 ミリ秒のステートメント数 : モニター・エレメント
- max_network_time_100_ms ネットワーク時間が 16 から 100 ミリ秒のステートメント数 : モニター・エレメント
- max_network_time_500_ms ネットワーク時間が 100 から 500 ミリ秒のステートメント数 : モニター・エレメント
- max_network_time_gt500_ms ネットワーク時間が 500 ミリ秒を超えるステートメント数 : モニター・エレメント
- network_time_top ステートメントの最大ネットワーク時間 : モニター・エレメント
- network_time_bottom ステートメントの最小ネットワーク時間 : モニター・エレメント
- xid トランザクション ID : モニター・エレメント
- elapsed_exec_time ステートメント実行経過時間 : モニター・エレメント
- host_response_time ホスト応答時間 : モニター・エレメント

- num_transmissions 伝送回数 : モニター・エレメント
- num_transmissions_group 伝送グループの回数 : モニター・エレメント
- con_response_time 接続の最新応答時間 : モニター・エレメント
- con_elapsed_time 最新の接続経過時間 : モニター・エレメント
- gw_comm_errors 通信エラー : モニター・エレメント
- gw_comm_error_time 通信エラー時刻 : モニター・エレメント
- blocking_cursor ブロック・カーソル : モニター・エレメント
- トランザクション・プロセッサー・モニターに関するモニター・エレメント

dcs_db_name DCS データベース名

エレメント ID dcs_db_name

エレメント・タイプ 情報

表 663. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl_info	基本

説明 DCS ディレクトリーにカタログされている DCS データベースの名前。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

関連資料:

- 426 ページの『host_db_name ホスト・データベース名』
- 427 ページの『gw_db_alias ゲートウェイでのデータベース別名』

host_db_name ホスト・データベース名

エレメント ID host_db_name

エレメント・タイプ 情報

表 664. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl_info	基本

説明 情報が収集されているホスト・データベースの実名、またはアプリケーションの接続先のホスト・データベースの実名。これは、このホスト・データベースが作成されたときに付けられた名前です。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

関連資料:

- 426 ページの『dcs_db_name DCS データベース名』
- 427 ページの『gw_db_alias ゲートウェイでのデータベース別名』

gw_db_alias ゲートウェイでのデータベース別名

エレメント ID gw_db_alias

エレメント・タイプ 情報

表 665. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 ホスト・データベースに接続するために DB2 Connect ゲートウェイで使用される別名。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

関連資料:

- 426 ページの『dcs_db_name DCS データベース名』
- 426 ページの『host_db_name ホスト・データベース名』

gw_con_time DB2 Connect ゲートウェイの最初の接続開始

エレメント ID gw_con_time

エレメント・タイプ タイム・スタンプ

表 666. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	タイム・スタンプ
DCS アプリケーション	dcs_appl	タイム・スタンプ

説明 ホスト・データベースへの最初の接続が DB2 Connect ゲートウェイから開始された日時。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

gw_connections_top ホスト・データベースへの同時接続の最大数

エレメント ID gw_connections_top

エレメント・タイプ 水準点

表 667. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本

説明 最初のデータベース接続以降、 DB2 Connect ゲートウェイが処理したホスト・データベースへの同時接続の最大数。

使用法 このエレメントは、DB2 Connect ゲートウェイでのアクティビティーのレベルと、関連したシステム・リソースの使用状況を知るのに役立ちます。

関連資料:

- 428 ページの『gw_total_cons DB2 Connect の接続試行合計回数』
- 428 ページの『gw_cur_cons DB2 Connect の現在の接続数』

gw_total_cons DB2 Connect の接続試行合計回数

エレメント ID gw_total_cons

エレメント・タイプ 水準点

表 668. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
DCS データベース	dcs_dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 最後の db2start コマンドまたは最後のリセット以降に、 DB2 Connect ゲートウェイから試行された接続の合計回数。

使用法 このエレメントは、DB2 Connect ゲートウェイでのアクティビティーのレベルと、関連したシステム・リソースの使用状況を知るのに役立ちます。

関連資料:

- 427 ページの『gw_connections_top ホスト・データベースへの同時接続の最大数』
- 428 ページの『gw_cur_cons DB2 Connect の現在の接続数』

gw_cur_cons DB2 Connect の現在の接続数

エレメント ID gw_cur_cons

エレメント・タイプ ゲージ

表 669. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
DCS データベース	dcs_dbase	基本

説明 DB2 Connect ゲートウェイが処理しているホスト・データベースへの現在の接続数。

使用法 このエレメントは、DB2 Connect ゲートウェイでのアクティビティーのレベルと、関連したシステム・リソースの使用状況を知るのに役立ちます。

関連資料:

- 427 ページの『gw_connections_top ホスト・データベースへの同時接続の最大数』
- 428 ページの『gw_total_cons DB2 Connect の接続試行合計回数』

gw_cons_wait_host ホストの応答を待機している接続の数

エレメント ID gw_cons_wait_host

エレメント・タイプ ゲージ

表 670. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
DCS データベース	dcs_dbase	基本

説明 DB2 Connect ゲートウェイが処理しているホスト・データベースへの接続のうち、ホストからの応答を待機している現在の接続数。

使用法 この値は頻繁に変わることがあります。実際のゲートウェイの使用率を把握するには、長期にわたって周期的にサンプリングを行わなければなりません。

関連資料:

- 428 ページの『gw_cur_cons DB2 Connect の現在の接続数』
- 429 ページの『gw_cons_wait_client クライアントの要求送信を待機している接続の数』

gw_cons_wait_client クライアントの要求送信を待機している接続の数

エレメント ID gw_cons_wait_client

エレメント・タイプ ゲージ

表 671. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース・マネージャー	db2	基本
DCS データベース	dcs_dbase	基本

説明 DB2 Connect ゲートウェイが処理しているホスト・データベースへの接続のうち、クライアントが要求を送信するのを待機している現在の接続数。

使用法 この値は頻繁に変わることがあります。実際のゲートウェイの使用率を把握するには、長期にわたって周期的にサンプリングを行わなければなりません。

関連資料:

- 428 ページの『gw_cur_cons DB2 Connect の現在の接続数』
- 429 ページの『gw_cons_wait_host ホストの応答を待機している接続の数』

gw_exec_time DB2 Connect ゲートウェイ処理の経過時間

エレメント ID gw_exec_time

エレメント・タイプ 時間

表 672. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl	ステートメント、タイム・スタンプ
DCS ステートメント	dcs_stmt	ステートメント、タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 DB2 Connect ゲートウェイがアプリケーションの要求を処理するのに要した時間 (接続が確立された以降)、または単一ステートメントを処理するのに要した時間 (秒およびマイクロ秒単位)。

使用法 このエレメントを使用して、全体の処理時間のうちどの部分が DB2 Connect ゲートウェイの処理によるものかを判別します。

sql_stmts 試行された SQL ステートメントの数

エレメント ID sql_stmts

エレメント・タイプ カウンター

表 673. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl	基本
データ伝送	stmt_transmissions	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 データ伝送スナップショットの場合、このエレメントは、ステートメント処理中に DB2 Connect ゲートウェイとホストの間で *n* 回のデータ伝送が行われる際の、 SQL ステートメントの数を表します。範囲 *n* は、*num_transmissions_group* エレメントで指定されます。

DCS DATABASE スナップショットの場合、このステートメントのカウントは、データベースが活動化された後のステートメントの数になります。

DCS APPLICATION スナップショットの場合、このステートメントのカウントは、データベースへの接続がこのアプリケーションによって確立された後のステートメントの数になります。

使用法 データベース・レベルまたはアプリケーション・レベルでは、このエレメントを使用してデータベース・アクティビティーを測定します。ある一定の期間について SQL ステートメントのスループットを計算するには、 2 つのスナップショットの間の経過時間の値でこのエレメントの値を割ります。

データ伝送レベルの場合: このエレメントを使用すると、処理中に 2、3、4 などのデータ伝送回数をいくつのステートメントが使用したかについて統計を得ることができます。 (1 つのステートメントを処理するには、少なくとも送信と受信の 2 回以上のデータ伝送が必要です。) この統計により、

データベース・レベルおよびアプリケーション・レベルでのデータベースやアプリケーションのアクティビティーやネットワーク・トラフィックの状態がより明確になります。

注:

1. *sql_stmts* モニター・エレメントは、 SQL ステートメントのサーバーへの送信が試行される回数を表します。
   - アプリケーション・レベルおよびデータベース・レベルでは、カーソル中の個々の SQL ステートメントは個別にカウントされる。
   - 伝送レベルでは、同一カーソル中のすべてのステートメントは単一の SQL ステートメントとしてカウントされる。

関連資料:

- 405 ページの『time_stamp スナップショット時刻』
- 458 ページの『num_transmissions_group 伝送グループの回数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

sql_chains 試行された SQL チェーンの数

エレメント ID sql_chains

エレメント・タイプ カウンター

表 674. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データ伝送	stmt_transmissions	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 ステートメント処理中に DB2 Connect ゲートウェイとホストの間で *n* 回のデータ伝送が行われる際の、 SQL ステートメントの数を表します。範囲 *n* は、 *num_transmissions_group* エレメントで指定されます。

例えば、チェーニングが ON の場合に、 PREP ステートメントと OPEN ステートメントがチェーニングされ、チェーンが合計 2 つの伝送を要する場合は、 *sql_chains* は「1」と報告され、 *sql_stmts* は「2」と報告されます。

チェーニングが OFF の場合は、 *sql_chains* のカウントは、 *sql_stmts* のカウントと等しくなります。

使用法 このエレメントを使用すると、処理中に 2、3、 4 などのデータ伝送回数をいくつのステートメントが使用したかについて統計を得ることができます。(1 つのステートメントを処理するには、少なくとも送信と受信の 2 回以上のデータ伝送が必要です。) この統計により、データベース・レベルおよびアプリケーション・レベルでのデータベースやアプリケーションのアクティビティーやネットワーク・トラフィックの状態がより明確になります。

注: *sql_stmts* モニター・エレメントは、 SQL ステートメントのサーバーへの送信が試行される回数を表します。伝送レベルでは、同一カーソル中のすべてのステートメントは単一の SQL ステートメントとしてカウントされます。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 「*コール・レベル・インターフェース ガイドおよびリファレンス 第 2 巻*」の『ステートメント属性 (CLI) のリスト』
- 458 ページの『num_transmissions_group 伝送グループの回数』

open_cursors オープン・カーソル数

エレメント ID open_cursors

エレメント・タイプ ゲージ

表 675. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl	ステートメント

説明 アプリケーションで現在開いているカーソルの数。

使用法 このエレメントを使用して、割り振られているメモリーの量を評価します。ターゲット・データベース上の DB2 クライアント、DB2 Connect、またはデータベース・エージェントに割り振られるメモリー量は、現在開いているカーソルの数と関連しています。この情報は、キャパシティー・プランニングに利用できます。例えば、ブロッキングを行っている各オープン・カーソルのバッファー・サイズは RQRIOBLK です。 *deferred_prepare* を使用可能にすると、 2 つのバッファーが割り振られます。

このエレメントには、早期クローズによって閉じられたカーソルは組み込まれていません。ホスト・データベースが最後のレコードをクライアントに戻すと、早期クローズが生じます。ホストとゲートウェイではカーソルが閉じられますが、クライアント側では依然として開いています。早期クローズ・カーソルは、 DB2 コール・レベル・インターフェースを使用して設定できます。

dcs_appl_status DCS アプリケーション状況

エレメント ID dcs_appl_status

エレメント・タイプ 情報

表 676. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 DB2 Connect ゲートウェイでの DCS アプリケーションの状況。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。次の値があります。

- SQLM_DCS_CONNECTPEND_OUTBOUND

  アプリケーションが DB2 Connect ゲートウェイからホスト・データベースへのデータベース接続を開始しましたが、要求はまだ完了していません。

- SQLM_DCS_UOWWAIT_OUTBOUND

  DB2 Connect ゲートウェイは、ホスト・データベースがアプリケーションの要求に応答するのを待っています。

- SQLM_DCS_UOWWAIT_INBOUND

  DB2 Connect ゲートウェイからホスト・データベースへの接続が確立され、ゲートウェイがアプリケーションの SQL 要求を待っています。あるいは、アプリケーション内の作業単位に代わって、DB2 Connect ゲートウェイが待機しています。通常、これはアプリケーションのコードが実行中であることを意味します。

関連資料:

- 433 ページの『host_ccsid ホスト・コード化文字セット ID』
- 434 ページの『outbound_comm_protocol アウトバウンド通信プロトコル』
- 434 ページの『outbound_comm_address アウトバウンド通信アドレス』
- 435 ページの『inbound_comm_address インバウンド通信アドレス』

agent_status DCS アプリケーション・エージェント

エレメント ID agent_status

エレメント・タイプ 情報

表 677. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 接続コンセントレーター環境では、この値は、関連エージェントを現在持っているアプリケーションを示します。

使用法 次の値があります。

- SQLM_AGENT_ASSOCIATED

  このアプリケーションに代わって作業中のエージェントがアプリケーションに関連付けられています。

- SQLM_AGENT_NOT_ASSOCIATED

  このアプリケーションに代わって作業していたエージェントは、もはやアプリケーションには関連付けられていません。現在はほかのアプリケーションで使用されています。この後、関連エージェントがないままこのアプリケーションの作業が行われると、エージェントが再び関連付けられます。

関連資料:

- 432 ページの『dcs_appl_status DCS アプリケーション状況』

host_ccsid ホスト・コード化文字セット ID

エレメント ID host_ccsid

エレメント・タイプ 情報

表 678. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 ホスト・データベースの コード化文字セット ID (CCSID) です。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

関連資料:

- 432 ページの『dcs_appl_status DCS アプリケーション状況』
- 434 ページの『outbound_comm_protocol アウトバウンド通信プロトコル』
- 434 ページの『outbound_comm_address アウトバウンド通信アドレス』
- 435 ページの『inbound_comm_address インバウンド通信アドレス』

outbound_comm_protocol アウトバウンド通信プロトコル

エレメント ID outbound_comm_protocol

エレメント・タイプ 情報

表 679. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl_info	基本

説明 DB2 Connect ゲートウェイとホストの間で使用される通信プロトコル。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。有効な値は次のとおりです。

- SQLM_PROT_APPC
- SQLM_PROT_TCPIP

関連資料:

- 432 ページの『dcs_appl_status DCS アプリケーション状況』
- 433 ページの『host_ccsid ホスト・コード化文字セット ID』
- 434 ページの『outbound_comm_address アウトバウンド通信アドレス』
- 435 ページの『inbound_comm_address インバウンド通信アドレス』

outbound_comm_address アウトバウンド通信アドレス

エレメント ID outbound_comm_address

エレメント・タイプ 情報

表 680. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
DCS アプリケーション	dcs_appl_info	基本

説明 これはターゲット・データベースの通信アドレスです。例えば、SNA ネット ID と LU パートナー名、または TCP/IP 用の IP アドレスとポート番号などです。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

関連資料:

- 432 ページの『dcs_appl_status DCS アプリケーション状況』
- 433 ページの『host_ccsid ホスト・コード化文字セット ID』
- 434 ページの『outbound_comm_protocol アウトバウンド通信プロトコル』
- 435 ページの『inbound_comm_address インバウンド通信アドレス』

inbound_comm_address インバウンド通信アドレス

エレメント ID inbound_comm_address

エレメント・タイプ 情報

表 681. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
DCS アプリケーション	dcs_appl_info	基本

説明 これはクライアントの通信アドレスです。例えば、SNA ネット ID と LU パートナー名、または TCP/IP 用の IP アドレスとポート番号などです。

使用法 このエレメントは、DCS アプリケーションに関する問題判別に使用します。

関連資料:

- 432 ページの『dcs_appl_status DCS アプリケーション状況』
- 433 ページの『host_ccsid ホスト・コード化文字セット ID』
- 434 ページの『outbound_comm_protocol アウトバウンド通信プロトコル』
- 434 ページの『outbound_comm_address アウトバウンド通信アドレス』

inbound_bytes_received 受信されたインバウンド・バイト数

エレメント ID inbound_bytes_received

エレメント・タイプ カウンター

表 682. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl	基本
DCS ステートメント	dcs_stmt	ステートメント

アプリケーション・レベルでのスナップショット・モニターの場合、このカウンターはリセットできます。その他のレベルではこのカウンターはリセットできません。

説明 DB2 Connect ゲートウェイがクライアントから受信したバイト数。ただし、通信プロトコルのオーバーヘッド (例えば TCP/IP や SNA のヘッダー) は含まれません。

使用法 このエレメントを使用して、クライアントから DB2 Connect ゲートウェイへのスループットを測定します。

関連資料:

- 436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
- 436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』
- 437 ページの『inbound_bytes_sent 送信されたインバウンド・バイト数』

outbound_bytes_sent 送信されたアウトバウンド・バイト数

エレメント ID outbound_bytes_sent

エレメント・タイプ カウンター

表 683. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本
DCS アプリケーション	dcs_appl	基本
DCS ステートメント	dcs_stmt	ステートメント
データ伝送	stmt_transmissions	ステートメント

ステートメント・レベルでのスナップショット・モニターの場合、このカウンターはリセットできません。その他のレベルではこのカウンターはリセットできます。

説明 DB2 Connect ゲートウェイがホストに送信したバイト数。通信プロトコルのオーバーヘッド (例えば TCP/IP や SNA のヘッダー) は含まれません。

データ伝送レベルの場合: このデータ伝送回数を使用したすべてのステートメントの処理で、 DB2 Connect ゲートウェイがホストに送信したバイト数。

使用法 このエレメントを使用して、 DB2 Connect ゲートウェイからホスト・データベースへのスループットを測定します。

関連資料:

- 435 ページの『inbound_bytes_received 受信されたインバウンド・バイト数』
- 436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』
- 437 ページの『inbound_bytes_sent 送信されたインバウンド・バイト数』

outbound_bytes_received 受信されたアウトバウンド・バイト数

エレメント ID outbound_bytes_received

エレメント・タイプ カウンター

表 684. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本

表 684. スナップショット・モニター情報 (続き)

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl	基本
DCS ステートメント	dcs_stmt	ステートメント
データ伝送	stmt_transmissions	ステートメント

ステートメント・レベルでのスナップショット・モニターの場合、このカウンターはリセットできません。その他のレベルではこのカウンターはリセットできます。

説明 DB2 Connect ゲートウェイがホストから受信したバイト数。ただし、通信プロトコルのオーバーヘッド (例えば TCP/IP や SNA のヘッダー) は含まれません。

データ伝送レベルの場合: このデータ伝送回数を使用したすべてのステートメントの処理で、 DB2 Connect ゲートウェイがホストから受信したバイト数。

使用法 このエレメントを使用して、ホスト・データベースから DB2 Connect ゲートウェイへのスループットを測定します。

関連資料:

- 435 ページの『inbound_bytes_received 受信されたインバウンド・バイト数』
- 436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
- 437 ページの『inbound_bytes_sent 送信されたインバウンド・バイト数』

inbound_bytes_sent 送信されたインバウンド・バイト数

エレメント ID inbound_bytes_sent

エレメント・タイプ カウンター

表 685. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl	基本
DCS ステートメント	dcs_stmt	ステートメント

アプリケーション・レベルでのスナップショット・モニターの場合、このカウンターはリセットできます。その他のレベルではこのカウンターはリセットできません。

説明 DB2 Connect ゲートウェイがクライアントに送信したバイト数。通信プロトコルのオーバーヘッド (例えば TCP/IP や SNA のヘッダー) は含まれません。

使用法 このエレメントを使用して、DB2 Connect ゲートウェイからクライアントへのスループットを測定します。

関連資料:

- 435 ページの『inbound_bytes_received 受信されたインバウンド・バイト数』
- 436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
- 436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』

outbound_bytes_sent_top 送信された最大アウトバウンド・バイト数

エレメント ID outbound_bytes_sent_top

エレメント・タイプ 水準点

表 686. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データ伝送	stmt_transmissions	ステートメント

説明 このデータベースに対するステートメントまたはチェーン、またはこの DCS アプリケーション内のステートメントまたはチェーンで、このデータ伝送回数を使用したすべてのステートメントまたはチェーンを処理している間に、 DB2 Connect ゲートウェイがホストに送信したステートメントまたはチェーン単位の最大バイト数。

使用法 このエレメントは、 DB2 Connect ゲートウェイからホスト・データベースへのスループットを示すもう 1 つのパラメーターである「送信されたアウトバウンド・バイト数」と組み合わせて使用します。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

outbound_bytes_received_top 受信された最大アウトバウンド・バイト数

エレメント ID outbound_bytes_received_top

エレメント・タイプ 水準点

表 687. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データ伝送	stmt_transmissions	ステートメント

説明 DB2 Connect ゲートウェイがホストから受信したステートメントまたはチェーン単位の最大バイト数。この DCS データベースに対するステートメントまたはチェーン、またはこの DCS アプリケーション内のステートメントまたはチェーンを処理している間に、このデータ伝送回数を使用したすべてのステートメントまたはチェーンが対象になります。

使用法 このエレメントは、 DB2 Connect ゲートウェイからホスト・データベースへのスループットを示すもう 1 つのパラメーターである「受信されたアウトバウンド・バイト数」と組み合わせて使用します。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

outbound_bytes_sent_bottom 送信された最小アウトバウンド・バイト数

エレメント ID outbound_bytes_sent_bottom

エレメント・タイプ 水準点

表 688. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データ伝送	stmt_transmissions	ステートメント

説明 DB2 Connect ゲートウェイがホストから受信したステートメントまたはチェーン単位の最小バイト数。この DCS データベースに対するステートメントまたはチェーン、またはこの DCS アプリケーション内のステートメントまたはチェーンを処理している間に、このデータ伝送回数を使用したすべてのステートメントまたはチェーンが対象になります。

使用法 このエレメントは、 DB2 Connect ゲートウェイからホスト・データベースへのスループットを示すもう 1 つのパラメーターである「送信されたアウトバウンド・バイト数」と組み合わせて使用します。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

outbound_bytes_received_bottom 受信された最小アウトバウンド・バイト数

エレメント ID outbound_bytes_received_bottom

エレメント・タイプ 水準点

表 689. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データ伝送	stmt_transmissions	ステートメント

説明 このデータベースに対するステートメントまたはチェーン、またはこの DCS アプリケーション内のステートメントでまたはチェーン、このデータ伝送回数を使用したすべてのステートメントまたはチェーンを処理している間に、 DB2 Connect ゲートウェイがホストから受信したステートメントまたはチェーン単位の最小バイト数。

使用法 このエレメントは、 DB2 Connect ゲートウェイからホスト・データベースへのスループットを示すもう 1 つのパラメーターである「受信されたアウトバウンド・バイト数」と組み合わせて使用します。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_128 送信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数

エレメント ID max_data_sent_128

エレメント・タイプ カウンター

表 690. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 1 から 128 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_128 受信アウトバウンド・バイト数が 1 から 128 バイトのステートメント数

エレメント ID max_data_received_128

エレメント・タイプ カウンター

表 691. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 1 から 128 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_256 送信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数

エレメント ID max_data_sent_256

エレメント・タイプ カウンター

表 692. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 129 から 256 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_256 受信アウトバウンド・バイト数が 129 から 256 バイトのステートメント数

エレメント ID max_data_received_256

エレメント・タイプ カウンター

表 693. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 129 から 256 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_512 送信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数

エレメント ID max_data_sent_512

エレメント・タイプ カウンター

表 694. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 257 から 512 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_512 受信アウトバウンド・バイト数が 257 から 512 バイトのステートメント数

エレメント ID max_data_received_512

エレメント・タイプ カウンター

表 695. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 257 から 512 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_1024 送信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数

エレメント ID max_data_sent_1024

エレメント・タイプ カウンター

表 696. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 513 から 1024 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_1024 受信アウトバウンド・バイト数が 513 から 1024 バイトのステートメント数

エレメント ID max_data_received_1024

エレメント・タイプ カウンター

表 697. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 513 から 1024 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_2048 送信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数

エレメント ID max_data_sent_2048

エレメント・タイプ カウンター

表 698. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 1025 から 2048 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_2048 受信アウトバウンド・バイト数が 1025 から 2048 バイトのステートメント数

エレメント ID max_data_received_2048

エレメント・タイプ カウンター

表 699. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 1025 から 2048 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_4096 送信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数

エレメント ID max_data_sent_4096

エレメント・タイプ カウンター

表 700. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 2049 から 4096 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_4096 受信アウトバウンド・バイト数が 2049 から 4096 バイトのステートメント数

エレメント ID max_data_received_4096

エレメント・タイプ カウンター

表 701. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 2049 から 4096 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_8192 送信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数

エレメント ID max_data_sent_8192

エレメント・タイプ カウンター

表 702. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 4097 から 8192 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_8192 受信アウトバウンド・バイト数が 4097 から 8192 バイトのステートメント数

エレメント ID max_data_received_8192

エレメント・タイプ カウンター

表 703. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 4097 から 8192 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_16384 送信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数

エレメント ID max_data_sent_16384

エレメント・タイプ カウンター

表 704. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 8193 から 16384 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_16384 受信アウトバウンド・バイト数が 8193 から 16384 バイトのステートメント数

エレメント ID max_data_received_16384

エレメント・タイプ カウンター

表 705. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 8193 から 16384 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_31999 送信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数

エレメント ID max_data_sent_31999

エレメント・タイプ カウンター

表 706. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 16385 から 31999 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_31999 受信アウトバウンド・バイト数が 16385 から 31999 バイトのステートメント数

エレメント ID max_data_received_31999

エレメント・タイプ カウンター

表 707. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 16385 から 31999 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_64000 送信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数

エレメント ID max_data_sent_64000

エレメント・タイプ カウンター

表 708. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 32000 から 64000 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_64000 受信アウトバウンド・バイト数が 32000 から 64000 バイトのステートメント数

エレメント ID max_data_received_64000

エレメント・タイプ カウンター

表 709. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 32000 から 64000 バイトだったステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_sent_gt64000 送信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数

エレメント ID max_data_sent_gt64000

エレメント・タイプ カウンター

表 710. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、送信アウトバウンド・バイト数が 64000 バイトを超えたステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_data_received_gt64000 受信アウトバウンド・バイト数が 64000 バイトを超えるステートメント数

エレメント ID max_data_received_gt64000

エレメント・タイプ カウンター

表 711. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、受信アウトバウンド・バイト数が 64000 バイトを超えたステートメントまたはチェーンの数を示します。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_network_time_1_ms ネットワーク時間が 1 ミリ秒以下のステートメント数

エレメント ID max_network_time_1_ms

エレメント・タイプ カウンター

表 712. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、ネットワーク時間が 1 ミリ秒以下だったステートメントまたはチェーンの数を示します (ネットワーク時間は、 1 つのステートメントまたはチェーンに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_network_time_4_ms ネットワーク時間が 1 から 4 ミリ秒のステートメント数

エレメント ID max_network_time_4_ms

エレメント・タイプ カウンター

表 713. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、ネットワーク時間が 1 ミリ秒を超えて 4 ミリ秒以下だったステートメントまたはチェーンの数を示します (ネットワーク時間は、 1 つのステートメントまたはチェーンに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_network_time_16_ms ネットワーク時間が 4 から 16 ミリ秒のステートメント数

エレメント ID max_network_time_16_ms

エレメント・タイプ カウンター

表 714. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、ネットワーク時間が 4 ミリ秒を超えて 16 ミリ秒以下だったステートメントまたはチェーンの数を示します (ネットワーク時間は、 1 つのステートメントまたはチェーンに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_network_time_100_ms ネットワーク時間が 16 から 100 ミリ秒のステートメント数

エレメント ID max_network_time_100_ms

エレメント・タイプ カウンター

表 715. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、ネットワーク時間が 16 ミリ秒を超えて 100 ミリ秒以下だったステートメントまたはチェーンの数を示します (ネットワーク時間は、 1 つのステートメントまたはチェーンに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_network_time_500_ms ネットワーク時間が 100 から 500 ミリ秒のステートメント数

エレメント ID max_network_time_500_ms

エレメント・タイプ カウンター

表 716. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、ネットワーク時間が 100 ミリ秒を超えて 500 ミリ秒以下だったステートメントまたはチェーンの数を示します (ネットワーク時間は、 1 つのステートメントまたはチェーンに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

max_network_time_gt500_ms ネットワーク時間が 500 ミリ秒を超えるステートメント数

エレメント ID max_network_time_gt500_ms

エレメント・タイプ カウンター

表 717. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント
データ伝送	stmt_transmissions	ステートメント

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、ネットワーク時間が 500 ミリ秒を超えたステートメントまたはチェーンの数を示します (ネットワーク時間は、 1 つのステートメントまたはチェーンに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 430 ページの『sql_stmts 試行された SQL ステートメントの数』
- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』
- 431 ページの『sql_chains 試行された SQL チェーンの数』

network_time_top ステートメントの最大ネットワーク時間

エレメント ID network_time_top

エレメント・タイプ 水準点

表 718. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント、タイム・スタンプ
DCS アプリケーション	dcs_appl	ステートメント、タイム・スタンプ
データ伝送	stmt_transmissions	ステートメント、タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、 DCS データベースに対してまたはこの DCS アプリケーション内で実行されたステートメントについて、あるいはこの回数のデータ伝送を使用したステートメントについて、最長ネットワーク時間を示します (ネットワーク時間は、1 つのステートメントに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。「タイム・スタンプ」スイッチが OFF のときは、このエレメントは収集されません。

関連資料:

- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』

network_time_bottom ステートメントの最小ネットワーク時間

エレメント ID network_time_bottom

エレメント・タイプ 水準点

表 719. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント、タイム・スタンプ
DCS アプリケーション	dcs_appl	ステートメント、タイム・スタンプ
データ伝送	stmt_transmissions	ステートメント、タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントは、 DCS データベースに対してまたはこの DCS アプリケーション内で実行されたステートメントについて、あるいはこの回数のデータ伝送を使用したステートメントについて、最小ネットワーク時間を示します (ネットワーク時間は、1 つのステートメントに対するホスト応答時間と実行経過時間の差です)。

使用法 このエレメントを使用すると、データベース・レベルおよびアプリケーション・レベルでのデータベース・アクティビティーやネットワーク・トラフィックの状態がより明確になります。

関連資料:

- 457 ページの『host_response_time ホスト応答時間』
- 393 ページの『total_exec_time ステートメント実行の経過時間』

xid トランザクション ID

エレメント ID xid

エレメント・タイプ 情報

表 720. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS アプリケーション	dcs_appl	作業単位

説明 トランザクション・マネージャーが 2 フェーズ・コミットのトランザクションで生成した、 (すべてのデータベースにおいて) ユニークなトランザクション ID。

使用法 この ID を使用すると、トランザクション・マネージャーが生成したトランザクションと複数のデータベースに対して実行されたトランザクションを関連付けることができます。トランザクション・マネージャーに問題があったときに、 2 フェーズ・コミット・プロトコルが関与するデータベース・ト

ランザクションとトランザクション・マネージャーが発信したトランザクションを対応させて、問題の診断に利用できます。

elapsed_exec_time ステートメント実行経過時間

エレメント ID elapsed_exec_time

エレメント・タイプ 時間

表 721. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase	ステートメント、タイム・スタンプ
アプリケーション	appl	ステートメント、タイム・スタンプ
DCS データベース	dcs_dbase	ステートメント、タイム・スタンプ
DCS アプリケーション	dcs_appl	ステートメント、タイム・スタンプ
DCS ステートメント	dcs_stmt	ステートメント、タイム・スタンプ
データ伝送	stmt_transmissions	ステートメント、タイム・スタンプ

ステートメント・レベルでのスナップショット・モニターの場合、このカウンターはリセットできません。その他のレベルではこのカウンターはリセットできます。

説明 DCS ステートメント・レベルでは、ホスト・データベース・サーバー上で SQL 要求の処理に要した経過時間を示します。この値はこのサーバーによって報告されています。 host_response_time エレメントと比較すると、このエレメントには DB2 Connect とホスト・データベース・サーバーの間のネットワーク経過時間が含まれていません。

ほかのレベルでは、この値は特定のデータベースやアプリケーションのために実行されたすべてのステートメント、あるいは特定の回数のデータ転送を使用したステートメントについてのホスト実行時間の合計を示します。

使用法 このエレメントと経過時間に関する他のモニター・エレメントを組み合わせて使用すると、データベース・サーバーでの SQL 要求の処理を評価したり、パフォーマンス上の問題を特定するときに利用できます。

host_response_time エレメントからこのエレメントの値を引くと、 DB2 Connect とホスト・データベース・サーバーの間のネットワーク経過時間を計算できます。

注: dcs_dbase、dcs_appl、dcs_stmt、および stmt_transmissions レベルの場合、 *elapsed_exec_time element* は z/OS データベースのみに適用されます。 DB2 Connect ゲートウェイが Windows、Linux、AIX、またはその他の UNIX データベースに接続している場合は、 *elapsed_exec_time* はゼロとして報告されます。

関連資料:

• 457 ページの『host_response_time ホスト応答時間』

host_response_time ホスト応答時間

エレメント ID host_response_time

エレメント・タイプ 時間

表 722. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	ステートメント
DCS アプリケーション	dcs_appl	ステートメント、タイム・スタンプ
DCS ステートメント	dcs_stmt	ステートメント、タイム・スタンプ
データ伝送	stmt_transmissions	ステートメント、タイム・スタンプ

ステートメント・レベルでのスナップショット・モニターの場合、このカウンターはリセットできません。その他のレベルではこのカウンターはリセットできます。

説明 DCS ステートメント・レベルでは、ステートメントが DB2 Connect ゲートウェイから処理のためにホストに送信された時刻と、ホストから結果を受信した時刻との間の経過時間です。 DCS データベースおよび DCS アプリケーションのレベルでは、特定のアプリケーションまたはデータベースに対して実行されたすべてのステートメントについての経過時間の合計です。データ伝送レベルでは、この多数のデータ伝送を使用したすべてのステートメントに関するホスト要求時間の合計です。

使用法 このエレメントと送信アウトバウンド・バイト数および受信アウトバウンド・バイト数を組み合わせて使用すると、次のようにしてアウトバウンド応答時間 (転送速度) を計算できます。

```
(送信されたアウトバウンド・バイト数 + 受信されたアウトバウンド・バイト数)
/ホストの応答時間
```

関連資料:

• 436 ページの『outbound_bytes_sent 送信されたアウトバウンド・バイト数』
• 436 ページの『outbound_bytes_received 受信されたアウトバウンド・バイト数』
• 456 ページの『elapsed_exec_time ステートメント実行経過時間』

num_transmissions 伝送回数

エレメント ID num_transmissions

エレメント・タイプ カウンター

表 723. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS ステートメント	dcs_stmt	ステートメント

説明

これは、DB2 UDB バージョン 8.1.2 以降には無関係なレガシー・モニター・エレメントです。 DB2 UDB バージョン 8.1.2 以降をご使用の場合は、 num_transmissions_group モニター・エレメントを参照してください。

DB2 Connect ゲートウェイとこの DCS ステートメントを処理するのに使用されたホストの間の伝送回数。 (1 回のデータ伝送は、送信 1 回または受信 1 回を示します。)

使用法 このエレメントを使用すると、特定のステートメントの実行時間が長くかかった理由が明確になります。例えば、照会のときに大きな結果セットを戻すと、完了するまでにたくさんのデータ伝送回数が必要になります。

関連資料:

• 458 ページの『num_transmissions_group 伝送グループの回数』

num_transmissions_group 伝送グループの回数

エレメント ID num_transmissions_group

エレメント・タイプ 情報

表 724. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS ステートメント	dcs_stmt	ステートメント

説明 この DCS ステートメントを処理するのに使われた、 DB2 Connect ゲートウェイとホストの間の伝送範囲。 (1 回のデータ伝送は、送信 1 回または受信 1 回を示します。)

使用法 このエレメントを使用すると、特定のステートメントの実行時間が長くかかった理由が明確になります。例えば、照会のときに大きな結果セットを戻すと、完了するまでにたくさんのデータ伝送回数が必要になります。

伝送範囲を表す定数は以下のように記述され、sqlmon.h で定義されています。

API 定数	説明
SQLM_DCS_TRANS_GROUP_2	2 回の伝送
SQLM_DCS_TRANS_GROUP_3TO7	3 回から 7 回までの伝送
SQLM_DCS_TRANS_GROUP_8TO15	8 回から 15 回までの伝送
SQLM_DCS_TRANS_GROUP_16TO64	16 回から 64 回までの伝送
SQLM_DCS_TRANS_GROUP_GT64	64 回を超える伝送

con_response_time 接続の最新応答時間

エレメント ID con_response_time

エレメント・タイプ 時間

表 725. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	タイム・スタンプ

説明 このデータベースに最後に接続された DCS アプリケーションにおける、接続処理の開始と実際の接続の確立との間の経過時間。

使用法 アプリケーションが特定のホスト・データベースに接続するために現在かかっている時間の標識として、このエレメントを使用します。

関連資料:

- 269 ページの『pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー』

con_elapsed_time 最新の接続経過時間

エレメント ID con_elapsed_time

エレメント・タイプ 時間

表 726. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	タイム・スタンプ

説明 このホスト・データベースから最後に切断された DCS アプリケーションが接続されていた経過時間。

使用法 アプリケーションがホスト・データベースへの接続を維持していた時間の長さの標識として、このエレメントを使用します。

関連資料:

- 269 ページの『pkg_cache_num_overflows パッケージ・キャッシュ・オーバーフロー』

gw_comm_errors 通信エラー

エレメント ID gw_comm_errors

エレメント・タイプ カウンター

表 727. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 DCS アプリケーションがホスト・データベースへの接続を試みたときに、または SQL ステートメントを処理していたときに通信エラー (SQL30081) が発生した回数。

使用法 時間をかけて通信エラーの数をモニターすることによって、 DB2 Connect ゲートウェイに特定のホスト・データベースとの接続の問題があるかどうかを評価できます。通常のエラーしきい値と考えられるものを設定できるため、エラーの数がこのしきい値を超えるたびに、通信エラーの調査を行うことができます。

管理用通知ログ に記録される通信エラー・ログとともに、このエレメントを問題判別に使用してください。

関連資料:

- 460 ページの『gw_comm_error_time 通信エラー時刻』
- 381 ページの『stmt_elapsed_time 最新のステートメント経過時間』

gw_comm_error_time 通信エラー時刻

エレメント ID gw_comm_error_time

エレメント・タイプ タイム・スタンプ

表 728. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
DCS データベース	dcs_dbase	タイム・スタンプ

説明 DCS アプリケーションがホスト・データベースへの接続を試みたときに、または SQL ステートメントを処理していたときに通信エラー (SQL30081) が最後に発生した日時。

使用法 このエレメントは、通信エラーおよび 管理用通知ログ に記録される通信エラー・ログと組み合わせて、問題判別に使用できます。

関連資料:

- 459 ページの『gw_comm_errors 通信エラー』

blocking_cursor ブロッキング・カーソル

エレメント ID blocking_cursor

エレメント・タイプ 情報

表 729. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	stmt	ステートメント
DCS ステートメント	dcs_stmt	ステートメント

表 730. イベント・モニター情報

イベント・タイプ	論理データ・グループ	モニター・スイッチ
デッドロックの詳細	event_detailed_dlconn	-
ステートメント	event_stmt	-

説明 このエレメントは、実行中のステートメントがブロッキング・カーソルを使用しているかどうかを示します。

使用法 照会のデータ転送にブロッキングを使用すると、パフォーマンスが改善されます。照会に使用される SQL はブロッキングの使用と関係があるため、多少の変更が必要になる場合があります。

トランザクション・プロセッサー・モニター

トランザクション・プロセッサー・モニターに関するモニター・エレメント

トランザクション・モニターまたはアプリケーション・サーバー (multi-tier) の環境では、アプリケーション・ユーザーは SQL 要求を直接的には発行しません。この場合、アプリケーション・ユーザーは、トランザクション・プロセッサー・モニター (UNIX または Windows NT サーバーで稼働する CICS、 TUXEDO、ENCINA など) あるいはアプリケーション・サーバーに対して、ビジネス・トランザクションの実行を要求します。ビジネス・トランザクションは、それぞれアプリケーションの一部で、これが SQL 要求をデータベース・サーバーに発行します。 SQL 要求は中間サーバーによって発行されるので、 SQL 要求の実行の原因となったクライアントについて、データベース・サーバーには情報がありません。

トランザクション・プロセッサー・モニター (TP モニター) トランザクションやアプリケーション・サーバー・コードの開発者は、 sqleseti - Set Client Information API を使用して、元のクライアントに関する情報をデータベース・サーバーに提供できます。この情報は以下に示すモニター・エレメントにあります。

- tpmon_client_userid TP モニター・クライアント・ユーザー ID : モニター・エレメント
- tpmon_client_wkstn TP モニター・クライアント・ワークステーション名 : モニター・エレメント
- tpmon_client_app TP モニター・クライアント・アプリケーション名 : モニター・エレメント
- tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング : モニター・エレメント

tpmon_client_userid TP モニター・クライアント・ユーザー ID

エレメント ID tpmon_client_userid

エレメント・タイプ 情報

表 731. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
DCS アプリケーション	dcs_appl	基本

説明 **sqleseti** API が使用された場合は、トランザクション・マネージャーが生成してサーバーに提供したクライアント・ユーザー ID。

使用法 アプリケーション・サーバーまたは TP モニター環境でこのエレメントを使用すると、トランザクションの実行対象になっているエンド・ユーザーを識別できます。

関連資料:

- 462 ページの『tpmon_client_wkstn TP モニター・クライアント・ワークステーション名』
- 462 ページの『tpmon_client_app TP モニター・クライアント・アプリケーション名』

- 463 ページの『tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング』

tpmon_client_wkstn TP モニター・クライアント・ワークステーション名

エレメント ID tpmon_client_wkstn

エレメント・タイプ 情報

表 732. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
DCS アプリケーション	dcs_appl	基本

説明 この接続で **sqleseti** API が発行された場合に、クライアントのシステムまたはワークステーション (CICS EITERMID など) を示します。

使用法 このエレメントを使用すると、ノード ID、端末 ID、またはその他の同様の ID を使用して、ユーザーのマシンを識別できます。

関連資料:

- 461 ページの『tpmon_client_userid TP モニター・クライアント・ユーザー ID』
- 462 ページの『tpmon_client_app TP モニター・クライアント・アプリケーション名』
- 463 ページの『tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング』

tpmon_client_app TP モニター・クライアント・アプリケーション名

エレメント ID tpmon_client_app

エレメント・タイプ 情報

表 733. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
DCS アプリケーション	dcs_appl	基本

説明 この接続で **sqleseti** API が発行された場合に、トランザクションを実行中のサーバー・トランザクション・プログラムを示します。

使用法 このエレメントは、問題判別およびアカウンティングの目的で使用します。

関連資料:

- 461 ページの『tpmon_client_userid TP モニター・クライアント・ユーザー ID』
- 462 ページの『tpmon_client_wkstn TP モニター・クライアント・ワークステーション名』
- 463 ページの『tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング』

tpmon_acc_str TP モニター・クライアント・アカウンティング・ストリング

エレメント ID tpmon_acc_str

エレメント・タイプ 情報

表 734. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
アプリケーション	appl_info	基本
DCS アプリケーション	dcs_appl	基本

説明 この接続で **sqleseti** API が発行された場合に、ロギングおよび診断の目的でターゲット・データベースに渡されたデータです。

使用法 このエレメントは、問題判別およびアカウンティングの目的で使用します。

関連資料:

- 461 ページの『tpmon_client_userid TP モニター・クライアント・ユーザー ID』
- 462 ページの『tpmon_client_wkstn TP モニター・クライアント・ワークステーション名』
- 462 ページの『tpmon_client_app TP モニター・クライアント・アプリケーション名』

フェデレーテッド・データベース・システム

フェデレーテッド・データベース・システムに関するモニター・エレメント

フェデレーテッド・システムは、リモート・データ・アクセスを提供するマルチデータベース・サーバーです。これにより、 IBM 製および他社製のリレーショナルおよび非リレーショナルの異なるプラットフォーム上に常駐するさまざまなデータ・ソースにクライアントがアクセスできるようになります。分散データへのアクセスを統合して、異機種混合環境で単一のデータベース・イメージをユーザーに提供します。

次のエレメントにより、 DB2 フェデレーテッド・システム内で実行される各アプリケーションからデータ・ソースへのすべてのアクセスに関する情報、およびフェデレーテッド・サーバー・インスタンス内で実行される特定のアプリケーションからデータ・ソースへのアクセスに関する情報が示されます。次のエレメントがあります。

- datasource_name データ・ソース名 : モニター・エレメント
- disconnects 切断回数 : モニター・エレメント
- insert_sql_stmts 挿入回数 : モニター・エレメント
- update_sql_stmts 更新回数 : モニター・エレメント
- delete_sql_stmts 削除回数 : モニター・エレメント
- create_nickname ニックネーム作成回数 : モニター・エレメント
- passthrus パススルー数 : モニター・エレメント

- stored_procs ストアード・プロシージャー数 : モニター・エレメント
- remote_locks リモート・ロック数 : モニター・エレメント
- sp_rows_selected ストアード・プロシージャーによって戻された行数 : モニター・エレメント
- select_time 照会応答時間 : モニター・エレメント
- insert_time 挿入応答時間 : モニター・エレメント
- update_time 更新応答時間 : モニター・エレメント
- delete_time 削除応答時間 : モニター・エレメント
- create_nickname_time ニックネーム作成応答時間 : モニター・エレメント
- passthru_time パススルー時間 : モニター・エレメント
- stored_proc_time ストアード・プロシージャー時間 : モニター・エレメント
- remote_lock_time リモート・ロック時間 : モニター・エレメント

datasource_name データ・ソース名

エレメント ID datasource_name

エレメント・タイプ 情報

表 735. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

説明 このエレメントには、フェデレーテッド・サーバーによってリモート・アクセス情報が表示されているデータ・ソースの名前が含まれています。このエレメントは、SYSCAT.SERVERS の 'SERVER' 列に対応しています。

使用法 このエレメントを使用すると、アクセス情報が収集されて戻されているデータ・ソースを識別できます。

disconnects 切断回数

エレメント ID disconnects

エレメント・タイプ カウンター

表 736. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースから切断した合計回数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、フェデレーテッド・サーバーがいずれかのア

プリケーションに代わってこのデータ・ソースから切断した合計回数を判別できます。このエレメントと CONNECT カウントを組み合わせて使用すると、データ・ソースに現在接続中であるとフェデレーテッド・サーバーのこのインスタンスが判断しているアプリケーションの数を判別できます。

insert_sql_stmts 挿入回数

エレメント ID insert_sql_stmts

エレメント・タイプ カウンター

表 737. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースに INSERT ステートメントを発行した合計回数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、フェデレーテッド・サーバーまたはアプリケーションによりこのデータ・ソースに対して行われたデータベース・アクティビティーのレベルを判別できます。

このエレメントを使用すると、次の公式を使用して、フェデレーテッド・サーバーまたはアプリケーションによるこのデータ・ソースへの書き込みアクティビティーのパーセンテージも判別できます。

```
書き込みアクティビティー =
   (INSERT ステートメント + UPDATE ステートメント + DELETE ステートメント )/
   (SELECT ステートメント + INSERT ステートメント + UPDATE ステートメント +
   DELETE ステートメント)
```

update_sql_stmts 更新回数

エレメント ID update_sql_stmts

エレメント・タイプ カウンター

表 738. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースに UPDATE ステートメントを発行した合計回数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始

- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、フェデレーテッド・サーバーまたはアプリケーションによりこのデータ・ソースに対して行われたデータベース・アクティビティーのレベルを判別できます。

このエレメントを使用すると、次の公式を使用して、フェデレーテッド・サーバーまたはアプリケーションによるこのデータ・ソースへの書き込みアクティビティーのパーセンテージも判別できます。

```
書き込みアクティビティー =
  (INSERT ステートメント + UPDATE ステートメント + DELETE ステートメント ) /
  (SELECT ステートメント + INSERT ステートメント + UPDATE ステートメント +
  DELETE ステートメント)
```

delete_sql_stmts 削除回数

エレメント ID delete_sql_stmts

エレメント・タイプ カウンター

表 739. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースに DELETE ステートメントを発行した合計回数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、フェデレーテッド・サーバーまたはアプリケーションによりこのデータ・ソースに対して行われたデータベース・アクティビティーのレベルを判別できます。

このエレメントを使用すると、次の公式を使用して、フェデレーテッド・サーバーまたはアプリケーションによるこのデータ・ソースへの書き込みアクティビティーのパーセンテージも判別できます。

```
書き込みアクティビティー =
  (INSERT ステートメント + UPDATE ステートメント + DELETE ステートメント ) /
  (SELECT ステートメント + INSERT ステートメント + UPDATE ステートメント +
  DELETE ステートメント)
```

create_nickname ニックネーム作成回数

エレメント ID create_nickname

エレメント・タイプ カウンター

表 740. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソース上にあるオブジェクトのニックネームを作成した合計回数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、このフェデレーテッド・サーバー・インスタンスまたはアプリケーションによるこのデータ・ソースへの CREATE NICKNAME アクティビティーの量を判別できます。 CREATE NICKNAME 処理を行うと、データ・ソース・カタログに対して複数の照会が実行されるので、このエレメントの値が大きい場合は、原因を突き止めるか、またはアクティビティーを制限する必要があります。

passthrus パススルー数

エレメント ID passthrus

エレメント・タイプ カウンター

表 741. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースに直接渡した SQL ステートメントの合計数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、フェデレーテッド・サーバーがネイティブに処理できる SQL ステートメントのパーセンテージ、およびパススルー・モードが必要なパーセンテージを判別できます。この値が大きい場合は、原因を突き止めて、ネイティブ・サポートをさらに効率的に使用する方法を検討してください。

stored_procs ストアード・プロシージャー数

エレメント ID stored_procs

エレメント・タイプ カウンター

表 742. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースで呼び出したストアード・プロシージャーの合計数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、フェデレーテッド・データベースでローカルに行われたストアード・プロシージャーの呼び出し数、またはアプリケーションがフェデレーテッド・データベースに対して呼び出したストアード・プロシージャーの呼び出し数を判別できます。

remote_locks リモート・ロック数

エレメント ID remote_locks

エレメント・タイプ カウンター

表 743. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にいずれかのアプリケーションに代わって、フェデレーテッド・サーバーがこのデータ・ソースで呼び出したリモート・ロックの合計数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントを使用すると、データ・ソースでリモート側で行われたリモート・ロックの数を判別できます。

sp_rows_selected ストアード・プロシージャーによって戻された行数

エレメント ID sp_rows_selected

エレメント・タイプ カウンター

表 744. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	基本
アプリケーション	appl_remote	基本

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのアプリケーションのストアード・プロシージャーの処理の結果として、データ・ソースからフェデレーテッド・サーバーに送信された行の数が含まれています。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

使用法 このエレメントは複数の目的に使用できます。次の公式を使用すると、ストアード・プロシージャー単位でデータ・ソースからフェデレーテッド・サーバーに送信された平均行数を計算できます。

```
    ストアード・プロシージャー単位の行数
= 戻された行数
/ 呼び出されたストアード・プロシージャーの数
```

このアプリケーションについて、データ・ソースからフェデレーテッド・サーバーに行を戻すときの平均時間も計算できます。

```
平均時間 = ストアード・プロシージャー応答合計時間/戻された行数
```

select_time 照会応答時間

エレメント ID select_time

エレメント・タイプ カウンター

表 745. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからの照会に対して、このデータ・ソースが応答に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間は、フェデレーテッド・サーバーがデータ・ソースから行を要求してからフェデレーテッド・サーバーがその行を利用できるようになるまでの時間です。

注: 照会ブロッキングが行われるため、フェデレーテッド・サーバーが行の読み取りを試行しても、その通信がすべて処理されるとは限りません。取得を要求した次の行は、戻される行のブロックに入っている可能性があります。そのため、照会応答合計時間は、データ・ソースでの処理を示すとは限らず、実際にはデータ・ソースまたはクライアントでの処理を示します。

使用法 このエレメントを使用すると、このデータ・ソースのデータを待機した実際の時間を判別できます。この情報は、キャパシティー・プランニング、CPU のチューニング、および SYSCAT.SERVERS の通信速度を調整するときに役に立ちます。これらのパラメーターを変更すると、オプティマイザーが要求をデータ・ソースに送信するかしないかに影響を与えます。

insert_time 挿入応答時間

エレメント ID insert_time

エレメント・タイプ カウンター

表 746. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからの INSERT に対して、このデータ・ソースが応答に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間とは、フェデレーテッド・サーバーが INSERT ステートメントをデータ・ソースにサブミットしてからデータ・ソースが INSERT を処理したことをフェデレーテッド・サーバーに応答するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースに対する INSERT が処理されるのを待機するために生じた実際の時間を判別できます。この情報は、キャパシティー・プランニングおよびチューニングを行うときに便利です。

update_time 更新応答時間

エレメント ID update_time

エレメント・タイプ カウンター

表 747. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからの UPDATE に対して、このデータ・ソースが応答に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間とは、フェデレーテッド・サーバーが UPDATE ステートメントをデータ・ソースにサブミットしてからデータ・ソースが UPDATE を処理したことをフェデレーテッド・サーバーに応答するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースに対する UPDATE が処理されるのを待機するために生じた実際の時間を判別できます。この情報は、キャパシティー・プランニングおよびチューニングを行うときに便利です。

delete_time 削除応答時間

エレメント ID delete_time

エレメント・タイプ カウンター

表 748. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからの DELETE に対して、このデータ・ソースが応答に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間とは、フェデレーテッド・サーバーが DELETE ステートメントをデータ・ソースにサブミットしてからデータ・ソースが DELETE を処理したことをフェデレーテッド・サーバーに応答するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースに対する DELETE が処理されるのを待機するために生じた実際の時間を判別できます。この情報は、キャパシティー・プランニングおよびチューニングを行うときに便利です。

create_nickname_time ニックネーム作成応答時間

エレメント ID create_nickname_time

エレメント・タイプ カウンター

表 749. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからの CREATE NICKNAME ステートメントに対して、このデータ・ソースが処理に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間は、フェデレーテッド・サーバーが CREATE NICKNAME ステートメントを処理するためにデータ・ソースから情報の検索を開始してから必要なデータの取り出しがすべて完了するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースでニックネームを作成するのに要した実際の時間を判別できます。

passthru_time パススルー時間

エレメント ID passthru_time

エレメント・タイプ カウンター

表 750. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからの PASSTHRU に対して、このデータ・ソースが応答に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間は、フェデレーテッド・サーバーが PASSTHRU ステートメントをデータ・ソースにサブミットしてからデータ・ソースが PASSTHRU ステートメントを処理したことを応答するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースでステートメントをパススルー・モードで処理するのに要した実際の時間を判別できます。

stored_proc_time ストアード・プロシージャー時間

エレメント ID stored_proc_time

エレメント・タイプ カウンター

表 751. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからのストアード・プロシージャー・ステートメントに対して、このデータ・ソースが応答に要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間は、フェデレーテッド・サーバーがストアード・プロシージャーをデータ・ソースにサブミットしてからデータ・ソースがストアード・プロシージャーを処理したことを応答するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースでストアード・プロシージャーの処理に要した実際の時間を判別できます。

remote_lock_time リモート・ロック時間

エレメント ID remote_lock_time

エレメント・タイプ カウンター

表 752. スナップショット・モニター情報

スナップショット・レベル	論理データ・グループ	モニター・スイッチ
データベース	dbase_remote	タイム・スタンプ
アプリケーション	appl_remote	タイム・スタンプ

スナップショット・モニターの場合、このカウンターはリセットできます。

説明 このエレメントには、次に示す時点以降にこのフェデレーテッド・サーバー・インスタンス上で実行されているすべてのアプリケーションまたは単一アプリケーションからのリモート・ロックに、このデータ・ソースが要した合計時間が含まれています (ミリ秒単位)。

- フェデレーテッド・サーバー・インスタンスの開始
- データベース・モニター・カウンターの最後のリセット

応答時間は、フェデレーテッド・サーバーがデータ・ソースにリモート・ロックをサブミットしてからフェデレーテッド・サーバーがデータ・ソースでリモート・ロックを解放するまでの時間です。

使用法 このエレメントを使用すると、このデータ・ソースでリモート・ロックに要した実際の時間を判別できます。

第 7 章 モニター・インターフェース

データベース・システム・モニター・インターフェース

モニター・タスク	API
スナップショットのキャプチャー	db2GetSnapshot - スナップショットの入手
自己記述型データ・ストリームの変換	db2ConvMonStream - モニター・ストリームの変換
データベース・システム・モニター・スイッチの表示	db2MonitorSwitches - モニター・スイッチの入手 / 更新
スナップショットのサイズ見積もり	db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり
モニター・スイッチの取得/更新	db2MonitorSwitches - モニター・スイッチの入手 / 更新
モニター・カウンターのリセット	db2ResetMonitor - モニターのリセット
データベース・システム・モニター・スイッチの更新	db2MonitorSwitches - モニター・スイッチの入手 / 更新

モニター・タスク	CLP コマンド
イベント・モニター出力の GUI ツールによる分析	db2eva - イベント・アナライザー・コマンド
スナップショットのキャプチャー	GET SNAPSHOT コマンド
データベース・マネージャー・モニター・スイッチの表示	GET DATABASE MANAGER MONITOR SWITCHES コマンド
モニター・アプリケーションのモニター・スイッチの表示	GET MONITOR SWITCHES コマンド
イベント・モニター・トレースのフォーマット	db2evmon - イベント・モニター生産性向上ツール・コマンド
表書き込み CREATE EVENT MONITOR ステートメントに関する SQL 例の生成	db2evtbl
アクティブなデータベースのリスト	LIST ACTIVE DATABASES コマンド
データベースに接続されたアプリケーションのリスト	LIST APPLICATIONS コマンド
DCS アプリケーションのリスト	LIST DCS APPLICATIONS コマンド
モニター・カウンターのリセット	RESET MONITOR コマンド
データベース・システム・モニター・スイッチの更新	UPDATE MONITOR SWITCHES コマンド

モニター・タスク	SQL ステートメント
イベント・モニターの活動化	SET EVENT MONITOR STATE ステートメント
イベント・モニターの作成	CREATE EVENT MONITOR ステートメント
イベント・モニターの非活動化	SET EVENT MONITOR STATE ステートメント
イベント・モニターの削除	DROP ステートメント
イベント・モニター値の書き込み	FLUSH EVENT MONITOR ステートメント

モニター・タスク	SQL 関数
イベント・モニターの状態の判別	EVENT_MON_STATE スカラー関数
データベース・マネージャー・レベルのスナップショットの取得	SNAPSHOT_DBM
データベース・マネージャー・レベルでの、現行のモニター・スイッチ設定値の取得	SNAPSHOT_SWITCHES
高速コミュニケーション・マネージャーのスナップショットの取得	SNAPSHOT_FCM
指定されたパーティションについての、高速コミュニケーション・マネージャーのスナップショットの取得	SNAPSHOT_FCM_NODE
データベース・レベルのスナップショットの取得	SNAPSHOT_DATABASE
アプリケーション・レベルのスナップショットの取得	SNAPSHOT_APPL
アプリケーション・レベルのスナップショットの取得	SNAPSHOT_APPL_INFO
ロック待機情報に関する、アプリケーション・レベルのスナップショットの取得	SNAPSHOT_LOCKWAIT
ステートメント情報に関する、アプリケーション・レベルのスナップショットの取得	SNAPSHOT_STATEMENT
エージェント情報に関する、アプリケーション・レベルのスナップショットの取得	SNAPSHOT_AGENT
サブセクション情報に関する、アプリケーション・レベルのスナップショットの取得	SNAPSHOT_SUBSECT
バッファー・プール・レベルのスナップショットの取得	SNAPSHOT_BP
表スペース・レベルのスナップショットの取得	SNAPSHOT_TBS
構成情報に関する、表スペース・レベルのスナップショットの取得	SNAPSHOT_TBS_CFG
コンテナー情報に関する、表スペース・レベルのスナップショットの取得	SNAPSHOT_TBS_CONTAINER
静止プログラム情報に関する、表スペース・レベルのスナップショットの取得	SNAPSHOT_QUIESCER
表スペース・マップの範囲に関する、表スペース・レベルのスナップショットの取得	SNAPSHOT_RANGES
表レベルのスナップショットの取得	SNAPSHOT_TABLE
ロック・レベルのスナップショットの取得	SNAPSHOT_LOCK
SQL ステートメントのキャッシュ情報のスナップショットの取得	SNAPSHOT_DYN_SQL

関連資料:

- 「*SQL リファレンス 第 1 巻*」の『EVENT_MON_STATE スカラー関数』
- 「*SQL リファレンス 第 2 巻*」の『CREATE EVENT MONITOR ステートメント』

- 「*SQL リファレンス 第 2 巻*」の『SET EVENT MONITOR STATE ステートメント』
- 「*SQL リファレンス 第 1 巻*」の『SYSCAT.EVENTMONITORS カタログ・ビュー』
- 「*SQL リファレンス 第 1 巻*」の『SYSCAT.EVENTS カタログ・ビュー』
- 「*管理 API リファレンス*」の『db2GetSnapshot - スナップショットの入手』
- 「*管理 API リファレンス*」の『db2MonitorSwitches - モニター・スイッチの入手 / 更新』
- 「*管理 API リファレンス*」の『db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり』
- 「*管理 API リファレンス*」の『db2ResetMonitor - モニターのリセット』
- 「コマンド・リファレンス」の『GET SNAPSHOT コマンド』
- 「コマンド・リファレンス」の『GET MONITOR SWITCHES コマンド』
- 「コマンド・リファレンス」の『GET DATABASE MANAGER MONITOR SWITCHES コマンド』
- 「コマンド・リファレンス」の『LIST APPLICATIONS コマンド』
- 「コマンド・リファレンス」の『LIST DCS APPLICATIONS コマンド』
- 「コマンド・リファレンス」の『RESET MONITOR コマンド』
- 「コマンド・リファレンス」の『LIST ACTIVE DATABASES コマンド』
- 「コマンド・リファレンス」の『db2eva - イベント・アナライザー・コマンド』
- 「コマンド・リファレンス」の『db2evmon - イベント・モニター生産性向上ツール・コマンド』
- 「*SQL リファレンス 第 2 巻*」の『FLUSH EVENT MONITOR ステートメント』
- 「*管理 API リファレンス*」の『db2ConvMonStream - モニター・ストリームの変換』

第 3 部 ヘルス・モニター・ガイド

第 8 章 ヘルス・モニターの紹介

ヘルス・モニターの概要

ヘルス・モニターとは、サーバー・サイドのツールの一種で、インスタンスとアクティブ・データベースの正常性を定期的にモニターして、例外による管理機能を追加します。ヘルス・モニターには、潜在的なシステムの正常性の問題についてデータベース管理者 (DBA) にアラートを出す機能もあります。ヘルス・モニターは、ハードウェア障害や、システム・パフォーマンスまたは機能の受諾不能に至りかねない問題を事前に検出します。ヘルス・モニターには事前の対策を講じる性質があるので、ユーザーは、システム・パフォーマンスに影響する問題に発展する前にその問題に取り組むことができます。

ヘルス・モニターは、ヘルス・インディケーターを使用してシステムの状態を検査し、アラートを発行する必要があるかどうかを判別します。アラートに応じて、事前構成済みのアクションが取られます。さらにヘルス・モニターは、管理通知ログにアラートを記録し、電子メールまたはページャーで通知を送信することもできます。この例外による管理モデルにより、アクティブ・モニターを必要とせずに、潜在的なシステムの正常性に関する問題に対するアラートを生成できるので、貴重な DBA リソースを解放できます。

ヘルス・モニターは、パフォーマンスを悪化させないインターフェースを使用して、システムの正常性に関する情報を収集します。情報を収集するのに、スナップショット・モニター・スイッチを ON にしません。

関連概念:

- 483 ページの『ヘルス・インディケーターのプロセスのサイクル』
- 489 ページの『ヘルス・モニター』

関連タスク:

- 485 ページの『ヘルス・アラート通知の使用可能化』

関連資料:

- 481 ページの『ヘルス・インディケーター』

ヘルス・インディケーター

ヘルス・モニターは、ヘルス・インディケーターを使用して、データベース・マネージャーやデータベースのパフォーマンスの特定の性質の正常性を評価します。ヘルス・インディケーターは、表スペースなど特定のクラスのデータベース・オブジェクトのある性質の正常性を測定します。基準は、正常性を判別するためのメジャーに適用されます。ここで適用される基準は、ヘルス・インディケーターのタイプに従属します。この基準に基づいて正常稼働ではないと判別されると、アラートが生成されます。

ヘルス・モニターによって、以下の 3 つのタイプのヘルス・インディケーターが戻されます。

- **しきい値ベースの**インディケーターは、オブジェクトの動作の (連続的な範囲の値の) 統計を表すメジャーである。警告およびアラームしきい値は、正常、警告、およびアラーム範囲の境界つまりゾーンを定義します。しきい値ベースのヘルス・インディケーターには、正常、警告、およびアラームの 3 つの有効な状態があります。
- **状態ベースの**インディケーターは、データベース・オブジェクトまたはリソースの操作が正常かどうかを定義するオブジェクトの、複数の異なる状態の限定集合を表すメジャーである。これらの状態の 1 つが通常で、その他の状態はすべて通常ではないと見なされます。状態ベースのヘルス・インディケーターには、通常およびアテンションの 2 つの有効な状態があります。
- **コレクション状態ベースの**インディケーターは、データベース中の 1 つ以上のオブジェクトのコレクション状態を表すデータベース・レベルのメジャーである。コレクション中のオブジェクトごとにデータがキャプチャーされ、これらのオブジェクトの間で最も重大な条件が集約状態として表されます。コレクション中の 1 つ以上のオブジェクトがアラートを必要とする状態である場合は、ヘルス・インディケーターはアテンションを表示します。コレクション状態ベースのヘルス・インディケーターには、通常およびアテンションの 2 つの有効な状態があります。

ヘルス・インディケーターは、インスタンス、データベース、表スペース、および表スペース・コンテナー・レベルです。

ヘルス・モニター情報には、ヘルス・センター、Web ヘルス・センター、 CLP、または API を介してアクセスできます。これらのツールを使用してヘルス・インディケーターの構成も行えます。

アラートは、正常の状態から正ではない状態への変化、または定義されたしきい値の境界に基づくヘルス・インディケーター値の警告またはアラームのゾーンへの変化に応答して生成されます。アラートには、アテンション、警告、およびアラームの 3 つがあります。

- 特定の状態を測定するヘルス・インディケーターの場合、通常でない状態が登録されると、アテンション・アラートが発行される。
- 値の連続範囲を計測するヘルス・インディケーターの場合は、正常、警告、およびアラームの各状態の境界やゾーンは、しきい値によって定義される。例えば、値がアラーム・ゾーンとして定義されている値のしきい値範囲に入ると、アラームのアラートが発行されて、問題に対して即時に対処が必要であることを示します。

関連概念:

- 489 ページの『ヘルス・モニター』

関連資料:

- 527 ページの『ヘルス・インディケーターの要約』
- 527 ページの『ヘルス・インディケーターの形式』

ヘルス・インディケーターのプロセスのサイクル

次の図は、ヘルス・インディケーターの評価プロセスを図示しています。ステップの集合は、特定のヘルス・インディケーターのリフレッシュ・インターバルが経過するたびに実行されます。

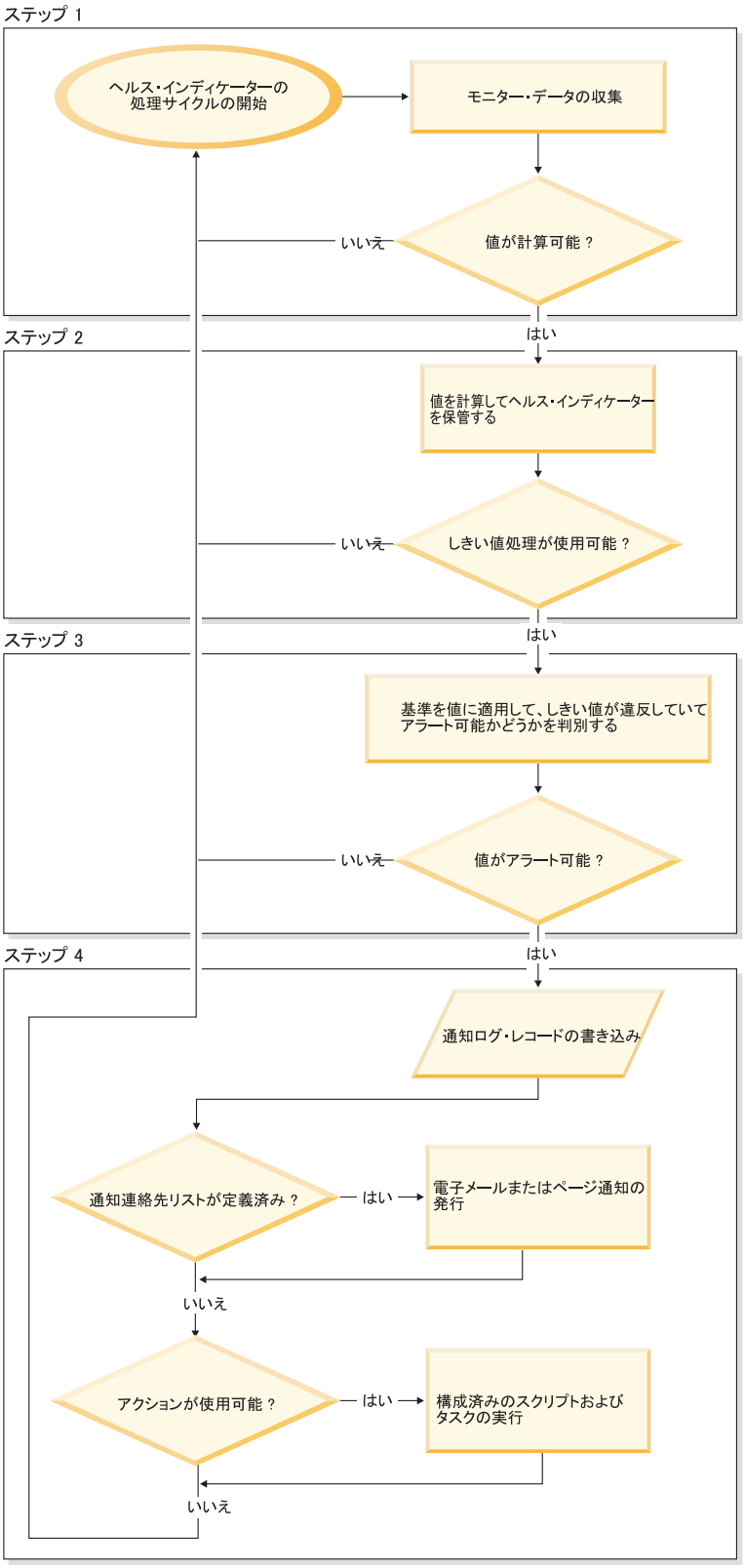

図 5. ヘルス・インディケーターのプロセスのサイクル

注:

1. NOTIFYLEVEL データベース・マネージャー構成パラメーターは、アラート通知が DB2® 管理通知ログと定義済みの連絡先に送信するかどうかを制御します。アラーム通知には重大度レベル 2 以上が必要です。警告およびアテンション・アラートを送信するには、重大度レベル 3 以上が必要です。
2. Windows® 上で DB2 UDB バージョン 7 のインストール内容にマイグレーションする際には、 NOTIFYLEVEL データベース・マネージャー構成パラメーターの値は更新されません。

関連資料:

- 「*コマンド・リファレンス*」の『UPDATE DATABASE MANAGER CONFIGURATION コマンド』
- 「*管理ガイド: パフォーマンス*」の『notifylevel - 「通知レベル」構成パラメーター』

ヘルス・アラート通知の使用可能化

アラートの生成時に電子メールまたはページャーで通知できるようにするには、構成パラメーターを設定して連絡先情報を指定しなければなりません。

前提条件:

連絡先リストのあるシステム上で、 DB2 Administration Server (DAS) が実行されている必要があります。例えば、CONTACT_HOST 構成パラメーターがリモート・システムに設定されている場合は、連絡先にアラートを通知するには、リモート・システム上で DAS が実行されていなければなりません。

手順:

ヘルス・アラート通知を使用可能にするには、次のようにします。

1. SMTP_SERVER パラメーターを指定します。

   DAS 構成パラメーター SMTP_SERVER は、電子メールとページャーの両方の通知メッセージを送信する際に使用するメール・サーバーの場所を指定します。DB2 UDB のインストール先のシステムが非認証 SMTP サーバーとして使用可能になっている場合は、このステップを省略してください。
2. CONTACT_HOST パラメーターを指定します。

   DAS 構成パラメーター CONTACT_HOST は、ローカル・システム上のすべてのインスタンス用の連絡先リストのリモート位置を指定します。このパラメーターを設定すると、複数のシステム間で 1 つの連絡先リストを共有できます。DB2 UDB のインストール先のローカル・システム上に連絡先リストを保持する場合は、このステップを省略してください。
3. ヘルス・モニター通知のデフォルトの連絡先を指定します。

   アラートの生成時にヘルス・モニターから電子メールまたはページャー通知を行えるようにするには、デフォルトの管理連絡先を指定しなければなりません。この情報を指定しないことを選択した場合は、アラート条件に関する情報メッセージを送信できません。

インストール中にデフォルトの管理連絡先情報を指定するか、またはインストールが完了するまでこの作業を遅らせることができます。

この作業を遅らせることを選択した場合や、通知リストにさらに連絡先かグループを追加したい場合は、 CLP、C API、またはヘルス・センターを使用して連絡先を指定できます。

CLP を使用して連絡先を指定するには、次のようにしてください。

ヘルス・モニター通知のデフォルトとして電子メールの連絡先を定義するには、次のコマンドを発行してください。

```
DB2 ADD CONTACT contact_name TYPE EMAIL ADDRESS
   email_address DESCRIPTION 'Default Contact'

DB2 UPDATE NOTIFICATION LIST ADD CONTACT contact_name
```

完全な構文の詳細については、「コマンド・リファレンス」を参照してください。

C API を使用して連絡先を指定するには、次のようにしてください。

次の C コードの抜粋は、正常性の通知の連絡先を定義する方法を図示しています。

```
...
#include <db2ApiDf.h>

SQL_API_RC rc = 0;
struct db2AddContactData addContactData;
struct sqlca sqlca;

char* userid = "myuser";
char* password = "pwd";
char* contact = "DBA1";
char* email = "dba1@mail.com";
char* desc = "Default contact";

memset(&addContactData, '¥0', sizeof(addContactData));
memset (&sqlca, '¥0', sizeof(struct sqlca));

addContactData.piUserid = userid;
addContactData.piPassword = password;
addContactData.piName = contact;
addContactData.iType = DB2CONTACT_EMAIL;
addContactData.piAddress = email;
addContactData.iMaxPageLength = 0;
addContactData.piDescription = desc;

rc = db2AddContact(db2Version810, &addContactData, &sqlca);

if (rc == 0) {
   db2HealthNotificationListUpdate update;
   db2UpdateHealthNotificationListData data;
   db2ContactTypeData contact;

   contact.pName = contact;
   contact.contactType = DB2CONTACT_EMAIL;

   update.iUpdateType = DB2HEALTHNOTIFICATIONLIST_ADD;
   update.piContact = &contact;

   data.iNumUpdates = 1;
   data.piUpdates = &update;
```

```
   rc = db2UpdateHealthNotificationList (db2Version810, &data, &ca);
}
...
```

ヘルス・センターを使用して連絡先を指定するには、次のようにしてください。

a. 正常性の通知リストを定義したいインスタンスを右マウス・ボタンでクリックします。

b. **「構成」**をクリックしてから、**「アラート通知」**をクリックします。「ヘルス・アラート通知の構成」ウィンドウが開きます。

c. ウィンドウの左側の**「選択可能」**リストに連絡先が表示されない場合は、**「連絡先の管理」**をクリックします。「連絡先」ウィンドウが開き、システム名が事前選択されています。

d. **「連絡先の追加」**をクリックします。「連絡先の追加」ウィンドウが開きます。

e. 名前と電子メール・アドレスを指定して、連絡先を定義します。ページャーの電子メール・アドレスを指定する場合は、**「アドレスはページャー用」**を選択します。

f. **「OK」**をクリックします。

g. 「連絡先」ウィンドウを閉じて、「ヘルス・アラート通知の構成」ウィンドウに戻ります。この時点で、新しい連絡先が**「選択可能な連絡先」**リストに表示されています。

h. 右矢印ボタンをクリックして、連絡先を**「ヘルス通知コンタクト・リスト」**に移動します。

i. **「OK」**をクリックして、正常性の通知リストに連絡先を組み込みます。

推奨 通知に関する障害が生じている場合は、「ヘルス通知コンタクト・リスト」の下の**「トラブルシューティング」**を選択してください。「ヘルス・アラート通知のトラブルシューティング」ウィザードが開きます。

関連資料:

- 「コマンド・リファレンス」の『UPDATE ADMIN CONFIGURATION コマンド』
- 「コマンド・リファレンス」の『ADD CONTACT コマンド』
- 「コマンド・リファレンス」の『UPDATE HEALTH NOTIFICATION CONTACT LIST コマンド』

第 9 章 ヘルス・モニターの使用法

ヘルス・モニター

ヘルス・モニターは、データベース・マネージャー、データベース、表スペース、および表スペース・コンテナーに関する情報をキャプチャーします。ヘルス・モニターは、データベース・システム・モニター・エレメント、オペレーティング・システム、および DB2® UDB から取り出されるデータに基づいて、ヘルス・インディケーターを計算します。ヘルス・モニターがデータベースとそのオブジェクトに関するヘルス・インディケーターを評価できるのは、データベースがアクティブの場合に限ります。 ACTIVATE DATABASE コマンドを使用してデータベースを開始するか、データベースに対する永続接続を保守することにより、データベースをアクティブにしておくこともできます。

ヘルス・モニターは、ヘルス・インディケーターごとに最大数の履歴レコードを保存します。この履歴は、 <instance path>¥hmonCache ディレクトリーに格納され、ヘルス・モニターの停止時に除去されます。ヘルス・モニターは、レコードの最大数に達した時点で、古い履歴レコードを自動的に整理します。

ヘルス・モニター・データにはヘルス・スナップショットを使用してアクセスできます。個々のヘルス・スナップショットは、最新のリフレッシュ・インターバルに基づいて、ヘルス・インディケーターごとに状況を報告します。スナップショットは、既存のデータベースの正常性に関する問題を検出したり、データベース環境で正常性が低くなる可能性を予測したりするのに便利です。ヘルス・スナップショットは、 C または C++ アプリケーション中の API を使用して CLP からキャプチャーしたり、グラフィック管理ツールを使用することによってキャプチャーしたりすることができます。

ヘルス・モニターでは、インスタンス・アタッチメントが必要です。 ATTACH TO コマンドを使用してインスタンスへのアタッチメントが確立されていない場合、ローカル・インスタンスへのデフォルトのインスタンス・アタッチメントが作成されます。

パーティション・データベース環境では、スナップショットは、インスタンスの任意のパーティションでとることも、単一のインスタンス接続を使用してグローバルにとることもできます。グローバル・スナップショットは、それぞれのパーティションで収集されたデータを集約して単一の値セットを戻します。

使用上の注意:

ヘルス・モニターは、DB2 UDB のすべてのエディションでサポートされています。

Windows® では、DB2 インスタンスのサービスは、 SYSADM 権限のあるアカウントの下で実行する必要があります。管理者特権のあるアカウントを使用するには、`db2icrt` コマンド上で「-u」オプションを使用するか、または Windows の「サービス」フォルダーを使用して、「ログオン」プロパティーを編集します。

Windows 上では、ヘルス・モニターのプロセスのことを DB2FMP と呼びます。

通知が送信され、アラート・アクションが実行されるには、ヘルス・モニターのあるシステム上で、 DB2 Administration server が実行されていなければなりません。リモートのスクリプト、タスク、または連絡先リストを使用する場合は、リモート・システム上で DB2 Administration server も開始しなければなりません。

ツール・カタログ・データベースは、タスクを作成する場合のみ必要です。ヘルス・インディケーターに関するアラート・タスク・アクションを使用しない場合は、ヘルス・モニターにはツール・カタログ・データベースは必要ありません。

関連概念:

- 481 ページの『ヘルス・モニターの概要』

関連タスク:

- 491 ページの『SQL 表関数を使用したデータベースのヘルス・スナップショットのキャプチャー』
- 493 ページの『CLP を使用したデータベースのヘルス・スナップショットのキャプチャー』
- 495 ページの『クライアント・アプリケーションからのデータベースのヘルス・スナップショットのキャプチャー』

関連資料:

- 499 ページの『ヘルス・モニターの出力例』
- 501 ページの『グローバル・ヘルス・スナップショット』

ヘルス・インディケーターのデータ

ヘルス・モニターは、個々のデータベース・パーティションに関するヘルス・インディケーターごとに、以下を含むデータの集合を記録します。

- ヘルス・インディケーター名
- 値
- 評価タイム・スタンプ
- アラート状態
- 公式 (該当する場合)
- 追加情報 (該当する場合)
- 最新のヘルス・インディケーター評価の履歴 (最大 10)。個々の履歴項目は、次のヘルス・インディケーター評価を、現行ヘルス・インディケーターの出力に至るまでキャプチャーします。
  - 値
  - 公式 (該当する場合)
  - アラート状態
  - タイム・スタンプ

ヘルス・モニターは、インスタンス、データベース、および表スペース・レベルで、最大重大度アラート状態の追跡も行います。それぞれのレベルで、このヘル

ス・インディケーターは、そのレベルと、そのレベルより低いすべてのレベルのヘルス・インディケーターに関する、既存の最大重大度アラートを表します。例えば、インスタンスに関する最大重大度アラート状態には、インスタンス、そのすべてのデータベース、およびそれらのデータベースごとのすべての表スペースと表スペース・コンテナーに関するヘルス・インディケーターが含まれます。

関連資料:

- 535 ページの『インスタンス最大重大度アラート状態』
- 536 ページの『データベース最大重大度アラート状態』

SQL 表関数を使用したデータベースのヘルス・スナップショットのキャプチャー

SQL 表関数を使用して、データベースのヘルス・スナップショットをキャプチャーすることができます。使用可能なそれぞれのヘルス・スナップショット表関数は、ヘルス・スナップショット要求タイプに対応しています。

手順:

SQL 表関数を使用して、データベースのヘルス・スナップショットをキャプチャーするには、次のようにします。

1. 使用しようとしている SQL 表関数を識別します。

   SQL 表関数には、以下の 2 つの入力パラメーターがあります。

   - VARCHAR(255): データベース名。
   - INT: パーティション番号 (0 から 999 の間の値)。モニターしたいパーティション番号に対応する整数を入力します。現在接続しているパーティションのスナップショットをキャプチャーする場合は、値 -1 を入力します。グローバル・スナップショットをキャプチャーするには、値 -2 を入力します。

   例外: データベース・マネージャーのスナップショット SQL 表関数だけはこの規則の例外です。それには 1 つのパラメーターしかありません。この 1 つのパラメーターは、パーティション番号のパラメーターです。データベース名パラメーターに NULL を入力すると、モニターは、表関数の呼び出しに使用される接続によって定義されたデータベースを使用します。

2. SQL ステートメントを発行します。

   以下の例では、現在接続されているパーティション、およびこの表関数呼び出しが行われた接続によって定義されたデータベース上についての、基本的なヘルス・スナップショットをキャプチャーします。

   ```
   SELECT * FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1))
          as HEALTH_DB_INFO
   ```

   戻り表から個々のモニター・エレメントを選択することもできます。戻り表の各列は、モニター・エレメントに対応しています。したがって、各モニター・エレメントの列名は、モニター・エレメントの名前に対応しています。次のステートメントの場合は、 db_path と server_platform のモニター・エレメントだけが戻されます。

```
SELECT db_path, server_platform
       FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1 ) )
       as HEALTH_DB_INFO
```

関連概念:

- 489 ページの『ヘルス・モニター』

関連資料:

- 492 ページの『ヘルス・モニター SQL 表関数』
- 549 ページの『ヘルス・モニター・インターフェース』

ヘルス・モニター SQL 表関数

次の表に、スナップショットの表関数をすべてリストします。それぞれの表関数は、ヘルス・スナップショット要求タイプに対応しています。

表 753. スナップショット・モニター SQL 表関数

モニター・レベル	SQL 表関数	戻される情報
データベース・マネージャー	HEALTH_DBM_INFO	データベース・マネージャー・レベルからのヘルス・スナップショットに関する基本情報
データベース・マネージャー	HEALTH_DBM_HI	データベース・マネージャー・レベルからのヘルス・インディケーター情報
データベース・マネージャー	HEALTH_DBM_HI_HIS	データベース・マネージャー・レベルからのヘルス・インディケーター履歴情報
データベース	HEALTH_DB_INFO	データベースからのヘルス・スナップショットに関する基本情報
データベース	HEALTH_DB_HI	データベースからのヘルス・インディケーター情報
データベース	HEALTH_DB_HI_HIS	データベースからのヘルス・インディケーター履歴情報
データベース	HEALTH_DB_HIC	データベースのコレクション・ヘルス・インディケーターに関するコレクション情報
データベース	HEALTH_DB_HIC_HIS	データベースのコレクション・ヘルス・インディケーターに関するコレクション履歴情報
表スペース	HEALTH_TBS_INFO	データベースの表スペースのヘルス・スナップショットに関する基本情報
表スペース	HEALTH_TBS_HI	データベースの表スペースに関するヘルス・インディケーター情報
表スペース	HEALTH_TBS_HI_HIS	データベースの表スペースに関するヘルス・インディケーター履歴情報
表スペース	HEALTH_CONT_INFO	データベースのコンテナーのヘルス・スナップショットに関する基本情報
表スペース	HEALTH_CONT_HI	データベースのコンテナーに関するヘルス・インディケーター情報
表スペース	HEALTH_CONT_HI_HIS	データベースのコンテナーに関するヘルス・インディケーター履歴情報

関連概念:

- 489 ページの『ヘルス・モニター』

関連タスク:

- 493 ページの『CLP を使用したデータベースのヘルス・スナップショットのキャプチャー』

関連資料:

- 「*SQL リファレンス 第 1 巻*」の『サポートされている関数と SQL 管理ルーチン』
- 549 ページの『ヘルス・モニター・インターフェース』

CLP を使用したデータベースのヘルス・スナップショットのキャプチャー

CLP から GET HEALTH SNAPSHOT コマンドを使用して、ヘルス・スナップショットをキャプチャーすることができます。コマンド構文は、ヘルス・モニターによってモニターされたさまざまなタイプのヘルス・スナップショット情報の取り出しをサポートします。

前提条件:

ヘルス・スナップショットをキャプチャーするには、インスタンス・アタッチメントがなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンスのスナップショットを取得するには、まずそのインスタンスにアタッチする必要があります。

手順:

CLP を使用して、データベースのヘルス・スナップショットをキャプチャーするには、次のようにします。

1. CLP から、GET HEALTH SNAPSHOT コマンドを必要なパラメーターを指定して発行します。

   次の例では、データベース・マネージャーの開始直後に、データベース・マネージャー・レベルのヘルス・スナップショットがキャプチャーされます。

   ```
   db2 get health snapshot for dbm
   ```

2. パーティション・データベース・システムの場合、特定のパーティションについてデータベース・スナップショットを固有にキャプチャーすることも、すべてのパーティションについてグローバルなスナップショットをキャプチャーすることもできます。特定のパーティション (例えば、パーティション番号 2) 上のデータベースのヘルス・スナップショットをキャプチャーするには、次のコマンドを発行します。

   ```
   db2 get health snapshot for db on sample at dbpartitionnum 2
   ```

   すべてのパーティション上のすべてのアプリケーションについてのデータベース・スナップショットをキャプチャーするには、次のコマンドを発行します。

   ```
   db2 get health snapshot for db on sample global
   ```

   次のコマンドは、公式、追加情報、およびヘルス・インディケーター履歴を含む追加の詳細情報のある、ヘルス・スナップショットをキャプチャーします。

```
db2 get health snapshot for db on sample show detail
```

3. コレクション状態ベースのヘルス・インディケーターの場合、状態にかかわらず、すべてのコレクション・オブジェクトについてデータベース・スナップショットをキャプチャーすることができます。正規の GET HEALTH SNAPSHOT FOR DB コマンドは、すべてのコレクション状態ベースのヘルス・インディケーターについて、アラートが必要なすべてのコレクション・オブジェクトを戻します。

   すべてのコレクション・オブジェクトをリストしてデータベースのヘルス・スナップショットをキャプチャーするには、次のコマンドを発行します。

```
db2 get health snapshot for db on sample with full collection
```

関連概念:

- 489 ページの『ヘルス・モニター』

関連資料:

- 「コマンド・リファレンス」の『GET HEALTH SNAPSHOT コマンド』
- 494 ページの『ヘルス・モニター CLP コマンド』
- 499 ページの『ヘルス・モニターの出力例』
- 501 ページの『グローバル・ヘルス・スナップショット』

ヘルス・モニター CLP コマンド

次の表に、サポートされているスナップショット要求のタイプをすべてリストします。

表 754. スナップショット・モニター CLP コマンド

モニター・レベル	CLP コマンド	戻される情報
データベース・マネージャー	`get health snapshot for dbm`	データベース・マネージャー・レベル情報。
データベース	`get health snapshot for all database`	データベース・レベル情報。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	`get health snapshot for database on` *database-alias*	データベース・レベル情報。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	`get health snapshot for all on` *database-alias*	データベース、表スペース、および表スペース・コンテナーの情報。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
表スペース	`get snapshot for tablespaces on` *database-alias*	データベースに接続されたアプリケーションがアクセスしている各表スペースの表スペース・レベルの情報。また、表スペース中の各表スペース・コンテナーに関する正常性の情報も含みます。

関連タスク:

- 493 ページの『CLP を使用したデータベースのヘルス・スナップショットのキャプチャー』

関連資料:

- 「コマンド・リファレンス」の『GET HEALTH SNAPSHOT コマンド』
- 549 ページの『ヘルス・モニター・インターフェース』

クライアント・アプリケーションからのデータベースのヘルス・スナップショットのキャプチャー

C または C++ アプリケーションでスナップショット・モニター API を使用して、ヘルス・スナップショットをキャプチャーすることができます。 db2GetSnapshot API にパラメーターを指定することにより、多数の異なるヘルス・スナップショット要求タイプにアクセスすることができます。

前提条件:

ヘルス・スナップショットをキャプチャーするには、インスタンスにアタッチしていなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンスのスナップショットを取得するには、まずそのインスタンスにアタッチする必要があります。

手順:

1. コードに sqlmon.h および db2ApiDf.h DB2 ライブラリーを組み込みます。これらのライブラリーは、 sqllib¥include ディレクトリーにあります。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. スナップショットのバッファー単位サイズを 50 KB に設定します。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 51200
   ```

3. sqlma、sqlca、sqlm_collected、および db2GetSnapshotData 構造体を宣言します。

   ```
   struct sqlma *pRequestedDataGroups;
   struct sqlca sqlca;
   memset (&sqlca, '¥0', sizeof(struct sqlca));
   struct sqlm_collected collectedData;
   memset (&sqlm_collected, '¥0', sizeof(struct sqlm_collected));
   db2GetSnapshotData getSnapshotParam;
   memset(&db2GetSnapshotData, '¥0', sizeof(db2GetSnapshotData));
   ```

4. スナップショット・バッファーを含み、バッファーのサイズを設定するようにポインターを初期化します。

   ```
   static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
   sqluint32 outputFormat;
   char *snapshotBuffer;
   ```

5. sqlma 構造体を初期化し、キャプチャーしようとしているスナップショットがデータベース・マネージャー・レベル情報のものであることを指定します。

   ```
   pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
   memset(&pRequestedDataGroups, '¥0', sizeof(struct pRequestedDataGroups));
   pRequestedDataGroups->obj_num = 1;
   pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
   ```

6. スナップショット・リスト出力を保持するバッファーを初期化します。

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (&snapshotBuffer, '¥0', sizeof(snapshotBuffer));
```

7. db2GetSnapshotData 構造体に、スナップショット要求タイプ (sqlma 構造体から)、バッファー情報、およびスナップショットをキャプチャーするために必要な他の情報を含めます。

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION8;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_HEALTH;
```

8. ヘルス・スナップショットをキャプチャーします。以下のパラメーターを渡します。
   - db2GetSnapshotData 構造体。これには、スナップショットをキャプチャーするのに必要な情報が含まれています。
   - スナップショット出力の宛先となるバッファーの参照。

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. バッファーのオーバーフローを処理するためのロジックを組み込みます。スナップショットが取られた後、バッファー・オーバーフローについて sqlcode がチェックされます。バッファー・オーバーフローが発生する場合には、バッファーがクリアされて再初期化され、スナップショットが再度取られます。

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
  snapshotBufferSize += SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("¥nMemory allocation error.¥n");
    return;
  }

  getSnapshotParam.iBufferSize = snapshotBufferSize;
  getSnapshotParam.poBuffer = snapshotBuffer;
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. スナップショット・モニターのデータ・ストリームを処理します。スナップショット・モニターのデータ・ストリームを参照するには、これらのステップの後の図を参照してください。
11. バッファーをクリアします。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

db2GetSnapshot API でヘルス・スナップショットをキャプチャーした後、ヘルス・スナップショット出力が自己記述型データ・ストリームとして戻されます。以下は、データ・ストリーム構造の例です。

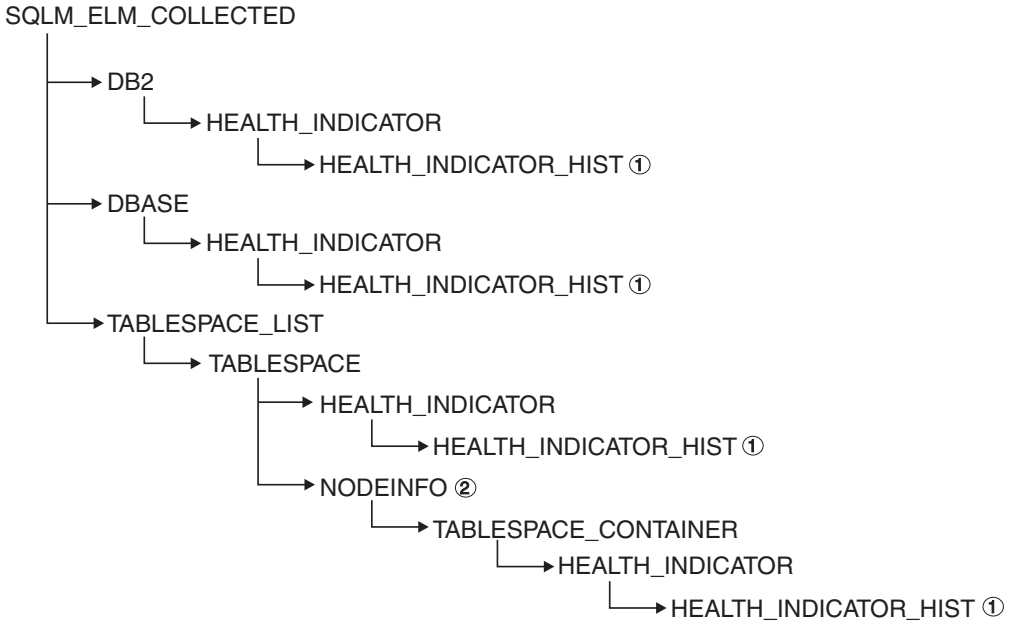

図 6. ヘルス・スナップショット自己記述型データ・ストリーム

凡例:

1. SQLM_CLASS_HEALTH_WITH_DETAIL スナップショット・クラスが使用される場合のみ使用可能。
2. DB2 UDB Enterprise Server Edition でのみ使用可能。それ以外の場合は、表スペース・コンテナー・ストリームになります。

次の階層は、ヘルス・スナップショット自己記述型データ・ストリーム中の特定のエレメントを示しています。

SQLM_ELM_HI の下のエレメントの階層:

```
SQLM_ELM_HI
   SQLM_ELM_HI_ID
   SQLM_ELM_HI_VALUE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
   SQLM_ELM_HI_ALERT_STATE
```

SQLM_ELM_HI_HIST の下のエレメントの階層 (SQLM_CLASS_HEALTH_WITH_DETAIL スナップショット・クラス使用時のみ使用可能):

```
SQLM_ELM_HI_HIST
   SQLM_ELM_HI_FORMULA
   SQLM_ELM_HI_ADDITIONAL_INFO
      SQLM_ELM_HEALTH_INDICATOR_HIST
         SQLM_ELM_HI_ID
         SQLM_ELM_HI_VALUE
            SQLM_ELM_HI_TIMESTAMP
               SQLM_ELM_SECONDS
               SQLM_ELM_MICROSEC
         SQLM_ELM_HI_ALERT_STATE
         SQLM_ELM_HI_FORMULA
         SQLM_ELM_HI_ADDITIONAL_INFO
```

SQLM_ELM_OBJ_LIST の下のエレメントの階層:

```
SQLM_ELM_HI_OBJ_LIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_DETAIL
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

SQLM_ELM_OBJ_LIST_HIST の下のエレメントの階層
(SQLM_CLASS_HEALTH_WITH_DETAIL スナップショット・クラス使用時のみ使用可能):

```
SQLM_ELM_HI_OBJ_LIST_HIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

関連概念:

- 6 ページの『システム・モニター出力 : 自己記述型データ・ストリーム』
- 46 ページの『スナップショット・モニター自己記述型データ・ストリーム』
- 489 ページの『ヘルス・モニター』

関連資料:

- 「*管理 API リファレンス*」の『db2GetSnapshot - スナップショットの入手』
- 「*管理 API リファレンス*」の『db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり』
- 498 ページの『ヘルス・モニター API 要求タイプ』

ヘルス・モニター API 要求タイプ

次の表に、サポートされているスナップショット要求のタイプをすべてリストします。

表 755. スナップショット・モニター API 要求タイプ

モニター・レベル	API 要求タイプ	戻される情報
データベース・マネージャー	SQLMA_DB2	データベース・マネージャー・レベル情報。
データベース	SQLMA_DBASE_ALL	データベース・レベル情報。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。
データベース	SQLMA_DBASE	データベース・レベル情報。情報が戻されるのは、最低 1 つのアプリケーションがデータベースに接続している場合だけです。

表 755. スナップショット・モニター API 要求タイプ (続き)

モニター・レベル	API 要求タイプ	戻される情報
表スペース	SQLMA_DBASE_TABLESPACES	データベースに接続されたアプリケーションがアクセスしている各表スペースの表スペース・レベルの情報。また、表スペース中の各表スペース・コンテナーに関する正常性の情報も含みます。

関連概念:

- 46 ページの『スナップショット・モニター自己記述型データ・ストリーム』

関連タスク:

- 495 ページの『クライアント・アプリケーションからのデータベースのヘルス・スナップショットのキャプチャー』

関連資料:

- 「*管理 API リファレンス*」の『db2GetSnapshot - スナップショットの入手』
- 「*管理 API リファレンス*」の『db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり』

ヘルス・モニターの出力例

次の例は、CLP を使用して取ったヘルス・スナップショットとそれらに対応する出力を示し、ヘルス・モニターの性質を図示します。この例の目的は、データベース・マネージャーの開始直後に全体の正常性に関する状況を検査することです。

例 1 の手順:

1. 次のように GET HEALTH SNAPSHOT コマンドを使用して、データベース・マネージャーのスナップショットを取ります。

   ```
   db2 get health snapshot for dbm
   ```

   CLP から GET HEALTH SNAPSHOT コマンドが発行された後、スナップショット出力が画面に送られます。

   ```
   Node name                                    =
   Node type                                    = Database Server with local
                                                  and remote clients

   Instance name                                = DB2
   Snapshot timestamp                           = 11-07-2002 12:43:23.613425

   Number of database partitions in DB2 instance  = 1
   Start Database Manager timestamp             = 11-07-2002 12:43:18.000108
   Instance highest severity alert state        = Not yet evaluated

   Health Indicators:

       Not yet evaluated
   ```

2. この出力を分析します。

   このヘルス・スナップショットから、インスタンスの最大重大度アラート状態が「Not yet evaluated」であることが分かります。ヘルス・モニターが開始されたばかりで、まだヘルス・インディケーターを評価していないので、インスタンスはこの状態です。

インスタンスの最大重大度アラート状態が変わっている場合、次のようにします。

- HEALTH_MON データベース・マネージャー構成パラメーターの値を検査して、ヘルス・モニターが ON かどうか判別する。
- HEALTH_MON=OFF の場合は、ヘルス・モニターが開始されていない。ヘルス・モニターを開始するには、 UPDATE DBM CFG USING HEALTH_MON ON コマンドを発行します。
- HEALTH_MON=ON の場合は、インスタンスにアタッチしてヘルス・モニターを活動化する。インスタンス・アタッチメントがある場合は、ヘルス・モニターをメモリー中にロードできなかった可能性があります。

CLP を使用してデータベースのヘルス・スナップショットをとる別の例を以下に概略します。

例 2 の前提条件:

- データベース接続が存在していなければならない。
- データベースが静止していなければならない。

例 2 の手順:

1. 次のように GET HEALTH SNAPSHOT コマンドを使用して、データベース・マネージャーのスナップショットを取ります。

   ```
   db2 get health snapshot for db on sample
   ```

   CLP から GET HEALTH SNAPSHOT コマンドが発行された後、スナップショット出力が画面に送られます。

   ```
                Database Health Snapshot

   Snapshot timestamp                          = 12-09-2002 11:44:37.793184

   Database name                               = SAMPLE
   Database path                               = E:¥DB2¥NODE0000¥SQL00002¥
   Input database alias                        = SAMPLE
   Operating system running at database server= NT
   Location of the database                    = Local
   Database highest severity alert state       = Attention

   Health Indicators:

    ...
      Indicator Name                              = db.log_util
         Value                                    = 60
         Unit                                     = %
         Evaluation timestamp                     = 12-09-2002 11:44:00.095000
         Alert state                              = Normal

      Indicator Name                              = db.db_op_status
         Value                                    = 2
         Evaluation timestamp                     = 12-09-2002 11:44:00.095000
         Alert state                              = Attention
   ```

2. この出力を分析します。

   このヘルス・スナップショットは、 *db.db_op_status* ヘルス・インディケーター上にアテンション・アラートがあることを示しています。値 2 は、データベースが静止状態であることを示しています。

関連概念:

- 489 ページの『ヘルス・モニター』

関連資料:

- 494 ページの『ヘルス・モニター CLP コマンド』

グローバル・ヘルス・スナップショット

パーティション・データベース・システムでは、現行パーティション、指定したパーティション、またはすべてのパーティションのヘルス・スナップショットをとることができます。パーティション・データベースのすべてのパーティションに渡ってグローバル・ヘルス・スナップショットをとる場合は、可能であれば結果が戻される前にデータが集約されます。

ヘルス・インディケーターの集約されたアラート状態は、すべてのデータベース・パーティション間の最大重大度アラート状態と等しくなります。追加情報と履歴データは、データベース・パーティション間で集約できないので使用できません。ヘルス・インディケーターの残りのデータは、以下の表に詳述されているように集約されます。

表 756. ヘルス・インディケーターの値、タイム・スタンプ、および公式データの集約

ヘルス・インディケーター	集約の詳細
• db2.db2_op_status • db2.sort_privmem_util • db2.mon_heap_util • db.db_op_status • db.sort_shrmem_util • db.spilled_sorts • db.log_util • db.log_fs_util • db.locklist_util • db.apps_waiting_locks • db.db_heap_util • db.db_backup_req • ts.ts_util	ヘルス・インディケーターの値は、最大値を含むパーティションから取得される。 評価タイム・スタンプと公式は、同じパーティションから取得される。
• db.max_sort_shrmem_util • db.pkgcache_hitratio • db.catcache_hitratio • db.shrworkspace_hitratio	ヘルス・インディケーターの値は、最小値を含むパーティションから取得される。 評価タイム・スタンプと公式は、同じパーティションから取得される。
• db.deadlock_rate • db.lock_escal_rate	ヘルス・インディケーターの値は、すべてのデータベース・パーティション間の値の合計になる。 評価タイム・スタンプと公式は集約できないので使用できない。
• ts.ts_op_status • tsc.tscont_op_status • tsc.tscont_util	ヘルス・インディケーターは集約されない。

表 756. ヘルス・インディケーターの値、タイム・スタンプ、および公式データの集約 (続き)

ヘルス・インディケーター	集約の詳細
• db.hadr_op_status • db.hadr_log_delay	これらのヘルス・インディケーターは、複数のパーティション・データベースではサポートされない。
• db.tb_reorg_req • db.tb_runstats_req • db.fed_nicknames_op_status • db.fed_servers_op_status	このヘルス・インディケーターは、1 つのパーティションのみに対して評価されるので、集約の必要はない。ヘルス・インディケーターを評価するパーティションからデータが戻されます。

注: 1 つのパーティション・オブジェクトに関するグローバル・スナップショットをとると、集約するパーティションがないので、出力にはすべての属性が含まれます。

関連概念:

- 489 ページの『ヘルス・モニター』

関連タスク:

- 491 ページの『SQL 表関数を使用したデータベースのヘルス・スナップショットのキャプチャー』
- 493 ページの『CLP を使用したデータベースのヘルス・スナップショットのキャプチャー』
- 495 ページの『クライアント・アプリケーションからのデータベースのヘルス・スナップショットのキャプチャー』

ヘルス・モニターのグラフィック・ツール

ヘルス・センター

ヘルス・センターは、例外による管理をサポートするために設計されたグラフィック管理ツールです。クライアントでカタログされているすべての Windows®、Linux、および UNIX® のインスタンスとデータベースの場合、ヘルス・センターは以下のものを備えます。

- すべてのインスタンスとそれらのデータベースのロールアップされたアラート状態を表示するセントラル・ロケーション
- インスタンス、データベース、およびそれらの子オブジェクトに関する現行アラートを表示するグラフィカル・インターフェース
- 現行アラートに関する詳細情報と推奨される解決方法にアクセスするグラフィカル・インターフェース

コマンド行からヘルス・センターを開始するには、 `db2hc` コマンドを入力してください。

Windows では、「スタート」メニューから、**「スタート」→「プログラム」→「IBM® DB2®」→「モニター・ツール」→「ヘルス・センター」**をクリックして、ヘルス・センターを開始することもできます。

ヘルス・センターの左側のパネルにはナビゲーション・ツリーがあり、右側のパネルにはアラート・ビューがあります。ナビゲーション・ビューの内容は、ナビゲーション・ビューの上部で選択したトグル・ボタンに基づいてフィルタリングされます。

ヘルス・センターは、**「任意のアラート状態のオブジェクト (Object in Any Alert State)」**トグル・ボタンが選択された状態で開きます。このボタンは、現行アラートと注意を払う必要のあるインスタンスを関連付けるのに役立ちます。**「すべてのオブジェクト」**トグル・ボタンを選択すると、クライアントでカタログされたすべての Windows、Linux、および UNIX インスタンスと、それぞれの状態が表示されます。アイコンのないインスタンスは、ヘルス・モニターを実行していないか、またはバージョン 8 より前のインスタンス (ヘルス・モニター機能のサポートがない) です。

インスタンスを選択すると、ヘルス・センターは、選択されたインスタンスに関するヘルス・モニターから状況を要求します。アラート・ビューには、インスタンス、そのすべてのデータベース、およびそれらのデータベースごとの表スペースと表スペース・コンテナーに関するすべての現行アラートが表示されます。ナビゲーション・ビュー内のインスタンスを展開して、子データベース・オブジェクトを選択すると、アラート・ビューは、選択されたデータベースとその表スペースまたは表スペース・コンテナーに関するアラートに制限されます。

ヘルス・センターの右上隅にリフレッシュ・アイコンがあります。リフレッシュ・アイコンをクリックして即時リフレッシュするか、特定のリフレッシュ・インターバルを設定すると、ヘルス・センターは現在の状況についてサーバー上のヘルス・モニターを照会します。この照会により、ヘルス・モニターがヘルス・インディケーター評価をリフレッシュすることはありません。個々のヘルス・インディケーターには、定義済みのリフレッシュ・インターバルがあります。ヘルス・インディケーターがアラート状態に関して再評価されるのは、リフレッシュ・インターバルが経過した場合だけです。ヘルス・センターの時間指定リフレッシュまたは要求リフレッシュのたびに、ヘルス・インディケーターの現在の状況のみ表示されます。

アラート・ビューには、特定のカスタマイズ列やソート順序を含むカスタマイズ・ビューを定義する機能があります。ヘルス・センターには、独自の命名およびカテゴリー化スキームにカスタマイズできる、事前定義済みのビューが 6 つあります。ウィンドウの下部にあるツールバーを使用するか、**「表示」**メニューで**「ユーザー設定表示」**を選択すると、事前定義済みのビューを選択できます。独自のカスタマイズ・ビューを定義するには、ウィンドウの下部にあるツールバーの**「表示」**ボタンをクリックするか、**「表示」**メニューを使用してください。アラート・ビューでデータを表示するために選択したビューは、次回のヘルス・センターの呼び出し時に記憶されています。

アラートに関する詳細情報を取得するには、アラート・ビューでアラート行を選択してください。**「選択」**メニューを使用するか、行を右クリックして、**「詳細表示」**を選択してください。「詳細」ウィンドウにはアラートに関する詳細情報が表示されます。この情報には、アラートが生じたオブジェクトやパーティション、公式 (該当する場合)、およびヘルス・インディケーターの値が含まれます。

しきい値ベースのヘルス・インディケーターの場合、アラート条件の判別に使用されたしきい値が表示されます。「詳細」ウィンドウにはヘルス・インディケーターに関する追加情報も表示されます。この情報には、構成パラメーターや、アラートのコンテキストを示す他のモニター・データが含まれる場合もあります。ヘルス・インディケーターの目的や測定対象の重要属性である理由を含む、ヘルス・インディケーターの説明が表示されます。

コレクション状態ベースのヘルス・インディケーターの場合、**「ヘルス・インディケーターのアラート状態 (Health Indicator Alert State)」**表の「オブジェクト」に、コレクション・オブジェクトのリストが表示されます。この表には、オブジェクト名、タイム・スタンプ、および詳細情報が表示されます。

詳細ページには**「履歴の表示」**ボタンがあります。 2 回目のヘルス・インディケーター評価のリフレッシュ以降、ヘルス・インディケーターの履歴レコードが保管されます。履歴レコードの保管後に限り、ヘルス・センター内の「履歴の表示」ダイアログに内容が表示されます。コレクション状態ベースのヘルス・インディケーターの場合、「履歴」ウィンドウ内で**「収集履歴の表示 (View Collection History)」**ボタンをクリックして、収集の履歴を表示できます。

ヘルス・センター状況ビーコン

ヘルス・センター状況ビーコンは、 DB2 管理ツール中で使用可能にできる表示標識です。他の DB2 管理ツールを処理している最中に、ヘルス・センターが開いていないと、このビーコンは現行アラートを通知します。このビーコンは、アラート条件のためにヘルス・センターを開くようユーザーにプロンプトを出すことを意図しています。

ヘルス・センター状況ビーコンには、2 種類の通知方式があります。 1 つ目の通知方式では、ポップアップ・メッセージが使用されます。もう 1 つの通知方式では、オープン・ウィンドウの状況表示行の右側に表示されるグラフィック・ビーコンが使用されます。このグラフィック・ビーコンには、1 回のクリックでヘルス・センターにアクセスできるボタンが組み込まれています。

両方のビーコン通知方式とも、「ツール設定」ダイアログを使用して使用可能にします。「 ポップアップによる通知」方式はポップアップ・メッセージ通知を制御し、「状況表示行による通知」方式は表示ビーコンを制御します。

Web ヘルス・センター

Web ヘルス・センターには、 Web ブラウザーか Web を使用できる携帯情報端末 (PDA) を使用してアクセスできます。表示されるインターフェースは、使用されるメディアに応じて変わりますが、内容は同じです。 Web ヘルス・センターは、通常ヘルス・センター全体を使用しており、現在は通常のアクセス・ポイントから離れているユーザーを対象としています。

Web ヘルス・センターには、特定のインスタンスとそのすべての子オブジェクトの正常性に関する情報を表示する機能が備えられています。 Web ヘルス・センターを使用してデータベース固有のコンテキストを選択することはできません。

Web ヘルス・センターを使用する際の最初のステップとして、インストール中にセットアップされた DB2 Web ツールのサイトでアプリケーションにログオンします。メインページから、「ヘルス・センター」タブ (または PDA 上のリンク) をクリックします。最初に、システムを選択して認証のためにユーザー ID とパスワードを入力するようプロンプトが出されます。次に、正常性に関する状況を表示したいインスタンスを選択しなければなりません。

DB2 Web ツールは、DB2 検索機能を使用して、ネットワーク上のシステム、インスタンス、およびデータベースを自動的にカタログできます。web.xml 中でフラグを false にして、これらの自動カタログを使用不可にすることを選択した場合は、アプリケーション・サーバー上のシステム、インスタンス、およびデータベースを手動でカタログしなければなりません。

インスタンスを選択すると、Web ヘルス・センターは、「現行アラート (Current® Alerts)」ページを開きます。このページには、選択したインスタンス、そのカタログされたデータベース、およびそれらのデータベース中の表スペースと表スペース・コンテナーで既存のアラートがすべてリストされます。アラートをクリックすると、ヘルス・センターで記述されたアラートの全詳細情報を入手できます。

PDA 中の Web ヘルス・センターの「記述」ビューにヘルス・インディケーターの詳細情報が表示されます。このビューは、「現行アラート (Current Alerts)」表示から最初にアクセスできる画面です。「記述」ビューの上部のリンクを使用して、ヘルス・インディケーターの推奨事項と履歴にアクセスできます。

ヘルス・インディケーターのアラートの解決

ヘルス・モニターには、ヘルス・アラートの解決や調査の際に行うステップを記述した推奨事項が備えられています。

推奨事項には、アラートに対して直接のアクションを取る際に実行できるスクリプトも含まれていることがあります。ヘルス・モニターから戻されるスクリプトは、ヘルス・モニターが稼働しているインスタンスで実行することを意図しています。クライアントがリモートに推奨事項を照会している場合は、SYSPROC.EXEC_DB2_SCRIPT ストアード・プロシージャーを使用して、意図されたインスタンスでスクリプトを実行できます。

推奨事項はサーバーから戻されます。クライアントのロケールのメッセージ・ファイルがサーバーにインストールされている場合は、推奨事項のテキストはクライアントのロケールになります。それ以外の場合は、推奨事項のテキストは英語で戻されます。

SQL による正常性の推奨事項の照会

SYSPROC.HEALTH_HI_REC ストアード・プロシージャーを使用して、 SQL で推奨事項を照会できます。 SYSPROC.HEALTH_HI_REC ストアード・プロシージャーを使用すると、推奨事項は次のような XML 文書で戻されます。

- sqllib¥misc ディレクトリー中にある正常性の推奨事項の XML スキーマ DB2RecommendationSchema.xsd に従ってフォーマットされている。
- UTF-8 でエンコードされ、クライアントの言語のテキストが含まれている。
- 推奨事項の集合を収集したものとして編成されている。個々の推奨事項の集合には、解決しようとしている問題 (ヘルス・インディケーター) が記述され、そのヘルス・インディケーターを解決する際の 1 つ以上の推奨事項が含まれます。この文書から検索できる情報に関する特定の詳細情報については、スキーマ定義を参照してください。

CLP を使用して入手できる情報はすべて、 SQL を使用して照会すると戻される XML 推奨文書でも入手できます。

SYSPROC.HEALTH_HI_REC ストアード・プロシージャーは、以下の引き数を取ります。

- ヘルス・インディケーター
- ヘルス・インディケーターがアラート状態になっているオブジェクトの定義

出力の推奨文書は BLOB として戻されます。したがって、CLP の場合は表示される出力の量が制限されるので、コマンド行からこのストアード・プロシージャーを処理しても効果的ではありません。戻された XML 文書を適切に解析してご希望のエレメントや属性を検索できる高水準言語 (C や Java™ など) を使用して、このストアード・プロシージャーを呼び出すことをお勧めします。

関連資料:

- 「*SQL Administrative Routines*」の『HEALTH_HI_REC プロシージャー』

CLP を使用した正常性の推奨事項の検索

CLP から GET RECOMMENDATIONS コマンドを使用して推奨事項を検索できます。このコマンド構文は、推奨事項を照会して、特定のオブジェクト上で現在アラート状態になっているヘルス・インディケーターなどの特定のヘルス・アラートを解決することをサポートしています。またこのコマンド構文は、特定のヘルス・インディケーターに関する推奨事項の完全集合の検索もサポートしています。このヘルス・インディケーターは、コマンドの実行時にアラート状態になっている必要はありません。特定のヘルス・インディケーターに関するアラートを解決する際の推奨事項は、単一パーティション・レベルかグローバル・レベルのいずれかで照会できます。

前提条件:

ヘルス・モニターから推奨事項を検索するには、インスタンス・アタッチメントがなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンス上のヘルス・モニターから推奨事項を取得するには、まずそのインスタンスにアタッチする必要があります。ヘルス・モニターから推奨事項を検索するには、特殊権限は必要ありません。

手順:

次の例では、 *db.db_op_status* ヘルス・インディケーターに関する推奨事項の全集合が戻されます。このコマンドを実行する際に、このヘルス・インディケーターはアラート状態になっている必要はありません。次のコマンドを実行して、*db.db_op_status* ヘルス・インディケーターに関するアラートを解決する場合に推奨できるアクションの全集合を参照できます。

```
db2 get recommendations for db.db_op_status

Recommendations:

Recommendation: Investigate rollforward progress.

A rollforward is in progress on the database due to an explicit
request from the administrator. You have to wait for the rollforward
to complete for the instance to return to active state.

Take one of the following actions:

Launch DB2 tool: Utility Status Manager

The Utility Status Manager allows you to monitor the progress
and change the priority of currently running utilities.

To open the Utility Status Manager:
1.  From the Control Center, expand the object tree until you find
    the database that you want.
2.  Right-click the database, and click Manage Utilities from the
    pop-up menu. The Utility Status Manager opens.

To view progress of the rollforward utility, right-click on the
rollforward utility and select View Progress Details.


From the Command Line Processor, issue the commands shown in the
following example to view the progress of the rollforward
utility:

LIST UTILITIES SHOW DETAIL


Recommendation: Unquiesce the database.

The database has been put into QUIESCE PENDING or QUIESCE state by
an explicit request from the administrator. If you have QUIESCE_CONNECT
authority, or are DBADM or SYSADM, you will still have access to the
database and will be able to use it normally. For all other users, new
connections to the database are not permitted and new units of work
cannot be started. Also, depending on the quiesce request, active
units of work will be allowed to complete or will be rolled back
immediately. You can issue an unquiesce to return to active state.

Take one of the following actions:

Launch DB2 tool: Control Center Unquiesce Database

The Control Center has an option on a database that can be used to
unquiesce the database.

To unquiesce a database:
1.  From the Control Center, expand the object tree until you find
    the database that you want.
2.  Right-click the database, and click Unquiesce from the pop-up
    menu. The database is unquiesced.


From the Command Line Processor, issue the commands shown in the
```

```
following example:

CONNECT TO DATABASE database-alias
UNQUIESCE DATABASE
```

この出力は、このヘルス・インディケーターに関する推奨事項が 2 つ考えられることを示しています。 1 つはデータベースの活動化で、もう 1 つはデータベースに関するロールフォワード進行状況の調査です。このコマンドを使用すると、特定のアラートの解決方法が要求されるのではなく、考えられる推奨事項がすべて照会されるので、ヘルス・モニターはこの事例の最善の推奨事項を識別できません。その結果、推奨事項の全集合が戻されます。

GET RECOMMENDATIONS コマンドを使用する別の例を次に示します。

データベース SAMPLE のヘルス・インディケーター *db.db_heap_util* がアラート状態になっていることに気付き、アラートの解決方法を判別したいとします。この場合、特定の問題を解決したいので、次の方法で GET RECOMMENDATIONS コマンドを発行できます。

```
db2 get recommendations for health indicator db.db_heap_util for database on sample

Problem:

    Indicator Name                           = db.db_heap_util
       Value                                 = 42
       Evaluation timestamp                  = 11/25/2003 19:04:54
       Alert state                           = Alarm
       Additional information                =

Recommendations:

Recommendation: Increase the database heap size.
Rank: 1

Increase the database configuration parameter dbheap sufficiently
to move utilization to normal operating levels. To increase the
value, set the new value of dbheap to be equal to
(pool_cur_size / (4096*U)) where U is the desired utilization rate.
For example, if your desired utilization rate is 60% of the warning
threshold level, which you have set at 75%, then
U = 0.6 * 0.75 = 0.45 (or 45%).

Take one of the following actions:

Execute the following scripts at the DB2 server (this can be done using
the EXEC_DB2_CMD stored procedure):

CONNECT TO DATABASE SAMPLE;
UPDATE DB CFG USING DBHEAP 149333;
CONNECT_RESET;


Launch DB2 tool: Database Configuration Window

The Database Configuration window can be used to view and update
database configuration parameters.

To open the Database Configuration window:
1.  From the Control Center, expand the object tree until you find
    the databases folder.
2.  Click the databases folder. Any existing database are displayed
    in the contents pane on the right side of the window.
3.  Right-click the database that you want in the contents pane, and
```

```
    click Configure Parameters in the pop-up menu. The Database
    Configuration window opens.

On the Performance tab, update the database heap size parameter as
suggested and click OK to apply the update.


Recommendation: Investigate memory usage of database heap.
Rank: 2

There is one database heap per database and the database manager uses
it on behalf of all applications connected to the database. The data
area is expanded as needed up to the maximum specified by dbheap.

For more information on the database heap, refer to the DB2
Information Center.

Investigate the amount of memory that was used for the database heap
over time to determine the most appropriate value for the database
heap configuration parameter. The database system monitor tracks the
highest amount of memory that was used for the database heap.

Take one of the following actions:

Launch DB2 tool: Memory Visualizer

The Memory Visualizer is used to monitor memory allocation within
a DB2 instance. It can be used to monitor overall memory usage, and
to update configuration parameters for individual memory components.

To open the Memory Visualizer:
1.  From the Control Center, expand the object tree until you find
    the instances folder.
2.  Click the instances folder. Any existing instances are displayed
    in the contents pane on the right side of the window.
3.  Right-click the instance that you want in the contents pane, and
    click View Memory Usage in the pop-up menu. The Memory Visualizer
    opens.

To start the Memory Visualizer from the command line issue the
db2memvis command.

The Memory Visualizer displays a hierarchical list of memory pools
for the database manager. Database Heap is listed under the Database
Manager Memory group for each database. On Windows, it is listed
under the Database Manager Shared Memory group.

Click the check box on the Show Plot column for the Database Heap
row to add the element to the plot.
```

この出力は、問題のサマリーと、問題を解決するための推奨事項の集合を示しています。ヘルス・モニターは、優先順位に従って推奨事項をランク付けしています。個々の推奨事項には、説明とアクションの集合が含まれていて、推奨アクションを実行する方法が示されています。

この出力と、前述の例で戻された出力を比較してください。特定のオブジェクト上のヘルス・アラートに関する推奨事項を照会すると、ヘルス・モニターは特定のアラートを解決しようとし、出力の問題のセクション中に解決しようとしたアラートに関する詳細情報を示すことができます。

またヘルス・モニターは、推奨事項のランキングを示したり、場合によってはアラートを解決するために実行できるスクリプトを生成するすることもできます。さらに、一部の推奨事項が特定の問題状態に該当しない場合に、ヘルス・モニターはそ

れらの推奨事項をリジェクトして非表示にできます。他方、最初の例のように、推奨事項がヘルス・インディケーター名のみで照会されると、考えられる推奨事項の全集合が常に戻されます。この場合、単に CLP コマンドは、ユーザーがアラートを示された場合に考慮する必要のあるアクションに関する情報を示します。

パーティション・データベース・システムの場合、特定のパーティション上のアラート状態になっているヘルス・インディケーターに関する推奨事項を照会することも、すべてのパーティションについてグローバルに照会することもできます。推奨事項をグローバルに照会すると、すべてのパーティション上のヘルス・インディケーターに適用される推奨事項の集合が戻されます。例えば、パーティション 1 と 3 上でヘルス・インディケーターがアラート状態になっている場合は、 2 つのスクリプトを収集したものが戻され、それぞれのスクリプトは別のパーティションに適用されます。

例:

次の例は、特定のパーティション (この例ではパーティション番号 2) 上のヘルス・インディケーターに関する推奨事項を照会する方法を示しています。

```
db2 get recommendations for health indicator db.db_heap_util
  for database on sample at dbpartitionnum 2
```

例:

次の例は、複数のパーティション上でアラート状態になっているヘルス・インディケーターを解決するための推奨事項の集合を検索する方法を示しています。

```
db2 get recommendations for health indicator db.db_heap_util
  for database on sample global
```

関連資料:

- 「コマンド・リファレンス」の『GET RECOMMENDATIONS コマンド』

クライアント・アプリケーションを使用した正常性の推奨事項の検索

C または C++ アプリケーションで db2GetRecommendations API を使用して、推奨事項を照会できます。 db2GetRecommendations API を使用すると、推奨事項は次のような XML 文書で戻されます。

- SQLLIB ディレクトリー中の MISC サブディレクトリー中にある正常性の推奨事項の XML スキーマ DB2RecommendationSchema.xsd に従ってフォーマットされている。
- UTF-8 でエンコードされ、クライアントの言語のテキストが含まれている。
- 推奨事項の集合を収集したものとして編成されている。個々の推奨事項の集合には、解決しようとしている問題 (ヘルス・インディケーター) が記述され、そのヘルス・インディケーターを解決する際の 1 つ以上の推奨事項が含まれます。この文書から検索できる情報に関する特定の詳細情報については、スキーマ定義を参照してください。

CLP を使用して入手できる情報はすべて、戻される XML 推奨文書でも入手できます。

前提条件:

ヘルス・スナップショットをキャプチャーするには、インスタンス・アタッチメントがなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンスに関する推奨事項を照会するには、まずそのインスタンスにアタッチする必要があります。

手順:

クライアント・アプリケーションを使用して正常性の推奨事項を検索するには、次のようにします。

1. sqlmon.h および db2ApiDf.h DB2 ヘッダー・ファイルを組み込みます。これらのファイルは、 sqllib¥include ディレクトリーにあります。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. sqlca および db2GetRecommendationsData 構造体を宣言します。

   ```
   struct sqlca sqlca ;
   db2GetRecommendationsData recData ;

   memset( &sqlca, '¥0', sizeof( struct sqlca ) ) ;
   memset( &recData, '¥0', sizeof( db2GetRecommendationsData ) ) ;
   ```

3. 推奨事項を検索するアラートに関する情報を、 db2GetRecommendationsData 構造体に取り込みます。次のコードの抜粋では、 Sample データベース上の *db2.db_heap_util* ヘルス・インディケーターに関する推奨事項が照会されます。

   ```
   recData.iSchemaVersion = DB2HEALTH_RECSCHEMA_VERSION8_2 ;
   recData.iNodeNumber = SQLM_CURRENT_NODE ;
   recData.iIndicatorID = SQLM_HI_DATABASE_HEAP_UTILIZATION  ;
   recData.iObjType = DB2HEALTH_OBJTYPE_DATABASE ;
   recData.piDbName = "SAMPLE" ;
   ```

4. db2GetRecommendations API を呼び出して、指定データベース上のこのヘルス・インディケーターのアラートに関する推奨事項を検索します。

   ```
   db2GetRecommendations(  db2Version820, &recData, &sqlca ) ;
   ```

5. sqlca 中に戻された sqlcode を検査して、発生したエラーをチェックします。 API 呼び出しが正常に実行された場合は、 db2GetRecommendationsData 構造体の poRecommendation フィールド中に戻された推奨事項の XML 文書を処理してください。 XML パーサーの選択項目を使用して、ご希望のエレメントまたは属性を抽出してください。 XML 文書から取り出せる情報に関する詳細は、 sqllib¥misc ディレクトリー中の DB2RecommendationSchema.xsd XML スキーマを参照してください。

6. db2GetRecommendations API によって割り振られたメモリーを解放します。こうすると、 db2GetRecommendationsData 構造体の poRecommendation フィールド中に戻された推奨事項の文書が解放されます。

   ```
   db2GetRecommendationsFree( db2Version820, &recData, &sqlca );
   ```

普通は、ヘルス・インディケーターがアラート状態になっているのを検出した時点で推奨事項を照会するので、通常は上記のコードとスナップショット API に対する呼び出しを組み合わせて、ヘルス・スナップショットを取ります。

関連資料:

- 「*管理 API リファレンス*」の『db2GetRecommendations - アラート状態のヘルス・インディケーターに関する推奨の入手』

ヘルス・センターを使用したアラートの解決

ヘルス・センターには、アラート条件に関する推奨アクションを検索してインプリメントするサポートが備えられています。

手順:

1. アラート・ビューでアラート行を選択します。**「選択」->「推奨アドバイザー」**をクリックするか、行を右マウス・ボタンでクリックして**「推奨アドバイザー」**をクリックします。推奨アドバイザーが開いて、「詳細表示」ウィンドウと同様の形式でアラートの詳細が表示されます。
2. 推奨アドバイザーのステップに従って、最適な推奨事項を選択します。推奨アドバイザーには、推奨事項をインプリメントする機能が備えられています。

推奨事項には、調査と推奨事項という 2 つのタイプがあります。推奨アドバイザーには、これらの推奨事項のタイプに関する次の 4 種類のアクションがサポートされています。

グラフィック管理ツールの立ち上げ
このオプションは、アラート条件の解決や調査を行うグラフィック・ツールを立ち上げます。このツールは、アラートが発生したオブジェクトのコンテキスト中で立ち上げられます。

構成パラメーターの更新
更新する必要のある構成パラメーターと、現行値および推奨値がリストされます。必要に応じて、推奨値を更新できます。

DB2 コマンド・スクリプトの実行
推奨アクションには、複数のコマンドが必要になる場合があります。 DB2 コマンド・スクリプトを使用すると、複数のコマンドを実行して、アラート条件を解決できます。例えば、再編成の必要性ヘルス・インディケーターには、ユーティリティーを実行する DB2 コマンド・スクリプト・アクションが備えられています。

代替解決方法のインプリメント
DB2 管理ツール・セットでアクションを実行できない場合に、代替方式を使用してアラート条件を解決するための指示が備えられています。

関連概念:

- 502 ページの『ヘルス・モニターのグラフィック・ツール』

Web ヘルス・センターを使用した構成パラメーター更新の適用

Web ヘルス・センターには、構成パラメーターの更新に関する推奨アクションを適用する直接リンク・サポートが備えられています。構成パラメーターの更新アクションには、アクションの完全記述と、 Web コマンド・センター中の推奨事項を適用する Web リンクが含まれます。

手順:

Web ヘルス・センターを使用して構成パラメーターの更新に関する推奨アクションを適用するには、次のようにします。

1. Web コマンド・センターを開くリンクを選択します。「コマンド」テキスト域に、構成パラメーターを更新するコマンド・テキストが表示されます。
2. 推奨事項を適用します。
3. 「ヘルス・センター」タブを選択して、元の推奨アクションのページに戻ります。

この他のタイプの推奨アクションは、Web ヘルス・センターではサポートされません。他のタイプの推奨アクションとしては、アクションの完全記述が備えられています。ユーザーに Web ツールを使用してアクションを適用できないことを警告する通知メッセージも備えられています。

関連概念:

- 502 ページの『ヘルス・モニターのグラフィック・ツール』

ヘルス・インディケーターの構成

ヘルス・インディケーターの構成

デフォルトのヘルス・モニター構成は、インストール時に備えられます。したがって、DB2® の開始直後に、ヘルス・モニターがデータベース環境の正常性を評価できることが保証されます。しかし、特定のユーザーの環境用の構成を使用して、ヘルス・モニターがヘルス・インディケーターを評価したりアラート状態に対応したりする動作を微調整できます。

構成を定義できるレベルは複数あります。 DB2 のインストール時に、ヘルス・インディケーターごとに工場出荷時設定のデフォルト構成が備えられます。初めてヘルス・モニターを開始する際に、工場出荷時設定のコピーにより、デフォルトのインスタンス設定とグローバル設定が備えられます。

インスタンス設定は、インスタンスに適用されます。グローバル設定は、インスタンス中のカスタマイズ設定が定義されていないデータベース、表スペース、および表スペース・コンテナーなどのオブジェクトに適用されます。

特定のデータベース、表スペース、または表スペース・コンテナーに関するヘルス・インディケーター設定を更新すると、更新されたヘルス・インディケーターのオブジェクト設定が作成されます。オブジェクト設定のデフォルトは、グローバル設定です。

ヘルス・モニターは、特定のデータベース、表スペース、表スペース・コンテナーに関するヘルス・インディケーターを処理する際に、オブジェクト設定を検査します。特定のヘルス・インディケーターの設定が一度も更新されていない場合は、デフォルトのグローバル設定を使用してヘルス・インディケーターが処理されます。ヘルス・モニターがインスタンスに関するヘルス・インディケーターを処理する際には、インスタンス設定が使用されます。

ヘルス・インディケーターごとに構成できる複数の属性を使用して、ヘルス・モニターの動作を変更できます。最初のパラメーターの集合 (評価フラグ、しきい値、

感度) は、ヘルス・モニターがヘルス・インディケーターに関するアラートを生成する時点を定義します。 2 番目のパラメーターの集合 (アクション・フラグ、アクション) は、アラート生成時のヘルス・モニターの実行内容を定義します。

評価フラグ

個々のヘルス・インディケーターには、アラート状態の評価を使用可能にしたり使用不可にしたりする評価フラグがあります。

警告およびアラームのしきい値

しきい値ベースのヘルス・インディケーターには、ヘルス・インディケーター値の警告およびアラーム領域を定義する設定があります。特定のデータベース環境に合わせて、警告およびアラームのしきい値に変更を加えることができます。

感度パラメーター

感度パラメーターは、アラートが生成される前に、ヘルス・インディケーター値がアラート状態になっていなければならない期間の最小値を定義します。感度値に関連した待機時間は、ヘルス・インディケーター値がアラート状態になった最初のリフレッシュ・インターバルの際に開始されます。この値を使用すると、リソースが一時的に使用できないだけでアラートが誤って生成されてしまうことを防止できます。

例:

ログ使用率 (*db.log_util*) ヘルス・インディケーターを使用する例を考慮しましょう。週単位で DB2 通知ログを検討するとします。第 1 週に、*db.log_util* の項目はアラーム状態になっています。この状態に関する通知を受け取ったことを思い出しましたが、 CLP からアラート状態を検査すると、ヘルス・インディケーターは正常な状態に戻っていました。第 2 週の後に、週の同じ時点で同じヘルス・インディケーターに関する 2 度目のアラーム通知項目を受け取りました。アラートが生成された 2 つの事例に関してデータベース環境のアクティビティーを調査して、コミットに長時間を要するアプリケーションが毎週実行されていたことが判明しました。このアプリケーションがコミットするまでの間に、短時間 (約 8 分から 9 分) ログが使用できなくなっていました。通知ログ中のアラーム通知レコード中の履歴項目から、 *db.log_util* ヘルス・インディケーターが 10 分ごとに評価されていたことを判別できます。アラートが生成されているので、アプリケーション時間はこのリフレッシュ・インターバルをまたいでいるはずです。そこで、 *db.log_util* パラメーターの感度を 10 分に設定します。これで、*db.log_util* の値が初めて警告またはアラームのしきい値の領域に入るたびに、アラートが生成されるにはその前にこの値が 10 分以上その領域に入り続けていなければなりません。アプリケーション時間は 8、9 分のみなので、この状態に関する通知項目が通知ログに記録されることはなくなります。

アクション・フラグ

アラート生成に関するアクションの実行は、アクション・フラグによって制御されます。アクション・フラグが使用可能な場合のみ、構成済みのアラートが実行されます。

アクション

スクリプト・アクションかタスク・アクションを構成して、アラートの発生時に実行できます。しきい値ベースのヘルス・インディケーターの場合、警

告またはアラームのしきい値に応じて実行するようにアクションを構成できます。状態ベースのヘルス・インディケーターの場合、通常以外のすべての条件に応じて実行するようにアクションを構成できます。アクションを実行するには、DB2 Administration Server を実行していなければなりません。

次の入力パラメーターが、すべてのオペレーティング・システムのコマンド・スクリプトに渡されます。

- <health indicator shortname>
- <object name>
- <value | state>
- <alert type>

スクリプト・アクションは、オペレーティング・システム上のデフォルトのインタープリターを使用します。デフォルト以外のインタープリターを使用したい場合は、タスク・センターでスクリプトの内容を使用してタスクを作成してください。複数のパーティションがある環境では、スクリプト・アクション中に定義されたスクリプトは、すべてのパーティションからアクセス可能でなければなりません。

ヘルス・モニターが個々のヘルス・インディケーターを検査するリフレッシュ・インターバルは構成できません。ヘルス・モニターによって考慮される推奨アクションも構成できません。

ヘルス・モニターの構成は、バイナリー・ファイル HealthRules.reg に保管されます。

- Windows® では、 HealthRules.reg は x:¥<SQLLIB_PATH>¥<INSTANCE_NAME> 中に保管される。例えば d:¥sqllib¥DB2 のようになります。
- UNIX® では、 HealthRules.reg は ~/<SQLLIB_PATH>/cfg 中に保管される。例えば ~/home/sqllib/cfg のようになります。

ヘルス・モニターの構成を、Linux、UNIX、または Windows サーバー上の他の DB2 バージョン 8 インスタンスに複製できます。このレプリケーションを行うには、このバイナリー構成ファイルを、ターゲット・インスタンス上の該当するディレクトリーにコピーします。

関連概念:

- 517 ページの『CLP を使用したヘルス・インディケーター構成の更新』

関連タスク:

- 518 ページの『クライアント・アプリケーションを使用したヘルス・インディケーターの構成』
- 521 ページの『ヘルス・センターを使用したヘルス・インディケーターの構成』
- 516 ページの『CLP を使用したヘルス・インディケーター構成の検索』
- 518 ページの『CLP を使用したヘルス・インディケーター構成のリセット』

CLP を使用したヘルス・インディケーターの構成

CLP を使用したヘルス・インディケーター構成の検索

GET ALERT CONFIGURATION コマンドを使用すると、工場出荷時設定、インスタンス設定、グローバル設定、およびオブジェクト設定を表示できます。

手順:

データベース・レベルのヘルス・インディケーターのグローバル設定を表示するには、次のコマンドを発行します。この設定は、ヘルス・インディケーターのカスタマイズ設定のないすべてのデータベースに適用されます。

```
DB2 GET ALERT CONFIGURATION FOR DATABASES
```

例:

次の例は、GET ALERT CONFIGURATION コマンドと結果の出力を示しています。

```
db2 get alert configuration for databases

            Alert Configuration

  Indicator Name                     = db.db_op_status
      Default                        = Yes
      Type                           = State-based
      Sensitivity                    = 0
      Formula                        = db.db_status;
      Actions                        = Disabled
      Threshold or State checking    = Enabled

  Indicator Name                     = db.sort_shrmem_util
      Default                        = Yes
      Type                           = Threshold-based
      Warning                        = 70
      Alarm                          = 85
      Unit                           = %
      Sensitivity                    = 0
      Formula                        = ((db.sort_shrheap_allocated/sheapthres_shr)
                                       *100);
      Actions                        = Disabled
      Threshold or State checking    = Enabled
  ...
```

個々のヘルス・インディケーターの設定の出力は、デフォルトから変更されたかどうかを示します。上記の例では、グローバル設定は更新されていません。したがって、デフォルトの工場出荷時設定と同じです。データベース・レベルのヘルス・インディケーターの工場出荷時設定を表示するには、上記の例と同じコマンドに DEFAULT キーワードを指定して発行してください。

例:

SAMPLE データベースのカスタム設定を表示するには、次のコマンドを発行してください。

```
DB2 GET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
```

指定したオブジェクト上に特定のヘルス・インディケーターに関する固有の設定がない場合は、すべてのデータベースのグローバル設定が表示されます。

特定のヘルス・インディケーターの設定を表示するには、前述の例に USING *health-indicator-name* 文節を追加してください。

関連資料:

- 「コマンド・リファレンス」の『GET ALERT CONFIGURATION コマンド』

CLP を使用したヘルス・インディケーター構成の更新

特定のヘルス・インディケーターに関するヘルス・インディケーター構成中で、グローバル設定または特定のオブジェクトのオブジェクト設定を更新できます。

UPDATE ALERT CONFIGURATION コマンドには、さまざまな更新オプションに対応した 4 つの副文節があります。個々の UPDATE ALERT CONFIGURATION コマンド中で使用できる副文節は 1 つのみです。複数のオプションを使用するには、複数の UPDATE ALERT CONFIGURATION コマンドを発行しなければなりません。

1 つ目の副文節 SET *parameter-name value* は、次のものの更新をサポートしています。

- 評価フラグ
- 警告およびアラームしきい値 (該当する場合)
- 感度フラグ
- アクション・フラグ

これらの設定のパラメーター名は、それぞれ次のとおりです。

- THRESHOLDSCHECKED
- WARNING および ALARM
- SENSITIVITY
- ACTIONSENABLED

他の 3 つの副文節は、スクリプト・アクションやタスク・アクションの追加、更新、および削除をサポートしています。

次のコマンドは、 SAMPLE データベース上の *db.spilled_sorts* ヘルス・インディケーターに関するしきい値ベースのヘルス・インディケーター構成を更新します。この更新により、警告しきい値が 25 に変更され、アクションが使用可能になり、スクリプト・アクションが追加されます。

```
DB2® UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
    SET WARNING 25, ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
    ADD ACTION SCRIPT c:¥myscript TYPE OS COMMAND LINE PARAMETERS 'space'
    WORKING DIRECTORY c:¥ ON ALARM USER dba1 PASSWORD dba1
```

次のコマンドは、 *ts.ts_util* ヘルス・インディケーターに関する状態ベースのヘルス・インディケーター構成中のグローバル設定を更新します。この更新により、表スペースがバックアップ・ペンディング状態にある際に実行するアクションが定義されます。

```
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    SET ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    ADD ACTION TASK 0.1 ON ATTENTION 0X20 ON localhost USER dba1 PASSWORD dba1
```

この更新は、このヘルス・インディケーターのカスタマイズ設定のないインスタンスの表スペースすべてに適用されます。

ヘルス・インディケーター構成にアクションを追加する場合、 ON *condition* 文節のオプションは、ヘルス・インディケーターのタイプに基づいて異なります。

- しきい値ベースのヘルス・インディケーターの場合、 WARNING および ALARM が有効な条件。
- 状態ベースのヘルス・インディケーターの場合、 ON ATTENTION *state* オプションを使用しなければならない。定義済みの、ヘルス・インディケーターにとって有効な数値の状態を使用する必要があります。データベース・マネージャーとデータベースの操作可能状態の値は、 sqllib¥include¥sqlmon.h 中にあります。表スペースと表スペース・コンテナーの操作可能値は、 sqllib¥include¥sqlutil.h 中にリストされています。

関連資料:

- 「コマンド・リファレンス」の『UPDATE ALERT CONFIGURATION コマンド』

CLP を使用したヘルス・インディケーター構成のリセット

CLP は、グローバル設定を工場出荷時設定にリセットすることをサポートしています。特定のオブジェクトのオブジェクト設定を、そのオブジェクト・タイプのカスタム設定にリセットすることもできます。

手順:

CLP を使用してヘルス・インディケーター構成をリセットするには、次のようにします。

SAMPLE データベースのオブジェクト設定を、データベースの現行のグローバル設定にリセットするには、次のコマンドを発行してください。

```
DB2 RESET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
```

データベースのグローバル設定を工場出荷時設定にリセットするには、次のコマンドを発行してください。

```
DB2 RESET ALERT CONFIGURATION FOR DATABASES
```

特定のヘルス・インディケーターの構成をリセットするには、前述の例に USING *health-indicator-name* 文節を追加してください。

関連資料:

- 「コマンド・リファレンス」の『RESET ALERT CONFIGURATION コマンド』

クライアント・アプリケーションを使用したヘルス・インディケーターの構成

ヘルス・モニターの構成は、C または C++ アプリケーション中の db2GetAlertCfg、 db2UpdateAlertCfg、および db2ResetAlertCfg API によってアクセスできます。これらの各 API は、工場出荷時設定、インスタンス設定、グローバル設定、およびオブジェクト設定にアクセスできます。

次のステップは、 SAMPLE データベース上のヘルス・インディケーターに関する特定のオブジェクト設定を取得する手順を詳述しています。 db2GetAlertCfgData 構造中の objType パラメーターと defaultType パラメーターを組み合わせて使用すると、さまざまなレベルのヘルス・インディケーター構成にアクセスできます。

表 757. objType および defaultType が構成レベルにアクセスする際の設定

設定	objType および defaultType
工場出荷時設定	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES |TABLESPACES | CONTAINERS} および defaultType = DB2ALERTCFG_DEFAULT
グローバル設定	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES | TABLESPACES | CONTAINERS} および defaultType = DB2ALERTCFG_NOT_DEFAULT または objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} および defaultType = DB2ALERTCFG_DEFAULT
オブジェクト設定	objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} および defaultType = DB2ALERTCFG_NOT_DEFAULT

前提条件:

ヘルス・モニター構成にアクセスするには、インスタンス・アタッチメントがなければなりません。インスタンスへのアタッチメントがない場合、デフォルトのインスタンス・アタッチメントが作成されます。リモート・インスタンスのヘルス・モニター構成にアクセスするには、まずそのインスタンスにアタッチする必要があります。

手順 (GET):

SAMPLE データベース上のヘルス・インディケーターに関する特定のオブジェクト設定を取得するには、次のようにします。

1. sqllib¥include ディレクトリー中にある db2ApiDf.h DB2 ヘッダー・ファイルを組み込みます。

   ```
   #include <db2ApiDf.h>
   ```

2. sqlca および db2GetAlertCfgData 構造体を宣言して初期化します。

   ```
   struct sqlca ca;
   memset (&sqlca, '¥0', sizeof(struct sqlca));

   char* objName = NULL;
   char* dbName = "SAMPLE";
   db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASE;
   db2Uint32 defaultType = DB2ALERTCFG_NOT_DEFAULT;

   db2GetAlertCfgData data = {objType, objName, defaultType, dbName, 0, NULL} ;
   ```

3. db2GetAlertCfg API を呼び出します。

   ```
   rc = db2GetAlertCfg (db2Version810, &data, &ca);
   ```

4. 戻された構成を処理し、API によって割り当てられたバッファーを解放します。

```
if (rc >= SQLO_OK) {
  if ((data.ioNumIndicators > 0) && (data.pioIndicators != NULL)) {
    db2GetAlertCfgInd *pIndicators = data.pioIndicators;

    for (db2Uint32 i=0; i < data.ioNumIndicators; i++) {
 //process the entry as necessary using fields defined in db2ApiDf.h
    }
  }

  db2GetAlertCfgFree (db2Version810, &data, &ca);
}
```

手順 (UPDATE):

次のステップは、 *db.sort_shrmem_util* ヘルス・インディケーターのアラート構成中で、データベース・オブジェクトのグローバル設定を更新し、警告しきい値を 80 に設定してタスク・アクション 1.1 を追加する手順を詳述しています。

1. sqllib¥include ディレクトリー中にある db2ApiDf.h DB2 ヘッダー・ファイルを組み込みます。

   ```
   #include <db2ApiDf.h>
   ```

2. sqlca および db2AlertTaskAction 構造体を宣言して初期化します。

   ```
   struct sqlca ca;
   memset (&sqlca, '¥0', sizeof(struct sqlca));

   db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASES;

   db2Uint32 taskCondition = DB2ALERTCFG_CONDITION_WARNING;
   char* taskname = "1.1";
   char* hostname = NULL;
   char* userid = "nobody";
   char* password = "nothing";

   db2AlertTaskAction newTask={taskname,taskCondition,userid,password,hostname};
   ```

3. db2UpdateAlertCfgData 構造体を宣言して初期化します。

   ```
   struct db2UpdateAlertCfgData setData;

   setData.iObjType = objType;
   setData.piObjName = NULL;
   setData.piDbName = NULL;

   setData.iIndicatorID = 1002;

   setData.iNumIndAttribUpdates = 1;
   setData.piIndAttribUpdates[0].iAttribID = DB2ALERTCFG_WARNING;
   setData.piIndAttribUpdates[0].piAttribValue == 80;

   setData.iNumActionUpdates = 0;
   setData.piActionUpdates = NULL;

   setData.iNumActionDeletes = 0;
   setData.piActionDeletes = NULL;

   setData.iNumNewActions = 1;
   setData.piNewActions[0].iActionType = DB2ALERTCFG_ACTIONTYPE_TASK;
   setData.piNewActions[0].piScriptAttribs = NULL;
   setData.piNewActions[0].piTaskAttribs = &newTask;
   ```

4. db2UpdateAlertCfg API を呼び出します。

   ```
   rc = db2UpdateAlertCfg(db2Version810, &setData, &ca);
   ```

手順 (RESET):

次のステップは、 SAMPLE データベース中の MYTS 表スペースのカスタム設定をリセットする手順を詳述しています。

1. sqllib¥include ディレクトリー中にある db2ApiDf.h DB2 ヘッダー・ファイルを組み込みます。

   ```
   #include <db2ApiDf.h>
   ```

2. sqlca および db2ResetAlertCfgData 構造体を宣言して初期化します。

   ```
   struct sqlca ca;
   memset (&sqlca, '¥0', sizeof(struct sqlca));

   char* objName = "MYTS";
   char* dbName = "SAMPLE";
   db2Uint32 objType = DB2ALERTCFG_OBJTYPE_TABLESPACE;

   db2ResetAlertCfgData data = {objType, objName, dbName};
   ```

3. db2ResetAlertCfg API を呼び出します。

   ```
   rc = db2ResetAlertCfg (db2Version810, &data, &ca);
   ```

関連資料:

- 「*管理 API リファレンス*」の『db2GetAlertCfg - アラート構成の入手』
- 「*管理 API リファレンス*」の『db2ResetAlertCfg - アラート構成のリセット』
- 「*管理 API リファレンス*」の『db2UpdateAlertCfg - アラート構成の更新』

ヘルス・センターを使用したヘルス・インディケーターの構成

ヘルス・センターには、ヘルス・インディケーターの構成の表示、更新、およびリセットを行うグラフィカル・インターフェースが備えられています。ヘルス・インディケーターの構成は、インスタンス中のヘルス・モニター中に保管されます。次のステップには、構成ウィンドウにアクセスする方法が示されています。

手順:

1. ヘルス・インディケーターを構成するインスタンスを選択します。
2. **「選択」**メニューまたは右クリック・メニューから、**「構成」**をクリックしてから、**「ヘルス・インディケーターの設定」**をクリックします。「ヘルス・インディケーター構成ランチパッド」が開きます。
3. このランチパッドには、更新できる構成設定のレベルごとにボタンがあります。表示、更新、またはリセットを行いたい構成のレベルに関するボタンを選択します。個々のボタンにより、選択した構成設定レベルの「ヘルス・インディケーターの構成」ウィンドウが立ち上げられます。
4. ヘルス・インディケーターの設定を更新するには、「現在のヘルス・インディケーター設定」表のヘルス・インディケーターの行を選択します。
5. **「選択」**メニューまたは右クリック・メニューから、**「編集」**を選択します。「ヘルス・インディケーターの構成」ウィンドウが開きます。「ヘルス・インディケーターの構成」には、次の情報が表示されます。
   - **「詳細情報」**をクリックすると、ヘルス・インディケーターの説明が表示される。
   - **「評価」**チェック・ボックスを使用すると、ヘルス・インディケーターの評価を使用可能したり使用不可にしたりできる。

注: 現行アラートに関する右クリック・メニュー・オプションを使用して、ヘルス・センターの「アラート」ビューから現行アラートの**「評価」**フラグを使用不可にすることもできます。このオプションは、次回ヘルス・モニター中のインディケーターをリフレッシュする際に、ヘルス・インディケーターの評価を使用不可にします。ヘルス・センター内でアラートに関する**「評価を使用不可にする」**を選択すると、ヘルス・インディケーターに関する評価フラグは false に設定されますが、次のイベントが起きるまで「アラート」ビューからアラートは除去されません。

- この特定のヘルス・インディケーターのヘルス・モニター・リフレッシュ・インターバルに達する。
- ヘルス・モニターがヘルス・インディケーター評価をリフレッシュする。
- ヘルス・センターが状況の表示をリフレッシュする。

- しきい値ベースのヘルス・インディケーターの場合、「アラート」ページで、警告とアラームのしきい値を更新できる。ヘルス・インディケーターの感度もこのページで設定できます。
- 「アクション」ページで、アラート発生時に実行するアクションを選択できる。しきい値ベースのヘルス・インディケーターの場合は警告またはアラーム条件に応じて実行し、状態ベースのヘルス・インディケーターの場合は正常以外の条件に応じて実行するようにアクションを構成できます。**「アクションを有効にする」**チェック・ボックスを選択したり選択解除したりして、アクションの実行を使用可能にしたり使用不可にしたりできます。アラート・アクションの追加、更新、または除去を行うには、**「スクリプト・アクション」**および**「タスク・アクション」**表の隣のボタンを使用してください。

インスタンスの工場出荷時設定を表示するには、次のようにします。

1. **「インスタンス設定」**をクリックします。
2. 「インスタンス・ヘルス・インディケーターの構成」ウィンドウで、**「デフォルトの表示」**をクリックします。

データベース、表スペース、または表スペース・コンテナーのグローバル設定を表示するには、次のようにします。

1. **「グローバル設定」**をクリックします。
2. 「グローバル・ヘルス・インディケーターの構成」ウィンドウで、オブジェクト・タイプを選択します。
3. **「デフォルトの表示」**をクリックします。

関連概念:

- 502 ページの『ヘルス・モニターのグラフィック・ツール』

第 4 部 ヘルス・モニター・リファレンス

第 10 章 ヘルス・モニターの論理データ・グループ

ヘルス・モニター・インターフェースの論理データ・グループへのマッピング

次の表に、サポートされているヘルス・スナップショット要求のタイプをすべてリストします。

表 758. ヘルス・モニター・インターフェースの論理データ・グループへのマッピング

API 要求タイプ	CLP コマンド	SQL 表関数	論理データ・グループ
SQLMA_DB2	get health snapshot for dbm	HEALTH_DBM_INFO	db2
		HEALTH_DBM_HI	health_indicator
	get health snapshot for dbm show detail	HEALTH_DBM_HI_HIS	health_indicator_history
SQLMA_DBASE	get health snapshot for database on *dbname*	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for database on *dbname* show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE with SQLM_HMON_OPT_COLL_FULL in the agent_id	get health snapshot for database on *dbname* with full collection	HEALTH_DB_HIC	health_indicator, hi_obj_list
	get health snapshot for database on *dbname* show detail with full collection	HEALTH_DB_HIC_HIST	health_indicator_history, hi_obj_list
SQLMA_DBASE_ALL	get health snapshot for all databases	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for all databases show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE_TABLESPACES	get health snapshot for tablespaces on *dbname*	HEALTH_TS_INFO	tablespace
		HEALTH_TS_HI	health_indicator
		HEALTH_CONT_INFO	tablespace_container
		HEALTH_CONT_HI	health_indicator
	get health snapshot for tablespaces on *dbname* show detail	HEALTH_TS_HI_HIS	health_indicator_history
		HEALTH_CONT_HI_HIS	health_indicator_history

以下の図は、論理データ・グループがヘルス・スナップショット・データ・ストリーム内に現れる順番を示しています。

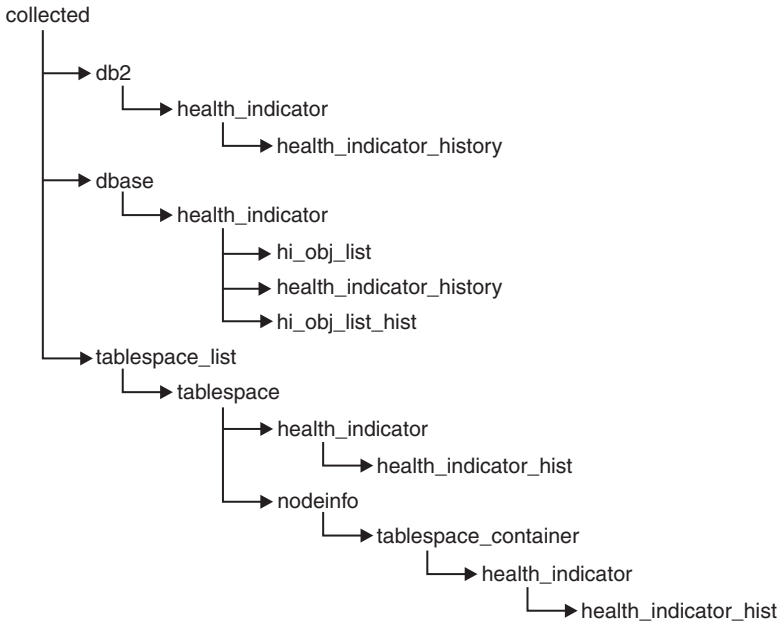

図 7. ヘルス・スナップショットの論理データ・グループ

関連資料:

- 「コマンド・リファレンス」の『GET HEALTH SNAPSHOT コマンド』
- 549 ページの『ヘルス・モニター・インターフェース』

第 11 章 ヘルス・インディケーター

ヘルス・インディケーターの形式

ヘルス・インディケーターのドキュメンテーションは次の標準形式で記述されます。

ID
ヘルス・インディケーターの名前。この ID は、CLP からの構成で使用されます。

ヘルス・モニター・レベル
ヘルス・モニターによってヘルス・インディケーターがキャプチャーされる際のレベル。

カテゴリー
ヘルス・インディケーターのカテゴリー。

タイプ
ヘルス・インディケーターのタイプ。以下の 4 つのタイプがあります。

- 上限しきい値ベース。アラートを生成する進行状況は正常、警告、アラームです。
- 下限しきい値ベース
- 状態ベース。 1 つの状態が正常で、その他の状態はすべて正常ではありません。
- 集合状態ベース。状態は、集合中のオブジェクトの状態の集約に基づいています。

単位
パーセンテージなどの、ヘルス・インディケーターで測定されるデータの単位。状態ベースや集合状態ベースのヘルス・インディケーターには該当しません。

説明
ヘルス・インディケーターによって収集されるデータの説明。

関連概念:

- 3 ページの『データベース・システム・モニター』
- 4 ページの『データベース・システム・モニターのデータ編成』

関連資料:

- 481 ページの『ヘルス・インディケーター』

ヘルス・インディケーターの要約

次の表には、すべてのヘルス・インディケーターが、カテゴリー別にグループ化されてリストされています。

表 759. 表スペース・ストレージのヘルス・インディケーター

名前	ID	追加情報
表スペース使用率	ts.ts_util	529 ページの『ts.ts_util 表スペース使用率』

表 759. 表スペース・ストレージのヘルス・インディケーター (続き)

名前	ID	追加情報
表スペース・コンテナー使用率	tsc.tscont_util	530 ページの『tsc.tscont_util 表スペース・コンテナー使用率』
表スペース操作可能状態	ts.ts_op_status	531 ページの『ts.ts_op_status 表スペース操作可能状態』
表スペース・コンテナー操作可能状態	tsc.tscont_op_status	531 ページの『tsc.tscont_op_status 表スペース・コンテナー操作可能状態』

表 760. ソートのヘルス・インディケーター

名前	ID	追加情報
専用ソート・メモリー使用率	db2.sort_privmem_util	532 ページの『db2.sort_privmem_util 専用ソート・メモリー使用率』
共有ソート・メモリー使用率	db.sort_shrmem_util	533 ページの『db.sort_shrmem_util 共有ソート・メモリー使用率』
オーバーフローしたソートのパーセンテージ	db.spilled_sorts	533 ページの『db.spilled_sorts オーバーフローしたソートのパーセンテージ』
長期共有ソート・メモリー使用率	db.max_sort_shrmem_util	534 ページの『db.max_sort_shrmem_util 長期共有ソート・メモリー使用率』

表 761. データベース・マネージャーのヘルス・インディケーター

名前	ID	追加情報
インスタンス操作可能状態	db2.db2_op_status	535 ページの『db2.db2_op_status インスタンス操作可能状態』
インスタンス最大重大度アラート状態	-	535 ページの『インスタンス最大重大度アラート状態』

表 762. データベースのヘルス・インディケーター

名前	ID	追加情報
データベース操作可能状態	db.db_op_status	536 ページの『db.db_op_status データベース操作可能状態』
データベース最大重大度アラート状態	-	536 ページの『データベース最大重大度アラート状態』

表 763. 保守のヘルス・インディケーター

名前	ID	追加情報
再編成の必要性	db.tb_reorg_req	537 ページの『db.tb_reorg_req 再編成の必要性』
統計収集の必要性ヘルス・インディケーター	db.tb_runstats_req	537 ページの『db.tb_runstats_req 統計収集の必要性』
データベース・バックアップの必要性	db.db_backup_req	538 ページの『db.db_backup_req データベース・バックアップの必要性』

表 764. 高可用性災害時リカバリーのヘルス・インディケーター

名前	ID	追加情報
HADR 操作可能状態ヘルス・インディケーター	db.hadr_op_status	538 ページの『db.hadr_op_status HADR 操作可能状態』
HADR ログ遅延ヘルス・インディケーター	db.hadr_delay	539 ページの『db.hadr_delay HADR ログ遅延』

表 765. ロギングのヘルス・インディケーター

名前	ID	追加情報
ログ使用率	db.log_util	539 ページの『db.log_util ログ使用率』
ログ・ファイル・システム使用率	db.log_fs_util	540 ページの『db.log_fs_util ログ・ファイル・システム使用率』

表 766. アプリケーション並行性のヘルス・インディケーター

名前	ID	追加情報
デッドロック率	db.deadlock_rate	541 ページの『db.deadlock_rate デッドロック率』
ロック・リスト使用率	db.locklist_util	542 ページの『db.locklist_util ロック・リスト使用率』
ロック・エスカレーション率	db.lock_escal_rate	542 ページの『db.lock_escal_rate ロック・エスカレーション率』
ロック待機中のアプリケーションのパーセンテージ	db.apps_waiting_locks	543 ページの『db.apps_waiting_locks ロック待機中のアプリケーションのパーセンテージ』

表 767. パッケージおよびカタログ・キャッシュ、およびワークスペースのヘルス・インディケーター

名前	ID	追加情報
カタログ・キャッシュ・ヒット率	db.catcache_hitratio	544 ページの『db.catcache_hitratio カタログ・キャッシュ・ヒット率』
パッケージ・キャッシュ・ヒット率	db.pkgcache_hitratio	544 ページの『db.pkgcache_hitratio パッケージ・キャッシュ・ヒット率』
共有ワークスペース・ヒット率	db.shrworkspace_hitratio	545 ページの『db.shrworkspace_hitratio 共有ワークスペース・ヒット率』

表 768. メモリーのヘルス・インディケーター

名前	ID	追加情報
モニター・ヒープ使用率	db2.mon_heap_util	545 ページの『db2.mon_heap_util モニター・ヒープ使用率』
データベース・ヒープ使用率	db.db_heap_util	546 ページの『db.db_heap_util データベース・ヒープ使用率』

表 769. フェデレーテッドのヘルス・インディケーター

名前	ID	追加情報
ニックネームの状態	db.fed_nicknames_op_status	546 ページの『db.fed_nicknames_op_status ニックネームの状態』
データ・ソース・サーバーの状態	db.fed_servers_op_status	547 ページの『db.fed_servers_op_status データ・ソース・サーバーの状態』

関連資料:

- 481 ページの『ヘルス・インディケーター』

表スペース・ストレージのヘルス・インディケーター

ts.ts_util 表スペース使用率

ID ts.ts_util

ヘルス・モニター・レベル 表スペース

カテゴリー 表スペース・ストレージ

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 このヘルス・インディケーターは、各 DMS 表スペースのストレージの使用量を追跡します。

すべてのコンテナーがフルの場合は、DMS 表スペースはフルであると考えられます。

標識は、次の公式を使用して計算されます。

```
(ts.used / ts.useable) * 100
```

ここで、 *ts.used* および *ts.useable* はそれぞれ、システム・モニター・エレメントの*表スペースにおける使用済みページ* と、*表スペースにおける使用可能ページ* です。

表スペース使用率は消費されたスペースのパーセントとして測定され、高いパーセントはこの標識の最適関数を下回っていることを示します。

この標識の追加情報に含まれる短期間と長期間の増加率は、現行増加率が短期間の例外的なものであるか、長期間にわたる一貫した増加であるかを判別するのに使用されます。

追加情報のフルになるまでの残り時間の計算は、すべてのフリー・スペースが消費されるまでの残りの時間を予測したものです。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 530 ページの『tsc.tscont_util 表スペース・コンテナー使用率』
- 531 ページの『ts.ts_op_status 表スペース操作可能状態』
- 531 ページの『tsc.tscont_op_status 表スペース・コンテナー操作可能状態』

tsc.tscont_util 表スペース・コンテナー使用率

ID	tsc.tscont_util
ヘルス・モニター・レベル	表スペース・コンテナー
カテゴリー	表スペース・ストレージ
タイプ	上限しきい値ベース
単位	パーセンテージ

説明 このヘルス・インディケーターは、各 SMS 表スペースのストレージの使用量を追跡します。コンテナーが定義されているファイル・システムにスペースが残されていない場合は、 SMS 表スペースはフルであると考えられます。

SMS コンテナーを拡張するためのフリー・スペースがファイル・システムで使用不可である場合は、関連する表スペースがフルになります。

フリー・スペースを消費するファイル・システムに定義されている各コンテナーに対してアラートが発行される場合があります。

標識は、次の公式を使用して計算されます。

```
(fs.used / fs.total)*100
```

ここで `fs` はコンテナーがあるファイル・システムです。

SMS 表スペース使用率は消費されたスペースのパーセントとして測定され、高いパーセントはこの標識の最適関数を下回っていることを示します。

この標識の追加情報に含まれる短期間と長期間の増加率は、現行増加率が短期間の例外的なものであるか、長期間にわたる一貫した増加であるかを判別するのに使用されます。

追加情報のフルになるまでの残り時間の計算は、すべてのフリー・スペースが消費されるまでの残りの時間を予測したものです。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 529 ページの『ts.ts_util 表スペース使用率』
- 531 ページの『ts.ts_op_status 表スペース操作可能状態』
- 531 ページの『tsc.tscont_op_status 表スペース・コンテナー操作可能状態』

ts.ts_op_status 表スペース操作可能状態

ID ts.ts_op_status

ヘルス・モニター・レベル 表スペース

カテゴリー 表スペース・ストレージ

タイプ 状態ベース

単位 該当なし

説明 表スペースの状態は、実行できるアクティビティーまたはタスクを制約します。正常から他の状態に変わると、アテンション・アラートが生成される場合があります。

関連資料:

- 「コマンド・リファレンス」の『LIST TABLESPACES コマンド』
- 322 ページの『tablespace_state 表スペースの状態』
- 481 ページの『ヘルス・インディケーター』
- 529 ページの『ts.ts_util 表スペース使用率』
- 530 ページの『tsc.tscont_util 表スペース・コンテナー使用率』
- 531 ページの『tsc.tscont_op_status 表スペース・コンテナー操作可能状態』

tsc.tscont_op_status 表スペース・コンテナー操作可能状態

ID tsc.tscont_op_status

ヘルス・モニター・レベル 表スペース・コンテナー

カテゴリー 表スペース・ストレージ

タイプ 状態ベース

単位 該当なし

説明 このヘルス・インディケーターは、表スペース・コンテナーのアクセス可能性を追跡します。コンテナーのアクセス可能性は、実行できるアクティビテ

ィーまたはタスクを制約します。コンテナーがアクセス不能の場合は、アテンション・アラートが生成される場合があります。

関連資料:

- 332 ページの『tablespace_num_containers 表スペース内のコンテナー数』
- 334 ページの『container_accessible コンテナーのアクセス可能性』
- 481 ページの『ヘルス・インディケーター』
- 529 ページの『ts.ts_util 表スペース使用率』
- 530 ページの『tsc.tscont_util 表スペース・コンテナー使用率』
- 531 ページの『ts.ts_op_status 表スペース操作可能状態』

ソートのヘルス・インディケーター

db2.sort_privmem_util 専用ソート・メモリー使用率

ID db2.sort_privmem_util

ヘルス・モニター・レベル データベース

カテゴリー ソート

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 ソートを行うためのヒープ・スペースが十分あり、ソートが不必要にオーバーフローしていなければ、ソートは正常稼働していると考えられます。

この標識は、専用ソート・メモリーの使用率を追跡します。db2.sort_heap_allocated (システム・モニター・エレメント) >= *sheapthres* (DBM 構成パラメーター) の場合は、ソートは *sortheap* パラメーターで定義されているソート・ヒープを十分に取得していない可能性があり、アラートが生成される場合があります。

標識は、次の公式を使用して計算されます。

```
(db2.sort_heap_allocated / sheapthres)*100
```

「ポストしきい値ソート」スナップショット・モニター・エレメントは、ソート・ヒープしきい値を超えた後に要求されたソート数を測定します。追加の詳細に表示されるこのインディケーターの値は、このヘルス・インディケーターの問題の重大度を示します。

「使用された最大専用ソート・メモリー」スナップショット・モニター・エレメントは、インスタンスの専用ソート・メモリー最高水準点を保持します。追加情報に示されるこの標識の値は、インスタンスが最後に再生された後の任意の時点で使用中だった専用ソート・メモリーの最大量を示します。この値は *sheapthres* の適切な値を決定するために利用できます。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 534 ページの『db.max_sort_shrmem_util 長期共有ソート・メモリー使用率』

- 533 ページの『db.sort_shrmem_util 共有ソート・メモリー使用率』
- 533 ページの『db.spilled_sorts オーバーフローしたソートのパーセンテージ』

db.sort_shrmem_util 共有ソート・メモリー使用率

ID db.sort_shrmem_util

ヘルス・モニター・レベル データベース

カテゴリー ソート

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 ソートを行うためのヒープ・スペースが十分あり、ソートが不必要にオーバーフローしていなければ、ソートは正常稼働していると考えられます。

この標識は、共有ソート・メモリーの使用率を追跡します。 *sheapthres_shr* データベース構成パラメーターは堅固な制限です。割り振りが制限に近くなると、アラートが生成される場合があります。

標識は、次の公式を使用して計算されます。

```
(db.sort_shrheap_allocated / sheapthres_shr)*100
```

sheapthres_shr が 0 に設定されると、 *sheapthres* は共有ソート・ヒープしきい値として使用されることに注意してください。

「使用された最大共有ソート・メモリー」スナップショット・モニター・エレメントは、データベースの共有ソート・メモリー最高水準点を保持します。追加情報に表示されるこの標識の値は、データベースがアクティブになった後の任意の時点で使用中であった共有ソート・メモリーの最大量を示します。この値は共有ソート・メモリーしきい値の適切な値を決定するために利用できます。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 534 ページの『db.max_sort_shrmem_util 長期共有ソート・メモリー使用率』
- 532 ページの『db2.sort_privmem_util 専用ソート・メモリー使用率』
- 533 ページの『db.spilled_sorts オーバーフローしたソートのパーセンテージ』

db.spilled_sorts オーバーフローしたソートのパーセンテージ

ID db.spilled_sorts

ヘルス・モニター・レベル データベース

カテゴリー ソート

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 ソートを行うためのヒープ・スペースが十分あり、ソートが不必要にオーバーフローしていなければ、ソートは正常稼働していると考えられます。

ディスクにオーバーフローするソートは、重大なパフォーマンス低下の原因となる可能性があります。これが起こると、アラートが生成される場合があります。

標識は、次の公式を使用して計算されます。

```
(db.sort_overflows_t - db.sort_overflows_t-1) /
(db.total_sorts_t - db.total_sorts_t-1) * 100
```

t は現在のスナップショットで、*t-1* は 1 時間前のスナップショットです。システム・モニター・エレメント db.sort_overflows はソート・ヒープを消費したソートの合計数であり、一時記憶域のディスク・スペースを必要とした可能性があります。エレメント db.total_sorts は実行されたソートの合計数です。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 534 ページの『db.max_sort_shrmem_util 長期共有ソート・メモリー使用率』
- 532 ページの『db2.sort_privmem_util 専用ソート・メモリー使用率』
- 533 ページの『db.sort_shrmem_util 共有ソート・メモリー使用率』

db.max_sort_shrmem_util 長期共有ソート・メモリー使用率

ID db.max_sort_shrmem_util

ヘルス・モニター・レベル データベース

カテゴリー ソート

タイプ 下限しきい値ベース

単位 パーセンテージ

説明 ソートを行うためのヒープ・スペースが十分あり、ソートが不必要にオーバーフローしていなければ、ソートは正常稼働していると考えられます。

この標識は構成済み共有ソート・ヒープを追跡し、 DB2 の他のどこかでの使用のために解放できるリソースがないかどうかを調べます。

使用量のパーセンテージが低い場合はアラートが生成される場合があります。

標識は、次の公式を使用して計算されます。

```
(db.max_shr_sort_mem / sheapthres_shr)*100
```

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 532 ページの『db2.sort_privmem_util 専用ソート・メモリー使用率』
- 533 ページの『db.sort_shrmem_util 共有ソート・メモリー使用率』
- 533 ページの『db.spilled_sorts オーバーフローしたソートのパーセンテージ』

データベース・マネージャー (DBMS) のヘルス・インディケーター

db2.db2_op_status インスタンス操作可能状態

ID db2.db2_op_status

ヘルス・モニター・レベル インスタンス

カテゴリー DBMS

タイプ 状態ベース

単位 該当なし

説明 インスタンス状態が実行されているアクティビティーまたはタスクを制約していないときは、インスタンスは正常稼働していると考えられます。

状態は、アクティブ、静止ペンディング、静止、またはダウンのいずれかになります。非アクティブの状態では、アテンション・アラートが生成される場合があります。

関連資料:

- 145 ページの『db2_status DB2 インスタンス状況』
- 481 ページの『ヘルス・インディケーター』
- 535 ページの『インスタンス最大重大度アラート状態』

インスタンス最大重大度アラート状態

ID 該当なし。このヘルス・インディケーターには、構成または推奨サポートはありません。

ヘルス・モニター・レベル インスタンス

カテゴリー DBMS

タイプ 状態ベース

単位 該当なし

説明 この標識は、モニター対象のインスタンスのロールアップ・アラート状態を表します。インスタンスのアラート状態は、インスタンスとそのデータベース、およびモニター対象のデータベース・オブジェクトの最高レベルのアラート状態です。アラート状態の順序は次のようになります。

- アラーム
- 警告
- アテンション
- 正常

インスタンスのアラート状態によって、DB2 が全体として正常稼働しているかどうかが決まります。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 535 ページの『db2.db2_op_status インスタンス操作可能状態』

データベースのヘルス・インディケーター

db.db_op_status データベース操作可能状態

ID db.db_op_status

ヘルス・モニター・レベル データベース

カテゴリー データベース

タイプ 状態ベース

単位 該当なし

説明 データベースの状態は、実行できるアクティビティーまたはタスクを制約します。状態は、アクティブ、静止ペンディング、静止、またはロールフォワードのいずれかになります。アクティブから他の状態に変わると、アテンション・アラートが生成される場合があります。

関連資料:

- 149 ページの『db_status データベース状況』
- 481 ページの『ヘルス・インディケーター』
- 536 ページの『データベース最大重大度アラート状態』

データベース最大重大度アラート状態

ID 該当なし。このヘルス・インディケーターには、構成または推奨サポートはありません。

ヘルス・モニター・レベル データベース

カテゴリー データベース

タイプ 状態ベース

単位 該当なし

説明 この標識は、モニター対象のデータベースのロールアップ・アラート状態を表します。データベースのアラート状態は、データベースとそのオブジェクトの最高レベルのアラート状態です。アラート状態の順序は次のようになります。

- アラーム
- 警告
- アテンション
- 正常

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 536 ページの『db.db_op_status データベース操作可能状態』

保守のヘルス・インディケーター

db.tb_reorg_req 再編成の必要性

ID db.tb_reorg_req

ヘルス・モニター・レベル データベース

カテゴリー データベース保守

タイプ 集合状態ベース

単位 該当なし

説明:

このヘルス・インディケーターは、データベース中の表や索引を再編成する必要性を追跡します。フラグメント化されたデータを除去するには、表か、表に定義されたすべての索引を再編成する必要があります。再編成するには、情報を圧縮して、行か索引データを再構成します。その結果、パフォーマンスが向上し、表や索引中のフリー・スペースが増えます。

SQL 照会を使用して、このヘルス・インディケーターによって考慮する表を限定できます。この照会のシステム表に対する副選択文節が、追加情報中の有効範囲に示されます。

アテンション・アラートが生成されて、再編成が必要なことが示される場合もあります。 AUTO_REORG データベース構成パラメーターを ON に設定すると、再編成を自動化できます。自動再編成を使用可能にすると、アテンション・アラートにより、 1 つ以上の自動再編成が正常に完了できなかったことが示されます。注意が必要なオブジェクトのリストについては、このヘルス・インディケーターの集合の詳細を参照してください。

関連資料:

- 「コマンド・リファレンス」の『REORG INDEXES/TABLE コマンド』
- 481 ページの『ヘルス・インディケーター』

db.tb_runstats_req 統計収集の必要性

ID db.tb_runstats_req

ヘルス・モニター・レベル データベース

カテゴリー データベース保守

タイプ 集合状態ベース

単位 該当なし

説明 このヘルス・インディケーターは、データベース中の表やそれらの索引の統計を収集する必要性を追跡します。照会の実行時間を改善するには、表および表に定義されたすべての索引に統計が必要です。

SQL 照会を使用して、このヘルス・インディケーターによって考慮する表を限定できます。この照会のシステム表に対する副選択文節が、追加情報中の有効範囲に示されます。

アテンション・アラートが生成されて、統計の収集が必要なことが示される場合もあります。 AUTO_RUNSTATS データベース構成パラメーターを ON に設定すると、統計を自動的に収集できます。統計の自動収集を使用可能にすると、アテンション・アラートにより、 1 つ以上の統計の自動収集アクションが正常に完了できなかったことが示されます。

関連資料:

- 「コマンド・リファレンス」の『RUNSTATS コマンド』
- 481 ページの『ヘルス・インディケーター』

db.db_backup_req データベース・バックアップの必要性

ID db.db_backup_req

ヘルス・モニター・レベル データベース

カテゴリー データベース保守

タイプ 状態ベース

単位 該当なし

説明 このヘルス・インディケーターは、データベースのバックアップの必要性を追跡します。ハードウェアやソフトウェアの障害の場合にデータが消失する可能性から保護するために、リカバリー計画の一部として、定期的にバックアップを取る必要があります。

このヘルス・インディケーターは、経過時間と、前回のバックアップ以後に変更されたデータの量に基づいて、データベースのバックアップが必要な時点を判別します。

アテンション・アラートが生成されて、データベースのバックアップが必要なことが示される場合もあります。 AUTO_DB_BACKUP データベース構成パラメーターを ON に設定すると、データベースのバックアップを自動化できます。自動データベース・バックアップを使用可能にすると、アテンション・アラートにより、 1 つ以上の自動データベース・バックアップが正常に完了できなかったことが示されます。

関連資料:

- 151 ページの『last_backup 最終バックアップ・タイム・スタンプ』

高可用性災害時リカバリーのヘルス・インディケーター

db.hadr_op_status HADR 操作可能状態

ID db.hadr_op_status

ヘルス・モニター・レベル データベース

カテゴリー 高可用性災害時リカバリー

タイプ 状態ベース

単位 該当なし

説明 このヘルス・インディケーターは、データベースの高可用性災害時リカバリ

ー (HADR) 操作可能状態を追跡します。 1 次サーバーとスタンバイ・サーバーの間の状態は、接続済み、混雑、または切断のいずれかになります。接続済みから他の状態に変わると、アテンション・アラートが生成される場合があります。

関連概念:

- 「*データ・リカバリーと高可用性 ガイドおよびリファレンス*」の『高可用性災害時リカバリーの概要』

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 539 ページの『db.hadr_delay HADR ログ遅延』
- 413 ページの『hadr_state HADR の状態 : モニター・エレメント』
- 414 ページの『hadr_connect_status HADR 接続状況 : モニター・エレメント』

db.hadr_delay HADR ログ遅延

ID db.hadr_delay

ヘルス・モニター・レベル データベース

カテゴリー 高可用性災害時リカバリー

タイプ 上限しきい値ベース

単位 分

説明 このヘルス・インディケーターは、1 次データベースに対するデータ変更と、スタンバイ・データベースに対するこれらの変更内容のレプリケーションの間の、現在の平均遅延 (分単位) を追跡します。遅延値が大きい場合は、 1 次データベースに障害が起きた後にスタンバイ・データベースにフェイルオーバーする際に、データ損失が生じる可能性があります。さらに遅延値が大きいと、 1 次データベースがスタンバイ・データベースより優先になっているためテークオーバーが必要な場合に、ダウン時間が長くなる可能性もあります。

関連概念:

- 「*データ・リカバリーと高可用性 ガイドおよびリファレンス*」の『高可用性災害時リカバリーの概要』

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 538 ページの『db.hadr_op_status HADR 操作可能状態』
- 「*管理ガイド: パフォーマンス*」の『hadr_syncmode - 対等状態にあるログ書き込みのための HADR 同期モード構成パラメーター』

ロギングのヘルス・インディケーター

db.log_util ログ使用率

ID db.log_util

ヘルス・モニター・レベル データベース

カテゴリー ロギング

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 このインディケーターは、データベースで使用されたアクティブ・ログ・スペースの合計量 (バイト数) を追跡します。

ログ使用率は消費されたスペースのパーセントとして測定され、パーセンテージが高い場合はアラートが生成される場合があります。

インディケーターは、次の公式を使用して計算されます。

```
(db.total_log_used / (db.total_log_used + db.total_log_available)) * 100
```

追加情報に示されるログ関連のデータベース構成パラメーターの値は、ログの現行割り振りを表示します。追加情報には、最も古いアクティブ・トランザクションを持つアプリケーションのアプリケーション ID も含まれます。このアプリケーションにログ・スペースの解放を強制することができます。

関連タスク:

- 「*管理ガイド: パフォーマンス*」の『構成パラメーターによる DB2 の構成』

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 540 ページの『db.log_fs_util ログ・ファイル・システム使用率』

db.log_fs_util ログ・ファイル・システム使用率

ID db.log_fs_util

ヘルス・モニター・レベル データベース

カテゴリー ロギング

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 ログ・ファイル・システム使用率は、トランザクション・ログが常駐するファイル・システムの使用率を追跡します。ファイル・システムに空きがなければ、DB2 は新規ログ・ファイルを作成できない可能性があります。

ログ使用率は消費されたスペースのパーセントとして測定されます。ファイル・システムのフリー・スペースの量が最小の場合 (つまり使用率のパーセンテージが高い場合) は、アラートが生成される場合があります。

インディケーターは、次の公式を使用して計算されます: *(fs.log_fs_used / fs.log_fs_total)*100*。ここで fs はログが常駐するファイル・システムです。

追加情報に示されるログ関連のデータベース構成パラメーターの値は、ログの現行割り振りを表示します。ユーザー出口が使用可能であるかどうかも追加の詳細情報に示されます。

追加の詳細情報に示される「ディスクがフルになった時はログをブロック (Block on Log Disk Full)」が「はい (yes)」に設定され、使用率が 100% で

ある場合、ログ・ファイルが正常に作成されるまでトランザクションをコミットできないアプリケーションへの影響を制限するために、アラートをできるだけ早く解決する必要があります。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 539 ページの『db.log_util ログ使用率』

アプリケーション並行性のヘルス・インディケーター

db.deadlock_rate デッドロック率

ID	db.deadlock_rate
ヘルス・モニター・レベル	データベース
カテゴリー	アプリケーション並行性
タイプ	上限しきい値ベース
単位	時間当たりのデッドロック数

説明 デッドロック率は、データベースでデッドロックが起きる率と、アプリケーションで競合問題が発生する度合いを追跡します。デッドロックは次の状態が原因で起こることがあります。

- データベースでロック・エスカレーションが発生している場合。
- システムが生成した行のロッキング数が十分なときに、アプリケーションが表を明示的にロッキングしている場合。
- アプリケーションがバインディングのときに不適切な分離レベルを使用している場合。
- カタログ表が反復可能読み取りのためにロックされている場合。
- 複数のアプリケーションが同じロックを異なる順序で獲得しているために、デッドロックになっている場合。

インディケーターは、次の公式を使用して計算されます。

$(db.deadlocks_t - db.deadlocks_{t-1})$

ここで 't' は現在のスナップショットで、't-1' は現在のスナップショットの 60 分前に取られた最後のスナップショットです。

デッドロック率が高くなると、アラートを生成する可能性のある競合の度合いも大きくなります。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 542 ページの『db.locklist_util ロック・リスト使用率』
- 542 ページの『db.lock_escal_rate ロック・エスカレーション率』
- 543 ページの『db.apps_waiting_locks ロック待機中のアプリケーションのパーセンテージ』

db.locklist_util ロック・リスト使用率

ID db.locklist_util

ヘルス・モニター・レベル データベース

カテゴリー アプリケーション並行性

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 このインディケーターは、使用されているロック・リスト・メモリーの量を追跡します。データベースごとにロック・リストが 1 つあり、データベースに現在接続しているすべてのアプリケーションが保持しているロックが入っています。ロック・リスト・メモリーには設定限界があります。一度その限界に達すると、次の状態が原因でパフォーマンスが低下します。

- ロック・エスカレーションにより行ロックから表ロックへの変換が行われ、その結果、データベースの共有オブジェクトにおける並行性が低下する。
- アプリケーションによる限定された表ロック待ちのため、アプリケーション間でさらに多くのデッドロックが起きる。その結果、トランザクションがロールバックされます。

ロック要求の最大数がデータベースの限界設定に達すると、アプリケーションにエラーが戻されます。

インディケーターは、次の公式を使用して計算されます。

```
(db.lock_list_in_use / (locklist * 4096)) * 100
```

使用率は消費されたメモリーのパーセントとして測定され、パーセンテージが高い場合は正常稼働ではない状態を示します。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 541 ページの『db.deadlock_rate デッドロック率』
- 542 ページの『db.lock_escal_rate ロック・エスカレーション率』
- 543 ページの『db.apps_waiting_locks ロック待機中のアプリケーションのパーセンテージ』

db.lock_escal_rate ロック・エスカレーション率

ID db.lock_escal_rate

ヘルス・モニター・レベル データベース

カテゴリー アプリケーション並行性

タイプ 上限しきい値ベース

単位 時間当たりのロック・エスカレーション数

説明 このインディケーターは、ロックが行ロックから表ロックにエスカレートされた率を追跡します。このエスカレーションの結果、トランザクションの並行性が影響を受けます。

アプリケーションが保留するロックの合計数がそのアプリケーションで使用可能なロック・リスト・スペースの最大量に達した場合、またはすべてのアプリケーションが使用するロック・リスト・スペースが合計ロック・リスト・スペースに近くなると、ロックはエスカレートされます。使用可能なロック・リスト・スペースの量は、 *maxlocks* および *locklist* データベース構成パラメーターによって決まります。

アプリケーションが許可されているロックの最大数に達し、エスカレートするロックがない場合は、アプリケーションは他のアプリケーションに割り振られたロック・リストのスペースを使用します。データベースごとにロック・リストが 1 つあり、データベースに現在接続しているすべてのアプリケーションが保持しているロックが入っています。ロック・リスト全体が満杯になるとエラーが起こります。

インディケーターは、次の公式を使用して計算されます。

$(db.lock\_escals_t - db.lock\_escals_{t-1})$

ここで 't' は現在のスナップショットで、't-1' は現在のスナップショットの 60 分前に取られた最後のスナップショットです。

デッドロック率が高くなると、アラートを生成する可能性のある競合の度合いも大きくなります。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 541 ページの『db.deadlock_rate デッドロック率』
- 542 ページの『db.locklist_util ロック・リスト使用率』
- 543 ページの『db.apps_waiting_locks ロック待機中のアプリケーションのパーセンテージ』

db.apps_waiting_locks ロック待機中のアプリケーションのパーセンテージ

ID	db.apps_waiting_locks
ヘルス・モニター・レベル	データベース
カテゴリー	アプリケーション並行性
タイプ	上限しきい値ベース
単位	パーセンテージ

説明 このインディケーターは、現在実行中でロック待ちのすべてのアプリケーションのパーセンテージを測定します。

パーセンテージが高い場合は、パフォーマンスに悪影響を及ぼす並行性の問題がアプリケーションで発生していることを示します。

インディケーターは、次の公式を使用して計算されます。

```
(db.locks_waiting / db.appls_cur_cons) *100)
```

関連資料:

- 481 ページの『ヘルス・インディケーター』

- 541 ページの『db.deadlock_rate デッドロック率』
- 542 ページの『db.locklist_util ロック・リスト使用率』
- 542 ページの『db.lock_escal_rate ロック・エスカレーション率』

パッケージおよびカタログ・キャッシュ、およびワークスペースのヘルス・インディケーター

db.catcache_hitratio カタログ・キャッシュ・ヒット率

ID db.catcache_hitratio

ヘルス・モニター・レベル データベース

カテゴリー パッケージおよびカタログ・キャッシュ、およびワークスペース

タイプ 下限しきい値ベース

単位 パーセンテージ

説明 ヒット率は、カタログ・キャッシュを使用できることによりディスク上のカタログに実際にアクセスしないで済んでいる程度を示すパーセンテージです。高い率は、実際のディスク入出力アクセスの回避に成功していることを示します。

インディケーターは、次の公式を使用して計算されます。

```
(1-(db.cat_cache_inserts/db.cat_cache_lookups))*100
```

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 544 ページの『db.pkgcache_hitratio パッケージ・キャッシュ・ヒット率』
- 545 ページの『db.shrworkspace_hitratio 共有ワークスペース・ヒット率』

db.pkgcache_hitratio パッケージ・キャッシュ・ヒット率

ID db.pkgcache_hitratio

ヘルス・モニター・レベル データベース

カテゴリー パッケージおよびカタログ・キャッシュ、およびワークスペース

タイプ 下限しきい値ベース

単位 パーセンテージ

説明 ヒット率は、パッケージ・キャッシュを使用できることによりシステム・カタログから静的 SQL のためのパッケージとセクションを再ロードしないで済み、また動的 SQL ステートメントを再コンパイルしないで済んでいる程度を示すパーセンテージです。高い率は、これらのアクティビティーの回避に成功していることを示します。

インディケーターは、次の公式を使用して計算されます。

```
(1-(db.pkg_cache_inserts/db.pkg_cache_lookups))*100
```

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 544 ページの『db.catcache_hitratio カタログ・キャッシュ・ヒット率』
- 545 ページの『db.shrworkspace_hitratio 共有ワークスペース・ヒット率』

db.shrworkspace_hitratio 共有ワークスペース・ヒット率

ID db.shrworkspace_hitratio

ヘルス・モニター・レベル データベース

カテゴリー パッケージおよびカタログ・キャッシュ、およびワークスペース

タイプ 下限しきい値ベース

単位 パーセンテージ

説明 ヒット率は、共有 SQL ワークスペースを使用できることにより、実行されようとしている SQL ステートメントのセクションの初期化が不要になっている程度を示すパーセンテージです。高い率は、このアクティビティーの回避に成功していることを示します。

インディケーターは、次の公式を使用して計算されます。

```
(1-(db.shr_workspace_section_inserts/db.shr_workspace_section_lookups))*100
```

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 544 ページの『db.catcache_hitratio カタログ・キャッシュ・ヒット率』
- 544 ページの『db.pkgcache_hitratio パッケージ・キャッシュ・ヒット率』

メモリーのヘルス・インディケーター

db2.mon_heap_util モニター・ヒープ使用率

ID db2.mon_heap_util

ヘルス・モニター・レベル インスタンス

カテゴリー メモリー

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 このインディケーターは、ID が SQLM_HEAP_MONITOR のメモリー・プールを基に、モニター・ヒープ・メモリーの使用量を追跡します。

使用率は次の公式を使用して計算されます。

```
(db2.pool_cur_size / db2.pool_max_size) * 100
```

これはメモリー・プール ID が SQLM_HEAP_MONITOR の場合です。

一度このパーセンテージが最大の 100% に達すると、モニター操作が失敗する場合があります。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 546 ページの『db.db_heap_util データベース・ヒープ使用率』

db.db_heap_util データベース・ヒープ使用率

ID db.db_heap_util

ヘルス・モニター・レベル データベース

カテゴリー メモリー

タイプ 上限しきい値ベース

単位 パーセンテージ

説明 このインディケーターは、ID が SQLM_HEAP_DATABASE のメモリー・プールを基に、モニター・ヒープ・メモリーの使用量を追跡します。

使用率は次の公式を使用して計算されます。

```
(db.pool_cur_size / db.pool_max_size) * 100
```

これはメモリー・プール ID が SQLM_HEAP_DATABASE の場合です。

一度このパーセンテージが最大の 100% に達すると、使用可能なヒープがないため、照会および操作が失敗する場合があります。

関連資料:

- 481 ページの『ヘルス・インディケーター』
- 545 ページの『db2.mon_heap_util モニター・ヒープ使用率』

フェデレーテッドのヘルス・インディケーター

db.fed_nicknames_op_status ニックネームの状態

ID db.fed_nicknames_op_status

ヘルス・モニター・レベル データベース

カテゴリー フェデレーテッド

タイプ 集合状態ベース

単位 該当なし

説明:

このヘルス・インディケーターは、フェデレーテッド・データベース中で定義されているニックネームをすべて検査し、無効なニックネームがあるかどうかを判別します。データ・ソース・オブジェクトがドロップされたり変更が加えられたりした場合や、ユーザー・マッピングが誤っている場合には、ニックネームが無効になることがあります。

フェデレーテッド・データベース中で定義されているニックネームが無効な場合は、アテンション・アラートが生成されることがあります。注意が必要なオブジェクトのリストについては、このヘルス・インディケーターの集合の詳細を参照してください。

このヘルス・インディケーターがニックネームの状況を検査するには、FEDERATED データベース・マネージャー・パラメーターを YES に設定しなければなりません。

関連資料:

- 「*SQL リファレンス 第 2 巻*」の『DROP ステートメント』
- 「*SQL リファレンス 第 2 巻*」の『ALTER NICKNAME ステートメント』
- 「*SQL リファレンス 第 2 巻*」の『CREATE NICKNAME ステートメント』
- 「*SQL リファレンス 第 2 巻*」の『CREATE USER MAPPING ステートメント』

db.fed_servers_op_status データ・ソース・サーバーの状態

ID	db.fed_servers_op_status
ヘルス・モニター・レベル	データベース
カテゴリー	フェデレーテッド
タイプ	集合状態ベース
単位	該当なし

説明:

このヘルス・インディケーターは、フェデレーテッド・データベース中で定義されているデータ・ソース・サーバーをすべて検査し、使用できないものがあるかどうかを判別します。データ・ソース・ソース・サーバーが停止したり、存在しなくなったり、構成が誤っていたりすると、データ・ソース・サーバーが使用できないことがあります。

フェデレーテッド・データベース中で定義されているデータ・ソース・ソース・サーバーが無効な場合は、アテンション・アラートが生成されることがあります。注意が必要なオブジェクトのリストについては、このヘルス・インディケーターの集合の詳細を参照してください。

このヘルス・インディケーターがデータ・ソース・サーバーの状況を検査するには、 FEDERATED データベース・マネージャー・パラメーターを YES に設定しなければなりません。

関連資料:

- 「*SQL リファレンス 第 2 巻*」の『ALTER SERVER ステートメント』
- 「*SQL リファレンス 第 2 巻*」の『ALTER USER MAPPING ステートメント』
- 「*SQL リファレンス 第 2 巻*」の『CREATE USER MAPPING ステートメント』

第 12 章 ヘルス・モニター・インターフェース

ヘルス・モニター・インターフェース

次の表は、API のヘルス・モニター・インターフェースをリストしています。

表 770. ヘルス・モニター・インターフェース: API

モニター・タスク	API
ヘルス・スナップショットのキャプチャー	db2GetSnapshot スナップショット・クラス SQLM_CLASS_HEALTH を指定してスナップショットを取得
集合オブジェクトの全リストを含むヘルス・スナップショットのキャプチャー	db2GetSnapshot スナップショット・クラス SQLM_CLASS_HEALTH および agent_id に SQLM_HMON_OPT_COLL_FULL を指定してスナップショットを取得
公式、追加情報、および履歴を含むヘルス・スナップショットのキャプチャー	db2GetSnapshot スナップショット・クラス SQLM_CLASS_HEALTH_WITH_DETAIL を指定してスナップショットを取得
公式、追加情報、履歴、および集合オブジェクトの全リストを含むヘルス・スナップショットのキャプチャー	db2GetSnapshot スナップショット・クラス SQLM_CLASS_HEALTH_WITH_DETAIL および agent_id に SQLM_HMON_OPT_COLL_FULL を指定してスナップショットを取得
自己記述型データ・ストリームの変換	db2ConvMonStream モニター・ストリームの変換
ヘルス・スナップショットのサイズ見積もり	db2GetSnapshotSize db2GetSnapshot 出力バッファーに必要なサイズの見積もり

次の表は、CLP コマンドのヘルス・モニター・インターフェースをリストしています。

表 771. ヘルス・モニター・インターフェース: CLP コマンド

モニター・タスク	CLP コマンド
ヘルス・スナップショットのキャプチャー	GET HEALTH SNAPSHOT コマンド
公式、追加情報、および履歴を含むヘルス・スナップショットのキャプチャー	GET HEALTH SNAPSHOT WITH DETAILS コマンド

次の表は、SQL 関数のヘルス・モニター・インターフェースをリストしています。

表 772. ヘルス・モニター・インターフェース: SQL 関数

モニター・タスク	SQL 関数
データベース・マネージャー・レベルの正常性に関する情報のスナップショット	HEALTH_DBM_INFO
データベース・マネージャー・レベルのヘルス・インディケーターのスナップショット	HEALTH_DBM_HI
データベース・マネージャー・レベルのヘルス・インディケーター履歴のスナップショット	HEALTH_DBM_HI_HIS
データベース・レベルの正常性に関する情報のスナップショット	HEALTH_DB_INFO
データベース・レベルのヘルス・インディケーターのスナップショット	HEALTH_DB_HI
データベース・レベルのヘルス・インディケーター履歴のスナップショット	HEALTH_DB_HI_HIS
データベース・レベルのヘルス・インディケーター集合のスナップショット	HEALTH_DB_HIC
データベース・レベルのヘルス・インディケーター収集履歴のスナップショット	HEALTH_DB_HIC_HIS
表スペース・レベルの正常性に関する情報のスナップショット	HEALTH_TBS_INFO
表スペース・レベルのヘルス・インディケーターのスナップショット	HEALTH_TBS_HI
表スペース・レベルのヘルス・インディケーター履歴のスナップショット	HEALTH_TBS_HI_HIS
表スペース・コンテナー・レベルの正常性に関する情報のスナップショット	HEALTH_CONT_INFO
表スペース・コンテナー・レベルのヘルス・インディケーターのスナップショット	HEALTH_CONT_HI
表スペース・コンテナー・レベルのヘルス・インディケーター履歴のスナップショット	HEALTH_CONT_HI_HIS

関連資料:

- 「*管理 API リファレンス*」の『db2GetSnapshot - スナップショットの入手』
- 「*管理 API リファレンス*」の『db2GetSnapshotSize - db2GetSnapshot 出力バッファーに必要なサイズの見積もり』
- 「*管理 API リファレンス*」の『db2ConvMonStream - モニター・ストリームの変換』
- 「*コマンド・リファレンス*」の『GET HEALTH SNAPSHOT コマンド』

第 5 部 付録

付録 A. バージョン 5 のモニター出力

バージョン 5 のシステム・モニター出力

DB2® バージョン 6 では、自己記述型データ・ストリームが、ファイルまたはパイプへのシステム・モニター出力という標準形になりました。以前は、システム・モニター・データは固定データ構造で戻されていました。

イベント・モニターは、デフォルトでは、自己記述型モニター形式でデータを書き込みます。これを個々のイベント・モニター用に上書きするには、レジストリー変数 *DB2OLDEVMON=evmon1,evmon2,...* を設定します (*evmon1* はデータを、バージョン 6 より前の形式で書き込むイベント・モニター)。

バージョン 8 のスナップショット要求が出されて、バージョン 6 より前のバージョンのスナップショット・データの集合がサーバー (例えば低いレベルのサーバー) から戻された場合、呼び出し側に SQLCODE +1627W が戻されます。モニター出力はバージョン 6 より前の形式になるため、バージョン 5 の方式を使用して解析する必要があります。

反対の場合、つまりバージョン 6 より前のスナップショット要求が出されて、バージョン 8 のスナップショット・データの集合が戻された場合には、db2ConvMonStream API を使用することができます。この API を使用して、論理データ・グループの新しいモニター形式を、対応するバージョン 6 より前のデータ構造に変換できます。

次の表に、バージョン 8 クライアントで使用できるスナップショットのシナリオをリストします。

表 773. クライアント/サーバーのスナップショットのシナリオ

要求されるスナップショットのバージョン	サーバーのバージョン	戻されるデータ形式	アクション
SQLM_DBMON_VERSION1 SQLM_DBMON_VERSION2 SQLM_DBMON_VERSION5 SQLM_DBMON_VERSION5_2	DB2 バージョン 6 から バージョン 8	固定サイズ構造	固定構造化方式でデータ・ストリームを解析。
SQLM_DBMON_VERSION6	DB2 バージョン 6、バージョン 7、およびバージョン 8	自己記述型	db2ConvMonStream() API を使用すると、バージョン 6 より前のモニター・アプリケーションをより簡単に移行できる。
SQLM_DBMON_VERSION7 SQLM_DBMON_VERSION8	DB2 バージョン 6、バージョン 7、およびバージョン 8	自己記述型	db2ConvMonStream() API を使用すると、バージョン 6 より前のモニター・アプリケーションをより簡単に移行できる。

関連概念:

- 51 ページの『イベント・モニター』
- 21 ページの『スナップショット・モニター』

関連資料:

- 「*管理 API リファレンス*」の『db2ConvMonStream - モニター・ストリームの変換』

付録 B. DB2 Universal Database の技術情報

DB2 資料とヘルプ

DB2® 技術情報は、以下のツールと方法を介して利用できます。

- DB2 インフォメーション・センター
  - トピック
  - DB2 ツールのヘルプ
  - サンプル・プログラム
  - チュートリアル
- ダウンロード可能な PDF ファイル、CD 上の PDF ファイル、および印刷された資料
  - ガイド
  - リファレンス・マニュアル
- コマンド行ヘルプ
  - コマンド・ヘルプ
  - メッセージ・ヘルプ
  - SQL 状態ヘルプ
- インストール済みソース・コード
  - サンプル・プログラム

ibm.com® にある技術資料、白書、 Redbooks™ その他の DB2 Universal Database™ 技術情報にオンラインでアクセスできます。 DB2 Information Management ソフトウェア・ライブラリー・サイト (www.ibm.com/software/data/pubs/) にアクセスしてください。

DB2 資料の更新

IBM® は、 DB2 インフォメーション・センターの資料のフィックスパックやその他の資料更新を定期的に発行しています。 DB2 インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/db2help/) にアクセスすれば、常に最新の情報が掲載されます。 DB2 インフォメーション・センターをローカル・インストールしている場合、更新記事を表示するには、まず手動で更新をインストールしてください。新しい情報が発表されたときに資料を更新することにより、 *DB2 インフォメーション・センター CD* からインストールした情報を更新することができます。

インフォメーション・センターの方が、PDF 資料やハードコピー資料よりも頻繁に更新されます。 DB2 の最新の技術情報を入手するには、資料更新が発行されたときにそれをインストールするか、または www.ibm.com サイトの DB2 インフォメーション・センターにアクセスしてください。

関連タスク:

- 576 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』

関連資料:

- 569 ページの『DB2 PDF 資料および印刷された資料』

DB2 インフォメーション・センター

DB2® インフォメーション・センターを使用すると、 DB2 Universal Database™、 DB2 Connect™、 DB2 Information Integrator および DB2 Query Patroller™ などの DB2 ファミリー製品を最大限に活用するのに必要なすべての情報にアクセスできます。また、DB2 インフォメーション・センターは、 DB2 の主な機能とコンポーネントに関する情報を提供します (レプリケーション、データウェアハウジング、および DB2 の種々の Extender など)。

Mozilla 1.0 以上または Microsoft® Internet Explorer 5.5 以上で表示する場合、 DB2 インフォメーション・センターには以下の機能があります。以下のいくつかの機能では、 JavaScript™ のサポートを使用可能にする必要があります:

柔軟なインストール・オプション
以下の中から、 ご使用の環境に最も適したオプションを使って DB2 資料を表示できます。

- 最新の資料を常に自動的に利用できるようにするには、 IBM® の Web サイト (http://publib.boulder.ibm.com/infocenter/db2help/) にある DB2 インフォメーション・センターからすべての資料に直接アクセスします。
- 更新処理を最小化し、イントラネット内のネットワーク・トラフィックだけに制限するには、イントラネット上の 1 つのサーバーに DB2 資料をインストールします。
- 柔軟性を改善し、ネットワーク接続への依存を軽減するには、個々のコンピューターに DB2 資料をインストールします。

検索 「検索」テキスト・フィールドに検索語を入力することにより、 DB2 インフォメーション・センターのすべてのトピックを検索できます。複数の語句を引用符で囲めば、完全一致を検索できます。また、ワイルドカード演算子 (*、?) とブール演算子 (AND、 NOT、OR) を使用して検索を絞り込むことができます。

タスク指向の目次
単一の目次の中から、DB2 資料のトピックを見付けることができます。目次は、主に実行するタスクの種類に従って編成されていますが、そのほかに製品概要、特定のゴール (目的) の情報、参照情報、索引、および用語集も含まれます。

- 製品概要では、DB2 ファミリーで使用可能な製品間の関係、そうした各製品で提供される機能、および各製品の最新リリース情報について説明されています。
- インストール、管理および開発などのゴール・カテゴリーには、タスクを迅速に完了し、そのための背景情報をよく理解できるようにするトピックが含まれています。
- 「参照」トピックでは、その対象に関する詳細な情報 (ステートメントとコマンドの構文、メッセージ・ヘルプ、構成パラメーターなど) が説明されています。

現在のトピックを目次に表示する

現在のトピックが目次のどの部分に該当するかを表示するには、目次フレーム内の**「リフレッシュ/現在のトピックの表示 (Refresh/Show Current Topic)」**ボタンをクリックするか、コンテンツ・フレーム内の**「目次に表示 (Show in Table of Contents)」**ボタンをクリックします。幾つかのファイルで関連トピックへの複数のリンクをたどった場合、または検索結果からトピックにアクセスした場合には、この機能が役立ちます。

索引 索引から、すべての資料にアクセスすることができます。索引では、用語が 50 音順に編成されています。

用語集 用語集を見れば、DB2 資料で使われているさまざまな用語の定義を調べることができます。用語集では、用語が 50 音順に編成されています。

組み込まれているローカライズ情報

DB2 インフォメーション・センターは、ブラウザーで設定された言語でトピックを表示します。設定された言語のトピックが利用できない場合、DB2 インフォメーション・センターにはそのトピックの英語版が表示されます。

iSeries™ 技術情報については、 IBM eServer™ iSeries Information Center (www.ibm.com/eserver/iseries/infocenter/) を参照してください。

関連概念:

- 557 ページの『DB2 インフォメーション・センターのインストール・シナリオ』

関連タスク:

- 567 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 568 ページの『DB2 インフォメーション・センターにおける特定の言語でのトピックの表示』
- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 560 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』
- 563 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

DB2 インフォメーション・センターのインストール・シナリオ

さまざまに異なる業務環境のもとでは、 DB2® 情報にどのようにアクセスするかの要件もそれぞれ異なります。 DB2 インフォメーション・センターにアクセスするには、 IBM® の Web サイト、サーバーまたは組織のネットワーク、あるいはコンピューターへのインストールという 3 つの方法が可能です。この 3 つのケースのいずれも、資料は DB2 インフォメーション・センター内に置かれます。インフォメーション・センターは、ブラウザーを使って表示できるように設計されたトピック・ベースの情報の Web サイトです。デフォルトでは、DB2 製品から、 IBM Web サイト上の DB2 インフォメーション・センターにアクセスします。これに対して、 イントラネット・サーバーまたはご自分のコンピューターから DB2 インフォメーション・センターにアクセスしたい場合、製品メディア・パック内にある

DB2 インフォメーション・センター CD から DB2 インフォメーション・センターをインストールする必要があります。以下では、DB2 資料へのアクセス・オプションの要約、および 3 つのインストール・シナリオを示します。これを参考にして、お客様の業務環境で DB2 インフォメーション・センターにアクセスするにはどの方法が最適か、どのようなインストール上の問題に配慮する必要があるかを判別してください。

DB2 資料にアクセスするオプションの要約:

以下の表は、お客様の実際の業務環境で、 DB2 インフォメーション・センターの DB2 製品情報にアクセスする方法としてどんなオプションが推奨されるかを示します。

インターネット・アクセス	イントラネット・アクセス	推奨されるアクション
はい	はい	IBM Web サイト上の DB2 インフォメーション・センターへのアクセス、またはイントラネット・サーバーにインストール済みの DB2 インフォメーション・センターへのアクセス
はい	いいえ	IBM Web サイト上の DB2 インフォメーション・センターへのアクセス
いいえ	はい	イントラネット・サーバーにインストール済みの DB2 インフォメーション・センターへのアクセス
いいえ	いいえ	ローカル・コンピューター上の DB2 インフォメーション・センターへのアクセス

シナリオ: コンピューター上の DB2 インフォメーション・センターへのアクセス:

Tsu-Chen 氏は小さな町で工場を経営していますが、その町には、インターネット・アクセスを提供する地元のインターネット・サービス・プロバイダーがありません。彼は、在庫、製品オーダー、銀行口座情報、および営業経費を管理するために DB2 Universal Database™ を購入しました。 Tsu-Chen 氏は以前に DB2 製品を利用したことがないので、 DB2 の使用方法を習得するために、DB2 製品資料を参照する必要があります。

Tsu-Chen 氏は 標準インストール・オプションを使って DB2 Universal Database を自分のコンピューターにインストールした後、 DB2 資料にアクセスしようとします。しかし、開こうとしているページが見つからないというエラー・メッセージがブラウザーから通知されました。 Tsu-Chen 氏は DB2 製品のインストール・マニュアルを調べた結果、 DB2 資料を自分のコンピューター上で利用するには、 DB2 インフォメーション・センターをインストールしなければならないことに気付きます。そしてメディア・パックの中にあった *DB2 インフォメーション・センター CD* を見つけ出して、インストールしました。

これで、Tsu-Chen 氏はオペレーティング・システムのアプリケーション・ランチャーから DB2 インフォメーション・センターにアクセスできるようになり、より良い業務成果をあげるために DB2 製品を利用する方法を習得できます。

シナリオ: IBM Web サイト上の DB2 インフォメーション・センターへのアクセス:

Colin は、あるセミナー企業に所属する情報技術コンサルタントです。彼の専門はデータベース・テクノロジーおよび SQL で、 DB2 Universal Database を使って北米一帯の企業を対象にこれらの科目のセミナーを開催しています。 Colin のセミナーでは、教材として DB2 資料も使用されます。たとえば、SQL の講習コースでは、データベース照会の基本構文と拡張構文を教えるために SQL に関する DB2 資料が使用されます。

Colin が教えている企業の大半はインターネット・アクセスを配備しています。このような状況から判断して、Colin は、最新バージョンの DB2 Universal Database を自分のモバイル・コンピューターにインストールしたとき、 IBM Web サイト上の DB2 インフォメーション・センターにアクセスするよう構成しました。この構成によって、Colin はセミナーで教えるときに最新の DB2 資料にオンライン・アクセスすることができます。

しかし、時折、Colin は移動中にインターネット・アクセスを利用できないことがあります。これは問題となります。担任するセミナーの準備のために DB2 資料にアクセスする必要のある場合には、とくにそうです。このような事態が起きないようにするために、 Colin は自分のモバイル・コンピューターに DB2 インフォメーション・センターのコピーをインストールしました。

こうして、Colin は常に DB2 資料のコピーを自在に活用できるようになりました。**db2set** コマンドを使って自分のモバイル・コンピューターのレジストリー変数を簡単に構成し、どこにいるかに応じて、IBM Web サイトまたは自分のモバイル・コンピューターから DB2 インフォメーション・センターにアクセスできます。

シナリオ: イントラネット・サーバー上の DB2 インフォメーション・センターへのアクセス:

Eva は、生命保険会社のデータベース上級管理者です。彼女は管理業務の一環として、会社の UNIX® データベース・サーバーに最新バージョンの DB2 Universal Database をインストールおよび構成します。彼女の会社は最近、セキュリティー上の理由から、インターネット・アクセスをもはや業務で利用できないようにすると社員に通知しました。同社はネットワーク環境を装備しているため、 Eva は DB2 インフォメーション・センターのコピーをイントラネット・サーバー上にインストールして、社内のデータウェアハウスを定期的に利用するすべての社員 (営業担当者、営業部長、および業務分析担当者) から DB2 資料へのアクセスを可能にすることにしました。

Eva は、応答ファイルを使って全社員のコンピューター上に最新バージョンの DB2 Universal Database をインストールするようデータベース・チームに指示します。その際、イントラネット・サーバーのホスト名とポート番号を使って DB2 インフォメーション・センターにアクセスできるよう、確実に各コンピューターを構成します。

しかし、Eva のチームの下級データベース管理者である Migual の誤解によって、数人の社員のコンピューター上で、イントラネット・サーバーの DB2 インフォメーション・センターにアクセスするよう DB2 Universal Database を構成する代わり

に、 DB2 インフォメーション・センターのコピーをそれらのコンピューターにインストールしてしまいました。これを訂正するために、Eva は、 **db2set** コマンドを使ってこれらのコンピューター上の DB2 インフォメーション・センターのレジストリー変数 (ホスト名は DB2_DOCHOST、ポート番号は DB2_DOCPORT) を変更するよう Migual に指示しました。これで、ネットワーク上の適切なすべてのコンピューターが DB2 インフォメーション・センターにアクセスできるようになり、社員は DB2 に関する質問の答えを DB2 資料から見つけることができます。

関連概念:

- 556 ページの『DB2 インフォメーション・センター』

関連タスク:

- 567 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 560 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』
- 563 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

関連資料:

- 「コマンド・リファレンス」の『db2set - DB2 プロファイル・レジストリー・コマンド』

DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)

DB2 製品資料にアクセスする方法として、 IBM Web サイト、イントラネット・サーバー、またはコンピューターにインストールしたバージョンの 3 つがあります。デフォルトでは、DB2 製品は IBM Web サイト上の DB2 資料にアクセスします。イントラネット・サーバーまたはコンピューター上の DB2 資料にアクセスしたい場合には、 *DB2 インフォメーション・センター CD* から資料をインストールする必要があります。 DB2 セットアップ・ウィザードを使用すれば、インストール設定を定義し、 UNIX オペレーティング・システムを使用するコンピューターに DB2 インフォメーション・センターをインストールできます。

前提条件:

このセクションでは、UNIX コンピューターに DB2 インフォメーション・センターをインストールするためのハードウェア、オペレーティング・システム、ソフトウェア、および通信の諸要件を一覧で示します。

- **ハードウェア要件**

  以下のいずれかのプロセッサーが必要です。

  - PowerPC (AIX)
  - HP 9000 (HP-UX)
  - Intel 32 ビット (Linux)
  - Solaris UltraSPARC コンピューター (Solaris オペレーティング環境)

- **オペレーティング・システム要件**

  以下のいずれかのオペレーティング・システムが必要です。

  – IBM AIX 5.1 (PowerPC 上)
  – HP-UX 11i (HP 9000 上)
  – Red Hat Linux 8.0 (Intel 32 ビット上)
  – SuSE Linux 8.1 (Intel 32 ビット上)
  – Sun Solaris バージョン 8 (Solaris オペレーティング環境の UltraSPARC コンピューター上)

  注: DB2 インフォメーション・センターは、 DB2 クライアントをサポートする UNIX オペレーティング・システム上で稼動します。このため、IBM Web サイトから DB2 インフォメーション・センターにアクセスするか、イントラネット・サーバーに DB2 インフォメーション・センターをインストールしてそれにアクセスすることをお勧めします。

- **ソフトウェア要件**

  – 以下のブラウザーがサポートされています。

    - Mozilla バージョン 1.0 以上

- DB2 セットアップ・ウィザードは、グラフィック・インストーラーです。ご使用のマシンで DB2 セットアップ・ウィザードのグラフィカル・ユーザー・インターフェースを表示可能にする X Window システム・ソフトウェアをインプリメントする必要があります。 DB2 セットアップ・ウィザードを実行する前に、ディスプレイを正しくエクスポートしたことを確認してください。たとえば、コマンド・プロンプトで

  ```
  export DISPLAY=9.26.163.144:0.
  ```

  というコマンドを入力します。

- **通信要件**

  – TCP/IP

手順:

DB2 セットアップ・ウィザードを使用して DB2 インフォメーション・センターをインストールするには、以下のようにします。

1. システムにログオンします。
2. DB2 インフォメーション・センター製品 CD を挿入してシステムにマウントします。
3. 次のコマンドを入力して、CD がマウントされているディレクトリーに移動します。

   ```
   cd /cd
   ```

   /cd は、CD のマウント・ポイントを表します。
4. **./db2setup** コマンドを入力して、 DB2 セットアップ・ウィザードを開始します。
5. IBM DB2 セットアップ・ランチパッドが開きます。 DB2 インフォメーション・センターのインストールに直接進むには、**「製品のインストール」**をクリックします。残りのステップについて説明しているオンライン・ヘルプを利用

できます。オンライン・ヘルプを呼び出すには、**「ヘルプ」**をクリックします。**「キャンセル」**をクリックすれば、いつでもインストールを終了できます。

6. **「インストールしたい製品を選択します」**ページでは、**「次へ」**をクリックします。
7. **「DB2 セットアップ・ウィザードにようこそ (Welcome to the DB2 Setup wizard)」**ページで、**「次へ」**をクリックします。 DB2 セットアップ・ウィザードは、プログラムのセットアップ操作を案内します。
8. インストールを続行するには、使用許諾条件に同意する必要があります。**「ご使用条件」**ページで、**「ご使用条件に同意します (I accept the terms in the license agreement)」** をクリックして、**「次へ」**をクリックします。
9. **「インストール・アクションの選択」**で、**「このコンピューターに DB2 インフォメーション・センターをインストールする (Install DB2 Information Center on this computer)」**を選択します。応答ファイルを使用して、このコンピューターまたは他のコンピューターに DB2 インフォメーション・センターをあとでインストールしたい場合には、**「設定を応答ファイルに保管する」**を選択します。**「次へ」**をクリックします。
10. **「インストールする言語の選択」**ページでは、 DB2 インフォメーション・センターをインストールする言語を選択します。**「次へ」**をクリックします。
11. **「DB2 インフォメーション・センター・ポートの指定」**ページでは、 DB2 インフォメーション・センターへの着信通信を構成します。**「次へ」**をクリックしてインストールを続けます。
12. **「ファイルのコピーの開始」**ページでは、インストールの選択項目を確認します。設定を変更するには、**「戻る」**をクリックします。**「インストール」**をクリックすると、 DB2 インフォメーション・センターのファイルがコンピューターにコピーされます。

このほか、応答ファイルを使って DB2 インフォメーション・センターをインストールすることもできます。

インストール・ログ db2setup.his、 db2setup.log、および db2setup.err は、デフォルトでは /tmp ディレクトリーに置かれます。

db2setup.log ファイルは、エラーも含めた DB2 製品のインストール情報をすべてキャプチャーします。 db2setup.his ファイルは、コンピューター上の DB2 製品インストール内容をすべて記録します。 DB2 は、db2setup.log ファイルを db2setup.his に付加します。 db2setup.err ファイルは、 Java から戻されるすべてのエラー出力 (例外やトラップの情報など) をキャプチャーします。

インストールが完了したら、ご使用の UNIX オペレーティング・システムに応じて、 DB2 は以下のいずれかのディレクトリーにインストールされます。

- AIX: /usr/opt/db2_08_01
- HP-UX: /opt/IBM/db2/V8.1
- Linux: /opt/IBM/db2/V8.1
- Solaris オペレーティング環境: /opt/IBM/db2/V8.1

関連概念:

- 556 ページの『DB2 インフォメーション・センター』
- 557 ページの『DB2 インフォメーション・センターのインストール・シナリオ』

関連タスク:

- 「*インストールおよび構成 補足*」の『応答ファイルによる DB2 のインストール (UNIX)』
- 567 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 568 ページの『DB2 インフォメーション・センターにおける特定の言語でのトピックの表示』
- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 563 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)

DB2 製品資料にアクセスする方法として、 IBM Web サイト、イントラネット・サーバー、またはコンピューターにインストールしたバージョンの 3 つがあります。デフォルトでは、DB2 製品は IBM Web サイト上の DB2 資料にアクセスします。イントラネット・サーバーまたはコンピューター上の DB2 資料にアクセスしたい場合には、 *DB2 インフォメーション・センター CD* から DB2 資料をインストールする必要があります。 DB2 セットアップ・ウィザードを使用すれば、インストール設定を定義し、 Windows オペレーティング・システムを使用するコンピューターに DB2 インフォメーション・センターをインストールできます。

前提条件:

このセクションでは、Windows に DB2 インフォメーション・センターをインストールするためのハードウェア、オペレーティング・システム、ソフトウェア、および通信の諸要件を一覧で示します。

- **ハードウェア要件**

  以下のいずれかのプロセッサーが必要です。

  – 32 ビット・コンピューター: Pentium または Pentium 互換の CPU

- **オペレーティング・システム要件**

  以下のいずれかのオペレーティング・システムが必要です。

  – Windows 2000

  – Windows XP

  注: DB2 インフォメーション・センターは、 DB2 クライアントをサポートする Windows オペレーティング・システム上で稼動します。このため、IBM Web サイトの DB2 インフォメーション・センターにアクセスするか、イントラネット・サーバーに DB2 インフォメーション・センターをインストールしてそれにアクセスすることをお勧めします。

- **ソフトウェア要件**

- 以下のブラウザーがサポートされています。
  - Mozilla 1.0 以上
  - Internet Explorer バージョン 5.5 または 6.0 (Windows XP の場合はバージョン 6.0)

- **通信要件**
  - TCP/IP

制約事項:

- DB2 インフォメーション・センターをインストールするには、管理権限をもつアカウントが必要です。

手順:

DB2 セットアップ・ウィザードを使用して DB2 インフォメーション・センターをインストールするには、以下のようにします。

1. DB2 インフォメーション・センターのインストールで定義したアカウントで、システムにログオンします。
2. CD をドライブに挿入します。自動実行機能が使用可能になっていれば、IBM DB2 セットアップ・ランチパッドが起動します。
3. DB2 セットアップ・ウィザードは、システム言語を判別して、その言語用のセットアップ・プログラムを立ち上げます。英語以外の言語でセットアップ・プログラムを実行したい場合、またはセットアップ・プログラムの自動始動が失敗した場合には、 DB2 セットアップ・ウィザードを手動で開始できます。

   次のようにして、DB2 セットアップ・ウィザードを手動で開始します。

   a. **「スタート」**をクリックし、**「ファイル名を指定して実行」**を選択します。
   b. **「開く」**フィールドで、以下のコマンドを入力します。

      ```
      x:¥setup.exe /i 2-letter language identifier
      ```

      ここで、*x:* は CD ドライブ、 *2-letter language identifier* (2 文字の言語識別子) はセットアップ・プログラムを実行する言語を表します。
   c. **「OK」**をクリックします。
4. IBM DB2 セットアップ・ランチパッドが開きます。 DB2 インフォメーション・センターのインストールに直接進むには、**「製品のインストール」**をクリックします。残りのステップについて説明しているオンライン・ヘルプを利用できます。オンライン・ヘルプを呼び出すには、**「ヘルプ」**をクリックします。**「キャンセル」**をクリックすれば、いつでもインストールを終了できます。
5. **「インストールしたい製品を選択します」**ページでは、**「次へ」**をクリックします。
6. **「DB2 セットアップ・ウィザードにようこそ (Welcome to the DB2 Setup wizard)」**ページで、**「次へ」**をクリックします。 DB2 セットアップ・ウィザードは、プログラムのセットアップ操作を案内します。
7. インストールを続行するには、使用許諾条件に同意する必要があります。**「ご使用条件」**ページで、**「ご使用条件に同意します (I accept the terms in the license agreement)」** をクリックして、**「次へ」**をクリックします。

8. **「インストール・アクションの選択」**で、**「このコンピューターに DB2 インフォメーション・センターをインストールする (Install DB2 Information Center on this computer)」**を選択します。応答ファイルを使用して、このコンピューターまたは他のコンピューターに DB2 インフォメーション・センターをあとでインストールしたい場合には、**「設定を応答ファイルに保管する」**を選択します。**「次へ」**をクリックします。
9. **「インストールする言語の選択」**ページでは、 DB2 インフォメーション・センターをインストールする言語を選択します。**「次へ」**をクリックします。
10. **「DB2 インフォメーション・センター・ポートの指定」**ページでは、 DB2 インフォメーション・センターへの着信通信を構成します。**「次へ」**をクリックしてインストールを続けます。
11. **「ファイルのコピーの開始」**ページでは、インストールの選択項目を確認します。設定を変更するには、**「戻る」**をクリックします。**「インストール」**をクリックすると、 DB2 インフォメーション・センターのファイルがコンピューターにコピーされます。

応答ファイルを使って DB2 インフォメーション・センターをインストールすることができます。また、**db2rspgn** コマンドを使って、既存のインストール内容に基づく応答ファイルを生成することもできます。

インストール時に検出されるエラーの詳細については、「マイ ドキュメント」¥DB2LOG¥ ディレクトリー内の db2.log ファイルと db2wi.log ファイルを参照してください。「マイ ドキュメント」 ディレクトリーの場所は、ご使用のコンピューターの設定によって異なります。

db2wi.log ファイルは、 DB2 の最新のインストール情報をキャプチャーします。db2.log は、DB2 製品のインストールの履歴をキャプチャーします。

関連概念:

- 556 ページの『DB2 インフォメーション・センター』
- 557 ページの『DB2 インフォメーション・センターのインストール・シナリオ』

関連タスク:

- 「*インストールおよび構成 補足*」の『応答ファイルによる DB2 製品のインストール (Windows)』
- 567 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 568 ページの『DB2 インフォメーション・センターにおける特定の言語でのトピックの表示』
- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 560 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』

関連資料:

- 「コマンド・リファレンス」の『db2rspgn - 応答ファイル生成プログラム・コマンド』

DB2 インフォメーション・センターの呼び出し

DB2 インフォメーション・センターは、Linux、UNIX、および Windows オペレーティング・システム用の DB2 製品 (DB2 Universal Database、 DB2 Connect、DB2 Information Integrator、 DB2 Query Patroller など) を使用するために必要なすべての情報を提供します。

DB2 インフォメーション・センターは、以下の場所から呼び出すことができます。

- DB2 UDB クライアントまたはサーバーがインストールされているコンピューター
- DB2 インフォメーション・センターがインストールされているイントラネット・サーバーまたはローカル・コンピューター
- IBM の Web サイト

前提条件:

DB2 インフォメーション・センターを呼び出すための要件は、以下のとおりです。

- *オプショナル*: 希望する言語でトピックを表示するようブラウザーを構成する
- *オプショナル*: コンピューターまたはイントラネット・サーバーにインストール済みの DB2 インフォメーション・センターを使用するよう DB2 クライアントを構成する

手順:

DB2 UDB クライアントまたはサーバーがインストールされているコンピューターから DB2 インフォメーション・センターを呼び出すには、以下のようにします。

- (Windows オペレーティング・システムの)「スタート」メニューから: **「スタート」→「プログラム」→「IBM DB2」→「情報」→「インフォメーション・センター」**をクリックします。
- コマンド行プロンプトから:
  - Linux および UNIX オペレーティング・システムの場合、 **db2icdocs** コマンドを発行します。
  - Windows オペレーティング・システムの場合、 **db2icdocs.exe** コマンドを発行します。

イントラネット・サーバーまたはローカル・コンピューターにインストール済みの DB2 インフォメーション・センターを Web ブラウザーで開くには、以下のようにします。

- Web ページ http://<host-name>:<port-number>/ を開きます (<host-name> はホスト名、 <port-number> は DB2 インフォメーション・センターを利用可能なポート番号)。

IBM Web サイトにある DB2 インフォメーション・センターを Web ブラウザーで開くには、以下のようにします。

- Web ページ publib.boulder.ibm.com/infocenter/db2help/ を開きます。

関連概念:

- 556 ページの『DB2 インフォメーション・センター』

関連タスク:

- 568 ページの『DB2 インフォメーション・センターにおける特定の言語でのトピックの表示』
- 576 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』
- 567 ページの『コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール』
- 577 ページの『コマンド行プロセッサーからメッセージ・ヘルプを呼び出す』
- 578 ページの『コマンド行プロセッサーからコマンド・ヘルプを呼び出す』
- 578 ページの『コマンド行プロセッサーから SQL 状態ヘルプを呼び出す』

コンピューターまたはイントラネット・サーバーへの DB2 インフォメーション・センターの更新インストール

http://publib.boulder.ibm.com/infocenter/db2help/ から利用できる DB2 インフォメーション・センターは、資料の新規追加または変更によって定期的に更新されます。さらに、更新された DB2 インフォメーション・センターをコンピューターまたはイントラネット・サーバーにダウンロードしてインストールできる場合もあります。DB2 インフォメーション・センターを更新しても、 DB2 クライアント製品またはサーバー製品は更新されません。

前提条件:

インターネットに接続されたコンピューターへのアクセスが必要です。

手順:

DB2 インフォメーション・センターの更新をコンピューターまたはイントラネット・サーバーにインストールするには、以下のようにします。

1. IBM の Web サイト (http://publib.boulder.ibm.com/infocenter/db2help/) にある DB2 インフォメーション・センターを開きます。
2. 「DB2 インフォメーション・センターにようこそ」ページの見出し「サービスおよびサポート」の「ダウンロード」セクションで、**「DB2 資料」**リンクをクリックします。
3. 最新のドキュメンテーション・イメージのレベルと、インストール済みのドキュメンテーション・レベルを比較して、 DB2 インフォメーション・センターを更新する必要があるかどうかを確認します。「DB2 インフォメーション・センターにようこそ」ページに、インストール済みのドキュメンテーションのレベルがリストされます。
4. より新しいバージョンの DB2 インフォメーション・センターが存在する場合、ご使用のオペレーティング・システムに対応する最新の *DB2* インフォメーション・センター・イメージをダウンロードします。
5. 最新の *DB2* インフォメーション・センター・イメージをインストールするには、 Web ページの指示に従ってください。

関連概念:

- 557 ページの『DB2 インフォメーション・センターのインストール・シナリオ』

関連タスク:

- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 560 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (UNIX)』
- 563 ページの『DB2 セットアップ・ウィザードを使用した DB2 インフォメーション・センターのインストール (Windows)』

DB2 インフォメーション・センターにおける特定の言語でのトピックの表示

DB2 インフォメーション・センターでは、ブラウザーの設定で指定した言語でのトピックの表示が試みられます。トピックがその指定言語に翻訳されていない場合は、 DB2 インフォメーション・センターでは英語でトピックが表示されます。

手順:

Internet Explorer Web ブラウザーで、指定どおりの言語でトピックを表示するには、以下のようにします。

1. Internet Explorer の**「ツール」**—> **「インターネット オプション」**—> **「言語...」**ボタンをクリックします。「言語の優先順位」ウィンドウがオープンします。
2. 該当する言語が、言語リストの先頭の項目に指定されていることを確認します。
   - リストに新しい言語を追加するには、**「追加...」**ボタンをクリックします。

     注: 言語を追加しても、特定の言語でトピックを表示するのに必要なフォントがコンピューターに備えられているとはかぎりません。
   - リストの先頭に新しい言語を移動するには、その言語を選択してから、その言語が言語リストに先頭に行くまで**「上へ」**ボタンをクリックします。
3. 使いたい言語で DB2 インフォメーション・センターを表示するには、ページをリフレッシュします。

Mozilla Web ブラウザーの場合に、使いたい言語でトピックを表示するには、以下のようにします。

1. Mozilla の**「編集」**—> **「設定」**—> **「言語」**ボタンをクリックします。「設定」ウィンドウに「言語」パネルが表示されます。
2. 該当する言語が、言語リストの先頭の項目に指定されていることを確認します。
   - リストに新しい言語を追加するには、**「追加...」**ボタンをクリックしてから、「言語を追加」ウィンドウで言語を選択します。
   - リストの先頭に新しい言語を移動するには、その言語を選択してから、その言語が言語リストに先頭に行くまで**「上に移動」**ボタンをクリックします。
3. 使いたい言語で DB2 インフォメーション・センターを表示するには、ページをリフレッシュします。

DB2 PDF 資料および印刷された資料

以下の表は、正式な資料名、資料番号、および PDF ファイル名を示しています。ハードコピー版の資料を注文するには、正式な資料名を知っておく必要があります。 PDF ファイルを印刷するには、PDF ファイル名を知っておく必要があります。

DB2 資料は、以下のカテゴリーに分類されています。

- DB2 中核情報
- 管理情報
- アプリケーション開発情報
- ビジネス・インテリジェンス情報
- DB2 Connect 情報
- 入門情報
- チュートリアル情報
- オプショナル・コンポーネント情報
- リリース・ノート

以下の表は、DB2 ライブラリー内の各資料について、その資料のハードコピー版を注文したり、PDF 版を印刷または表示したりするのに必要な情報を示しています。DB2 ライブラリー内の各資料に関する詳細な説明については、www.ibm.com/shop/publications/order にある IBM Publications Center にアクセスしてください。

DB2 の基本情報

こうした資料の情報は、すべての DB2 ユーザーに基本的なもので、プログラマーおよびデータベース管理者にとって役立つ情報であるとともに、DB2 Connect、DB2 Warehouse Manager、または他の DB2 製品を使用するユーザーにとっても役立つ内容です。

表 774. DB2 の基本情報

資料名	資料番号	PDF ファイル名
「*IBM DB2 Universal Database コマンド・リファレンス*」	SC88-9140	db2n0j81
「*IBM DB2 Universal Database 用語集*」	資料番号なし	db2t0j81
「*IBM DB2 Universal Database メッセージ・リファレンス 第 1 巻*」	GC88-9152 (ハードコピーなし)	db2m1j81
「*IBM DB2 Universal Database メッセージ・リファレンス 第 2 巻*」	GC88-9153 (ハードコピーなし)	db2m2j81
「*IBM DB2 Universal Database 新機能*」	SC88-9158	db2q0j81

管理情報

これらの資料の情報は、DB2 データベース、データウェアハウス、およびフェデレーテッド・システムを効果的に設計し、インプリメントし、保守するために必要なトピックを扱っています。

表 775. 管理情報

資料名	資料番号	PDF ファイル名
「*IBM DB2 Universal Database 管理ガイド: プランニング*」	SC88-9135	db2d1j81
「*IBM DB2 Universal Database 管理ガイド: インプリメンテーション*」	SC88-9133	db2d2j81
「*IBM DB2 Universal Database 管理ガイド: パフォーマンス*」	SC88-9134	db2d3j81
「*IBM DB2 Universal Database 管理 API リファレンス*」	SC88-9136	db2b0j81
「*IBM DB2 Universal Database データ移動ユーティリティー ガイドおよびリファレンス*」	SC88-9142	db2dmj81
「*IBM DB2 Universal Database データ・リカバリーと高可用性 ガイドおよびリファレンス*」	SC88-9143	db2haj81
「*IBM DB2 Universal Database データウェアハウス・センター 管理ガイド*」	SC88-9165	db2ddj81
「*IBM DB2 Universal Database SQL リファレンス 第 1 巻*」	SC88-9155	db2s1j81
「*IBM DB2 Universal Database SQL リファレンス 第 2 巻*」	SC88-9156	db2s2j81
「*IBM DB2 Universal Database システム・モニター ガイドおよびリファレンス*」	SC88-9157	db2f0j81

アプリケーション開発情報

これらの資料の情報は、 DB2 Universal Database (DB2 UDB) のアプリケーション開発者またはプログラマーが特に関心を持つ内容です。サポートされるさまざまなプログラミング・インターフェース (組み込み SQL、ODBC、JDBC、SQLJ、CLI など) を使用して DB2 UDB にアクセスするのに必要な資料とともに、サポートされる言語およびコンパイラーについても紹介されています。また、DB2 インフォメーション・センターをご使用の場合には、サンプル・プログラムのソース・コードの HTML バージョンにアクセスすることもできます。

表 776. アプリケーション開発情報

資料名	資料番号	PDF ファイル名
「*IBM DB2 Universal Database アプリケーション開発ガイド アプリケーションの構築および実行*」	SC88-9137	db2axj81
「*IBM DB2 Universal Database アプリケーション開発ガイド クライアント・アプリケーションのプログラミング*」	SC88-9138	db2a1j81
「*IBM DB2 Universal Database アプリケーション開発ガイド サーバー・アプリケーションのプログラミング*」	SC88-9139	db2a2j81
「*IBM DB2 Universal Database コール・レベル・インターフェース ガイドおよびリファレンス 第 1 巻*」	SC88-9159	db2l1j81
「*IBM DB2 Universal Database コール・レベル・インターフェース ガイドおよびリファレンス 第 2 巻*」	SC88-9160	db2l2j81
「*IBM DB2 Universal Database データウェアハウス・センター アプリケーション統合ガイド*」	SC88-9166	db2adj81
「*IBM DB2 Universal Database XML Extender 管理およびプログラミングのガイド*」	SC88-9172	db2sxj81

ビジネス・インテリジェンス情報

これらの資料の情報は、さまざまなコンポーネントを使用して、 DB2 Universal Database のデータウェアハウジング機能および分析機能を拡張する方法を説明しています。

表 777. ビジネス・インテリジェンス情報

資料名	資料番号	PDF ファイル名
「*IBM DB2 Warehouse Manager Standard Edition インフォメーション・カタログ・センター 管理ガイド*」	SC88-9167	db2dij81
「*IBM DB2 Warehouse Manager Standard Edition インストール・ガイド*」	GC88-9164	db2idj81

表 777. ビジネス・インテリジェンス情報 (続き)

資料名	資料番号	PDF ファイル名
「*IBM DB2 Warehouse Manager Standard Edition DB2 Warehouse Manager を使用時の ETI ソリューション・コンバージョン・プログラムの管理*」	SC88-9894	iwhe1mstx80

DB2 Connect 情報

このカテゴリーの情報は、 DB2 Connect Enterprise Edition または DB2 Connect Personal Edition を使用して、メインフレーム・サーバーおよびミッドレンジ・サーバー上のデータにアクセスする方法を説明しています。

表 778. DB2 Connect 情報

資料名	資料番号	PDF ファイル名
「*IBM コネクティビティー 補足*」	資料番号なし	db2h1j81
「*IBM DB2 Connect Enterprise Edition 概説およびインストール*」	GC88-9145	db2c6j81
「*IBM DB2 Connect Personal Edition 概説およびインストール*」	GC88-9146	db2c1j81
「*IBM DB2 Connect ユーザーズ・ガイド*」	SC88-9147	db2c0j81

入門情報

このカテゴリーの情報は、サーバー、クライアント、および他の DB2 製品をインストールして構成する場合に役立ちます。

表 779. 入門情報

資料名	資料番号	PDF ファイル名
「*IBM DB2 Universal Database DB2 クライアント機能 概説およびインストール*」	GC88-9144 (ハードコピーなし)	db2itj81
「*IBM DB2 Universal Database DB2 サーバー機能 概説およびインストール*」	GC88-9148	db2isj81
「*IBM DB2 Universal Database DB2 Personal Edition 概説およびインストール*」	GC88-9150	db2i1j81
「*IBM DB2 Universal Database インストールおよび構成 補足*」	GC88-9149 (ハードコピーなし)	db2iyj81

表 779. 入門情報 (続き)

資料名	資料番号	PDF ファイル名
「*IBM DB2 Universal Database DB2 Data Links Manager 概説およびインストール*」	GC88-9141	db2z6j81

チュートリアル情報

チュートリアル情報は、DB2 機能を紹介し、さまざまなタスクを実行する方法を示します。

表 780. チュートリアル情報

資料名	資料番号	PDF ファイル名
「*ビジネス・インテリジェンス・チュートリアル: データウェアハウス・センターの紹介*」	資料番号なし	db2tuj81
「*ビジネス・インテリジェンス・チュートリアル: データウェアハウジングの上級者向けガイド*」	資料番号なし	db2taj81
「*インフォメーション・カタログ・センター チュートリアル*」	資料番号なし	db2aij81
「*Video Central for e-business チュートリアル*」	資料番号なし	db2twj81
「*Visual Explain チュートリアル*」	資料番号なし	db2tvj81

オプショナル・コンポーネント情報

このカテゴリーの情報は、 DB2 のオプショナル・コンポーネントを使用する方法について説明しています。

表 781. オプショナル・コンポーネント情報

資料名	資料番号	PDF ファイル名
「*IBM DB2 Cube Views Guide and Reference*」	SC18–7298	db2aax81
「*IBM DB2 Query Patroller インストール、管理、使用法のガイド*」	GC88–9154	db2dwj81
「*IBM DB2 Spatial Extender and Geodetic Extender ユーザーズ・ガイドおよびリファレンス*」	SC88-9171	db2sbj81

表 781. オプショナル・コンポーネント情報 (続き)

資料名	資料番号	PDF ファイル名
「*IBM DB2 Universal Database Data Links Manager 管理ガイドおよびリファレンス*」	SC88-9169	db2z0x82
「*DB2 Net Search Extender 管理およびユーザーズ・ガイド*」 **注:** この資料の HTML 版は、HTML ドキュメンテーション CD からインストールされません。	SH88-8546	N/A

リリース・ノート

リリース・ノートは、ご使用の製品のリリースおよびフィックスパック・レベルに特有の追加情報を紹介します。また、リリース・ノートには、各リリース、アップデート、およびフィックスパックで組み込まれた資料上の更新の要約も含まれています。

表 782. リリース・ノート

資料名	資料番号	PDF ファイル名
「*DB2 リリース・ノート*」	「注」を参照。	「注」を参照。
「*DB2 インストール情報*」	製品 CD-ROM でのみ参照可能。	使用できません。

注: リリース・ノートは以下の形式で入手できます。

- XHTML およびテキスト形式 (製品 CD 内)
- PDF 形式 (PDF ドキュメンテーション CD 内)

さらに、リリース・ノートの中で、『*既知の問題と予備手段*』および『*リリース間の非互換性*』に関する部分は DB2 インフォメーション・センターにも表示されます。

UNIX ベースのプラットフォームでテキスト形式でリリース・ノートを確認するには、 Release.Notes ファイルを参照してください。このファイルは、DB2DIR/Readme/*%L* ディレクトリーに収録されています。 *%L* はロケール名を表しています。 DB2DIR は以下になります。

- AIX オペレーティング・システムの場合: /usr/opt/db2_08_01
- その他のすべての UNIX ベースのオペレーティング・システムの場合: /opt/IBM/db2/V8.1

関連概念:

- 555 ページの『DB2 資料とヘルプ』

関連タスク:

- 575 ページの『PDF ファイルからの DB2 資料の印刷方法』

- 576 ページの『DB2 の印刷資料の注文方法』
- 576 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』

PDF ファイルからの DB2 資料の印刷方法

DB2 PDF ドキュメンテーション CD に収録されている DB2 資料を印刷することができます。 Adobe Acrobat Reader を使用すれば、資料全体または特定のページを印刷できます。

前提条件:

Adobe Acrobat Reader がインストールされていることを確認してください。 Adobe Acrobat Reader をインストールする必要がある場合、 Adobe Web サイト (www.adobe.com) から入手できます。

手順:

PDF ファイルから DB2 資料を印刷するには以下のようにします。

1. *DB2 PDF* ドキュメンテーション CD をドライブに挿入します。 UNIX オペレーティング・システムの場合、 DB2 PDF ドキュメンテーション CD をマウントします。 UNIX オペレーティング・システムで CD をマウントする方法については、「*概説およびインストール*」を参照してください。
2. index.htm を開きます。ブラウザー・ウィンドウにファイルが開きます。
3. 参照したい PDF のタイトルをクリックします。 Acrobat Reader で PDF が開きます。
4. **「ファイル」 → 「印刷」**を選択して、所要の資料の任意の部分を印刷します。

関連概念:

- 556 ページの『DB2 インフォメーション・センター』

関連タスク:

- 「*DB2 Universal Database サーバー機能 概説およびインストール*」の『CD-ROM のマウント (AIX)』
- 「*DB2 Universal Database サーバー機能 概説およびインストール*」の『HP-UX 上での CD-ROM のマウント』
- 「*DB2 Universal Database サーバー機能 概説およびインストール*」の『CD-ROM のマウント (Linux)』
- 576 ページの『DB2 の印刷資料の注文方法』
- 「*DB2 Universal Database サーバー機能 概説およびインストール*」の『CD-ROM のマウント (Solaris)』

関連資料:

- 569 ページの『DB2 PDF 資料および印刷された資料』

DB2 の印刷資料の注文方法

ハードコピー版の資料を望む場合には、以下のいずれかの方法で注文できます。

印刷資料の注文方法:

一部の国または地域では、印刷された資料を注文することもできます。お客様がお住まいの国または地域でこのサービスが利用可能かどうかを確認するには、お住まいの国または地域の IBM Publications Web サイトをご覧ください。資料のご注文が可能な場合、以下のようにすることができます。

- 正規の IBM 製品販売業者または営業担当員に連絡してください。お客様がお住まいの地域の IBM 担当員の情報については、お手数ですが IBM の Web サイト (www.ibm.com/planetwide) の IBM Worldwide Directory of Contacts で確認してください。
- IBM Publications Center (http://www.ibm.com/shop/publications/order) にアクセスしてください。なお、IBM Publications Center から資料を注文できない国もあります。

DB2 製品がご利用可能になった時点で、印刷された資料は *DB2 PDF* ドキュメンテーション *CD* にある PDF 形式の資料と同じものです。さらに、 *DB2* インフォメーション・センター *CD* に収録されている印刷された資料の内容もまた、これらと同じです。ただし、DB2 インフォメーション・センター CD には、 PDF 資料にない追加情報も含まれます (たとえば、SQL 管理作業や HTML サンプル)。 DB2 PDF ドキュメンテーション CD に収録されている資料の中には、ハードコピーとしてご注文できない資料もあります。

注: DB2 インフォメーション・センターは、 PDF またはハードコピー の資料よりも頻繁に更新されます。ドキュメンテーションの更新が入手可能になった時点でインストールするか、 DB2 インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/db2help/) を参照して最新の情報を入手してください。

関連タスク:

- 575 ページの『PDF ファイルからの DB2 資料の印刷方法』

関連資料:

- 569 ページの『DB2 PDF 資料および印刷された資料』

DB2 ツールからコンテキスト・ヘルプを呼び出す

コンテキスト・ヘルプは、特定のウィンドウ、ノートブック、ウィザード、またはアドバイザーに関連したタスクまたはコントロールの情報を提供します。コンテキスト・ヘルプは、グラフィカル・ユーザー・インターフェースのある DB2 管理ツールおよび開発ツールから利用できます。コンテキスト・ヘルプには、以下の 2 種類があります。

- それぞれのウィンドウまたはノートブックにある**「ヘルプ」**ボタンからアクセス可能なヘルプ

- infopop (ポップアップ情報ウィンドウ)。これは、マウス・カーソルを特定のフィールドまたはコントロール上に置いたとき、またはウィンドウ、ノートブック、ウィザード、アドバイザー内でフィールドまたはコントロールを選択して F1 を押すと表示されます。

「ヘルプ」ボタンを押すと、概説、前提条件、およびタスク情報が表示されます。infopop は、それぞれのフィールドおよびコントロールについて説明します。

手順:

コンテキスト・ヘルプを呼び出すには、以下のようにします。

- ウィンドウおよびノートブックのヘルプを表示するには、いずれかの DB2 ツールを開始して、任意のウィンドウまたはノートブックを開きます。ウィンドウまたはノートブックの右下隅にある**「ヘルプ」**ボタンをクリックして、コンテキスト・ヘルプを呼び出します。

  また、それぞれの DB2 ツール・センターの上部にある**「ヘルプ」**メニュー項目からコンテキスト・ヘルプにアクセスすることもできます。

  ウィザードおよびアドバイザーでは、最初のページの「タスクの概要」リンクをクリックすると、コンテキスト・ヘルプを表示できます。
- ウィンドウまたはノートブック上の各コントロールの infopop ヘルプを表示するには、コントロールをクリックしてから、**F1** を押します。コントロールの詳細情報を示すポップアップ情報が、黄色いウィンドウに表示されます。

  注: フィールドまたはコントロールにマウス・カーソルを置いておくだけで infopops が表示されるようにするには、「ツール設定」ノートブックの**「文書 (Documentation)」**ページの**「infopops の自動表示」**チェック・ボックスを選択します。

  infopop に似た別のコンテキスト・ヘルプに、診断ポップアップ情報があります。これにはデータ入力規則が示されます。診断ポップアップ情報は、無効または不十分なデータが入力されたとき、紫色のウィンドウに表示されます。診断ポップアップ情報は、以下に関して表示されます。
  - 必須フィールド。
  - 日付フィールドのように、正確なフォーマットを必要とするデータのフィールド。

関連タスク:

- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 577 ページの『コマンド行プロセッサーからメッセージ・ヘルプを呼び出す』
- 578 ページの『コマンド行プロセッサーからコマンド・ヘルプを呼び出す』
- 578 ページの『コマンド行プロセッサーから SQL 状態ヘルプを呼び出す』

コマンド行プロセッサーからメッセージ・ヘルプを呼び出す

メッセージ・ヘルプは、メッセージが出された原因と、エラーへの応答として実行すべきアクションを説明します。

手順:

メッセージ・ヘルプを呼び出すには、コマンド行プロセッサーを開いて以下のように入力します。

```
? XXXnnnnn
```

ここで、*XXXnnnnn* は有効なメッセージ ID を表します。

たとえば、`? SQL30081` と入力すると、メッセージ SQL30081 に関するヘルプを表示します。

関連概念:

- 「メッセージ・リファレンス *第 1 巻*」の『メッセージの概要』

関連資料:

- 「コマンド・リファレンス」の『db2 - コマンド行プロセッサーの呼び出しコマンド』

コマンド行プロセッサーからコマンド・ヘルプを呼び出す

コマンド・ヘルプは、コマンド行プロセッサーでのコマンドの構文を説明します。

手順:

コマンド・ヘルプを呼び出すには、コマンド行プロセッサーを開いて以下のように入力します。

```
? command
```

ここで *command* はキーワードまたはコマンド全体を表します。

たとえば、`? catalog` と入力すると、すべての CATALOG コマンドに関するヘルプが表示され、 `? catalog database` と入力すると、 CATALOG DATABASE コマンドのヘルプだけが表示されます。

関連タスク:

- 576 ページの『DB2 ツールからコンテキスト・ヘルプを呼び出す』
- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 577 ページの『コマンド行プロセッサーからメッセージ・ヘルプを呼び出す』
- 578 ページの『コマンド行プロセッサーから SQL 状態ヘルプを呼び出す』

関連資料:

- 「コマンド・リファレンス」の『db2 - コマンド行プロセッサーの呼び出しコマンド』

コマンド行プロセッサーから SQL 状態ヘルプを呼び出す

DB2 Univerrsal Database は、 SQL ステートメントの結果の原因となったと考えられる条件の SQLSTATE 値を戻します。 SQLSTATE ヘルプは、SQL 状態および SQL 状態クラス・コードの意味を説明します。

手順:

SQL 状態ヘルプを呼び出すには、コマンド行プロセッサーを開いて以下のように入力します。

```
? sqlstate または ? class code
```

ここで、*sqlstate* は有効な 5 桁の SQL 状態を、*class code* は SQL 状態の最初の 2 桁を表します。

たとえば、`? 08003` を指定すると SQL 状態 08003 のヘルプが表示され、 `? 08` を指定するとクラス・コード 08 のヘルプが表示されます。

関連タスク:

- 566 ページの『DB2 インフォメーション・センターの呼び出し』
- 577 ページの『コマンド行プロセッサーからメッセージ・ヘルプを呼び出す』
- 578 ページの『コマンド行プロセッサーからコマンド・ヘルプを呼び出す』

DB2 チュートリアル

DB2® チュートリアルは、 DB2 Universal Database のさまざまな機能について学習するのを支援します。このチュートリアルでは、アプリケーションの開発、SQL 照会のパフォーマンス調整、データウェアハウスの処理、メタデータの管理、および DB2 を使用した Web サービスの開発の各分野で、段階的なレッスンが用意されています。

はじめに:

インフォメーション・センター (http://publib.boulder.ibm.com/infocenter/db2help/) から、このチュートリアルの XHTML 版を表示できます。

チュートリアルの中で、サンプル・データまたはサンプル・コードを使用する場合があります。個々のタスクの前提条件については、それぞれのチュートリアルを参照してください。

DB2 Universal Database チュートリアル:

以下に示すチュートリアルのタイトルをクリックすると、そのチュートリアルを表示できます。

ビジネス・インテリジェンス・チュートリアル: データウェアハウス・センターの紹介
: データウェアハウス・センターを使用して簡単なデータウェアハウジング・タスクを実行します。

ビジネス・インテリジェンス・チュートリアル: データウェアハウジングの上級者向けガイド
: データウェアハウス・センターを使用して高度なデータウェアハウジング・タスクを実行します。

インフォメーション・カタログ・センター・チュートリアル
: インフォメーション・カタログを作成および管理して、インフォメーション・カタログ・センターを使用してメタデータを配置し使用します。

Visual Explain チュートリアル
Visual Explain を使用して、パフォーマンスを向上させるために SQL ステートメントを分析し、最適化し、調整します。

DB2 トラブルシューティング情報

DB2® 製品を使用する際に役立つ、トラブルシューティングおよび問題判別に関する広範囲な情報を利用できます。

DB2 ドキュメンテーション
トラブルシューティング情報は、DB2 インフォメーション・センター、および DB2 ライブラリーに含まれる PDF 資料の中でご利用いただけます。DB2 インフォメーション・センターで、 (ブラウザー・ウィンドウの左側の) ナビゲーション・ツリーの「サポートおよびトラブルシューティング (Support and troubleshooting)」ブランチを参照すると、 DB2 トラブルシューティング・ドキュメンテーションの詳細なリストが見つかります。

DB2 Technical Support の Web サイト
現在問題が発生していて、考えられる原因とソリューションを検索したい場合は、 DB2 Technical Support の Web サイトを参照してください。Technical Support サイトには、最新の DB2 出版物、TechNotes、プログラム診断依頼書 (APAR)、フィックスパック、DB2 内部エラー・コードの最新リスト、その他のリソースが用意されています。この知識ベースを活用して、問題に対する有効なソリューションを探し出すことができます。

DB2 Technical Support の Web サイト (http://www.ibm.com/software/data/db2/udb/winos2unix/support) にアクセスしてください。

DB2 Problem Determination Tutorial Series
DB2 製品で作業中に直面するかもしれない問題を素早く識別し、解決する方法に関する情報を見つけるには、 DB2 Problem Determination Tutorial Series の Web サイトを参照してください。あるチュートリアルでは、使用可能な DB2 問題判別機能およびツールを紹介し、それらをいつ使用すべきかを判断する助けを与えます。別のチュートリアルは、『データベース・エンジン問題判別 (Database Engine Problem Determination)』、『パフォーマンス問題判別 (Performance Problem Determination)』、『アプリケーション問題判別 (Application Problem Determination)』などの関連トピックを扱っています。

DB2 Technical Support (http://www.ibm.com/software/data/support/pdm/db2tutorials.html) には、 DB2 問題判別チュートリアルがすべて揃っています。

関連概念:

- 556 ページの『DB2 インフォメーション・センター』
- 「*問題判別の手引き*」の『Introduction to Problem Determination - DB2 テクニカル・サポートのチュートリアル』

アクセス支援

アクセス支援機能は、身体に障害のある (身体動作が制限されている、視力が弱いなど) ユーザーがソフトウェア製品を十分活用できるように支援します。 DB2® バージョン 8 製品に備わっている主なアクセス支援機能は、以下のとおりです。

- すべての DB2 機能は、マウスの代わりにキーボードを使ってナビゲーションできます。詳細については、『キーボードによる入力およびナビゲーション』を参照してください。
- DB2 ユーザー・インターフェースのフォント・サイズおよび色をカスタマイズすることができます。詳細については、『アクセスしやすい表示』を参照してください。
- DB2 製品は、Java™ Accessibility API を使用するアクセス支援アプリケーションをサポートします。詳細については、 582 ページの『支援テクノロジーとの互換性』を参照してください。
- DB2 資料は、アクセスしやすい形式で提供されています。詳細については、 582 ページの『アクセスしやすい資料』を参照してください。

キーボードによる入力およびナビゲーション

キーボード入力

キーボードだけを使用して DB2 ツールを操作できます。マウスを使って実行できる操作は、キーまたはキーの組み合わせによっても実行できます。標準のオペレーティング・システム・キー・ストロークを使用して、標準のオペレーティング・システム操作を実行できます。

キーまたはキーの組み合わせによって操作を実行する方法について、詳しくは キーボード・ショートカットおよびアクセラレーター: Common GUI help を参照してください。

キーボード・ナビゲーション

キーまたはキーの組み合わせを使用して、 DB2 ツールのユーザー・インターフェースをナビゲートできます。

キーまたはキーの組み合わせによって DB2 ツールをナビゲートする方法の詳細については、キーボード・ショートカットおよびアクセラレーター: Common GUI help を参照してください。

キーボード・フォーカス

UNIX® オペレーティング・システムでは、アクティブ・ウィンドウの中で、キー・ストロークによって操作できる領域が強調表示されます。

アクセスしやすい表示

DB2 ツールには、視力の弱いユーザー、その他の視力障害をもつユーザーのためにアクセシビリティーを向上させる機能が備わっています。これらのアクセシビリティー拡張機能には、フォント・プロパティーのカスタマイズを可能にする機能も含まれています。

フォントの設定

「ツール設定」ノートブックを使用して、メニューおよびダイアログ・ウィンドウに使用されるテキストの色、サイズ、およびフォントを選択できます。

フォント設定に関する詳細情報は、メニューおよびテキストのフォントを変更する: Common GUI help を参照してください。

色に依存しない

本製品のすべての機能を使用するために、ユーザーは必ずしも色を識別する必要はありません。

支援テクノロジーとの互換性

DB2 ツールのインターフェースは、Java Accessibility API をサポートします。これによって、スクリーン・リーダーその他の支援テクノロジーを DB2 製品で利用できるようになります。

アクセスしやすい資料

DB2 形式は、ほとんどの Web ブラウザーで表示可能な XHTML 1.0 形式で提供されています。 XHTML により、ご使用のブラウザーに設定されている表示設定に従って資料を表示できます。さらに、スクリーン・リーダーや他の支援テクノロジーを使用することもできます。

シンタックス・ダイアグラムはドット 10 進形式で提供されます。この形式は、スクリーン・リーダーを使用してオンライン・ドキュメンテーションにアクセスする場合にのみ使用できます。

関連概念:

• 582 ページの『ドット 10 進シンタックス・ダイアグラム』

ドット 10 進シンタックス・ダイアグラム

スクリーン・リーダーを使用してインフォメーション・センターを利用するユーザーのために、シンタックス・ダイアグラムがドット 10 進形式で提供されます。

ドット 10 進形式では、各シンタックス・エレメントは別々の行に書き込まれます。複数のシンタックス・エレメントが常に同時に存在する (または常に同時に不在の) 場合、単一のコンパウンド・シンタックス・エレメントとみなせるので同一行に表示できます。

各行は、ドット 10 進数で開始します。たとえば、3 または 3.1 ないしは 3.1.1 です。こうした数を適切に聞き取るには、スクリーン・リーダーが句読点を読み取るように設定されていることを確認してください。同じドット 10 進数を持つすべてのシンタックス・エレメント (たとえば、 3.1 という数値を持つすべてのシンタックス・エレメント) は、相互に排他的な代替エレメントです。 3.1 USERID および 3.1 SYSTEMID という行を聞き取る場合、シンタックスには両方ではなく USERID または SYSTEMID のどちらかが含まれることが分かります。

ドット 10 進レベルは、ネストのレベルを表示します。たとえば、ドット 10 進数 3 のシンタックス・エレメントの後に、一連のドット 10 進数 3.1 のシンタックス・エレメントが続きます。 3.1 の番号が付されたシンタックス・エレメントすべては、番号 3 の付されたシンタックス・エレメントに従属します。

シンタックス・エレメントに関する情報を追加するため、ドット 10 進数の次に特定のワードおよびシンボルが使用されます。 時折、こうしたワードおよびシンボルはエレメントの最初に表示される場合もあります。 簡単に識別するため、ワードやシンボルがシンタックス・エレメントの一部である場合には、円記号 (¥) 文字が先頭に付きます。 * シンボルはドット 10 進数の次に使用でき、シンタックス・エレメントが反復することを示します。 たとえば、ドット 10 進数 3 のシンタックス・エレメント *FILE は、3 ¥* FILE という形式になります。 3* FILE という形式は、シンタックス・エレメント FILE が反復されることを示します。 3* ¥* FILE という形式は、シンタックス・エレメント * FILE が反復されることを示します。

シンタックス・エレメントのストリングを分離するのに使用されるコンマなどの文字は、シンタックス内の分離する項目の直前に表示されます。 こうした文字は、それぞれの項目と同一行に表示するか、同じドット 10 進数を持つ関連する項目のある別の行に表示できます。 またその行には、シンタックス・エレメントに関する情報を提供する別のシンボルを表示することも可能です。 たとえば、複数の LASTRUN および DELETE シンタックス・エレメントを使用している場合には、5.1*、5.1 LASTRUN、および 5.1 DELETE という行は、エレメントをコンマで区切る必要があります。区切り文字が指定されないと、各シンタックス・エレメントを区切るのにブランクが使用されると想定されます。

シンタックス・エレメントの前に % シンボルが付く場合、他の箇所で定義されている参照であることを示します。 % シンボルの後のストリングは、リテラルではなくシンタックス・フラグメントの名前です。 たとえば、2.1 %OP1 という行は別のシンタックス・フラグメント OP1 を参照すべきことを意味します。

以下のワードおよびシンボルが、ドット 10 進数の次に使用されます。

- ? は、オプションのシンタックス・エレメントであることを表します。? シンボルが後に続くドット 10 進数は、対応するドット 10 進数のシンタックス・エレメント、および任意の従属のシンタックス・エレメントがオプションであることを示します。ドット 10 進数の付いたシンタックス・エレメントが 1 つしかない場合、 ? シンボルはそのシンタックス・エレメントと同じ行に表示されます (たとえば、5? NOTIFY)。ドット 10 進数の付いたシンタックス・エレメントが複数ある場合、 ? シンボルだけで行に表示され、その後にオプションのシンタックス・エレメントが続きます。たとえば、「5 ?、5 NOTIFY、および 5 UPDATE」という行を聞き取る場合、シンタックス・エレメント NOTIFY および UPDATE がオプションである、つまりそのいずれかを選択でき、どちらも選択しないこともできることが分かります。 ? シンボルは、線路型ダイアグラムのバイパス線に相当します。
- ! は、デフォルトのシンタックス・エレメントであることを表します。! シンボルおよびシンタックス・エレメントが後に続くドット 10 進数は、そのシンタックス・エレメントが、同じドット 10 進数を共用するシンタックス・エレメントすべてのデフォルト・オプションであることを示します。同じドット 10 進数を共用するシンタックス・エレメントのうち 1 つだけに、 ! シンボルを指定できます。たとえば、「2? FILE、2.1! (KEEP)、および 2.1 (DELETE)」という行を聞き

取る場合、 FILE キーワードのデフォルト・オプションは (KEEP) になります。この例では、FILE キーワードを含めてもオプションを指定しない場合には、デフォルト・オプション KEEP が適用されます。デフォルト・オプションは、次に高位のドット 10 進数にも適用されます。 この例の場合、FILE キーワードが省略されると、デフォルトの FILE(KEEP) が使用されます。しかし、「2? FILE、2.1、2.1.1! (KEEP)、および 2.1.1 (DELETE)」という行を聞き取る場合、デフォルト・オプション KEEP は次に高位のドット 10 進数 2.1 (関連キーワードを持っていない) にのみ適用され、 2? FILE には適用されません。キーワード FILE が省略されると、どれも使用されません。

- * は、0 回以上反復できるシンタックス・エレメントを示します。 * シンボルが後に続くドット 10 進数は、このシンタックス・エレメントが 0 回以上使用できること、つまりオプションであり、なおかつ反復できることを表します。たとえば、5.1* データ域という行を聞き取る場合、 1 つまたは複数のデータ域を含めるか、またはデータ域を全く含めないことが可能です。「3*、3 HOST、および 3 STATE」という行を聞き取る場合、 HOST、STATE をどちらか一方または両方同時に含めるか、どちらも含めないことができます。

  注:

  1. ドット 10 進数の後にアスタリスク (*) が付き、ドット 10 進数の付いた項目が 1 つしかない場合には、同じ項目を複数回反復できます。
  2. ドット 10 進数の後にアスタリスクが付き、ドット 10 進数の付いた項目が複数ある場合、リストから複数の項目を使用できますが、各項目を複数回使用することはできません。 前述の例では、HOST STATE と書くことはできますが、HOST HOST とは書けません。
  3. * シンボルは、線路型シンタックス・ダイアグラムのループバック線に相当します。

- + は、1 回以上含める必要のあるシンタックス・エレメントであることを示します。 + シンボルが後に続くドット 10 進数は、このシンタックス・エレメントを 1 回以上含める必要があること、つまり少なくとも 1 回は含める必要があり、反復できることを表します。たとえば、「6.1+ データ域」という行を聞き取る場合、データ域を少なくとも 1 回は含めなければなりません。「2+、2 HOST、および 2 STATE」という行を聞き取る場合には、 HOST、STATE、またはその両方を含める必要があります。 * シンボルと同様に、+ シンボルは、ドット 10 進数の付いた項目が 1 つしかない場合に限り、その特定の項目のみを反復できます。 * シンボルと同様、 + シンボルは線路型シンタックス・ダイアグラムのループバック線に相当します。

関連資料:

- 「*SQL リファレンス 第 2 巻*」の『構文図の見方』

DB2 Universal Database 製品の共通基準認証

DB2 Universal Database は、 Common Criteria の評価検定レベル 4 (EAL4) で認証の評価を受けています。 Common Criteria の詳細については、以下の Common Criteria の Web サイトを参照してください。 http://niap.nist.gov/cc-scheme/

付録 C. 特記事項

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。 IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム(本プログラムを含む)との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Canada Limited

Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、 IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。 IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

商標

以下は、IBM Corporation の商標です。

ACF/VTAM
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
BookManager
C Set++
C/370
CICS
Database 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Information Integrator
DB2 Query Patroller
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eServer
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
iSeries
LAN Distance
MVS
MVS/ESA
MVS/XA
Net.Data
NetView
OS/390
OS/400
PowerPC
pSeries
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/400
SQL/DS
System/370
System/390
SystemView
Tivoli
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WebSphere
WIN-OS/2
z/OS
zSeries

以下は、それぞれ各社の商標または登録商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

Pentium は、Intel Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

［ア行］

[カ行]

[サ行]

［タ行］

［ナ行］

［ハ行］

[マ行]

[ヤ行]

［ラ行］

［ワ行］

［数字］

A

B

C

D

E

F

G

H

I

L

M

N

O

P

Q

R

S

T

U

W

X

IBM と連絡をとる

技術上の問題がある場合は、お客様サポートにご連絡ください。

製品情報

DB2 Universal Database 製品に関する情報は、
http://www.ibm.com/software/data/db2/udb から入手できます。

このサイトには、技術ライブラリー、資料の注文方法、製品のダウンロード、ニュースグループ、フィックスパック、ニュース、および Web リソースへのリンクに関する最新情報が掲載されています。

米国以外の国で IBM に連絡する方法については、 IBM Worldwide ページ (www.ibm.com/planetwide) にアクセスしてください。

Printed in Japan

SC88-9157-01

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12