SYBASE®
An SAP Company

パフォーマンス&チューニング・シリーズ：
sp_sysmon による Adaptive Server の監視

Adaptive Server® Enterprise

15.7

ドキュメント ID：DC01075-01-1570-01
改訂：2011 年9 月

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

目次

第 1 章

sp_sysmon の概要

この章では、sp_sysmon の使用と呼び出し方法について説明します。

sp_sysmon の使用

sp_sysmon を使用すると、システムのアクティビティに関する詳細レポートを作成できます。また、収集情報の種類を指定する多くの方法、レポート・データの収集間隔の設定、およびレポートの作成に関するその他のオプションが用意されています。

sp_sysmon のレポートはいくつかのセクションで構成されています。sp_sysmon は完全なレポートを作成することも、個別セクションの 1 つだけを作成することもできます。時間間隔を指定し、その期間の最初と最後にレポートを作成したり、ストアド・プロシージャを実行したりすることもできます。

sp_sysmon はサンプリング期間にモニタしたデータのみをレポートの対象とします。チューニング時は代表的なデータに基づいて判断を下すことになります。たとえば、スピンロックのチューニングを行う場合は、ピーク使用率のレポートのデータに基づいて判断します。しかし、エンジンの数を減らす場合は、典型的かつピークの運用負荷を示すサンプルの数に基づいて判断します。

レポートのデータは、Adaptive Server® が管理する一連のモニタ・カウンタから収集したものです。これらのカウンタは他のアプリケーションでも使用します。デフォルトでは、sp_sysmon レポートが呼び出し時にこれらのカウンタをクリアします。カウンタがクリアされると、カウンタを使用している他のアプリケーションに影響を与え、レポートするデータが無効になることがあります。

警告！ sp_sysmon にカウンタをクリアさせるかどうかは、`noclear` オプションによって決まります。`noclear` オプションをオンにすると、sp_sysmon はカウンタをクリアしなくなり、sp_sysmon が他の sp_sysmon セッションと同時に実行できるようになります。デフォルトでは、noclear は、sp_sysmon をサンプル・インターバルを使用して実行した場合に有効となり、sp_sysmon を *begin_* パラメータと *end_sample* パラメータを使用して実行した場合に無効となります。詳細については、「noclear オプションの使用」(7 ページ) を参照してください。

sp_sysmon をシングル CPU のサーバ上で実行すると 5 ～ 7% のオーバヘッドが発生しますが、マルチプロセッサ・サーバ上で実行するとそれ以上のオーバヘッドが発生します (パーセンテージはサイトによって異なります)。オーバヘッドの量は CPU の数に伴って増加します。sp_sysmon の `noclear` は同じ内部カウンタを使用しています。オプションを使用せずに実行すると、sp_sysmon はこれらのカウンタを 0 にリセットします。

sp_sysmon とモニタリング・テーブルで共有しているカウンタがあります。最初の sp_sysmon が実行中に、2 番目を起動するとすべてのカウンタがクリアされるので、最初の sp_sysmon セッションが作成したレポートは不正確になります。

sp_sysmon を他のアプリケーションと一緒に使用した場合の詳細については、「noclear オプションの使用」(7 ページ) を参照してください。

---

sp_sysmon のパフォーマンスのチューニングに関するヒントは、sp_sysmon に指定したサンプル間隔に基づいています。推奨事項はシステム要件を基に十分に検討してから運用システムに取り入れてください。いかなる推奨事項でも、取り入れる前に自分のデータでテスト領域を設定し、どんな変更についてもテストすることを強くおすすめします。

sp_sysmon はシステムのスナップショット・ビューを提供するので、運用負荷が変化した場合は推奨事項の再検討が必要となることがあります。

注意 デフォルト・サイズの tempdb を使用した Adaptive Server 上では、sp_sysmon を実行できません。Adaptive Server が一時的なデータベース上でログの領域不足にならないように、少なくとも 2MB の単位で tempdb のサイズを増やしてください。

---

sp_sysmon を実行する状況

Adaptive Server 設定パラメータのチューニング前後で sp_sysmon を実行すれば、データを収集して比較することができます。このデータは、パフォーマンスをチューニングするときの基礎になります。また、このデータを使用して設定変更の結果を観察できます。

調査を必要とする動作をシステムが示した場合に、sp_sysmon を使用します。たとえば、通常の負荷がかかる状況でのシステムの動作を知りたい場合は、一般的な状況で通常の負荷がかかるときに sp_sysmon を実行します。たとえば、営業時間帯の OLTP ユーザがほとんど帰宅してからバッチ・ジョブが開始されるまでの合間となる午後 7:00 からの 10 分間に sp_sysmon を実行することに意味があるかどうか考えてみてください。そうではなく、通常の OLTP の負荷がかかりバッチ・ジョブが動作中に sp_sysmon を実行するべきでしょう。

多くのテストでは、まず最初にアプリケーションを起動し、キャッシュが安定状態になったときに、sp_sysmon を起動するのが最善の方法です。処理能力を計測しようとする場合は、テストの時間帯にサーバをビジー状態に保てるだけの作業量を必ず与えてください。

サンプリング時間中にサーバのアイドル時間があると、多くの情報、特に 1 秒当たりのデータを計測した統計値がきわめて低くなる可能性があります。

一般に、次のような場合に sp_sysmon を使用すると、役に立つ情報が生成されます。

- キャッシュ設定やプール設定の変更の前後。
- 特定の sp_configure の変更の前後 (たとえば、メモリ・サイズ、キャッシュ、ディスク I/O に関連したオプションの変更)。
- 混合アプリケーションへの新しいクエリの追加前と追加後。
- Adaptive Server エンジン数の増加や減少の前後。
- 新しいディスク・デバイスの追加時と、追加したディスク・デバイスへのオブジェクトの割り当て時。
- ピークの時間帯に競合またはボトルネックを調べる場合。
- 予想される最大のアプリケーション負荷を与えた場合の Adaptive Server の設定を調べるストレス・テストの最中。
- パフォーマンスが低下している場合や異常な動作がある場合。

sp_sysmon がクエリの使用中またはアプリケーション開発中でも役立つことに気が付くことでしょう。次に、例を示します。

- deferred_varcol として報告される特定の更新が、直接更新と遅延更新のどちらで実行されるかを調べるために、インデックスと更新を使用して作業する場合。
- 特定のクエリや混合クエリのキャッシングの動作をチェックする場合。
- 並列インデックス作成のためのパラメータやキャッシュ設定をチューニングする場合。

sp_sysmon の呼び出し

sp_sysmon では、enable monitoring を true (有効) に設定し、ユーザが mon_role を持つようにする必要があります。

```
sp_configure 'enable monitoring', 1
grant role mon_role to sa
```

次のように sp_sysmon を使用します。

- 固定した時間間隔を使用し、実行時間を分単位で指定してサンプリングを提供する。
- begin_sample パラメータを使用してサンプリングを開始し、end_sample パラメータを使用してサンプリングを終了する。

必要な情報が提供されるように、出力を次のようにカスタマイズします。次の処理を実行できます。

- レポート全体を出力する。
- “Cache Management”や“Lock Management”など、レポートの一部のセクションだけを出力する。
- 名前付きのアプリケーション (isql、bcp) と、名前付きのアプリケーションおよびユーザ名の組み合わせを指定し、アプリケーション・レベルでの詳細なレポートも一緒に出力する。デフォルトではこのセクションは省略される。

一定の時間間隔

sp_sysmon を呼び出すには、isql を使用して以下を実行します。

```
sp_sysmon interval [, section  [, applmon]]
```

interval は “hh:mm:ss”の形式でなければなりません。たとえば、sp_sysmon を 10 分間実行するには、次のように記述します。

```
sp_sysmon "00:10:00"
```

applmon パラメータの詳細については、「アプリケーション詳細パラメータの指定」(6 ページ) を参照してください。

begin_sample と *end_sample* を使用する実行

begin_sample パラメータと end_sample パラメータを使用すると、sp_sysmon を呼び出して、サンプリングの開始、クエリの発行、サンプリングの終了、任意の時点での結果の出力を実行できます。次に例を示します。

```
sp_sysmon begin_sample
execute proc1
execute proc2
select sum(total_sales) from titles
sp_sysmon end_sample
```

注意　多数の CPU が搭載されたアクティビティの高いシステムでは、サンプリング時間が長すぎると、カウンタがオーバフローすることがあります。

sp_sysmon の出力に負の数が含まれていた場合は、サンプル時間を短くしてください。

出力されるレポート・セクションの指定

レポートの 1 つのセクションを印刷するには、表 1-1 に一覧表示されている *section* パラメータの値のいずれかを使用してください。

表 1-1: sp_sysmon レポート・セクション

レポート・セクション	パラメータ
Application Management	appmgmt
Cache Wizard	cache wizard
Data Cache Management	dcache
Disk I/O Management	diskio
ESP Management	esp
Housekeeper Task Activity	housekeeper
Index Management	indexmgmt
Kernel Utilization	kernel
Lock Management	locks
Memory Management	memory
Metadata Cache Management	mdcache*
Monitor Access to Executing SQL	monaccess
Network I/O Management	netio
Parallel Query Management	parallel
Procedure Cache Management	pcache
Recovery Management	recovery
RepAgent	repagent
Task Management	taskmgmt
Transaction Management	xactmgmt
Transaction Profile	xactsum
Worker Process Management	wpm

また、sp_monitorconfig を使用すると、sp_sysmon mdcache を介して使用可能な情報のほとんどを取得できます。『リファレンス・マニュアル：プロシージャ』を参照してください。

アプリケーション詳細パラメータの指定

applmon パラメータを sp_sysmon に指定すると、レポートにはアプリケーションまたはアプリケーションとログイン名からの詳細な情報が含まれます。このパラメータは、レポート全体を出力する場合か、section パラメータに appmgmt を指定した場合にのみ有効です。アプリケーションの詳細パラメータを指定してレポートの別のセクションを要求した場合は、アプリケーションの詳細パラメータが無視されます。

3 番目のパラメータには、次のいずれかを指定してください。

パラメータ	レポートされる情報
appl_only	アプリケーション名ごとの CPU、I/O、優先度の変更、リソースの制限違反。
appl_and_login	アプリケーション名とログイン名ごとの CPU、I/O、優先度の変更、リソースの制限違反。すべてのセクションで使用可能。
no_appl	レポートのアプリケーションとログインのセクションを省略する。これがデフォルト値です。

次の例では、sp_sysmon が 5 分間実行され、アプリケーションとログインについての詳細な情報を含む「Application Management (アプリケーション・管理)」セクションが出力されます。

```
sp_sysmon "00:05:00", appmgmt, appl_and_login
```

出力例については、「アプリケーションごと、またはアプリケーションとログインごとのアプリケーション情報」(57 ページ) を参照してください。

noclear オプションの使用

デフォルトでは、sp_sysmon はレポートのソース・データとして使用されるモニタ・カウンタをクリアしません。他のアプリケーションや sp_sysmon レポートのインスタンスの実行中にカウンタをクリアすると、レポートされるデータは不正になります。

モニタ・カウンタをクリアしないようにするには、sp_sysmon でオプションの noclear パラメータを使用します。次に例を示します。

```
sp_sysmon "00:15:00", noclear
```

この例では、15 分のサンプル・インターバルの間、モニタ・カウンタをクリアせずに sp_sysmon レポートを実行します。noclear の値は他のパラメータ値と組み合わせて使用できます。

```
@section
@applmon
@filter
@dumpcounters
```

たとえば、noclear オプションを使用しつつカーネル・リソース使用のレポートを作成するには、次のようにします。

```
sp_sysmon "00:15:00", kernel, noclear
```

sp_sysmon を noclear オプションを付けて実行すると、サンプリング期間の最初に I/O アクティビティ量がわずかに増加することがあります。これは、レポートがカウンタの初期値を保存するテンポラリ・テーブルを作成する必要があるからです。このアクティビティが sp_sysmon のレポート・データに与える影響は、静止状態にあるほとんどのシステムを除けば、無視しても問題ありません。

noclear を使用すると、sp_sysmon レポートのセッションを複数個同時に実行することもできます。他に実行中のシステム監視アプリケーションで、モニタ・カウンタを使用するものがないか検討してください。もしあれば、noclear オプションを使用してください。

'begin_sample' を実行したセッションとは異なるセッションから sp_sysmon 'end_sample' を実行した場合、sp_sysmon はモニタ・カウンタの値を減らしません。また、Adaptive Server はクライアントにエラー・メッセージ番号 19374 を返します。つまり、アプリケーションがモニタ・カウンタを有効にし、それを無効にせずにログアウトした場合、使用カウントはアプリケーションのログアウト前と同じユーザ数を示します。

ファイルへの出力のリダイレクト

完全な sp_sysmon レポートは、数百行もの出力になります。出力をファイルに保存するには、isql の入出力リダイレクト・フラグを使用してください。たとえば、このコマンドを含んだ *sysmon_in* というファイルを作成し sp_sysmon を 10 秒間実行するには、次のようにします。

```
sp_sysmon '00:00:10'
go
```

出力を *sysmon_out* というファイルに引き渡すには、isql -o パラメータを使用してください。

```
$SYBASE/$SYBASE_OCS/bin/isql -Usa -P -Sbig_server -isysmon_in -osysmon_out
```

isql の詳細については、『ユーティリティ・ガイド』を参照してください。

第 2 章

sp_sysmon を利用したパフォーマンスのモニタリング

この章では、sp_sysmon の出力内容を解釈したり、示している事態を推測したりするためのヒントも含めた sp_sysmon の出力について説明します。

使用している Adaptive Server 環境やそのアプリケーションについて理解している場合は、sp_sysmon の出力は非常に役立ちます。

レポートの使用方法

sp_sysmon を使用すると、チューニング前後の Adaptive Server システムの動作について情報を取得できます。レポートの全体についてよく学習し、行った変更が与えるすべての影響について理解してください。パフォーマンスのボトルネックを 1 つ取り除くと、別のボトルネックが明らかになることがあります。同様に、チューニング作業によって、ある領域でのパフォーマンスが向上したとしても、実際には別の領域でパフォーマンスが低下していることもあります。

sp_sysmon からの出力は、チューニング作業が必要な領域を示すだけでなく、さらにチューニングを行ってもこれ以上パフォーマンスを改善できなくなる時点を調べるためにも役立ちます。Adaptive Server のチューニングをやめる時期、また別の場所で問題が発生する時期を知ることは、チューニングする対象を知ることと同じように重要です。

sp_sysmon を 1 回実行すると、特定の時間間隔でのリソースの使用状況が示されます。使用する時間間隔がチューニングする Adaptive Server の作業負荷と状況を明確に表していることを確認してください。

ほかにも次のような情報が、sp_sysmon からの出力の解釈に役立ちます。

- sp_configure または設定ファイルから得られる、使用中の設定パラメータについての情報。
- sp_cacheconfig と sp_helpcache から得られる、キャッシュの設定やバインドについての情報。
- ディスク・デバイス、セグメント、およびディスク・デバイスに格納されるオブジェクトについての情報。

ヘッダ情報

sp_sysmon ヘッダには、Adaptive Server のバージョン、sp_sysmon の実行日、開始時刻と終了時刻、サンプルの長さ、サーバに名前を付けたかどうかなど、現在の sp_sysmon の実行に関する一般情報が含まれています。ヘッダは、sp_sysmon を実行したモード (noclear オプションを含めたかどうか) とカウンタが最後にクリアされた日付 (カウンタについては、「sp_sysmon の使用」(1 ページ) を参照) を示します。

ヘッダは、設定パラメータを設定してモニタリング・テーブル情報を収集する必要があるかどうか、および mon_role の役割をシステム管理者に付与する必要があるかどうかを示します。

出力の読み込み

sp_sysmon では、パフォーマンス統計値が見やすい表形式で表示されます。たとえば、7 つの Adaptive Server エンジンが実行されている SMP 環境では、通常、次のように出力されます。

```
Engine Utilization (Tick %)        CPU Busy   I/O Busy         Idle
------------------------          --------   --------     --------
Engine 0                            68.7 %      2.5 %       28.8 %
Engine 1                            61.9 %      3.3 %       34.8 %
Engine 2                            67.0 %      2.4 %       30.6 %
Engine 3                            69.0 %      3.8 %       27.2 %
Engine 4                            60.2 %      2.7 %       37.2 %
Engine 5                            55.7 %      3.2 %       41.1 %
Engine 6                            53.8 %      3.2 %       43.0 %
------------------------          --------   --------     --------
Summary          Total             436.3 %     21.1 %      242.6 %
Average                             62.3 %      3.0 %       34.7 %
```

ロー

大部分のローには、ロックの取得やストアド・プロシージャの実行など、具体的なアクティビティやイベントの種類が示されます。CPU に関連するデータの場合、ローには SMP 環境での各 Adaptive Server エンジンについてのパフォーマンス情報が示されます。関連する複数のローが 1 つのグループとして表示される場合は、通常、最後のローは総数と平均値の概要です。

sp_sysmon レポートには、インデントされたローがあります。これは、あるカテゴリが別のカテゴリのサブカテゴリであることを示します。次の例では、“Found in Wash”は“Cache Hits”のサブカテゴリであり、“Cache Hits”は“Cache Searches”のサブカテゴリです。

```
Cache Searches                   per sec       per xact       count  % of total
                              ----------      ---------    -------   --------
   Cache Hits                    32731.6       153429.6     9819492     100.0 %
     Found in Wash                 288.4         1351.7       86508       0.9 %
Cache Misses                        10.9           50.9        3257       0.0 %
---------------------       ------------      ---------   ---------
Total Cache Searches             32742.5       153480.5      9822749
```

“count”の値が 0 のローは、多くの場合出力されません。

出力のカラム

特に記述がないかぎり、カラムは次のパフォーマンス統計値を示します。

- “per sec” － サンプル間隔での、1 秒当たりの平均値。
- “per xact” － サンプル間隔での、コミットされたトランザクション当たりの平均値。

- “count” – サンプル間隔での総数。
- “% of total” – コンテキストによって異なる。これは出現した箇所ごとに説明する。

データの解釈

Adaptive Server のチューニングが成功した場合は、基本的な測定値が、スループットでは増加し、アプリケーションの応答時間では減少して出力されます。しかし、Adaptive Server のチューニングでこれらの 2 つの値を出力させることはできません。

ほとんどの場合、Adaptive Server のアクティビティの概要を把握し、クエリとアプリケーションについてチューニングと分析を注意深く行い、オブジェクトごとのロックとアクセスをモニタリングしながら、繰り返しチューニングしてください。

1 秒当たりのデータとトランザクション当たりのデータ

測定する環境とカテゴリについて、1 秒当たりのデータとトランザクション当たりのデータを検討してください。トランザクション当たりのデータは、一般に負荷が明確なベンチマークやテスト環境の方が、大きな意味を持ちます。

テスト・データを比較する場合は、1 秒当たりのデータよりもトランザクション当たりのデータの方がより意味を持ちます。これは、ベンチマーク・テスト環境ではトランザクション数が明確で、簡単に比較できるためです。トランザクション当たりのデータは、パーセンテージで表される結果の妥当性を検証する場合にも役立ちます。

トランザクション内で実行されない大量のクエリに関わる作業負荷を分析するときに、1 秒当たりの平均を調べるとさらに役立つ場合があります (たとえば、select 文)。1 秒当たりの分析を使用すると、これらのクエリに影響するチューニングを比較できます。

総数に対するパーセンテージとカウント・データ

“% of total”データの意味は、イベントのコンテキストやカテゴリごとの総数によって異なります。パーセンテージを解釈する場合、これらの値は一般的な傾向を理解するために役立つものではありますが、単独で使用すると判断を誤る可能性があることに注意してください。たとえば、200 個のイベントの 50% は、2 つのイベントの 50% よりも意味を持ちます。

“count”データは、サンプル間隔中に発生したイベントの総数です。カウント・データを使って、パーセンテージで表される結果の妥当性を検証できます。

エンジン当たりのデータ

ほとんどの場合、あるカテゴリのエンジン当たりのデータは、すべてのエンジンでかなり均等なアクティビティを示しますが、次の 2 つの例外があります。

- プロセスの数が CPU より少ない場合は、アクティビティが示されないエンジンがある。
- ほとんどのプロセスで、単純な挿入や短い選択など、完全に同一のアクティビティが実行されていて、あるプロセスで大規模なバルク・コピーなどの I/O 集約操作が行われている場合は、バランスの悪いネットワークやディスク I/O が示される。

総数データまたは計算データ

計算ローでは総数と平均値がレポートされ、Adaptive Server エンジンのアクティビティの概要が示されます。

データが偏っていると、実際の結果を間違って解釈する可能性があるので、平均値を解釈する場合は注意してください。たとえば、ある時間の Adaptive Server エンジンの稼働率が 98% で、別のエンジンの稼働率が 2% の場合、平均値は 49% と示されますが、これを誤って解釈する可能性があります。

キャッシュ・ウィザード

“Cache Wizard”セクションは、最適のパフォーマンスを得るためにデータ・キャッシュを監視および設定するのに役立ちます。

sp_sysmon cache wizard を使用する場合は、object lockwait timing 設定パラメータを有効にする必要があります。

キャッシュ・ウィザードによって次のことを識別できます。

- ホット・オブジェクト (アクセス頻度の高いオブジェクト)。名前付きキャッシュまたはデフォルト・データ・キャッシュの論理読み込み回数によって、出力がランク付けされる。
- キャッシュ上のスピンロック競合。
- キャッシュとバッファ・プールの使用率。
- キャッシュ、バッファ・プール、オブジェクトの各レベルのヒット率。
- 大容量 I/O の効果。
- 非同期プリフェッチ (APF) の効果。
- さまざまなオブジェクトによるキャッシュ占有。

sp_sysmon の出力に“Cache Wizard”セクションが表示されるのは、cache wizard を sp_sysmon 構文で指定した場合のみです。キャッシュ・ウィザードでは、topN と filter の 2 つのパラメータを指定できます。cache wizard が構文に含まれている場合、sp_sysmon には他のセクション (“Kernel Utilization”、“Worker Process Management” など) は含まれません。

sp_syntax では、レポートの最後に推奨事項セクションが出力されます。

キャッシュ・ウィザードの構文

キャッシュ・ウィザードから出力を取得する構文は次のとおりです。

```
sp_sysmon begin_sample
sp_sysmon { end_sample | interval }
[, 'cache wizard' [, top_N [, filter] ] ]
```

パラメータ

- top_N – varchar データ型であり、指定された間隔における論理読み込み回数 (LR/sec カラムの表示) のランク付け基準に基づいて、オブジェクト・セクションに出力されるオブジェクトのリストを限定する。

  ランク付けの順序は、正の整数を指定すると昇順、負の整数を指定すると降順になります。0 (ゼロ) の値を指定し、間隔の終了時にキャッシュを占有していた全オブジェクトのリストを取得します。デフォルト値は 10 です。

- filter – varchar データ型であり、レポートに含めるキャッシュのパターンを指定する。

  たとえば、filter を default data cache と指定した場合は、デフォルト・データ・キャッシュの情報だけがレポートされます。emp% の filter の値を指定すると、すべてのキャッシュで名前がこのパターンに一致する情報が出力されます。

  すべてのキャッシュに対する出力を表示するには、filter を空白のままにしておきます。最初に default data cache のキャッシュが表示された後、それ以外のキャッシュの情報がアルファベット順に表示されます。

詳細については、「出力例」(15 ページ) を参照してください。

例

次の例では、キャッシュ・ウィザードがデフォルト・データ・キャッシュに対して 5 分間隔で実行され、キャッシュ内の上位 15 のオブジェクトを示しています。

```
sp_sysmon '00:05:00','cache wizard','15','default data cache'
```

キャッシュ・ウィザードの実行準備

sp_sysmon は、キャッシュ・ウィザードに必要な情報をモニタリング・テーブルとモニタ・カウンタから取得します。

詳細については、『パフォーマンス&チューニング・シリーズ : モニタリング・テーブル』を参照してください。

キャッシュ・ウィザードを使用するには、次の設定パラメータを有効にする必要があります。有効になってない場合、これらの設定パラメータは sp_sysmon によって自動的に間隔の開始時に有効に設定され、終了時に無効に設定されます (これらのオプションはすべて動的です)。

- enable monitoring － 1 に設定する。
- per object statistics active － 1 に設定する。
- object lockwait timing － 1 に設定する。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

出力例

キャッシュ・ウィザード・セクションの出力には、キャッシュごとに 3 つの主要なセクションがあります。これに推奨事項のセクションが続き、レポートの最後に凡例セクションが出力されます。

- キャッシュ・セクション。特定のキャッシュについての概要統計が表示される。

```
default data cache
------------------
Run Size          :   100.00Mb  Usage%               :  2.86
LR/sec            :    41.10    PR/sec:             : 22.57  Hit%:45.9
Cache Partitions  :     4       Spinlock Contention% :  0.00
```

Usage% － ページがキャッシュに取り込まれるたびに、Adaptive Server によってそのページが参照 (使用) されたかどうかが追跡されます。ページがキャッシュから削除されると、このカウントは減ります。Usage% は、キャッシュの現在の使用状況をキャッシュ・サイズに対するパーセンテージで示します。

LR/sec － (1 秒あたりの論理読み込み) 論理読み込みとは、キャッシュからの読み込み (ヒット) またはディスクからの読み込み (ミス) における、すべての読み込みのことです。LR/sec は、サンプル間隔の長さで分割された間隔内におけるキャッシュでの論理読み込み回数を示します。

PR/sec － (1 秒あたりの物理読み込み) 物理読み込みとは、ディスクからの読み込み (ミス) のことです。PR/sec は、サンプル間隔で分割された間隔内におけるキャッシュでの物理読み込み回数を示します。

Hit% － LR/sec に対する (LR/sec - PR/sec) の割合などの、キャッシュ読み込みの総数に対するヒット率です。

- バッファ・プール・セクション ― 「キャッシュ・セクション」の統計情報がキャッシュのバッファ・プール別に表示される。

```
Buffer Pool Information
-------------------------------------------------------------------------
IO Size Wash Size Run Size APF%   LR/sec   PR/sec   Hit%   APF-Eff% Usage%
------- ---------- -------- ------ -------- -------- ------ -------- ------
4 Kb     3276 Kb    16.00 Mb  10.00     0.47     0.13  71.43    n/a     0.20
2 Kb    17200 Kb    84.00 Mb  10.00    40.63    22.43  44.79    n/a     3.37
```

APF-Eff% ― 非同期プリフェッチが使用するページ数と非同期プリフェッチが取り込む合計ページ数。

Usage% は、バッファ・プールに取り込まれたページが参照されたかどうかを追跡し、バッファ・プールの実行サイズに対する、バッファ・プールで参照されたページの割合を示します。

- オブジェクト・セクション ― 間隔の終了時にキャッシュを占有しているオブジェクトについての統計をレポートする。topN パラメータを使用して、このセクションのサイズを制限する。オブジェクトは PR/sec の値を基準にして昇順に表示される。

```
Object Statistics
-------------------------------------------------------------------------
Object                       LR/sec  PR/sec   Hit%  Obj_Cached% Cache_Occp%
------------------------------ ------- ------- ------ ----------- -----
empdb.dbo.t1                   0.57    0.30  47.06      56.25       0.02
empdb.dbo.t2                   0.30    0.30   0.00      56.25       0.02
empdb.dbo.t3                   0.30    0.30   0.00      56.25       0.02

Object                              Obj Size  Size in Cache
------------------------------ -----------   ------------
empdb.dbo.t1                           32 Kb          18 Kb
empdb.dbo.t2                           32 Kb          18 Kb
empdb.dbo.t3                           32 Kb          18 Kb
```

- 推奨事項セクション ― サンプル間隔で収集されたデータをもとに、適用可能なさまざまな推奨事項が示される。

```
The various recommendations are as follows:
Usage% for 'default data cache' is low (< 5%)
Usage% for  4k buffer pool in cache:default data cache is low (< 5%)
Consider using Named Caches or creating more cache partitions for
'default data cache' or both
Consider increasing the 'wash size' of the 2k pool for 'default data cache'
Consider adding a large I/O pool for 'default data cache'
```

- 凡例セクション ― 出力で使用された各種用語の説明が表示される。出力で使用された用語の一部がここでより詳しく説明される。

キャッシュ・ウィザードの出力例

```
sp_sysmon '00:00:30', 'cache wizard'

================================================================================
Cache Wizard
================================================================================

------------------
default data cache
------------------
Run Size        :   100.00 Mb   Usage%              :      2.86
LR/sec          :    41.10      PR/sec             :     22.57   Hit%:  45.09
Cache Partitions:        4      Spinlock Contention%:      0.00

Buffer Pool Information
--------------------------------------------------------------------------------
IO Size Wash Size  Run Size    APF%   LR/sec   PR/sec   Hit%   APF-Eff% Usage%
------- ---------- ----------- ------ -------- -------- ------ -------- ------
4 Kb     3276 Kb    16.00 Mb  10.00     0.47     0.13  71.43    n/a      0.20
2 Kb    17200 Kb    84.00 Mb  10.00    40.63    22.43  44.79    n/a      3.37

(1 row affected)

Object Statistics

--------------------------------------------------------------------------------
Object                          LR/sec  PR/sec  Hit%   Obj_Cached% Cache_Occp%
------------------------------- ------- ------- ------ ----------- -----------
tempdb.dbo.t1                     0.57    0.30  47.06       56.25        0.02
tempdb.dbo.t2                     0.30    0.30   0.00       56.25        0.02
tempdb.dbo.t3                     0.30    0.30   0.00       56.25        0.02
tempdb.dbo.t4                     0.30    0.30   0.00       56.25        0.02
tempdb.dbo.t5                     0.30    0.30   0.00       56.25        0.02
tempdb.dbo.t6                     0.30    0.30   0.00       56.25        0.02
tempdb.dbo.t8                     0.30    0.30   0.00       56.25        0.02
tempdb.dbo.t7                     0.57    0.20  64.71       62.50        0.02
tempdb.dbo.tempcachedobjstats     3.63    0.00 100.00       50.00        0.01
tempdb.dbo.tempobjstats           0.47    0.00 100.00       25.00        0.00

Object                          Obj Size    Size in Cache
------------------------------- ----------- -------------
tempdb.dbo.t1                         32 Kb        18 Kb
tempdb.dbo.t2                         32 Kb        18 Kb
tempdb.dbo.t3                         32 Kb        18 Kb
tempdb.dbo.t4                         32 Kb        18 Kb
tempdb.dbo.t5                         32 Kb        18 Kb
tempdb.dbo.t6                         32 Kb        18 Kb
tempdb.dbo.t8                         32 Kb        18 Kb
tempdb.dbo.t7                         32 Kb        20 Kb
tempdb.dbo.tempcachedobjstats         16 Kb         8 Kb
```

```
tempdb.dbo.tempobjstats                16 Kb         4 Kb

---------
company_cache
---------
Run Size         :     1.00 Mb   Usage%               :      0.39
LR/sec           :     0.07      PR/sec               :      0.07   Hit%:   0.00
Cache Partitions:        1      Spinlock Contention%:      0.00

Buffer Pool Information
------------------------------------------------------------------------------
IO Size Wash Size  Run Size    APF%   LR/sec   PR/sec   Hit%   APF-Eff% Usage%
------- ---------- ----------- ------ -------- -------- ------ -------- ------
2 Kb      204 Kb     1.00 Mb  10.00     0.07     0.07   0.00    n/a     0.39


Object Statistics
------------------------------------------------------------------------------

Object               LR/sec  PR/sec  Hit%   Obj_Cached% Cache_Occp%
-------------------- ------- ------- ------ ----------- -----------
empdb.dbo.history      0.07    0.07   0.00       25.00        0.39

Object               Obj Size    Size in Cache
-------------------- ----------- -------------
empdb.dbo.history         16 Kb         4 Kb

-------------
companydb_cache
-------------
Run Size         :     5.00 Mb   Usage%               :    100.00
LR/sec           :   380.97      PR/sec               :     56.67   Hit%:  85.13
Cache Partitions:        1      Spinlock Contention%:      0.00

Buffer Pool Information
------------------------------------------------------------------------------
IO Size Wash Size  Run Size    APF%   LR/sec   PR/sec   Hit%   APF-Eff% Usage%
------- ---------- ----------- ------ -------- -------- ------ -------- ------
2 Kb     1024 Kb     5.00 Mb  10.00   380.97    56.67  85.13  98.42   100.00


Object Statistics
------------------------------------------------------------------------------

Object                         LR/sec  PR/sec  Hit%   Obj_Cached% Cache_Occp%
------------------------------ ------- ------- ------ ----------- -----------
company_db.dbo.emp_projects     41.07   22.80  44.48       19.64        9.45
company_db.dbo.dept_det         93.03   20.67  77.79       99.08       54.53
company_db.dbo.emp_perf        116.70    2.63  97.74       97.77       34.18
company_db.dbo.dept_locs         0.43    0.17  61.54       50.00        0.16
```

```
Object                          Obj Size    Size in Cache
------------------------------ ----------- -------------
company_db.dbo.emp_projects       2464 Kb      484 Kb
company_db.dbo.dept_det           2818 Kb     2792 Kb
company_db.dbo.emp_perf           1790 Kb     1750 Kb
company_db.dbo.dept_locs            16 Kb        8 Kb

TUNING RECOMMENDATIONS
-----------------------------------------------------------------------
Usage% for 'default data cache' is low (< 5%)
Usage% for  4k buffer pool in cache:default data cache is low (< 5%)
Usage% for  2k buffer pool in cache:default data cache is low (< 5%)

Usage% for 'company_cache' is low (< 5%)
Usage% for  2k buffer pool in cache:company_cache is low (< 5%)


 Consider adding a large I/O pool for 'companydb_cache'

LEGEND
------
LR/sec     - number of logical reads per second, i.e. sum of cache & disk reads
PR/sec     - number of physical reads per second i.e. disk reads
Run Size- size of cache or buffer pool in Kilobytes

Cache Partitions- number of cache partitions
Spinlock Contention%- Percentage spinlock contention for the cache
Hit%       - ratio of hits to total searches

Usage%     - ratio of pages referenced to Run Size

Wash Size- wash size of buffer pool in Kilobytes
APF%       - asynchronous prefetch % for this buffer pool
APF-Eff%- Ratio of buffers found in cache and brought in because
                of APF to the number of APF disk reads performed

Object     - combination of db, owner, object and index name
Obj Size- size of the object in Kilobytes
Size in Cache- size occupied in cache in Kilobytes at the end of sample
Obj_Cached%- Ratio of 'Size in Cache' to 'Obj Size'
Cache_Occp%- Ratio of 'Size in Cache' to 'Run Size' of cache
```

カーネルの使用率

注意 “Kernel Utilization” セクションでは、Adaptive Server の実行モードがスレッド・モードまたはプロセス・モードであるかによって、レポートされる情報が異なります (プロセス・モードについては、「Kernel Utilization － プロセス・モード」(29 ページ) を参照)。

Adaptive Server のアクティビティをレポートします。CPU が Adaptive Server に対して利用可能だった時点での、Adaptive Server エンジンのビジー状態の程度、CPU がオペレーティング・システムに解放された頻度、エンジンがネットワークとディスク I/O をチェックした回数、および各チェック時に待機中だった I/O の平均値が示されます。

Kernel Utilization － スレッド・モード

出力例

次の例は、スレッド・プール syb_default_pool が 1 つ、Adaptive Server エンジンが 3 つある環境での sp_sysmon からの “Kernel Utilization” の出力を示します。

```
Kernel Utilization

------------------
 Engine Utilization (Tick %)   User Busy   System Busy    I/O Busy      Idle
 -----------------------  ------------  ------------  ----------  --------
 ThreadPool : syb_default_pool
    Engine 0                    8.7 %         0.8 %      31.1 %    59.5 %
    Engine 1                    8.0 %         0.1 %      33.2 %    58.7 %
    Engine 2                    8.8 %         0.3 %      32.7 %    58.2 %
    Engine 3                   15.0 %         0.3 %      32.6 %    52.0 %
   -------------------------  ------------  ------------  ----------  -------
   Server Summary      Total        40.4 %         1.6 %     129.6 %  228.4 %
                     Average        10.1 %         0.4 %      32.4 %   57.1 %

 Average Runnable Tasks              1 min        5 min     15 min  % of total
 -------------------------       ----------    ----------   -------   --------

 ThreadPool : syb_default_pool
    Global Queue                     0.0           0.0        0.0      0.0 %
    Engine 0                         0.3           0.2        0.1     61.7 %
    Engine 1                         0.0           0.0        0.0      0.7 %
    Engine 2                         0.1           0.1        0.0     17.3 %

   -------------------------  ------------  ------------  ---------
   Pool Summary        Total          0.5           0.3        0.1
                     Average          0.1           0.1        0.0
```

```
   -------------------------  ------------  ------------  ----------
   Server Summary      Total           0.5           0.3         0.1
                     Average           0.1           0.1         0.0

 CPU Yields by Engine                per sec      per xact      count % of total
 -------------------------          --------    ----------   --------  --------
ThreadPool : syb_default_pool
 Engine 0
    Full Sleeps                        20.4          13.8        2442     4.2 %
    Interrupted Sleeps                 81.7          55.4        9800    16.7 %
 Engine 1
    Full Sleeps                        22.8          15.5        2740     4.7 %
    Interrupted Sleeps                 89.1          60.4       10697    18.3 %
 Engine 2
    Full Sleeps                        19.9          13.5        2390     4.1 %
    Interrupted Sleeps                 93.4          63.3       11202    19.1 %
 Engine 3
    Full Sleeps                        17.8          12.1        2133     3.6 %
    Interrupted Sleeps                142.6          96.7       17113    29.2 %
 -------------------------   ------------  ------------  ----------
 Pool Summary                      487.6         330.6       58517

 -------------------------   ------------  ------------  ----------
 Total CPU Yields                  487.6         330.6       58517

Thread Utilization (OS %)       User Busy   System Busy        Idle
 -------------------------   ------------  ------------  ----------
 ThreadPool : syb_blocking_pool : no activity during sample

 ThreadPool : syb_default_pool
  Thread 6    (Engine 0)            0.0 %         0.0 %      100.0 %
  Thread 7    (Engine 1)            0.2 %         0.0 %       99.8 %
  Thread 8    (Engine 2)            0.0 %         0.0 %      100.0 %
 -------------------------   ------------  ------------  ----------
 Pool Summary        Total           0.2 %         0.0 %      299.9 %
                   Average           0.1 %         0.0 %       99.9 %

 ThreadPool : syb_system_pool
  Thread 10   (NetController)        0.1 %         0.0 %       99.9 %
  -------------------------   ------------  ------------  ----------
 Pool Summary       Total           1.1 %         0.0 %      400.0 %
                  Average           0.0 %         0.0 %      100.0 %

 -------------------------   ------------  ------------  ----------
 Server Summary      Total           0.2 %         0.0 %     1099.8 %
                   Average           0.0 %         0.0 %      100.0 %

Adaptive Server threads are consuming 0.0 CPU units.
Throughput (committed xacts per CPU unit) : 275.0
```

```
 Page Faults at OS                per sec      per xact       count % of total
 -------------------------       --------    ----------    --------  --------
  Minor Faults                        0.6           0.4          69    100.0 %
  Major Faults                        0.0           0.0           0      0.0 %
 -------------------------       --------    ----------    --------  --------
 Total Page Faults                    0.6           0.4          69    100.0 %


 Context Switches at OS           per sec      per xact       count % of total
 -------------------------       --------    ----------    --------  --------
  ThreadPool : syb_blocking_pool
    Voluntary                         0.0           0.0           0      0.0 %
    Non-Voluntary                     0.0           0.0           0      0.0 %
  ThreadPool : syb_default_pool
    Voluntary                        43.6         130.7        1307     56.3 %
    Non-Voluntary                     0.2           0.6           6      0.3 %
  ThreadPool : syb_system_pool
    Voluntary                        33.7         101.0        1010     43.5 %
    Non-Voluntary                     0.0           0.0           0      0.0 %
   -------------------------  -----------  ------------  ----------  ---------
 Total Context Switches              77.4         232.3        2323    100.0 %

 CtlibController Activity         per sec      per xact      count  % of total
 -------------------------       --------    ----------    --------  --------
  Polls                               1.0           0.7         120       n/a
  Polls Returning Events              0.0           0.0           0      0.0 %

DiskController Activity           per sec      per xact       count % of total
 -------------------------       --------    ----------    --------  --------
  Polls                           35019.1       23741.8     4202292       n/a
  Polls Returning Events            351.2         238.1       42145      1.0 %
  Polls Returning Max Events          0.0           0.0           0      0.0 %
  Total Events                      374.9         254.1       44984       n/a
  Events Per Poll                     n/a           n/a       0.011       n/a

NetController Activity            per sec      per xact       count % of total
 -------------------------       --------    ----------    --------  --------
  Polls                            7919.5        5369.1      950338       n/a
  Polls Returning Events           2537.8        1720.5      304536     32.0 %
  Polls Returning Max Events          0.0           0.0           0      0.0 %
  Total Events                     2537.8        1720.5      304536       n/a
  Events Per Poll                     n/a           n/a       0.320       n/a

 Blocking Call Activity           per sec      per xact       count % of total
 -------------------------       --------    ----------    --------  --------
  Total Requests                      0.0           0.0           0       n/a
```

エンジンの使用率 (チック %)

(アイドル状態だった時間ではなく) 各 Adaptive Server エンジンでタスクを実行しているために、Adaptive Server カーネルがビジー状態だった時間のパーセンテージが内部測定 (“チック”) を使用して報告されます。出力はスレッド・プールに応じてグループ化されます。“Server Summary”には、すべてのプールの合計時間が表示されます。

“Engine Utilization (Tick %)”は、Adaptive Server エンジンが多すぎるかまたは少なすぎるかを調べる場合に役立ちます。

“Engine Utilization (Tick %)”にレポートされる値は、オペレーティング・システムのツールによってレポートされる CPU 使用率の値とは異なります。Adaptive Server によって処理されるタスクがない場合、Adaptive Server はループに入り、スリープ状態になる前に作業を探します。alter thread pool ... idle timeout パラメータは、Adaptive Server エンジンが CPU を解放する前に実行できるタスクを検索してループする時間 (ミリ秒単位) を制御します。

Adaptive Server が実行できるタスクのチェックに要する時間を減らすには、idle timeout スレッド・プール・パラメータの値を小さくします。ただし、idle timeout の値を小さくすると、遅延時間が増えて、パフォーマンスが低下する可能性があります。

sp_sysmon がカウンタをサンプリングする場合 (デフォルトでは、100 ミリ秒ごと)、各エンジンは現在実行中の動作を示します。sp_sysmon は、エンジンがどの程度ビジーであるかをレポートし、ユーザとシステム・タスクを区別します。

- “User Busy” － ユーザ・タスク (ユーザ接続数など)。
- “System Busy” － 内部タスク (ハウスキーピングなど)。

たとえば、タスクを実行している場合は "CPU Busy"、アイドル中の場合は、"Idle" とそれぞれ表示します。アイドル中に Adaptive Server で保留中の I/O がある場合は、"I/O Busy" としてカウントされます。3 つのエンジンがアイドル中に保留中の I/O が 1 つある場合、各エンジンは "I/O Busy" としてカウントされます。

sp_sysmon の出力をチェックして問題がないかどうかを調べ、競合を緩和するためにチューニングを行うことで、エンジンが 80 ～90% ビジー状態にあっても、高いレベルの応答時間を維持できます。値が一貫して非常に大きい (90% 以上) 場合は、エンジンを追加することで、応答時間とスループットを向上させることができます。

“Engine Utilization (Tick %)”の値はサンプル間隔中の平均値であるため、この平均値が非常に大きい場合は、サンプル間隔のある時点でエンジンが 100% ビジー状態になっている可能性があります。スピンロック競合が発生すると CPU の使用率が高くなるため、サーバで CPU の使用率が高い場合は確認を行ってください。メモリ内テーブル・スキャンによっても CPU の使用率が高くなります。使用率を低下させるには、クエリを適切にチューニングします。

エンジンの使用率がきわめて高い場合は、housekeeper ウォッシュ・タスクによってディスクに書き込まれるページはほとんどありません (このタスクはアイドル CPU サイクルの間にだけ実行されるためです)。これは、チェックポイントがディスクに書き込む必要がある多くのページを発見し、チェックポイントのプロセス、大規模なバッチ・ジョブ、データベース・ダンプのいずれかによって、ある期間の CPU 使用率が 100% になり、応答時間の著しい低下を引き起こすことを意味します。

“Engine Utilization (Tick %)”のパーセンテージが一貫して高いときに、Adaptive Server エンジンを追加して応答時間とスループットを向上させたい場合は、各エンジンを追加したあと、ほかの領域でリソースの競合が増加していないかどうかをチェックしてください。

Adaptive Server が多数のユーザを処理している環境では、通常、パフォーマンスはエンジンの間でかなり均等に分散されます。しかし、タスクよりもエンジンの数が多い場合、いくつかのエンジンの使用率が高くなり、ほかのエンジンがアイドル状態になることがあります。

例：

```
Engine Utilization (Tick %)      User Busy   System Busy    I/O Busy     Idle
   -------------------------  ------------  ------------  ----------  -------
   ThreadPool : Marketing_pool
    Engine 3                         78.0 %         4.5 %       3.4 %   18.0 %

   ThreadPool : syb_default_pool
    Engine 0                          8.7 %          .8 %       1.3 %   94.8 %
    Engine 1                         87.0 %         5.2 %       3.5 %   19.4 %
    Engine 2                         65.7 %         3.7 %       3.7 %   12.8 %
```

SMP 環境では、タスクはエンジンにゆるやかに結び付けられています。グローバル実行キュー内にタスクを配置することがある、ロック競合などのほかのアクティビティがなければ、そのタスクは同じエンジンで実行し続けます。

平均実行可能なタスク

実行可能なタスクの 1 分間、5 分間、15 分間平均を出すには、monSysLoad モニタリング・テーブルの実行可能なタスク平均を使用します。エンジンで実行中の (実行可能な) タスクはすべてこの平均に含まれます。分間隔は固定されており、設定した sp_sysmon サンプル時間によって変化しません。

実行可能なタスクの数によって、Adaptive Server がどのくらいビジーかを正確に測定できます。たとえば、平均“エンジン使用率”が 90% で実行可能なタスクの数が多いサーバは、平均エンジン使用率が 90% で実行可能なタスクが少ないサーバよりも、追加エンジンのメリットを利用できる可能性が大きくなります。

1 分間、5 分間、15 分間サンプルを比較すると、サーバの負荷が増加、安定、または減少しているかどうかを調べるのに役立ちます。

エンジンごとの CPU の解放

各 Adaptive Server エンジンがオペレーティング・システムに CPU を解放した回数をレポートします。エンジンは解放されると、短時間の間スリープします。ウェイクアップ時に実行可能なタスクがない場合は、再度スリープ・モードに入ります。出力はスレッド・プールとそれに関連するエンジンに応じて配置されます。各エンジンでは、sp_sysmon によって次のことがレポートされます。

- Full Sleeps － エンジンがフル・スリープ・モード中も継続してスリープ状態にあったことを示す (つまり、スリープ中に実行可能なタスクを受け取らなかったことを示す)。フル・スリープ・モードに入ると、別のスリープが発生する可能性が高くなる。ただし、フル・スリープによって遅延が発生することはない。
- Interrupted Sleeps － 実行可能なタスクによってスリープモードが中断され、エンジンがウェイクアップさせられたことを示す。スリープ状態が割り込みにより終了すると、スケジュール・タスクが発生する可能性が高くなる。スリープ状態が割り込みにより終了すると、遅延が発生する場合がある。

“% of total”のデータは、すべてのエンジンが解放した回数の合計に対して、1 つのエンジンが解放した回数の割合をパーセンテージで示します。

“Total CPU Yields”では、すべてのエンジンにわたって合計されたデータがレポートされます。

エンジンがビジー状態でない場合は、エンジンは idle timeout パラメータを使用して、指定した時間を過ぎたあとに CPU を解放します。

- Engine Utilization Low/CPU Yields Low － I/O Busy% カラムの値が CPU Busy% カラムの値よりも大きい場合は、保留中の I/O が大量に存在することを示す。I/O 処理でボトルネックを解決するには、オペレーティング・システム・レベルで加えることができる変更点について考慮する。
- Engine Utilization Low/CPU Yields High － エンジンは非アクティブである。
- Engine Utilization High/CPU Yields Low － Adaptive Server は非常にビジーで、複数のジョブを実行する必要がある。エンジンを追加するとパフォーマンスの改善に役立つ可能性がある。この値は、“Average Runnable Tasks”の値が高いことにも相関する。
- Engine Utilization High/CPU Yields High － 簡単なクエリを実行しているユーザ接続数が標準的な場合に発生する。“Engine Utilization”が非常に高くないか、“Average Runnable”タスクが高くない限り、エンジンを追加しても役に立つ可能性はほとんどない。
- ほとんどのエンジンの解放では“full sleeps”が利用される。“interrupted sleeps”がエンジン全体の解放の 20% 以上を占めると、全体の遅延時間が増加する場合がある。遅延時間を短くするために、idle timeout の値を大きくすることを検討する。ただし、これにより、CPU 使用量が増加する。

『リファレンス・マニュアル：コマンド』のalter thread pool を参照してください。

スレッド使用率 (OS %)

各 Adaptive Server のスレッドがオペレーティング・システム・レベルで使用している時間を表示します。スレッド使用率とは、スレッドが実際に CPU 上にあった時間の部分を指します。“Pool Summary”は、スレッド・プールでスレッドが使用されるパーセンテージを示します。“Server Summary”は、Adaptive Server でスレッドが使用されるパーセンテージを示します。

“User time”では、スレッドがユーザ空間でコードを実行していた (つまり、Adaptive Server で実行していた) sp_sysmon サンプル間隔中の時間の割合が、パーセンテージでレポートされます。“System time”では、スレッドがシステム空間でコードを実行していた (つまり、オペレーティング・システム・カーネルで実行していた) sp_sysmon サンプル間隔中の時間の割合が、パーセンテージでレポートされます。この区別は、Engine Utilization (Tick %) セクションでレポートされた“User Busy”と“System Busy”とは無関係です。

注意 sp_sysmon では、Adaptive Server のデータ収集方法が原因で、使用率が 100% を超えるスレッドが表示される場合があります。

“Thread Utilization (OS %)”出力を調べる場合は、次のことを考慮してください。

- “Thread Utilization (OS %)”の値は“Engine Utilization (Tick %)”の値よりも大きくなければならない。両方の値の差が最も大きくなるのは、idle timeout が高い値に設定され、作業負荷が断続的である場合である。たとえば、100% アイドル状態にあるが idle timeout の値が -1 であるエンジンでは、エンジン使用率は 0% だが、スレッド使用率は 100% として表示される。
- “Engine Utilization (Tick %)”の値が“Thread Utilization (OS %)”の値よりも大きい場合は、ホストで CPU リソースが少なくなっている可能性がある。これは一般的に、エンジンに必要な CPU 時間が与えられなかったことを示す。これにより、パフォーマンスは大幅に低下する可能性がある。スピンロック競合も関係している場合は、パフォーマンスが著しく低下する可能性がある。
- ディスクまたはネットワーク・タスクの“Thread Utilization (OS %)”の値が高い場合は、追加のディスクまたはネットワーク・タスクが必要である可能性がある。たとえば、ネットワーク・タスク・スレッドは 80% ビジー状態にあるが、エンジン使用率が低い場合は、飽和状態のネットワーク・タスクが作業用のエンジンを必要としている可能性がある。sp_configure "number of network tasks" を使ってネットワーク・タスクを追加すると、この状況が緩和される場合がある。

“Thread Utilization (OS %)”の最後に、sp_sysmon では、Adaptive Server スレッドが使用している CPU ユニットと各 CPU でコミットされたトランザクションのスループットの数がレポートされます。次に例を示します。

```
Adaptive Server threads are consuming 5.6 CPU units.
Throughput is 8238.0 commited xacts per CPU unit.
```

Adaptive Server が使用できる物理ハードウェアに使用される CPU ユニットの数を比較できます。たとえば、8 つのコアと 1 つのコアにつき 2 つのスレッドがある場合は、合計 16 の CPU ユニットを使用できます。Adaptive Server が 6 つの CPU ユニットを使用している場合は、追加の作業を実行できます。ただし、Adaptive Server が 14 の CPU ユニットを使用している場合は、キャパシティがほとんど残っていないため、サブコア・スレッドを使用する頻度が高くなります。

OS でのページ・フォールト

メジャーおよびマイナー・ページ・フォールトの数とすべてのページ・フォールトの概要をレポートします。sp_sysmon では、レポートされたページ・フォールトがある場合に限り、このセクションが出力されます。

注意　Windows の場合、“Page Faults at OS”は表示されません。

- マイナー・フォールト － メモリ管理ユニットで「使用可能」とマーク付けされていなかったが、物理メモリで使用可能であったメモリ・ページを Adaptive Server が必要とした場合に発生する。
- メジャー・フォールト － 物理メモリで使用できなかったメモリ・ページを Adaptive Server が必要とし、ディスクからページを取得しなければならなかった場合に発生する。

マイナーおよびメジャー・ページ・フォールトが発生した場合は、物理メモリが不足している可能性があります。メジャー・ページ・フォールトは、マイナー・ページ・フォールトよりもはるかに大きなパフォーマンスの低下につながります。メジャー・ページ・フォールトが発生した場合は、Adaptive Server のメモリ設定がホストに対して高すぎるか、その他のプロセス (ファイル・システム・キャッシュなど) によって Adaptive Server が予期する物理ページが使用されています。

OS でのコンテキストの切り替え

Adaptive Server スレッドがオペレーティング・システム・レベルで実行する自発的および自発的でないコンテキストの切り替え回数をレポートします。sp_sysmon では、レポートされたコンテキストの切り替えがある場合とオペレーティング・システムでデータ収集がサポートされている場合に限り、このセクションが出力されます。

注意　Windows、Red Hat 5、または SLES 11 では、“Page Faults at OS”は表示されません。

Adaptive Server は、I/O ポーリングやエンジンのスリープなどのさまざまな理由で、「自発的な」コンテキストの切り替えを実行する場合があります。コンテキストの切り替え回数は、作業負荷によって異なります。「自発的でない」コンテキストの切り替えは、スレッドが削除を要求せずに、オペレーティング・システムによって Adaptive Server が CPU から削除される場合 (オペレーティング・システムによってスレッドが先取りされる場合など) に発生します。「自発的でない」コンテキストの切り替えが発生すると、CPU リソースが残り少なくなり、Adaptive Server のパフォーマンスが著しく低下する可能性があります。自発的でないコンテキストの切り替え回数は少ないはずなので問題はありません。

次に、設定済みでないシステムからのサンプル出力を示します。「自発的でない」コンテキストの切り替え回数は非常に少ないです。

```
Context Switches at OS          per sec      per xact      count   % of total
 ------------------------  -----------  ------------  ---------   ----------
  Voluntary                       278.7         278.7       2787        99.7 %
  Non-Voluntary                     0.9           0.9          9         0.3 %
 ------------------------  -----------  ------------  ---------   ----------
Total Context Switches            279.6         279.6       2796       100.0 %
```

以下は、「自発的でない」コンテキストの切り替え回数が非常に多い過負荷になったマシンからのサンプル出力を示します。

```
Context Switches at OS          per sec      per xact      count   % of total
 ------------------------  -----------  ------------  ---------   ----------
  Voluntary                      2683.7       16370.4     163704        18.4 %
  Non-Voluntary                 11893.0       72547.4     725474        81.6 %
 ------------------------  -----------  ------------  ---------   ----------
Total Context Switches          14576.7       88917.8     889178       100.0 %
```

この状況を改善するには、エンジン数を減らし、Adaptive Server 以外の作業負荷をホストから削除し、さらに CPU を追加するか、追加の処理能力を持つホストにマイグレートします。

CtlibController、DiskController、および NetController アクティビティ

“CtlibController Activity”は、Client Library (CTLIB) コントローラ・イベントを Adaptive Server がチェックする頻度 (`Polls` ローに表示) と Client Library がイベントを返す回数 (`Polls Returning Events` ローに表示) を示します。

“DiskController Activity”は、ディスク・コントローラ・イベントを Adaptive Server がチェックする頻度 (`Polls` ローに表示) とディスクがイベントを返す回数 (`Polls Returning Events` ローに表示) を示します。

“NetController Activity”は、ネットワーク・コントローラ・イベントを Adaptive Server がチェックする頻度 (`Polls` ローに表示) とネットワークがイベントを返す回数 (`Polls Returning Events` ローに表示) を示します。

これらのセクションには次のローがあります。

- Polls － このコントローラが、完了までにオペレーティング・システムをポーリングした回数。
- Polls Returning Events － このコントローラが、オペレーティング・システムをポーリングし、少なくとも 1 つのイベントを受け取った回数。
- Polls Returning Max Events － オペレーティング・システムのポーリングによって最大イベント数が返された回数。
- Total Events － 返された I/O イベントの総数。
- Events Per Poll － 各ポーリングで返された平均イベント数。

複数のディスクまたはネットワーク・タスクを設定する場合、表示されるデータはそのコントローラのすべてのタスクからなる集合体です。

次の場合に、ディスクまたはネットワーク・タスクの追加を検討してください。

- “Polls Returning Max Events”が 0 より多い。
- “Events per Poll”の値が 3 より多い。

ただし、ディスクまたはネットワーク・タスクをさらに追加する前に、コントローラの“Thread Utilization”の値とシステム全体の負荷を確認してください。

呼び出しアクティビティのブロック

呼び出しタスクのブロック (つまり、syb_blocking_pool への要求) についてレポートします。キューイングされた要求のパーセンテージが大きい場合や待機時間がかなり長い場合は、syb_blocking_pool サイズ変更を検討してください。

Kernel Utilization － プロセス・モード

Adaptive Server のアクティビティをレポートします。CPU が Adaptive Server に対して利用可能だった時点での、Adaptive Server エンジンのビジー状態の程度、CPU がオペレーティング・システムに解放された頻度、エンジンがネットワークとディスク I/O をチェックした回数、および各チェック時に待機中だった I/O の平均値が示されます。

出力例

次の例は、Adaptive Server エンジンが 8 つある環境での、sp_sysmon からの“Kernel Utilization”の出力を示します。

```
Kernel Utilization
------------------

Your Runnable Process Search Count is set to 2000
and I/O Polling Process Count is set to 10

Engine Busy Utilization       CPU Busy    I/O Busy      Idle
```

```
------------------------      --------    --------  --------
Engine 0                         3.3 %      13.7 %     83.0 %
Engine 1                         2.4 %       9.2 %     88.4 %
Engine 2                         4.9 %      19.9 %     75.2 %
Engine 3                          .8 %      19.1 %     76.1 %
Engine 4                         2.8 %       7.0 %     90.2 %
Engine 5                         1.9 %       7.3 %     90.8 %
Engine 6                         2.5 %       7.6 %     89.9 %
Engine 7                         2.0 %       8.1 %     89.9 %
------------------------      --------    --------   --------
Summary              Total      24.6 %      91.9 %    683.5 %
Average                          3.1 %      11.5 %     85.4 %

CPU Yields by Engine           per sec      per xact       count   % of total
-------------------------  -----------   -----------   ---------   ---------
Engine 0                          31.5           0.0        2802      11.6 %
Engine 1                          38.1           0.0        3388      14.0 %
Engine 2                          21.8           0.0        1936       8.0 %
Engine 3                          30.2           0.0        2689      11.1 %
Engine 4                          37.9           0.0        3372      13.9 %
Engine 5                          38.4           0.0        3421      14.1 %
Engine 6                          36.1           0.0        3217      13.3 %
Engine 7                          38.4           0.0        3420      14.1 %
-------------------------   ----------   -----------    --------
Total CPU Yields                 272.4           0.2       24245

Network Checks
Non-Blocking                  434722.4         366.8    38690292      99.9 %
Blocking                         272.4           0.2       24247       0.1 %
-------------------------   ----------   -----------    --------
Total Network I/O Checks      434994.8         367.1    38714539
Avg Net I/Os per Check             n/a           n/a     0.00003         n/a

Disk I/O Checks
Total Disk I/O Checks         435855.6         367.8    38791149         n/a
Checks Returning I/O          125358.1         105.8    11156875      28.8 %
Avg Disk I/Os Returned              n/a           n/a      0.00313         n/a
```

エンジンのビジー状態での使用率

(アイドル状態だった時間ではなく) 各 Adaptive Server エンジンでタスクを実行しているために、Adaptive Server カーネルがビジー状態だった時間のパーセンテージが報告されます。計算ローには、すべてのエンジンのアクティブ時間を合わせた総数と平均値が示されます。

Engine Busy Utilization によってレポートされる値は、オペレーティング・システムのツールによってレポートされる CPU 使用率の値とは異なります。Adaptive Server によって処理されるタスクがない場合、Adaptive Server はネットワーク I/O、完了したディスク I/O、実行キュー内のタスクを定期的にチェックするループに入ります。

Adaptive Server の内部からは行えない測定の 1 つに、Adaptive Server が CPU を制御した時間と、CPU がオペレーティング・システムに使用されていた時間とを比較したパーセンテージがあります。オペレーティング・システムは CPU 時間を Adaptive Server に表示します。sp_sysmon は、Adaptive Server がオペレーティング・システムによって表示される時間をどのように使用するかをレポートします。両方のシステムの読み取り値を検出するには、sp_sysmon をオペレーティング・システムのモニタ・ツールとともに必ず使用してください。

正しいコマンドについては、オペレーティング・システムのマニュアルで確認してください。

アイドル中の Adaptive Server が I/O のチェックに要する時間を減らすには、sp_configure パラメータの runnable process search count の値を減らします。このパラメータは、Adaptive Server エンジンが CPU を解放する前に実行できるタスクを検索してループする時間を指定します。ただし、runnable process search count の値を小さくすると、遅延時間が増えて、パフォーマンスが低下する可能性があります。

“Engine Busy Utilization”では、与えられた CPU 時間の間に Adaptive Server エンジンがビジー状態だった時間が測定されます。10 分間のサンプル間隔のうち、Adaptive Server がエンジンを 80% 利用できて、“Engine Busy Utilization”が 90% だったときは、Adaptive Server が 7 分 12 秒間ビジー状態であり、48 秒間アイドル状態だったことを意味します。

sp_sysmon がカウンタをサンプリングする場合 (デフォルトでは、100ミリ秒ごと)、各エンジンは現在実行中の動作を示します。タスクを実行している場合は "CPU Busy"、アイドル中の場合は "Idle"、アイドル中に保留中の非同期ディスク I/O が少なくとも 1 つある場合は "IO Busy" とそれぞれ表示されます。

“Engine Busy Utilization”は、Adaptive Server エンジンが多すぎるかまたは少なすぎるかを調べる場合に役立ちます。リソースの競合を回避するチューニング可能なメカニズムによって、Adaptive Server の高いスケーラビリティが提供されます。

sp_sysmon の出力をチェックして問題がないかどうかを調べ、競合を緩和するためにチューニングを行うことで、“Engine Busy”の値が 80 ～ 90% 以内であっても、高いレベルの応答時間を維持できます。値が一貫して非常に大きい (90% 以上) 場合は、エンジンを追加することで、応答時間とスループットを向上させることができます。

“Engine Busy Utilization”の値はサンプル間隔中の平均値であるため、この平均値が非常に大きい場合は、サンプル間隔のある時点でエンジンが 80% ビジー状態になっている可能性があります。スピンロック競合が発生すると CPU の使用率が高くなるため、サーバで CPU の使用率が高い場合は確認を行ってください。メモリ内テーブル・スキャンによっても CPU の使用率が高くなります。使用率を低下させるには、クエリを適切にチューニングします。

エンジンの使用率がきわめて高い場合は、housekeeper ウォッシュ・タスクによってディスクに書き込まれるページはほとんどありません (このタスクはアイドル CPU サイクルの間にだけ実行されるためです)。これは、チェックポイントがディスクに書き込む必要がある多くのページを発見し、チェックポイントのプロセス、大規模なバッチ・ジョブ、データベース・ダンプのいずれかによって、ある期間の CPU 使用率が 100% になり、応答時間の著しい低下を引き起こすことを意味します。

“Engine Busy Utilization”のパーセンテージが一貫して高いときに、Adaptive Server エンジンを追加して応答時間とスループットを向上させたい場合は、各エンジンを追加したあと、ほかの領域でリソースの競合が増加していないかどうかをチェックしてください。

Adaptive Server が多数のユーザを処理している環境では、通常、パフォーマンスはエンジンの間でかなり均等に分散されます。しかし、タスクよりもエンジンの数が多い場合、いくつかのエンジンの使用率が高くなり、ほかのエンジンがアイドル状態になることがあります。次に例を示します。

```
Engine Busy Utilization          CPU Busy       I/O Busy        Idle
------------------------         ---------      -------      --------
Engine 0                            68.7 %          2.5 %       28.8 %
Engine 1                            61.9 %          3.3 %       34.8 %
Engine 2                            67.0 %          2.4 %       30.6 %
Engine 3                            69.0 %          3.8 %       27.2 %
Engine 4                            60.2 %          2.7 %       37.2 %
Engine 5                            55.7 %          3.2 %       41.1 %
Engine 6                            53.8 %          3.2 %       43.0 %
------------------------         --------       --------    --------
Summary                Total       436.3 %          21.1 %     242.6 %
Average                              62.3 %           3.0 %      34.7 %
```

SMP 環境では、タスクはエンジンにゆるやかに結び付けられています。グローバル実行キュー内にタスクを配置することがある、ロック競合などのほかのアクティビティがなければ、そのタスクは同じエンジンで実行し続けます。

エンジンごとの CPU の解放

各 Adaptive Server エンジンがオペレーティング・システムに対して CPU を解放した回数。“% of total”のデータは、すべてのエンジンが解放した回数の合計に対して、1 つのエンジンが解放した回数の割合をパーセンテージで示します。

“Total CPU Yields”では、すべてのエンジンにわたって合計されたデータがレポートされます。ただし、1 つ以上の保留中の非同期ディスク I/O を含んでいるエンジンは、runnable process search count を使い終わった場合でも解放されません。

“Engine Busy Utilization”のデータがエンジンの使用率が低いことを示す場合は、“CPU Yields by Engine”のデータを使用して、実際にアクティブでないのか、それともオペレーティング・システムによって頻繁に CPU を使用できないようにされているのかを調べます。

エンジンがビジー状態でない場合は、エンジンは runnable process search count パラメータを使用して、指定した時間を過ぎたあとに CPU を解放します。“CPU Yields by Engine”の値が大きい場合は、エンジンが自発的に解放したことを示します。

- Engine Busy Low/CPU Yields Low － I/O Busy% カラムの値が CPU Busy% カラムの値よりも大きい場合は、保留中の I/O が大量に存在することを示す。I/O 処理でボトルネックを解決するには、オペレーティング・システム・レベルで加えることができる変更点について考慮する。Adaptive Server のシステム CPU 時間が大きすぎる場合は (オペレーティング・システムの出力に基づく)、runnable process search count の値を減らす。
- Engine Busy Low/CPU Yields High － エンジンは非アクティブである。
- Engine Busy High/CPU Yields Low － Adaptive Server は非常にビジーで、複数のジョブを実行する必要がある。エンジンを追加するとパフォーマンスの改善に役立つ可能性がある。
- Engine Busy High/CPU Yields High － 簡単なクエリを実行しているユーザ接続数が標準的な場合に発生する。runnable process search count を低くしても効果はほとんどないか、まったくない可能性がある。“Engine Busy Utilization”が非常に高くない限り、エンジンを追加しても役に立つ可能性はほとんどない。

『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

```
CPU Yields by Engine            per sec      per xact      count   % of total
------------------------   ------------   -----------   --------    ---------
Engine 0                          73.5           1.8      44087      26.2 %
Engine 1                          40.4           1.0      24224      14.4 %
Engine 2                          23.7           0.6      14208       8.4 %
Engine 3                          36.2           0.9      21746      12.9 %
Engine 4                          38.7           1.0      23207      13.8 %
Engine 5                          41.3           1.0      24760      14.7 %
Engine 6                          26.8           0.7      16108       9.6 %
----------------------     ------------    ----------   --------
Total CPU Yields                 280.6           6.9      168340
```

ネットワーク・チェック

ブロッキング・ネットワーク I/O チェックと非ブロッキング・ネットワーク I/O チェック、間隔での I/O チェックの総数、ネットワーク・チェック当たりのネットワーク I/O の平均値についての情報をレポートします。

Adaptive Server には、ネットワーク I/O をチェックする方法として、ブロッキング・モードと非ブロッキング・モードがあります。

```
Network Checks
Non-Blocking                166371.5        4098.3    99822899      99.8 %
Blocking                       279.0           6.9      167388       0.2 %
-------------------------   ---------      --------   ----------
Total Network I/O Checks    166650.5        4105.2    99990287
Avg Net I/Os per Check           n/a           n/a     0.00476        n/a
```

非ブロッキング

Adaptive Server が非ブロッキング・ネットワーク・チェックを実行した回数。非ブロッキング・ネットワーク・チェックでは、エンジンがネットワークの I/O をチェックし、I/O が待機中であるかどうかにかかわらず、処理が継続されます (通常、エンジンはこの方法でネットワーク I/O をチェックします)。

ブロッキング

Adaptive Server がブロッキング・ネットワーク・チェックを実行した回数。こうして、エンジンはオペレーティング・システムに CPU を解放し、解放の数に等しくなります (タイミングの問題で異なる場合もあります)。

エンジンがキュー内のすべての実行可能タスクを完了した後、エンジンは短時間ループし、ネットワークのポーリングと次のクライアント要求のチェックを行います。一定の回数 (sp_configure パラメータの runnable process search count によって決定されます) だけループした後で、実行可能なタスクが検出されず、保留中の非同期ディスク I/O がない場合、エンジンはブロッキング・ネットワーク I/O をポーリングし、CPU をオペレーティング・システムに解放します。この時点で、エンジンはスリープしていることになります。

スリープ状態のエンジンはクロック・チックごとに 1 回の割合でウェイクアップし、ルーチン・ハウスキーピング・タスクを実行し、実行可能なタスクをチェックします。実行可能なタスクがない場合、エンジンは別のブロッキング・ネットワーク・チェックを行い、スリープ状態に戻ります。

Adaptive Server は、別のエンジンによってネットワーク I/O が完了すると、次のクロック・チックの前にスリープ中のエンジンをウェイクアップする場合があります。ウェイクアップされると、エンジンは処理後の I/O 作業を実行し、通常はタスクが実行可能になります (I/O 作業が処理されるまでは、このタスクがエンジンを再び解放する可能性はほとんどありません)。

ネットワーク I/O チェックの総数

エンジンがパケットを送受信するソケットをポーリングする回数。このカテゴリを“CPU Yields by Engine”と合わせて使用すると、役に立ちます。

エンジンがアイドル状態の場合、エンジンはネットワーク・パケットをチェックする間ループします。“Network Checks”の値が低く、“CPU Yields by Engine”の値が大きい場合、エンジンが CPU を解放している頻度が高く、ネットワークをチェックする頻度が不足しています。システムにオーバヘッドを負担する余裕がある場合は、解放の頻度が低くなっても許容されます。

エンジンは runnable process search count で定義した回数だけループした後で解放されます (ただし、エンジンに少なくとも 1 つの保留中のディスク I/O が必要です)。その結果、エンジンを解放する頻度が高すぎると、runnable process search count の設定値が低くなりすぎるため、パフォーマンス低下が発生する場合があります。

チェック当たりのネットワーク I/O の平均

サンプル間隔の間にすべての Adaptive Server エンジンが行うチェック 1 回当たりのネットワーク I/O (送信と受信) についての平均回数。

sp_configure パラメータの i/o polling process count では、スケジューラがディスクやネットワークの I/O が完了したかどうかをチェックする前に、Adaptive Server が実行するプロセスの最大数を指定します。i/o polling process count をチューニングすると、Adaptive Server の応答時間とスループットの両方に影響します。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

Adaptive Server エンジンによるネットワーク I/O のチェックは頻繁に行われているが、検索頻度が低い場合は、ネットワーク I/O チェックの頻度を下げてみてください。

ディスク I/O チェック

ディスク I/O チェックの総数と I/O を返したチェック数をレポートします。

```
ディスク I/O チェック
Total Disk I/O Checks        171839.7        4233.0    103103833       n/a
Checks Returning I/O          31783.1         782.9     19069887    18.5 %
Avg Disk I/Os Returned            n/a           n/a      0.01722       n/a
```

- `Checks Returning I/O` ― 実際のチェック数。
- `Avg Disk I/Os Returned` ― 1 つ以上の完了した IO を生成した `Checks Returning I/O` のパーセンテージ。

ディスク I/O チェックの総数

エンジンがルーチンを入力してディスク I/O をチェックした回数。io polling process count 設定パラメータはこのチェックの頻度を制御します。

タスクが I/O を実行する必要がある場合は、そのタスクを実行している Adaptive Server エンジンがただちに I/O 要求を発行し、I/O の完了を待機する間はタスクをスリープ状態にします。エンジンは別のタスクを処理する場合もありますが、I/O の完了も引き続きチェックします。エンジンは、完了した I/O を見つけると、タスクをスリープ・キューから実行キューへ移動します。

I/O を返したチェック

I/O がディスク・チェック・ルーチンの入力時に未処理であった回数。

Adaptive Server エンジンは "`io polling process count`" タスクの値がサーバの 1 チック内または "`idle loops`" 中に実行された場合に、ネットワーク I/O をポーリングします。ただし、保留中のディスク I/O が少なくとも 1 つない限り、エンジンはディスク I/O をポーリングしません。"`Checks Returning IO`" では、エンジンがディスク I/O をチェックする回数と Adaptive Server によるチェックで保留中のディスク I/O が少なくとも 1 つあった回数が示されます。

たとえば、あるエンジンが予期される I/O を 100,000 回チェックすると、この平均値では、実際に I/O が実行された回数のパーセンテージが示されます。100,000 回のチェックのうち、I/O が 10,000 回完了された場合は、10% のチェックが実際に有効であり、残りの 90% はオーバヘッドです。

ただし、1 回のチェック当たりに返された I/O の平均値、およびサンプル間隔中にエンジンがどの程度ビジーだったかもチェックしてください。サンプルにアイドル時間が含まれていたり、I/O トラフィックが散発的である場合は、チェックがビジー時間内に I/O を返す割合が高くなっている可能性があります。

このカテゴリの結果が低すぎるか高すぎると思われる場合は、i/o polling process count を設定して、チェックの頻度を増減できます。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

返されたディスク I/O の平均

1 つ以上の完了した IO を生成した“Checks Returning I/O”のパーセンテージ。

Adaptive Server エンジンが各チェックの間で待機する時間を増やすと、スループットが向上します。これは、I/O のチェックに費やす時間が短くなると、Adaptive Server エンジンが処理を行うための時間が長くなるからです。ただし、使用している環境でこれを確認する必要があります。I/O Busy% の値が高い場合は、頻繁にチェックを行い、I/O を完了と同時に取得する必要があります。i/o polling process count 設定パラメータは、Adaptive Server がこのチェックを実行する頻度を制御します。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

ワーカー・プロセス管理

付与されたワーカー・プロセスと拒否されたワーカー・プロセスの数、およびワーカー・プロセスに対するメモリ要求が成功した数と失敗した数など、ワーカー・プロセスの使用をレポートします。

この出力を「並列クエリ管理」(40 ページ)の出力を使用して分析します。

出力例

```
Worker Process Management
-------------------------
                                per sec      per xact     count    % of total
                            ------------  ------------  --------    ---------
Worker Process Requests
Requests Granted                    0.1           0.4        23      100.0 %
Requests Denied                     0.0           0.0         0        0.0 %
------------------------    ------------  ------------    ------
Total Requests                      0.1           0.4        23

Requests Terminated                 0.0           0.0         0        0.0 %

Worker Process Usage
Total Used                          0.2           1.1        69        n/a
Max Ever Used During Sample         0.0           0.1         9        n/a

Memory Requests for Worker Processes
Succeeded                           2.2          10.1       646      100.0 %
Failed                              0.0           0.0         0        0.0 %
------------------------     -----------  ------------   -------
Total Requests                      2.2          10.1       646

Avg Mem Ever Used by a WP (in bytes)  n/a           n/a     1208.9        n/a
```

ワーカー・プロセス要求

ワーカー・プロセスに対する要求とワーカー・プロセスのメモリをレポートします。並列クエリは、ワーカー・プロセスに対して複数の要求を行います。たとえば、ソートが必要な並列クエリの場合、1 番目はデータにアクセスするための要求で、2 番目は並列ソートの要求になります。

“Requests Granted”と“Requests Denied”のローには、ワーカー・プロセスが付与された要求の数と、実行時に利用できるワーカー・プロセスが不足していたために拒否された要求の数が示されます。

ワーカー・プロセス数に対して行われた調整の数を調べるには、「並列クエリの使用率」(41 ページ) を参照してください。

“Requests Terminated”では、クエリをキャンセルする [Ctrl + c] キーなどのユーザ・アクションによって停止された要求の回数がレポートされます。

ワーカー・プロセスの使用率

サンプル間隔中に使用されたワーカー・プロセスの総数をレポートします。“Max Ever Used During Sample”では、sp_sysmon のサンプリング時間中、いずれかの時点で使用されたワーカー・プロセスの最大数がレポートされます。“Max Ever Used During Sample”を使用して設定パラメータ number of worker processes を設定できます。

ワーカー・プロセスのメモリ要求

ワーカー・プロセスのメモリ割り付けのために行われた要求の数、これらの要求が成功した数、および失敗した数。ワーカー・プロセスのメモリはパラメータ memory per worker process を使用して設定されたメモリ・プールから割り付けられます。

“Failed”の値が 0 以外の場合は、memory per worker process の値を増やす必要があります。

ワーカー・プロセスによって使用されたメモリの平均

サンプル間隔のいずれかの時点での、すべてのアクティブなワーカー・プロセスで使用されたメモリの最大値の平均をレポートします。各ワーカー・プロセスには、主に調整メッセージをやりとりするためのメモリが必要です。このメモリは、Adaptive Server によって、グローバル・メモリ・プールから割り付けられます。

このメモリ・プールのサイズは、2 つの設定パラメータ number of worker processes と memory per worker process を乗算したものです。

number of worker processes を 50 に設定し、memory per worker process をデフォルト値である 1024 バイトに設定すると、50K のメモリ・プールを利用できます。memory for worker process を 2048 バイトに増やすと、さらに 50K のメモリが必要になります。

起動時には、各ワーカー・プロセスのために静的な構造体が作成されます。ワーカー・プロセスが使用中の間、メモリの割り付けは、必要になると追加され、不要になると解除されます。出力される平均値は、すべてのワーカー・プロセスに対して静的および動的に割り付けられたすべてのメモリを、サンプル間隔中に実際に使用されたワーカー・プロセスの数で除算した平均値です。

数多くのワーカー・プロセスが設定されたにもかかわらず、サンプル間隔中に少ししか使用されない場合は、使用されないプロセスのための静的メモリも平均値の計算対象になるので、実際よりも大きな値が出力されます。

“Avg Mem”が memory per worker process によって設定された値に近く、“Max Ever Used During Sample”のワーカー・プロセス数が設定された数に近い場合は、パラメータの値を増やす必要があります。

ワーカー・プロセスがメモリ・プールからのメモリを必要としたときに利用できるメモリがない場合、プロセスはエラー・メッセージを出力して終了します。

注意 ほとんどの並列クエリ処理では、デフォルト値の 1024 で十分です。

ただし、dbcc checkstorage は例外です。dbcc checkstorage では、設定されているワーカー・プロセスが 1 つだけである場合、1792 バイトまで使用する可能性があります。dbcc checkstorage を使用する場合に number of worker processes を 1 に設定してあるときは、memory per worker process を増やす必要があります。

並列クエリ管理

並列クエリの実行をレポートします。並列クエリの総数、ワーカー・プロセス数が実行時に調整された回数、およびマージやソート中に付与されたロックがレポートされます。

出力例

```
Parallel Query Management
-------------------------

Parallel Query Usage            per sec     per xact      count   % of total
------------------------   ------------   ----------  ---------   ----------
Total Parallel Queries              0.1          0.3         19          n/a
WP Adjustments Made
Due to WP Limit                     0.0          0.0          0        0.0 %
Due to No WPs                       0.0          0.0          0        0.0 %

Merge Lock Requests             per sec     per xact      count   % of total
------------------------   ------------   ----------  ---------    ----------
Network Buffer Merge Locks
Granted with no wait                1.1          5.0        320       21.3 %
Granted after wait                  3.9         18.4       1179       78.7 %

Result Buffer Merge Locks
Granted with no wait                0.0          0.0          0        0.0 %
Granted after wait                  0.0          0.0          0        0.0 %

Work Table Merge Locks
    Granted with no wait            0.0          0.0          0        0.0 %
    Granted after wait              0.0          0.0          0        0.0 %
------------------------   ------------   ----------  ---------    ----------
Total # of Requests                 5.0         23.4       1499
```

```
Sort Buffer Waits              per sec     per xact      count   % of total
-----------------------   ------------   ----------  ---------   ----------
Producer Waits                     0.0          0.1          8      100.0 %
Consumer Waits                     0.0          0.0          0        0.0 %
-----------------------   ------------   ----------  ---------    ----------
Total # of Waits                   0.0          0.1          8
```

並列クエリの使用率

並列実行に適したクエリの総数をレポートします。オプティマイザが最善のプランを決定し、クエリを逐次実行するか並列実行するかということと、並列クエリで使用するワーカー・プロセスの数を決定します。

“WP Adjustments Made”では、オプティマイザによって推奨されるワーカー・プロセス数が、実行時に調整されなければならなかった回数がレポートされます。可能性のある原因として、次の 2 つがレポートされます。

- “Due to WP Limit” － set parallel_degree または set scan_parallel_degree で設定されたセッションレベルの制限値のために、キャッシュされたクエリ・プランのワーカー・プロセス数が調整された回数が示される。

  “Due to WP Limit”が 0 以外の値の場合は、セッションレベルの制限値を設定するアプリケーションを調べてください。

- “Due to No WPs” － 使用できるワーカー・プロセスが不足していたために、ワーカー・プロセスの数が減らされた要求の数が示される。これらのクエリは、逐次実行することもあれば、オプティマイザが推奨する数よりも少ないワーカー・プロセスを使用して並列実行することもある。このことは、適切に最適化されていないプランを使用してクエリが実行していることを意味する。

  “Due to No WPs”が 0 以外の値であり、サンプリング時間中のシステムの負荷が一般的な量のときには、number of worker processes 設定パラメータを増やすか、いくつかのクエリでセッションレベルの制限値を設定する必要があります。

  調整されたプランを使用するログインの fid (ファミリ ID) で sp_showplan を実行すると、キャッシュされたプランだけが示され、調整されたプランは示されません。

  ログインが調整されたプランを実行している場合、sp_who では、sp_showplan の結果によって示される数とは異なる fid のワーカー・プロセス数が表示されます。

マージ・ロック要求

並列マージ・ロックが要求された数、すぐに付与された数、マージの各種類について待機しなければならなかった要求の数をレポートします。マージには、次の 3 つの種類があります。

- “Network Buffer Merge Lock” − クライアントに結果を返すネットワーク・バッファの競合がレポートされる。
- “Result Buffer Merge Locks” − グループ化されていない集約の結果と、ソートも集約もされていない変数割り当ての結果の処理に使用される結果バッファの競合がレポートされる。
- “Work Table Merge Locks” − ワーク・テーブルからの結果をマージしている間に発生したロックの競合がレポートされる。

“Total # of Requests” では、3 種類のマージ要求の総数がレポートされます。

ソート・バッファの待機

並列ソートに使用されるソート・バッファの競合をレポートします。並列ソート・バッファは次のワーカー・プロセスによって使用されます。

- プロデューサ − 並列スキャンからのローを返すワーカー・プロセス。
- コンシューマ − 並列ソートを実行するワーカー・プロセス。

待機の値が大きい場合は、number of sort buffers の値を大きく設定できます。

ガイドラインについては『パフォーマンス&チューニング・シリーズ：クエリ処理と抽象プラン』の「第 10 章 パフォーマンス改善のための統計値の使用」を参照してください。

タスク管理

オープンされている接続、エンジンごとのタスク・コンテキストの切り替え、原因ごとのタスク・コンテキストの切り替えについての情報を出力します。`Task Management` は、タスクが実行されなくなった理由を明らかにします (つまり、レポートにはリソース不足と競合が示されます)。

出力例

```
Task Management               per sec      per xact      count   % of total
--------------------------  ------------   ----------   --------  ----------

Connections Opened                   0.2           0.0        154         n/a
```

```
Task Context Switches by Engine
Engine 0                          269.9           3.4     240477       7.3 %
Engine 1                          209.4           2.7     186574       5.7 %
Engine 2                          198.4           2.5     176730       5.4 %
Engine 3                          198.5           2.5     176859       5.4 %
Engine 4                          278.4           3.5     248054       7.5 %
Engine 5                          187.1           2.4     166697       5.1 %
Engine 6                          202.7           2.6     180640       5.5 %
Engine 7                          312.2           4.0     278129       8.5 %
Engine 8                          215.7           2.7     192198       5.8 %
Engine 9                          260.3           3.3     231940       7.1 %
Engine 10                         235.1           3.0     209447       6.4 %
Engine 11                         409.2           5.2     364604      11.1 %
Engine 12                         225.8           2.9     201166       6.1 %
Engine 13                         484.9           6.1     432020      13.1 %
-------------------------   ------------  ------------  ----------
Total Task Switches:             3687.5          46.7    3285535

Task Context Switches Due To:
Voluntary Yields                  860.8          10.9     766929      23.3 %
Cache Search Misses               852.0          10.8     759122      23.1 %
System Disk Writes                 12.7           0.2      11353       0.3 %
Exceeding I/O batch size          184.7           2.3     164550       5.0 %
Logical Lock Contention             0.3           0.0        258       0.0 %
Address Lock Contention             0.0           0.0         12       0.0 %
Latch Contention                    0.7           0.0        587       0.0 %
Log Semaphore Contention           31.1           0.4      27737       0.8 %
PLC Lock Contention                 0.6           0.0        577       0.0 %
Group Commit Sleeps                 6.9           0.1       6122       0.2 %
Last Log Page Writes               71.4           0.9      63619       1.9 %
Modify Conflicts                   19.9           0.3      17728       0.5 %
I/O Device Contention               0.0           0.0          0       0.0 %
Network Packet Received           152.1           1.9     135483       4.1 %
Network Packet Sent               643.7           8.1     573528      17.5 %
Network services                  127.8           2.3       7564       8.9 %
Other Causes                       850.7          10.8     757930      23.1%
```

オープンされた接続

Adaptive Server に対してオープンされた接続数。ここには、クライアント接続やリモート・プロシージャ・コールなどすべての種類の接続が含まれます。カウントされる接続は、サンプル間隔の間に開始された接続だけです。サンプル間隔が開始される前に確立された接続は、アクティブでリソースを使用していても、カウントされません。

“Connections Opened”によって、Adaptive Server 環境とサンプル間隔中の作業負荷についての概要が提供されます。さらに、このデータをアプリケーションの動作を理解するために役立てることもできます。つまり、アプリケーションが接続のオープンとクローズを繰り返しているか、1 回の接続で複数のトランザクションを実行しているかを調べる場合に、このデータが役立ちます。

コミットされたトランザクションの詳細については、「トランザクション・プロファイル」(62 ページ) を参照してください。

エンジンごとのタスク・コンテキスト切り替え

Adaptive Server で実行中の各プロセスには独自の「コンテキスト」または環境があり、現在のデータベース、クライアントの文字セット、現在実行されているクエリなどの情報が含まれています。「コンテキスト切り替え」は、Adaptive Server が 1 つのクライアントまたはシステム・プロセスを実行しているが、別のクライアントまたはシステム・プロセスに切り替える必要がある場合に発生します。この切り替えでは、エンジンが実行しているコンテキストを変更する必要があります。

“Task Context Switches by Engine”では、各 Adaptive Server エンジンがあるユーザ・タスクから別のユーザ・タスクへコンテキストを切り替えた回数がレポートされます。“% of total”では、すべての Adaptive Server エンジンのタスク切り替えの合計回数に対して、各 Adaptive Server エンジンによるタスク切り替え回数の割合が、パーセンテージでレポートされます。

“Total Task Switches”では、SMP サーバ上のすべてのエンジンのタスク切り替えについて、アクティビティが要約されます。“Total Task Switches”を使用して、再設定の効果を観察できます。キャッシュ検索ミスでタスクがブロックされたり、タスクが頻繁に切り替えられているように思われる場合は、キャッシュを再設定したりメモリを追加したりできます。次に、データを調べて、タスクが切り替えられる頻度が高くなったか低くなったかの傾向を確認します。

タスク・コンテキスト切り替えの原因

Adaptive Server がいくつかの一般的な理由でコンテキストを切り替えた回数。“% of total”では、すべての Adaptive Server エンジンのタスク・コンテキスト切り替え回数の合計に対する特定の理由によるコンテキスト切り替え回数の割合が、パーセンテージでレポートされます。

“Task Context Switches Due To”では、タスクがエンジンからスイッチ・オフされた原因の概要が提供されます。このセクションで示される、発生の可能性があるパフォーマンス上の問題は、このセクションの下に示される原因の説明など、そのほかの sp_sysmon からの出力をチェックすることで調査できます。

たとえば、ほとんどのタスク切り替えの原因が物理 I/O である場合は、メモリを追加したりキャッシュを再設定したりして物理 I/O を最小にします。ただし、ロック競合がほとんどのタスク切り替えの原因である場合は、ロックについてのレポートのセクションをチェックします。

詳細については、「ロック管理」(92 ページ) を参照してください。

自発的解放

タスクが解放されずに完了した回数。解放を完了したタスクは、`Network Packet Sent` カラムで示されます。自発的解放の数値が大きい場合、競合はほとんど発生していません。

設定パラメータ time slice では、プロセスを実行できる時間の長さが設定されます。CPU 集約タスクは、コード内の特定の「解放ポイント」で CPU を解放する以外の原因によっては、ほかのプロセスに CPU を使用させるための切り替えを行いません。タスクが割り当てられた時間を使い切っているかどうかを特定するには、「割り当てられたスライスの不足」(56 ページ) を参照してください。

詳細については、『パフォーマンス&チューニング・シリーズ：基本』の「エンジンと CPU の使用方法」を参照してください。

キャッシュ検索ミス

必要なページがキャッシュ内に見つからず、それをディスクから読み込むためにタスクを切り替えた回数。データ・ページとインデックス・ページの場合、タスクは物理読み込みの実行中に切り替えられます。

sp_sysmon の出力のキャッシュに関連する部分の詳細については、「データ・キャッシュ管理」(102 ページ) を参照してください。

I/O バッチ・サイズの超過

Adaptive Server が I/O バッチ制限値を超えたために I/O 集約タスクがエンジンによって切り替えられた回数。ただし、“Exceeding I/O batch size”では、ログを取らない I/O バッチからのコンテキストの切り替えのみがレポートされます。Adaptive Server は、大量の I/O を実行する操作の間に、ディスク I/O サブシステムがあふれないようにするためにディスク書き込みを調整します。たとえば、チェックポイント・タスクは大量のデータ・ページをディスクに書き込みます。Adaptive Server では、チェックポイント・タスクは切り替えられ、書き込みのバッチが完了するまでスリープ状態になってから、ウェイクアップして別のバッチを実行します。

i/o batch size パラメータを使用してバッチ・サイズ (ログを取らない I/O とログ I/O の両方) を設定します。

デフォルトでは、バッチ当たりの書き込みの数は 100 に設定されています。次の場合は、バッチ当たりの書き込みの数を増やす必要があります。

- 環境のスループットが高く、大きなデータ・キャッシュを使った高度なトランザクション環境の場合。
- システムが I/O にバインドされていない場合。

システム・ディスクへの書き込み

次の原因でタスクが切り替えられた回数。

- ログ I/O バッチ・サイズ (10 ページ) を超過していた
- ディスク書き込みを実行する必要があった
- ハウス・キーピング・プロセスやチェックポイント・プロセスなどの別のプロセスによって書き込み中のページにアクセスする必要があった

Adaptive Server の書き込みは、ほとんどが非同期で発生しますが、ページ分割の書き込み、リカバリ、OAM ページの書き込みの間は、プロセスがスリープ状態になります。

“System Disk Writes”の値が大きい場合は、ページ分割の値をチェックし、データ・ページやインデックス・ページの分割によって問題が発生していないかどうかを調べます。

「ページ分割」(82 ページ) を参照してください。

システム・ディスク書き込みの値が、ページ分割が原因で大きくなったのではない場合、Adaptive Server をチューニングしてこの値を変更することはできません。

論理ロック競合

テーブル、データ・ページ、またはデータ・ローでのロックの競合のために、タスクが切り替えられた回数。

トランザクションの詳細とロック管理に関するレポートのセクションをチェックし、ロック競合の問題を調査してください。

- 詳細については、「トランザクションの詳細」(65 ページ) と「ロック管理」(92 ページ) を参照。
- クエリが遅延更新や高負荷の直接更新を行っていないかどうか調べる。これらの更新は、余分なインデックス・ロックを引き起こす可能性がある。「更新」(67 ページ) を参照。
- sp_object_stats を使用して、オブジェクトごとに情報をレポートする。

  詳細については、『パフォーマンス＆チューニング・シリーズ：ロック』の「第 3 章 ロックのレポート」を参照してください。

そのほかにも、次の項目を参照すると、ロックとロック競合に役立ちます。

- サーバ・レベルまたはクエリ・レベルで使用するロックの種類については、『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「ロックの種類」を参照。
- ロック競合の減少については、『パフォーマンス&チューニング・シリーズ：ロック』の「第 2 章 ロックの設定とチューニング」を参照。
- インデックスとクエリのチューニングについては、『パフォーマンス&チューニング・シリーズ：ロック』の「第 1 章 ロックの概念」を参照。特に、インデックスを使用して、更新や削除によるテーブル・スキャンや排他テーブル・ロックを発生させない方法についての説明がある。

アドレス・ロック競合

アドレス・ロックによってタスクが切り替えられた回数。Adaptive Server は、全ページロック・テーブルのインデックス・ページに対するアドレス・ロックを取得します。これらのロックによって、データ・ページへのアクセスがブロックされます。たとえば、インデックス・ページの排他アドレス・ロックにより、別のシステム・プロセス ID (spid) がそのページを読み取ったり、データに対するそのインデックスを使用できないようにします。ただし、タスクは別のアクセス方法でデータにアクセスすることもできます。また、リーフ・レベルのインデックス・ページが排他的にロックされている場合は、ロックされたページ上のインデックス・ローによって示されるデータに変更を加えることはできません。これは、インデックス・ローにも変更が必要である可能性があるためです。

ラッチ競合

タスクがラッチを待機する必要があったためにタスクが切り替えられた回数。

ユーザ・テーブルで全ページ・ロックだけが使用されている場合、ラッチ競合は、データオンリーロック・システム・テーブルかアロケーション・ページのどちらかで発生します。

アプリケーションでデータオンリー・ロックが使用されている場合、ここでレポートされる競合には、インデックス・ページ、OAM ページ、アロケーション・ページでのラッチの待機を含む、すべてのラッチの待機が含まれます。

ページ割り付け中の競合の削減

挿入とローの長さを増加させる更新が非常に多い SMP 環境では、ページ割り付けが頻繁に発生するため、アロケーション・ページ・ラッチの競合によってパフォーマンスが低下することがあります。通常、Adaptive Server は、すでにオブジェクトが使用していて、空き領域があることがわかっているアロケーション・ユニットのオブジェクトに新しいページを割り付けます。

各オブジェクトについて、Adaptive Server は、そのオブジェクトにページを割り付けるのに必要なタスクのためのヒントとして、このアロケーション・ページ番号を追跡します。1 回に複数のタスクが同じアロケーション・ユニットでページを割り付ける必要がある場合、2 番目以降のタスクはそのアロケーション・ページでラッチをブロックします。

「貪欲な割り付け」スキームを指定すると、Adaptive Server はテーブルに対するページ割り付けのための 8 つのアロケーション・ヒントのリストを保持できます。

次のコマンドを使用すると、データベース 6 の salesdetail テーブルに対して貪欲な割り付けを有効にできます。

```
dbcc tune(des_greedyalloc, 6, salesdetail, "on")
```

貪欲な割り付けを無効にするには、次のように指定します。

```
dbcc tune(des_greedyalloc, 6, salesdetail, "off")
```

dbcc tune(des_greedyalloc) の効果は永続的なものではないので、Adaptive Server の再起動後にはこのコマンドを再発行する必要があります。

dbcc tune(des_greedyalloc) コマンドは、次の条件がすべて満たされている場合のみ使用してください。

- 複数のエンジンを持っている場合。エンジンが 4 未満の場合はほとんど役に立たない。
- 大量のページがオブジェクトに割り付けられている場合。sp_spaceused または optdiag を使用して、ページ数を追跡できる。
- ラッチ競合カウンタが競合を示している場合。

テーブルに専用のセグメントが割当てられている場合、貪欲な割り付けはさらに役立ちます。同じセグメントの複数のテーブルに対して貪欲な割り付けを有効にすると、複数のテーブルで同じアロケーション・ヒントが使用されることがあります。

貪欲な割り付けは、master データベースと tempdb データベースでは許可されません。また、システム・テーブルでも許可されません。貪欲なページ割り付けは、分割されたテーブルには適用できません。

ヒントは、空き領域が存在する可能性があるアロケーション・ページです。ヒントの最大保持数は 16 です。

物理ロック、論理ロック、およびオブジェクト・ロックの遷移

- 物理ロックの遷移 – 物理ロックの取得に関連するコンテキストの切り替えを特定する。
- 論理ロックの遷移 – クラスタ論理ロックの取得に関連するコンテキストの切り替えを特定する。

- オブジェクト・ロックの遷移 ― オブジェクト・ロックの取得に関連するコンテキストの切り替えを特定する。

これらの 3 つのローでは、`% total` はクラスタ・メッセージングによるコンテキストの切り替え回数の合計を表します。このコンテキストの `% total` の値が大きい場合、クラスタ・メッセージ・アクティビティは活発です。この数値を `Number of Cluster Lock Requests` と比較して、ロック要求あたりのクラスタ・メッセージ数を特定します。コンテキストの切り替えの詳細については、monSysWaits モニタリング・テーブルにクエリしてください。

ロック要求によって作成されるコンテキストの切り替え数が多すぎる場合は、アプリケーション分割を評価します。コンテキストの切り替え回数が多い場合は、クラスタ・メッセージ・バッファに追加の要件があります。CIPC メッセージ・バッファが不足している場合は、monCIPC で ReceiveCount と TransmitCount の値が大きい可能性があります。この問題を解決するには、CIPC メッセージ・バッファのサイズを増やします。

ログ・セマフォ競合

セマフォは、2 番目のタスクが現在使用されているデータ構造体にアクセスするのを防ぐ、簡単な内部ロック・メカニズムです。Adaptive Server では、複数のプロセスが ULC のレコードにアクセスし、強制的にフラッシュできるので、セマフォを使用してユーザ・ログ・キャッシュが保護されます。

“Log Semaphore Contention”では、別のタスクが保持しているトランザクション・ログ・セマフォの取得を必要としたために、タスクが切り替えられた回数がレポートされます。これは、SMP システムだけに適用されます。

ログ・セマフォ競合の値が大きい場合は、「トランザクション管理」(69 ページ)を参照してください。

トランザクション・ログによって使用されるディスクでのディスク・キューイングをチェックしてください。「ディスク I/O 管理」(125 ページ) を参照してください。

さらに、「エンジンの使用率 (チック %)」(23 ページ) も参照してください。レポートされているエンジン使用率の値が低く、応答時間が容認できる範囲内である場合は、エンジンの数を減らすことを検討してください。実行するエンジンの数を減らすと、同時にログにアクセスしようとするタスクの数が減るので、競合が減ります。

ログ・セマフォ競合が多い場合は、通常、ロギングまたは I/O サブシステムに問題があることを示しています。ただし、高スループットを使用する OLTP 環境では、細かくチューニングされたシステムでもログ・セマフォで競合が多く発生する場合があります。

PLC ロック競合

ユーザ・ログ・キャッシュでのロックに対する競合をレポートします。

グループ・コミット・スリープ

トランザクションのコミットが実行され、ログが別のタスクによってディスクに書き込まれるまでスリープ状態になったタスクの数。

あるタスクがコミットすると、そのタスクのログ・レコードは、ユーザ・ログ・キャッシュからキャッシュ内のトランザクション・ログの現在のページへフラッシュされます。あるページ (ログ I/O のサイズが大きく設定されている場合は複数のページ) が満杯でない場合は、タスクが切り替えられて実行キューの最後に置かれます。次の場合に、ページに対するログ書き込みが実行されます。

- 別のプロセスがログ・ページへの書き込みを行い、ログをフラッシュした場合。これは、グループ・コミット・スリープまたは I/O の速度調整として表される。
- タスクが実行キューの先頭に到達し、ログ・ページをフラッシュするプロセスがほかにない場合。これは、最後のログ・ページ書き込みとして表される。

スループットの高い環境では、ログ I/O のサイズを大きくすると、ログ・デバイスでのディスク・キューイングの問題を回避することができます。“Group Commit Sleeps”のパーセンテージが大きくても問題とはみなされません。

最後のログ・ページ書き込み

最後のログ・ページへの書き込み中にスリープ状態になったためにタスクが切り替えられた回数。

「グループ・コミット・スリープ」(50 ページ) で説明したように、ほかのタスクがログ・ページに書き込んでいるのを待っている間は、スリープしているのとは対照的に、上記のタスクは最後のログ・ページへの書き込みをしなければならないために切り替えられます。

この値が大きい場合は、「ログ・ページ当たりの平均書き込み数」(79 ページ) を参照して、Adaptive Server が同一の最終ログ・ページに繰り返し書き込んでいるかどうかを調べてください。ログ I/O のサイズが 2K よりも大きい場合は、ログ I/O のサイズを小さくすると不要なログ書き込みの数を減らすことができます。

詳細については、『Optimizing Transaction Performance in Adaptive Server Enterprise 12.5』ホワイト・ペーパー (http://www.sybase.com) を参照してください (Sybase ホーム・ページの検索フィールドに“Optimizing Transaction Performance”と入力します)。

変更競合

タスクが、特殊なライトウェイト保護メカニズムのもとで、別のタスクが保持しているページへの排他アクセスを取得しようとした回数。特定の操作では、Adaptive Server はライトウェイト保護メカニズムを使って、実際のページ・ロックを使用せずにページへの排他アクセスを取得します (たとえば、いくつかのシステム・テーブルへのアクセスやダーティ・リードがあります)。これらのプロセスは、ページを変更しない場合でも、ページへの排他アクセスを必要とします。

I/O デバイス競合

特定のデバイスのためのセマフォの待機中にタスクがスリープ状態になった回数。

タスクが物理 I/O を実行する必要がある場合、Adaptive Server は、I/O 構造体を満杯にして、I/O 構造体をエンジンごとの I/O キューにリンクします。2 つの Adaptive Server エンジンが、同じデバイスから I/O 構造体を同時に要求した場合は、エンジンの 1 つはセマフォを待機している間スリープします。

デバイス競合は、ディスク・ミラーリングを有効にした場合 (デフォルトでは「無効」) または I/O が遅延される場合にのみ発生します。遅延 I/O の詳細については、「ディスク I/O 管理」(125 ページ) を参照してください。

I/O デバイスのセマフォに対する競合が多く発生している場合は、テーブルを複数のデバイスに再分配したり、デバイスを追加してテーブルやインデックスをそのデバイスへ移動したりすることで競合を減らしてください。

受信されたネットワーク・パケット

タスク切り替えが“Network Packet Received”によってレポートされた場合、それは次のいずれかに起因しています。

- タスクがマルチ・パケットのバッチの一部を受信し、クライアントからバッチの次のパケットが送信されるまで待機するために切り替えられた。
- タスクが現在のバッチの処理を完全に終了し、クライアントからの次のコマンドやパケットの受信を待機しながら、受信スリープ状態になった。これは、“Network Packet Received”によってタスク・コンテキスト切り替えが発生する最も一般的な原因である。

“Network Packet Received”の値が大きい場合は、「ネットワーク I/O 管理」(131 ページ) を参照してください。すべての接続、または特定の接続がより大きなパケット・サイズを使用してログインできるように、ネットワーク・パケット・サイズを設定できます。

ネットワーク・パケット・サイズの詳細については、『パフォーマンス&チューニング・シリーズ：基本』の「第 1 章 基本の概要」を参照してください。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

送信されたネットワーク・パケット

ネットワークが各パケットをクライアントへ送信するのを待機しながら、タスクが送信スリープ状態になった回数。ネットワーク・モデルでは、1 つの接続について未処理のまま残っているパケットは、どの時点でも 1 つだけと決まっています。タスクでは、タスク・コンテキスト切り替えを回避するように、ネットワーク I/O タスクへの送信要求を通知せずにネットワーク・パケットを送信することができます。

多くの送信データがあり、タスクが小さなパケット (1 パケット当たり 512 バイト) を数多く送信している場合、そのタスクは結局何度もスリープすることになります。データのパケット・サイズは設定可能であり、クライアントごとに異なるパケット・サイズを要求できます。クエリが完了すると、この動作は `Network Packet Send` カラムに含まれます。

ネットワーク・パケットの詳細については、『パフォーマンス&チューニング・シリーズ：基本』の「第 2 章 ネットワークとパフォーマンス」を参照してください。

“Network Packet Sent” がタスク切り替えの主な原因である場合は、「ネットワーク I/O 管理」(131 ページ) を参照してください。

ネットワーク・サービス

(send と receive オペレーション以外の) ネットワーク・オペレーション上でスリープすることが原因でコンテキストを切り替えた回数を計測します。

そのほかの原因

上記以外の原因で切り替えられたタスクの数 (その値は、“Network Services” による待機の数だけ減ります)。うまくチューニングされたサーバでは、チューニング可能なタスク切り替えの原因が減るにつれて、この値が高くなります。

アプリケーション管理

ユーザ・タスクの実行情報をレポートします。このセクションは、リソース制限を使用する場合や、実行属性を設定したりエンジンとの結び付きを割り当てたりしてアプリケーションをチューニングする場合に役立ちます。アプリケーション、ログイン、ストアド・プロシージャに対する調整を行う前に、一般的な負荷がかかる時間帯に sp_sysmon を実行し、このセクションの情報についてよく理解してください。

関連する背景情報については、『パフォーマンス&チューニング・シリーズ：基本』を参照してください。

詳細なアプリケーション情報の要求

3 番目の sp_sysmon パラメータ (*applmon*) を使用して特定のタスクについての情報を要求すると、sp_sysmon からの出力によって、概要情報に加えて各アプリケーション固有の情報が個別に出力されます。次の 2 つのうちいずれかの方法で、詳細なアプリケーション情報を表示できます。

- アプリケーションとログインの情報 (sp_sysmon パラメータの appl_and_login を使用) － sp_sysmon は、実行中のログインとアプリケーションごとにセクションを分けて出力する。
- アプリケーション情報だけ (sp_sysmon パラメータの appl_only を使用) － sp_sysmon は、アプリケーションごとにセクションを分けて出力する。このセクションでは、アプリケーションを実行中のすべてのログインのデータが組み合わされている。

たとえば、10 人のユーザが isql を使ってログインし、5 人のユーザが sales_reports というアプリケーションを使ってログインしている場合、「アプリケーションとログインの情報」を要求すると、15 個の詳細なセクションが出力されます。「アプリケーション情報だけ」を要求すると、2 つの詳細なセクションが出力されます。1 つは isql ユーザのアクティビティに関する要約情報、もう 1 つは sales_reports ユーザのアクティビティに関する要約情報です。

appl_and_login は、sp_sysmon のすべてのセクションで使用できます。次に構文の例を示します。

```
sp_sysmon “00:05:00”, @applmon=appl_and_login
```

「アプリケーション詳細パラメータの指定」(6 ページ) を参照してください。

出力例

```
Application Management
----------------------

Application Statistics Summary (All Applications)
-------------------------------------------------
Priority Changes                 per sec      per xact      count   % of total
-------------------------  ------------   ----------   --------  ----------
To High Priority                     3.4           0.1       2058       18.6 %
To Medium Priority                   9.6           0.2       5774       52.3 %
To Low Priority                      5.4           0.1       3217       29.1 %
-----------------------   ------------   ----------   --------
Total Priority Changes              18.4           0.5      11049

Allotted Slices Exhausted        per sec       per xact      count   % of total
-------------------------  -----------    ----------    -------  ----------
High Priority                        0.0           0.0          0       0.0 %
Medium Priority                     13.8           0.3       8251     100.0 %
Low Priority                         0.0           0.0          0       0.0 %
-------------------------  -----------    ----------   --------  ----------
Total Slices Exhausted              13.8           0.3       8251

Skipped Tasks By Engine          per sec       per xact      count  % of total
-------------------------  -----------    ----------   --------  ----------
Total Engine Skips                   0.0           0.0          0         n/a

Engine Scope Changes                 0.0           0.0          0         n/a
```

次の例は、アプリケーションとログインの出力を示します。1 つのアプリケーションとログインについての情報だけが含まれています。最初の行は、アプリケーション名 (矢印の前) とログイン名 (矢印のあと) を示しています。

```
-------------------------------------------------------------------------
  Application->Login:   ctisql->adonis

  Application Activity           per sec      per xact      count   % of total
-------------------------  ------------   ----------   --------  ----------
CPU Busy                             0.1           0.0         27       2.8 %
    I/O Busy                         1.3           0.1        461      47.3 %
    Idle                             1.4           0.2        486      49.9 %

Number of Times Scheduled            1.7           0.2        597         n/a

 Application Priority Changes     per sec      per xact      count   % of total
-------------------------  ------------   ----------   --------  ----------
To High Priority                     0.2           0.0         72       50.0 %
    To Medium Priority               0.2           0.0         72       50.0 %
    To Low Priority                  0.0           0.0          0        0.0 %
-------------------------  ------------   ----------   --------  ----------
Total Priority Changes               0.4           0.0        144
```

```
  Application I/Os Completed     per sec      per xact      count   % of total
--------------------------  ------------    ----------   --------  ----------
Disk I/Os Completed                 0.6           0.1        220       53.9 %
    Network I/Os Completed          0.5           0.1        188       46.1 %
--------------------------   ------------       -------   --------
    Total I/Os Completed            1.1           0.1        408

 Resource Limits Violated        per sec      per xact      count   % of total
--------------------------  ------------    ----------   --------  ----------
IO Limit Violations
    Estimated                       0.0           0.0           0        0.0 %
    Actual                          0.1           4.0           4       50.0 %
  Time Limit Violations
    Batch                           0.0           0.0           0        0.0 %
    Xact                            0.0           0.0           0        0.0 %
  RowCount Limit Violations         0.1           4.0           4       50.0 %

--------------------------   -----------       -------    --------
Total Limits Violated               0.1           8.0            8
```

アプリケーション情報の概要 (すべてのアプリケーション)

sp_sysmon で、概要セクション (すべてのアプリケーションについての情報) に出力される統計情報は、リソースの使用に何らかの異常がないかどうかを調べる場合に役立ちます。異常があれば、詳細なレポートでより詳しく調べることができます。

アプリケーション情報には次の項目が表示されます。

- タスクの優先度が別のレベルに切り替えられていないかどうか。
- タスクを適切に実行できる時間が割り当てられているかどうか。
- low の優先度を割り当てたタスクが CPU 時間を必要としているかどうか。
- ロード・バランスをとるためのエンジンのバインドは正しいかどうか。

“Application Statistics Summary”には、ユーザ・タスクだけでなくシステム・タスクのデータも含まれます。リソースに問題があることが概要のレポートに示され、アプリケーション情報やアプリケーションとログインの情報に、それを裏付ける証拠が見つからない場合は、レポートの sp_sysmon カーネル・セクションを調べます。詳細については、「カーネルの使用率」(20 ページ) を参照してください。

優先度の変更

サンプル間隔中に各優先度の実行キュー内のすべてのユーザ・タスクに対して行われた優先度の変更をレポートします。システム関連のアクティビティによって、いくつかの優先度が切り替えられていても問題はありません。たとえば次のような場合に、こうした優先度の変更が発生します。

- ロックを待機してタスクがスリープする場合 − Adaptive Server は、一時的にそのタスクの優先度を上げる。
- ハウスキーピング・タスクがスリープする場合 − Adaptive Server は、ハウスキーピング・ウォッシュ・タスクおよびハウスキーピング・チョア・タスクがウェイクアップするときに優先度を medium に引き上げ、スリープ状態に戻るときに low に戻す。
- タスクがストアド・プロシージャを実行する場合 − タスクは、ストアド・プロシージャの優先度を使用し、ストアド・プロシージャを実行したあと、以前の優先度レベルに戻る。

論理プロセス管理を使用していて、安定した状態での値と比べて優先度変更後の値が大きい場合は、アプリケーションまたはそのアプリケーションに関連するユーザ・タスクの優先度が頻繁に変更されていることを示します。各アプリケーションの優先度変更データを確認し、アプリケーションとログインが、予期したとおりに動作しているかどうかを確認してください。

優先度の変更率が高くなっている原因が、アプリケーションでも関連するタスクでもない場合は、システム・アクティビティが原因であると思われます。

優先度変更の総数

サンプリング時間中の優先度変更の総数をレポートします。このセクションでは、実行キューの優先度変更が数多く発生しているかどうかをすばやく調べることができます。

割り当てられたスライスの不足

各実行キューのユーザ・タスクが、実行用に割り当てられた時間を超過した回数。ユーザ・タスクがいったんエンジンへのアクセスを取得すると、ユーザ・タスクは与えられた時間内でだけ、実行が許されます。タスクが、その時間内にエンジンを解放しなかった場合は、重要なリソースを保持することなくできるだけ早く解放するように、Adaptive Server から要求されます。エンジンを解放したあと、タスクは実行キューに戻されます。

“Allocated Slices Exhausted”は、実行属性やエンジンとの結合をチューニングする必要がある CPU 集約アプリケーションがあるかどうかを調べる場合に役立ちます。これらの数値が大きい場合は、アプリケーションが CPU 集約であることを示します。アプリケーション・レベルでの情報は、どのアプリケーションをチューニングする必要があるかを判断する場合に役立ちます。いくつかのタスク、特に大規模なソート・オペレーションを実行するタスクは、CPU 集約です。

“Alloted slices exhausted”を使用して、新しいハードウェアの条件を評価します。CPU 速度が速いと効果的であることが多く、タイムスライスの使用量が多いほど、使用しているサイトで高速 CPU にアップグレードできるメリットが大きくなります。

エンジンによって省略されたタスク

エンジンが実行キューの先頭にあるユーザ・タスクを省略した回数。これは、実行キューの先頭にあるタスクがエンジン・グループと結び付いており、エンジン・グループの一部でないエンジンによってキューでバイパスされているときに起こります。

この値は、エンジン・グループや複数のエンジン・グループのバインドの設定に影響を受けます。より重要なタスクのために low 優先度のタスクがバイパスされる場合は、このカテゴリの数値が高くても問題はありません。エンジン・グループがバインドされており、実行の準備が整ったタスクが互換性のあるエンジンを見つけられない可能性もあります。この場合、タスクはエンジンがアイドル状態になるまで実行せずに待機します。エンジン・グループとエンジン・グループのバインド方法を調べ、ロード・バランスをチェックしてください。

エンジン・スコープの変更

サンプル間隔中にユーザがユーザ・タスクのエンジン・グループのバインドを変更した回数。

アプリケーションごと、またはアプリケーションとログインごとのアプリケーション情報

このセクションでは、特定のアプリケーションとログインのタスクによって、または各アプリケーションのすべてのユーザによって使用されるシステム・リソースについて、詳しい情報が出力されます。

アプリケーションのアクティビティ

アプリケーションが I/O 集約であるか CPU 集約であるかを調べる場合に役立ちます。このセクションでは、アプリケーション内のすべてのユーザ・タスクが実行された時間、I/O が実行された時間、またはアイドル状態で使用した時間がレポートされます。さらに、このセクションでは、タスクがスケジューリングされ、実行するために選択された回数もレポートされます。

CPU ビジー

サンプル間隔中にユーザ・タスクが実行されていたクロック・チック数。このカテゴリの数値が大きい場合は、このアプリケーションが CPU 集約であることを示しています。これが問題になる場合は、エンジンにバインドすることで解決できます。

I/O ビジー

サンプル間隔中にユーザ・タスクが I/O を実行していたクロック・チック数。このカテゴリの数値が大きい場合は、このアプリケーションが I/O 集約であることを示しています。また、アイドル時間が大きい場合もアプリケーションが I/O 集約である可能性があります。

より高い優先度の割り当て、負荷の低いエンジンやエンジン・グループへのバインド、アプリケーション・データの複数のデバイスへの分割のいずれかを行うと、アプリケーションのスループットが向上します。

アイドル

サンプル間隔中にユーザ・タスクがアイドル状態だったクロック・チック数。

スケジュールされた回数

ユーザ・タスクがスケジューリングされ、エンジンで実行するために選択された回数。このデータは、アプリケーションに十分なリソースがあるかどうかを調べる場合に役立ちます。通常、十分な CPU 時間を必要とするタスクでこの数値が低い場合は、リソース不足を示しています。十分なエンジン・リソースを持つ負荷の高いシステムでは、優先度の変更を検討してください。

アプリケーションの優先度変更

サンプル間隔中に、このアプリケーションの優先度が変更された回数。

“Application Management”カテゴリに問題が示されている場合は、このセクションを使って原因を正確に特定してください。

完了されたアプリケーションの I/O

サンプル間隔中に、このアプリケーションによって完了されたディスク I/O とネットワーク I/O をレポートします。

このカテゴリには、完了したディスク I/O とネットワーク I/O の総数が示されます。

I/O の完了に問題があると思われる場合は、「ディスク I/O 管理」(125 ページ)と「ネットワーク I/O 管理」(131 ページ)を参照してください。

リソースの制限違反

次の制限に対する違反の数と種類がレポートされます。

- I/O 制限違反 − 予測と実際。
- 時間制限 − バッチとトランザクション。
- RowCount 制限違反。
- 制限違反の総数。

サンプル時間中に制限の超過がなかった場合は、総数の行だけが出力されます。

ESP 管理

このセクションでは、拡張ストアド・プロシージャの使用がレポートされます。

出力例

```
===============================================================================

ESP Management                per sec     per xact       count   % of total
--------------------------  ------------  ----------  ----------  ---------
ESP Requests                        0.0         0.0           0         n/a
```

ESP 要求

サンプル間隔中の拡張ストアド・プロシージャを呼び出した数。

ESP を実行した平均時間

サンプル間隔中に実行されたすべての拡張ストアド・プロシージャの実行時間の平均値をレポートします。

ハウスキーピング・タスク・アクティビティ

ハウスキーピング・タスクについてレポートします。設定パラメータ housekeeper free write percent を 0 に設定すると、housekeeper task HK WASH は実行されません。enable housekeeper GC を 1 ～ 5 の値に設定すると、領域の再利用を有効にできます。enable housekeeper GC を 0 に設定すると、領域の再利用を無効にできます。

以下を参照してください。

- 『パフォーマンス&チューニング・ガイド：基本』の「第 3 章 エンジンと CPU の使用方法」
- 『システム管理ガイド 第 1 巻』の「システムの問題の診断」

出力例

```
===============================================================================

Housekeeper Task Activity
-------------------------
                         per sec    per xact     count        % of total
                         ---------  --------  --------        -----------
Buffer Cache Washes
   Clean                     228.0       5.6    136828            100.0 %
   Dirty                       0.0       0.0        12              0.0 %
                         ---------  --------  --------
Total Washes                 228.1       5.6    136840
Garbage Collections            1.3       0.0       791               n/a
Pages Processed in GC          0.0       0.0         0               n/a
Statistics Updates            12.0       0.3      7196                 n/a
```

バッファ・キャッシュのウォッシング

このセクションでは、次の情報がレポートされます。

- ハウスキーピング・ウォッシュ・タスクによって調査されるバッファの数。
- クリーンであるバッファの数。
- ダーティであるバッファの数。

Adaptive Server がディスクに書き込んでいない変更がデータ・キャッシュで表すデータ・ページに含まれている場合、バッファは「ダーティ」とみなされます。バッファが表すデータ・ページがディスク上のデータのコピーである場合、バッファは「クリーン」とみなされます。

ダーティ・バッファの数には、ウォッシュ・マーカを超えたために、すでに書き込みが開始されたバッファも含まれます。

sp_sysmon の “Recovery Management” セクションでは、ハウスキーピング・ウォッシュ・タスクがデータベースのすべてのダーティ・バッファを書き込みできた回数がレポートされます。詳細については、「リカバリ管理」(122 ページ) を参照してください。

ガーベジ・コレクション

コミットされた削除があるかどうか、つまりデータ・ページに再利用できる領域があるかどうかをハウスキーピング・ガーベジ・コレクション・タスクが確認した回数。

GC で処理されたページ

データオンリーロック・テーブルでハウスキーピング・ガーベジ・コレクション・タスクが未使用の領域の再利用に成功したページ数。

統計の更新

統計に書き込みが必要かどうかをハウスキーピング・チョア・タスクが確認した回数。

実行 SQL へのモニタ・アクセス

次の項目をレポートします。

- sp_showplan がクエリ・プランにアクセスしたときに発生した競合。

  注意　モニタリング・テーブルの問い合わせ (たとえば、monProcessSQLText と monSysSQLText) では、これらのカウンタは更新されません。

- SQL バッチ・テキスト・バッファでのオーバフローの数と、サンプル間隔中に送信された SQL バッチ・テキストの最大サイズ。

出力例

```
Monitor Access to Executing SQL
-------------------------------
                                  per sec    per xact     count    % of total
                               -----------  ----------   -------    ----------
Waits on Execution Plans               0.0         0.0         0         n/a
Number of SQL Text Overflows           8.6          .2      5138         n/a
Maximum SQL Text Requested             n/a         n/a    588307         n/a
(since beginning of sample)
```

実行プランの待機

sp_showplan を使おうとしたプロセスが、クエリ・プランへの読み込みアクセスを取得するまで待機しなければならなかった回数。コンパイル済みのプランが完了する前、またはクエリ・プランの実行が終了したあとに sp_showplan が実行される場合、クエリ・プランは利用できなくなることがあります。この場合、Adaptive Server はプランに 3 回アクセスを試みてから、ユーザにメッセージを返します。

SQL テキスト・オーバフローの数

SQL バッチ・テキストがテキスト・バッファのサイズを超えた回数。

要求された SQL テキストの最大サイズ

サンプル間隔が開始された時点以降の、SQL テキストのバッチの最大サイズ。この値を使用して、設定パラメータ max SQL text monitored を設定できます。

『システム管理ガイド：第 1 巻』の「システムの問題の診断」を参照してください。

トランザクション・プロファイル

データ変更をコマンドの種類とテーブルのロック・スキームごとにレポートします。

出力例

```
===============================================================================
Transaction Profile
-------------------

  Transaction Summary           per sec    per xact     count        % of total
------------------------      --------    --------   -------        ---------
    Committed Xacts               40.6         n/a     24357             n/a

  Transaction Detail            per sec    per xact     count        % of total
  -------------------         --------    --------   -------        ---------
  Inserts
    Fully Logged
      APL Heap Table               0.0         0.0         0            0.0 %
      APL Clustered Table          0.0         0.0         0            0.0 %
      Data Only Lock Table      1337.3       906.7    160479          100.0 %
      Fast Bulk Insert             0.0         0.0         0            0.0 %
    Minimally Logged
      APL Heap Table               0.0         0.0         0            0.0 %
      APL Clustered Table          0.0         0.0         0            0.0 %
      Data Only Lock Table         0.0         0.0         0            0.0 %
  ----------------------  -----------   ---------   -------        ---------
  Total Rows Inserted           1337.3       906.7    160479           100.0 %

  Updates
    Total Rows Updated             0.0         0.0         0             n/a
  ----------------------  -----------   ---------   -------        ---------
```

```
Total Rows Updated               0.0          0.0          0              0.0 %

Data Only Locked Updates
  Total Rows Updated             0.0          0.0          0               n/a
----------------------  ------------   ----------   -------        ----------
Total DOL Rows Updated           0.0          0.0          0              0.0 %

Deletes

  Fully Logged
    APL Deferred                 0.0          0.0          0              0.0 %
    APL Direct                   0.0          0.0          0              0.0 %
    DOL                          0.3          0.2         36            100.0 %
  Minimally Logged
    APL Direct                   0.0          0.0          0              0.0 %
    DOL                          0.0          0.0          0              0.0 %
----------------------  ------------   ----------   -------        ----------
Total Rows Deleted               0.3          0.2         36              0.0 %

======================  ============  ===========  ========
Total Rows Affected           1337.6        906.9    160515
======================  ============  ===========  ========
```

トランザクションの概要

コミットされたトランザクションをレポートします。“Committed Xacts”では、サンプル間隔中にコミットされたトランザクションの数がレポートされます。

カウントされたトランザクションには、暗黙的および明示的なトランザクションと連鎖 (ANSI 形式) および非連鎖トランザクションが含まれます。

- 明示的トランザクションでは、insert、update、delete などのデータ変更コマンドが実行される。begin transaction 文を指定しなかった場合は、Adaptive Server によって各操作が個別のトランザクションとして解釈される。明示的な commit transaction 文は必要ない。たとえば、次のコマンドは 3 つのトランザクションとしてカウントされる。

  ```
  1> insert …
  2> go
  1> insert …
  2> go
  1> insert …
  2> go
  ```

- 暗黙的トランザクションでは、データ変更コマンドが begin transaction 文と commit transaction 文で囲まれ、トランザクションの数はコミット文の数を使ってカウントされる。たとえば、次の文は、1 つのトランザクションとしてカウントされる。

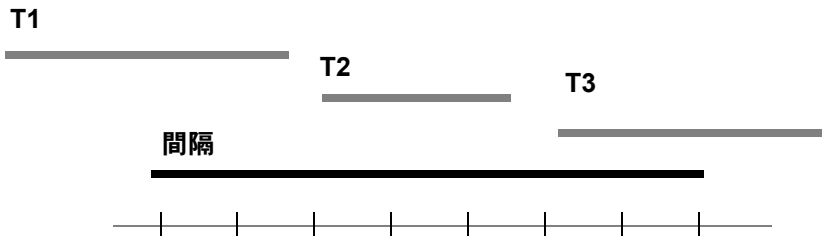
```
1> begin transaction
2> insert …
3> insert …
4> insert …
5> commit transaction
6> go
```

- ANSI トランザクション・モデルでは、select またはデータ変更コマンドによってトランザクションが開始されるが、commit transaction 文を使用してトランザクションを完了する必要がある。sp_sysmon は、commit transaction 文の数によってトランザクションの数をカウントする。たとえば、次の文は、1 つのトランザクションとしてカウントされる。

```
1> insert …
2> insert …
3> insert …
4> commit transaction
5> go
```

サンプル間隔の前に開始され、サンプル間隔中に完了するトランザクションがある場合、“Committed Xacts”では、サンプル間隔中に開始され完了したトランザクションの数よりも多くのトランザクションがレポートされます。トランザクションがサンプル間隔中に完了しない場合は、“Total # of Xacts”にはそのトランザクションは含まれません。図 2-1 では、T1 と T2 はカウントされますが T3 はカウントされません。

図 2-1: トランザクションがカウントされる仕組み



マルチ・データベース・トランザクションのカウント方法

マルチ・データベース・トランザクションもカウントされます。たとえば、3 つのデータベースを変更するトランザクションは、3 つのトランザクションとしてカウントされます。

マルチデータベース・トランザクションには、シングル・データベース・トランザクションよりも多くのオーバヘッドがかかります。マルチデータベース・トランザクションでは、より多くのログ・レコードとより多くのユーザ・ログ・キャッシュ (ULC) フラッシュが必要になり、データベース間で 2 フェーズ・コミットが実行されます。

可能であればいつでも、マルチ・データベース・トランザクションの数を減らしてパフォーマンスを改善できます。

トランザクションの詳細

データ変更オペレーションについて、詳しい統計情報が種類ごとに出力されます。ロールバックされたトランザクションによって実行される作業もレポートに含まれますが、このトランザクションは、トランザクションの数としてはカウントされません。

挿入、更新、削除の“Total Rows”の“% of total”カラムでは、すべてのトランザクションに対するそのトランザクション・タイプの割合が、パーセンテージとしてレポートされます。

遅延と直接の挿入、更新、削除の詳細については、『パフォーマンス＆チューニング・シリーズ：クエリ処理と抽象プラン』の「更新オペレーションの実行方法」を参照してください。

このセクションの出力では、APL は全ページロック・テーブルを示し、DOL はデータオンリーロック・テーブルを示します。

挿入

すべての insert 操作、update 操作、delete 操作に対するヒープ・テーブル (分割されたヒープ・テーブルを含む) とクラスタード・テーブルで実行された挿入の種類についての詳細な情報を、パーセンテージで出力します。“Inserts”では、次のテーブルで実行された挿入の数が表示されます。

- 全ページロック・ヒープ・テーブル
- クラスタード・インデックスのある全ページロック・テーブル
- データオンリーロック・テーブル

Insert 統計値には、高速バルク・コピーによる挿入は含まれません。これは、高速バルク・コピーは通常の挿入メカニズムやロギング・メカニズムを使用せずにデータ・ページとディスクへ直接書き込まれるためです。

完全なロギングと最低限のロギング

完全なロギング操作に対しては“Fully Logged”、最低限のロギング操作に対しては“Minimally Logged”についてレポートします。項目は“Fully Logged”サブセクションにあるが“Minimally Logged”にはない場合があります。これは、項目が最低限のロギングに適用されていないためです。

APL ヒープ・テーブル

全ページロック・ヒープ・テーブル、つまりクラスタード・インデックスがないすべてのテーブルで実行されたロー挿入の数。このセクションには、次の項目についての情報が含まれます。

- 分割されたヒープ・テーブル

- 分割されていないヒープ・テーブル
- ヒープ・テーブルへの低速バルク・コピー挿入
- select into コマンド
- 作業テーブルへの挿入。アプリケーションがクラスタード・インデックスなしで APL テーブルを使用する頻度が高い場合以外は、これが最も一般的な使用法である。

“% of total”カラムでは、挿入の総数に対する、ヒープ・テーブルへのロー挿入数の割合が、パーセンテージで示されます。

ヒープ・テーブルに対して多くの挿入が実行されている場合は、これらの挿入によって競合が発生していないかどうかを確認してください。

sp_sysmon のレポートをチェックし、「ロックの詳細」(95 ページ) で説明するヒープ上での最終ページ・ロックについてのデータを調べてください。競合の問題が示されている場合は、モニタリング・テーブルをクエリしてどのテーブルに問題があるかを発見できます。

多くの場合、挿入アクティビティをランダム化するクラスタード・インデックスを作成すると、ヒープに対するパフォーマンス上の問題を解決できます。しかし、分割されていないテーブルの分割を作成したり、分割されたテーブルの分割数を増加したりすることが必要になる場合もあります。

APL クラスタード・テーブル

クラスタード・インデックスを使った全ページロック・テーブルにローを挿入した数。“% of total”カラムでは、ロー挿入の総数に対する、クラスタード・インデックスを使ったテーブルへのロー挿入数の割合が、パーセンテージでレポートされます。

全ページロック・クラスタード・テーブルへの挿入によって、ページ分割が発生する可能性があります。

Adaptive Server は、クラスタード・インデックスを持つワークテーブルを使用して、ベクトル集合関数を実行する場合があります (たとえば、group by)。ただし、Adaptive Server リリース 15.0 以降ではハッシュベースのアルゴリズムがより一般的です。

詳細については、「クラスタ分割からのロー ID の更新」(82 ページ) と 「ページ分割」(82 ページ) を参照してください。

データオンリー・ロック・テーブル

すべてのデータオンリーロック・テーブルに挿入された数。“% of total”カラムは、すべての挿入に対する、データオンリーロック・テーブルへの挿入の割合が、パーセンテージでレポートされます。

高速バルク挿入

一部のログを取る高速 bcp による挿入の数。

挿入されたローの総数

すべてのテーブルへのすべてのロー挿入をレポートします。ここには、1 秒当たりのすべての挿入数の平均値、トランザクション当たりのすべての挿入数の平均値、挿入の総数が出力されます。“% of total”では、データ変更オペレーションによって影響を受けたローの総数に対する、ロー挿入の割合がパーセンテージでレポートされます。

更新と更新の詳細セクション

“Updates”には、“Updates”と“Data Only Locked Updates”の 2 つのセクションがあります。

完全なロギング操作に対しては“Fully Logged”、最低限のロギング操作に対しては“Minimally Logged”についてのレポートを含みます。項目は“Fully Logged”サブセクションにあるが“Minimally Logged”にはない場合があります。これは、項目が最低限のロギングに適用されていないためです。

更新

遅延ロー更新および直接ロー更新の数。“% of total”カラムには、ロー更新の総数に対する各種類の更新の割合がパーセンテージでレポートされます。sp_sysmon では、次の種類の更新がレポートされます。

- APL 遅延更新
- APL 直接置き換え
- APL 低コストの直接更新
- APL 高コストの直接更新
- DOL 遅延更新
- DOL 直接更新

“Fully Logged”セクションの“APL Direct Cheap”は、完全にログされた全ページ・ロック直接更新の数を示します。“Minimally Logged”セクションの“APL Direct Cheap”は、最低限のログを取る全ページ・ロック直接更新の実行回数を指します。

直接更新では、ログ・スキャンの回数が制限され、ロックが減少し、B ツリー・インデックスをたどる回数も少なくなる (結果としてロック競合が減ります) ので、直接更新は遅延更新よりも負荷が低く、一般に高速です。さらに、Adaptive Server がページをプリフェッチし、ログ・レコードに従って変更を実行する必要がないので、I/O を節約できます。

更新の種類の詳細については、『パフォーマンス＆チューニング・シリーズ：クエリ処理と抽象プラン』の「第 1 章 クエリ処理の概要」を参照してください。

更新されたローの総数

遅延更新と直接更新をすべてレポートします。“% of total”カラムでは、変更されたすべてのローに対する、更新されたローの割合が、パーセンテージで示されます。

データオンリーロック・テーブルの更新

データオンリーロック・テーブルに行われた更新について、次の詳細な情報をレポートします。

- DOL 置換 − 更新によってローの長さが変更されなかった。一部またはすべてのローが変更されたが、ローの長さは更新前と変わらなかった。
- DOL 縮小 − 更新によりローの長さが短くなり、ページ上に連続しない空き領域ができた。この領域は、集めて再利用される。
- DOL 低コストの拡張 − ローの長さが長くなったが、ページ上の最後のローだったので、ページ上のほかのローを移動する必要がなかった。
- DOL 高コストの拡張 − ローの長さが長くなり、ページ上のほかのローを動かす必要があった。
- DOL 拡張と先送り − ローが長くなりページに収まらなかったため、新しい位置に先送りされた。
- DOL 先送りされたローを元に戻す − 先送りされたローが更新によりページの元の位置に収まるようになったため、そのページに戻された。

完全なロギング操作に対しては“Fully Logged”、最低限のロギング操作に対しては“Minimally Logged”についてのレポートを含みます。項目は“Fully Logged”サブセクションにあるが“Minimally Logged”にはない場合があります。これは、項目が最低限のロギングに適用されていないためです。

“Total DOL Rows Updated”でレポートされる合計は、セクションの最後にある“Total Rows Affected”の合計には含まれません。このグループ内の更新は、“DOL Deferred”と“DOL Direct”ですでにレポートされている更新を異なる分類で提供しているためです。

削除

完全なロギングと最低限のロギング

完全なロギング操作に対しては“Fully Logged”、最低限のロギング操作に対しては“Minimally Logged”についてレポートします。項目は“Fully Logged”サブセクションにあるが“Minimally Logged”にはない場合があります。これは、項目が最低限のロギングに適用されていないためです。

削除されたローの総数

遅延更新と直接削除をすべてレポートします。“% of total”カラムには、挿入、更新、削除のいずれかが行われたすべてのローの数に対する、削除されたロー数の割合が、パーセンテージでレポートされます。

トランザクション管理

ULC (ユーザ・ログ・キャッシュ) のトランザクション・ログへのフラッシュ、ULC ログ・レコード、ULC セマフォ要求、ログ・セマフォ要求、トランザクション・ログの書き込み、トランザクション・ログの割り付けなどのトランザクション管理アクティビティをレポートします。

出力例

```
Transaction Management
----------------------

    ULC Flushes to Xact Log     per sec      per xact       count  % of total
    --------------------  ------------  ------------  ----------  ----------
    Any Logging Mode DMLs
      by End Transaction            1.6           1.1         189       1.0 %
      by Change of Database         0.0           0.0           4       0.0 %
      by Unpin                      0.0           0.0           2       0.0 %
      by Log markers                0.0           0.0           0       0.0 %

    Fully Logged DMLs
      by Full ULC                 145.5          98.6       17458      95.9 %
      by Single Log Record          4.5           3.1         544       3.0 %

    Minimally Logged DMLs
      by Full ULC                   0.0           0.0           0       0.0 %
      by Single Log Record          0.0           0.0           0       0.0 %
      by Start of Sub-Command       0.0           0.0           0       0.0 %
      by End of Sub-Command         0.0           0.0           0       0.0 %
```

```
    --------------------  ------------  ------------  ----------
    Total ULC Flushes             151.6         102.8       18197

    ULC Flushes Skipped          per sec      per xact       count   % of total
    --------------------  ------------  ------------  ----------  ----------
    Fully Logged DMLs
      by ULC Discards               0.2           0.1          21        100 %
    Minimally Logged DMLs
      by ULC Discards               0.0           0.0           0        0.0 %
    --------------------  ------------  ------------  ----------
    Total ULC Flushes Skips         0.2           0.1          21

    ULC Log Records              per sec       per xact       count   % of total
    --------------------  ------------  ------------  ----------  ----------
    Fully Logged DMLs            6662.3        4516.8      799476       100.0
    Minimally Logged DMLs           0.0           0.0           0         0.0
    --------------------  ------------  ------------  ----------
    Total ULC Log Records        6662.3        4516.8      799476

    Max ULC Size During Sample
    --------------------------
    Fully Logged DMLs               n/a           n/a       16384         n/a
    Minimally Logged DMLs           n/a           n/a           0         n/a

    ML-DMLs Sub-Command Scans   per sec      per xact       count  % of total
    --------------------  ------------  ------------  ----------  ----------
---
    Total Sub-Command Scans         0.0           0.0           0         n/a

    ML-DMLs ULC Efficiency      per sec      per xact       count  % of total
    --------------------  ------------  ------------  ----------  ----------
    Total ML-DML Sub-Commands       0.0           0.0           0         n/a

    ULC Semaphore Requests
      Granted                   13727.0        9306.4     1647239     100.0 %


      Waited                        0.0           0.0           0       0.0 %
    --------------------  ------------  ------------  ----------
    Total ULC Semaphore Req     13727.0        9306.4     1647239

    Log Semaphore Requests
      Granted                     184.0         124.7       22076     100.0 %
      Local Waited                  0.0           0.0           0       0.0 %
      Global Waited                 0.0           0.0           0       0.0 %
    --------------------  ------------  ------------  ----------
    Total Log Semaphore Req       184.0         124.7       22076

    Transaction Log Writes        167.5         113.6       20102         n/a
    Transaction Log Alloc         167.4         113.5       20092         n/a
    Avg # Writes per Log Page       n/a           n/a     1.00050         n/a
```

```
Tuning Recommendations for Transaction Management
------------------------------------------------
- Consider increasing the 'user log cache size'
  configuration parameter.
```

トランザクション・ログへの ULC のフラッシュ

ULC (ユーザ・ログ・キャッシュ) がトランザクション・ログへフラッシュされた回数の合計をレポートします。“% of total”カラムでは、ULC フラッシュの総数に対するフラッシュが実行された回数の割合が、カテゴリごとにパーセンテージでレポートされます。このカテゴリは、アプリケーション内にある、ULC フラッシュの問題の原因となる領域を特定する場合に役立ちます。

設定されたユーザ接続ごとに、1 つの ULC があります。Adaptive Server は ULC を使用して、トランザクション・ログ・レコードをバッファリングします。SMP およびシングルプロセッサのどちらのシステムでも、ULC はトランザクション・ログ I/O を減らすために役立ちます。SMP システムでは、トランザクション・ログの現在のページでの競合を減らすために役立ちます。

設定パラメータ user log cache size を使用して ULC のサイズを設定できます。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

ULC フラッシュは、次のアクティビティによって起こります。

- “by Full ULC” － プロセスの ULC が満杯になった。
- “by End Transaction”－ トランザクションが終了した (rollback または commit の明示的トランザクションまたは暗示的トランザクション)。
- “by Change of Database”－ トランザクションが異なるデータベースのオブジェクトを変更した (マルチ・データベース・トランザクション)。
- “by System Log Record” － システム・トランザクション (OAM ページ割り付けなど) がユーザ・トランザクション内で発生した。
- “by Unpin”－ 2 つのトランザクションが DOL テーブルの同じページを更新しようとしている。
- “by Start of Sub-Command”または“by End of Sub-Command” － ULC を最低限のログを取る DML のサブコマンドの開始または終了時点でフラッシュする必要があった。
- “by Other” － ディスクへの書き込みが必要になった場合など、そのほかの理由。

これらのアクティビティのいずれかによって ULC フラッシュが発生すると、Adaptive Server は、すべてのログ・レコードをユーザ・ログ・キャッシュからデータベース・トランザクション・ログへコピーします。

“Total ULC Flushes”では、サンプル間隔中に発生したすべての ULC フラッシュの総数がレポートされます。

Any Logging Mode DMLs

トランザクションの終了

“by End Transaction”の値が大きい場合は、実行時間の短い単純なトランザクションが数多くあることを示します。

データベースの変更

ULC は、データベースが変更されるたびにフラッシュされます。“by Change of Database”の値が大きい場合は、2K 以上の ULC のサイズを減らすことを検討してください。

ピン解除

「ピン解除」は、2 つのトランザクションが同じデータ・ページを更新しようとする場合に発生します。このため、2 番目のトランザクションによって最初のトランザクションのユーザ・ログ・キャッシュがフラッシュされ、データ・ページが最初のトランザクションの ULC からピン解除されます。

両方のトランザクションが同じページを更新しようとするため、Adaptive Server は通常、ロー・レベルのロックを使用して同時実行性を向上させます。

ピン解除は次の場合に発生します。

1 トランザクション T1 が行 R1 を更新する。

2 Adaptive Server が P1 の ULC で更新のログを取り、このデータ・ページが T1 の ULC に確実にピンされるようにする。

3 T2 が P1 ページの異なる行を更新しようとする場合に、このページが T1 の ULC にピンされていることが判明する。

4 ページを更新するために、T2 が T1 の ULC をフラッシュしてデータ・ページをピン解除する。

ピン解除の数を減らすには、次のようにします。

- 可能な限り、アプリケーションを再設計して、別のデータ・ページを直接要求できるようにする。
- ロー・ロックが避けられない場合は、sp_chgattribute max_rows_per_page を使用してデータを複数のデータ・ページに分散する。

ログ・マーカ

“by Log markers”の値が大きい場合は、Adaptive Server が永続的なログ・マーカ・スキャンのために ULC をフラッシュしていることを示します (永続的なログ・マーカ・スキャンは、syslogs にログ・レコードが含まれることを示します)。Adaptive Server はログ・レコードをトリガ、ロールバック、アボートの実行などの操作に使用します。Adaptive Server では、必要とする永続的なログ・マーカがない場合、ULC をフラッシュして新しいログ・マーカが作成されます。

“by Log markers”に大部分の ULC フラッシュが表示された場合は、不要または冗長なトリガ、ロールバック、またはアボートの回数を減らす必要が生じる場合があります。

システム・ログ・レコードとそのほかの理由

“by System Log Record”または“by Other”のいずれかの値が 20% を超えているときに、ULC のサイズが 2048 を超えている場合は、ULC サイズを減らすことを検討してください。

sp_sysmon のレポートのログ・アクティビティに関連するセクションで、次の内容をチェックします。

- ユーザ・ログ・キャッシュでのセマフォの競合 (SMP のみ)。「ULC セマフォ要求」(77 ページ) を参照。
- ログ・セマフォの競合(SMP のみ)。「ログ・セマフォ要求」(78 ページ) を参照。
- トランザクション・ログの書き込み数。「トランザクション・ログ書き込み」(78 ページ) を参照。

完全なログを取る DML

満杯の ULC

“by Full ULC”の値が大きい場合、Adaptive Server は、ULC のフラッシュをトランザクション当たり 2 回以上行っており、ユーザ・ログ・キャッシュのパフォーマンスのメリットを十分利用できていません。“by Full ULC”の“% of total”の値が 20% を超える場合は、user log cache size パラメータのサイズを増やすことを検討してください。

ULC のサイズを増やすと、各ユーザ接続に必要なメモリの量が増えます。そのため、大規模なトランザクションでは、ULC サイズを設定してパーセンテージを減らす必要はありません。

単一のログ・レコード

単一のログ・レコードは ULC 内に存在できないため、ULC フラッシュが発生する必要があります。

最低限のログを取る DML

“Minimally Logged DMLs”の“ULC Flushes to Xact Log”の値が高い場合、ULC は最低限のログを取る DML を構成する各サブコマンドのログ・レコードを完全に含むことはできません。この例では、ログ・レコードを破棄できないため、Adaptive Server のパフォーマンスは最低限のロギングのメリットを完全に得ることができません。user log cache size 設定パラメータのサイズを増やしてから sp_sysmon を再実行して、“ULC Flushes to Xact Log”の値が小さくなったことを確認してください。小さな値は、ULC を syslogs にフラッシュする必要がないことを示します。

満杯の ULC

最低限のログを取る DML の単一のサブコマンドが ULC に格納しきれないログ・レコード数を取ったため、syslogs にフラッシュする必要があることを示します。最低限のロギングは、サブコマンドのすべてのログ・レコードを単一の ULC 内でログに記録できる場合にのみ効果的です。サブコマンドが完了し、すべてのログ・レコードが ULC 内に収まると、ログ・レコードは破棄され、syslogs にフラッシュされません。

単一のログ・レコード

単一のログ・レコードは ULC 内に存在できないため、ULC フラッシュが発生する必要があります。

サブコマンドの開始

同じトランザクションで最低限のログを取る DML として実行した完全にログされたコマンドから得たログ・レコードは、最低限のログを取る DML を開始する前に ULC から syslogs にフラッシュする必要がありました。

サブコマンドの終了

最低限のログを取る DML の単一のサブコマンドは、前の完全な ULC や単一のログ・レコードなどのために、ログに記録する必要がありました。ULC では、サブコマンドのすべてのログ・レコードを保持するために、サブコマンドの終了時にフラッシュが必要でした。

ULC フラッシュの総数

サンプル間隔中の ULC フラッシュの総数をレポートします。

省略された ULC フラッシュ

Adaptive Server が省略するユーザ・ログ・キャッシュの数。

完全なログを取る DML

完全なログを取る DML の実行時にトランザクションのすべてのログ・レコードが ULC に収まる場合、Adaptive Server はすべてのログ・レコードを破棄できます。

“Fully Logged DMLs” の値が大きい場合は、パフォーマンスが向上します。

最低限のログを取る DML

最低限のログを取る DML のサブコマンドのすべてのログ・レコードが ULC 内に収まるために、ULC にあるログ・レコードを破棄できる回数。

“Minimally Logged DMLs” の値が大きい場合は、パフォーマンスが向上します。

省略される ULC フラッシュの総数

ユーザ・ログ・キャッシュのフラッシュの総数をレポートします。合計数はカラムごとの数です。

ULC ログ・レコード

“ULC Log Records” では、トランザクション当たりのログの平均値が出力されます。このローは、ベンチマークの場合や、管理された開発環境で、ULC に書き込まれるトランザクション当たりのログ・レコードの数を調べる場合に役立ちます。

複数のインデックス、または遅延更新や遅延削除に影響を及ぼすような多くのトランザクションでは、1 回のデータ更新のために複数のログ・レコードが必要です。多数のローを変更するクエリでは、ローごとに 1 つまたは複数のレコードを使用します。

このデータが通常の大きさでない場合は、次のサンプル中の最大 ULC サイズの項のデータについて検討し、長時間実行されるトランザクションや、数多くのローを変更するトランザクションを実行するアプリケーションがないか調べてください。

完全なログを取る DML

完全なログを取る DML からのトランザクションごとのログ・レコードの数を示します。

最低限のログを取る DML

最低限のログを取る DML からのトランザクションごとのログ・レコードの数を示します。

ULC ログ・レコードの合計数

ユーザ・ログ・キャッシュのログ・レコードの総数をレポートします。合計数はカラムごとの数です。

サンプル中の最大 ULC サイズ

“count”カラムの値は、すべての ULC のうち、どの ULC でも使用される最大バイト数です。このデータは、ULC のサイズが正しく設定されているかどうかを調べる場合に役立ちます。

Adaptive Server では、トランザクションが完了したときに ULC がフラッシュされるので、ULC に割り付けられているが使用されないメモリは無駄になります。“count”カラムの値が、user log cache size 設定パラメータに定義した値よりも一貫して小さい場合は、user log cache size を“count”カラムの値まで減らしてください (ただし、2048 バイト未満は不可)。

“Max ULC Size During Sample”がユーザ・ログ・キャッシュのサイズと等しい場合は、ユーザ・ログ・キャッシュへの書き込みを行うトランザクションによるフラッシュの数をチェックします (「満杯の ULC」(73 ページ) を参照してください)。満杯の ULC のためにログがフラッシュされた回数が 20% を超えた場合は、user log cache size 設定パラメータを増やすことを検討してください。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

完全なログを取る DML

完全なログを取る DML のユーザ・ログ・キャッシュに使用される最大バイト数。

最低限のログを取る DML

最低限のログを取る DML のユーザ・ログ・キャッシュに使用される最大バイト数。

ML-DML サブコマンド・スキャン

最低限のログを取るコマンドを実行する場合、Adaptive Server ではサブコマンドのログ・レコードをスキャンしなければならないことがあります。“ML-DMLs Sub-Command Scans”の値は、Adaptive Server が実行したスキャンの回数を指します。

ULC スキャン

ULC にはスキャンされるすべてのログ・レコード含まれているため、ULC 内全体に対してスキャンを実行できます。

syslogs スキャン

ULC にスキャンされるログ・レコードがなくなったため、syslogs を使用してスキャンを実行する必要がありました。

サブコマンド・スキャンの総数

ULC と syslogs スキャン回数の総計を表示します。

ML-DML ULC 効率

破棄されたサブコマンド

Adaptive Server が最低限のログを取る DML のサブコマンドからログ・レコードを破棄できる (つまり、ログ・レコードが ULC から syslogs にフラッシュされなかった) 回数。

“Discarded Sub-Commands”の数が大きいほど、ロギングの量を最小限にできるため、Adaptive Server は最低限のログを取る DML をより効果的に実行します。

ログに記録されたサブコマンド

Adaptive Server が最低限のログを取る DML のサブコマンドからログ・レコードを破棄できなかった (つまり、ULC のログ・レコードを syslogs にフラッシュする必要があった) 回数。

ML-DML サブコマンドの総数

“Discarded Sub-Commands”と“Logged Sub-Commands”の総数。

ULC セマフォ要求

ユーザ・タスクがすぐにセマフォを付与された回数、またはセマフォを待機しなければならなかった回数。“% of total”では、ULC セマフォ要求の総数に対して、セマフォを付与されたタスク数とセマフォを待機したタスク数の割合が、パーセンテージで示されます。この値は、SMP 環境の場合にだけ意味を持ちます。

“ULC Semaphore Requests”では、次の情報が出力されます。

- 付与 – タスクが要求してから、すぐに ULC セマフォを付与された回数。ULC に対する競合は発生しない。
- ローカル待機 – タスクが ULC への書き込みを行おうとして、セマフォ競合を発生させた回数。

- ULC セマフォ要求の総数 − サンプル間隔中に行われた ULC セマフォ要求の総数。この数には、付与された要求と待機しなければならなかった要求が含まれる。

ログ・セマフォ要求

キャッシュにあるトランザクション・ログの現在のページを保護するログ・セマフォの競合をレポートします。このデータは、SMP 環境でだけ意味を持ちます。

このカテゴリでは、次の情報が出力されます。

- 付与 − タスクがログ・セマフォを要求してから、すぐに付与された回数。“% of total”では、ログ・セマフォ要求の総数に対して、すぐに付与された要求の数の割合が、パーセンテージでレポートされる。
- ローカル待機 − 別のタスクによってログ・セマフォが既に取得されていたために、ULC をフラッシュしようとしたタスクがログ・セマフォを待機しなければならなかった回数。“% of total”では、ログ・セマフォ要求の総数に対して、ログ・セマフォを待機しなければならなかったタスク数の割合が、パーセンテージでレポートされる。
- グローバル待機 − 別のタスクがこのノードまたは別のノードでログ・セマフォを既に取得していたために、共有ディスク・クラスタ内で ULC をフラッシュしようとしたタスクがログ・セマフォを待機しなければならなかった回数。
- ログ・セマフォ要求の総数 − タスクがログ・セマフォを要求した回数。この数には、ただちに付与された要求と待機しなければならなかった要求が含まれる。

トランザクション・ログ書き込み

Adaptive Server がトランザクション・ログ・ページをディスクに書き込んだ回数の合計をレポートします。トランザクション・ログ・ページは、トランザクションがコミットされたとき (グループ・コミット・スリープの待機後) や現在のログ・ページが満杯になったときにディスクへ書き込まれます。

トランザクション・ログの割り付け

そのほかのページがトランザクション・ログに割り付けられた回数。このデータは、このセクションにあるほかのデータと比較する場合や、トランザクション・ログの成長率を追跡する場合に役立ちます。

ログ・ページ当たりの平均書き込み数

各ログ・ページがディスクに書き込まれた平均回数をレポートします。値は“count”カラムに出力されます。

スループットの高いアプリケーションでは、この数値ができるかぎり低くなるようにしてください。1 回のログ I/O は 2 ページのログを書き込めるので、トランザクション・ログが 2K の I/O を使用する場合の最低の値は 1 であり、4K のログ I/O では、最低の値は 0.5 であると考えられます。

スループットの低いアプリケーションでは、この数値が非常に高くなります。スループットが非常に低い環境では、トランザクションの完了ごとに 1 回の書き込みが発生する場合と同じくらい高くなります。

トランザクション管理の推奨事項の調整

sp_sysmon では、“Transaction Management”セクションの結果に基づいた推奨する調整について説明します。

インデックス管理

ノンクラスタード管理、ページ分割、インデックスの縮小などのインデックス管理アクティビティをレポートします。

出力例

```
Index Management
----------------

  Nonclustered Maintenance      per sec      per xact       count   % of total
  -------------------------  ------------  ------------  ----------  ----------
  Ins/Upd Requiring Maint         1996.3          25.3     1778674         n/a
    # of NC Ndx Maint              694.9           8.8      619122         n/a
    Avg NC Ndx Maint / Op            n/a           n/a     0.34808         n/a

  Deletes Requiring Maint         1922.5          24.3     1712946         n/a
    # of NC Ndx Maint              621.1           7.9      553387         n/a
Avg NC Ndx Maint / Op                n/a           n/a     0.32306         n/a

  RID Upd from Clust Split           0.0           0.0           0         n/a
    # of NC Ndx Maint                0.0           0.0           0         n/a

  Upd/Del DOL Req Maint              9.0           0.1        7989         n/a
    # of DOL Ndx Maint               9.2           0.1        8238         n/a
    Avg DOL Ndx Maint / Op           n/a           n/a     1.03117         n/a
```

```
  Page Splits                      11.7           0.1       10452         n/a
    Retries                         0.0           0.0           0        0.0 %
    Deadlocks                       0.0           0.0           0        0.0 %
    Add Index Level                 0.0           0.0           0        0.0 %

Page Shrinks                        0.1           0.0         124         n/a
Deadlocks                             %
Deadlock Retries Exceeded           0.0           0.0           0        0.0 %

  Index Scans                     per sec      per xact       count   % of total
  ----------------------  ------------  ------------  ---------  ----------
    Ascending Scans             69751.6         883.1    62148709       29.3 %
    DOL Ascending Scans        168653.5        2135.3   150270303       70.7 %
    Descending Scans                8.8           0.1        7884        0.0 %
    DOL Descending Scans           23.9           0.3       21271        0.0 %
                           ------------  ------------  ---------
  Total Scans                  238437.9        3018.8   212448167
```

ノンクラスタード管理

“Nonclustered Maintenance”では、1 つまたは複数のインデックスをメンテナンスする必要がある操作、または必要になる可能性がある操作の数がレポートされます。つまり、このカテゴリでは、Adaptive Server がインデックスの更新が必要かどうかを最低限チェックする必要がある操作の数がレポートされます。出力には、更新されたインデックスの数と 1 回の操作でメンテナンスされるインデックス数の平均値が含まれます。

クラスタード・インデックスと 1 つ以上のノンクラスタード・インデックスがあるテーブルでは、すべての挿入、すべての削除、一部の更新オペレーション、任意のデータ・ページ分割について、ノンクラスタード・インデックスを変更する必要があります。インデックス・メンテナンスの値が大きい場合は、インデックスをメンテナンスすることで Adaptive Server のパフォーマンスに対してどの程度の影響があるかを評価してください。インデックスを利用するデータ検索は高速になりますが、インデックスのメンテナンスを伴う更新処理は低速になります。メンテナンスする場合は、処理、I/O、インデックス・ページのロックがさらに必要になります。

ノンクラスタード・インデックスを評価する場合は、sp_sysmon の次の出力も関連します。

- 更新の総数、挿入と削除、およびページ分割の数と種類についての情報。「トランザクションの詳細」(65 ページ) と「ページ分割」(82 ページ) を参照。
- ロック競合についての情報。「ロックの詳細」(95 ページ) を参照。
- アドレス・ロック競合についての情報。「アドレス・ロック競合」(47 ページ) および「排他アドレスと共有アドレス」(97 ページ) 参照。

たとえば、実行された挿入の数を、発生したメンテナンス操作の数と比較できます。メンテナンス操作、ページ分割、リトライが比較的多く発生している場合は、アプリケーション内でインデックスが有用かどうかを検討してください。

詳細については、『パフォーマンス＆チューニング・シリーズ：ロックと同時実行制御』の「第 5 章 インデックス」を参照してください。

メンテナンスが必要な挿入/更新

“Ins/Upd Requiring Maint”セクションのデータによって、挿入オペレーションや更新オペレーションが、全ページロック・テーブルのインデックスにどのように影響を及ぼすかについての情報が提供されます。たとえば、3 つのノンクラスタード・インデックスがあるクラスタード・テーブルに対する挿入では、3 つのインデックスすべてを更新する必要があるので、ノンクラスタード・インデックスに対するメンテナンスで発生する操作数の平均値は 3 です。

しかし、同じテーブルに対する更新に必要なメンテナンス操作は、1 つだけ (キーの値が変更されたインデックスに対する操作) です。

- Ins/Upd Requiring Maint － 1 つまたは複数のインデックスの変更を必要とする可能性があり、インデックスを持つテーブルに対する挿入または更新オペレーション操作の数がレポートされる。
- # of NC Ndx Maint － 挿入または更新オペレーションの結果、メンテナンスが必要になるノンクラスタード・インデックスの数がレポートされる。
- Avg NC Ndx Maint/Op － 挿入または更新オペレーション 1 回当たりの、メンテナンスが必要になるノンクラスタード・インデックスの数の平均値がレポートされる。

データオンリーロック・テーブルでは、“Ins/Upd Requiring Maint”で挿入がレポートされ、“Upd/Del DOL Req Maint”で削除と挿入がレポートされます。

メンテナンスが必要な削除

“Deletes Requiring Maint”セクションでは、削除オペレーションがどのように全ページロック・テーブルのインデックスに影響したかがレポートされます。

- Deletes Requiring Maint － 1 つ以上のインデックスに対する変更を必要とする可能性のある削除オペレーションの数がレポートされる。
- # of NC Ndx Maint － 削除オペレーションの結果、メンテナンスが必要になったノンクラスタード・インデックスの数がレポートされる。
- Avg NC Ndx Maint/Op － 削除オペレーション 1 回当たりの、メンテナンスが必要になるノンクラスタード・インデックスの数の平均値がレポートされる。

「削除」(69 ページ) を参照してください。

クラスタ分割からのロー ID の更新

クラスタード・インデックスがある全ページロック・テーブル内におけるページ分割によって発生した、インデックスのメンテナンス・アクティビティをレポートします。このような分割では、新しいデータ・ページに移動されたすべてのローに対してノンクラスタード・インデックスを更新する必要があります。

- RID Upd from Clust Split － ノンクラスタード・インデックスのメンテナンスが必要になるページ分割の総数がレポートされる。
- # of NC Ndx Maint － ロー ID 更新操作の結果、メンテナンスが必要になるノンクラスタード・ローの数がレポートされる。
- Avg NC Ndx Maint/Op － 各ページ分割で更新されたノンクラスタード・インデックスのエントリ数の平均値がレポートされる。

データオンリーロック・テーブルでのメンテナンスが必要な更新および削除

“Upd/Del DOL Req Maint”セクションでは、更新および削除オペレーションがどのようにデータオンリーロック・テーブルのインデックスに影響したかがレポートされます。

- “Upd/Del DOL Req Maint”では、1 つ以上のインデックスに対する変更を必要とする可能性のある更新および削除オペレーションの数がレポートされる。
- “# of DOL Ndx Main”では、更新または削除オペレーションの結果、メンテナンスが必要になったインデックスの数がレポートされる。
- “Avg DOL Ndx Maint/Op”では、更新または削除オペレーション 1 回当たりの、メンテナンスが必要になるインデックスの数の平均値がレポートされる。

ページ分割

新しいロー用の十分な空きがなかったために、データ・ページ、クラスタード・インデックス・ページ、ノンクラスタード・インデックス・ページのいずれかに対して実行されたページ分割の数。

クラスタード・インデックスのある全ページロック・テーブルに、データ・ローを挿入する場合、ローはキーの値に従って物理的な順序で配置される必要があります。インデックス・ローも同様に物理的な順序でページに配置される必要があります。新しいロー用の空き領域が不足している場合は、Adaptive Server によってページが分割され、新しいページが割り付けられていくつかのローが新しいページに移動されます。ページ分割では、付加されたページ上の親のインデックス・ページとページ・ポインタが更新され、ロック競合が追加されるので、オーバヘッドが発生します。クラスタード・インデックスの場合は、ページ分割でも、新しいページ上のローを参照するすべてのノンクラスタード・インデックスが更新されます。

詳細については、『パフォーマンス＆チューニング・シリーズ：ロックと同時実行制御』の「第 5 章 インデックス」を参照してください。

昇順キー挿入でのページ分割の削減

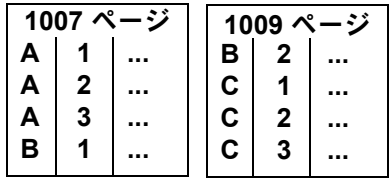
“Page Splits”の値が大きいときに、作成したアプリケーションが、複合キー上にクラスタード・インデックスがある全ページロック・テーブルに値を挿入すると、これらのインデックス・ページ分割ポイントを変更する特殊な最適化が行われ、ページ分割の数を減らすことができる場合があります。

特殊な最適化では、ページ分割を減らし、データ・ページの全体に対して書き込みが実行されることを目的としています。この最適化は、1 つ目のキーがテーブルで使用され、2 つ目のカラムが値を増やす場合の単位となっている複合キーがあるクラスタード・インデックスにだけ影響します。

デフォルトのデータ・ページ分割

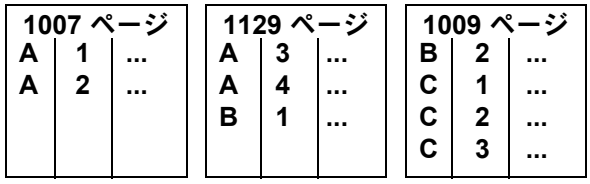
テーブル sales には、store_id と customer_id にクラスタード・インデックスがあります。3 つの店舗 (A、B、C) があり、各店舗では昇順の数値を使用して顧客レコードを追加します。テーブルには、キーの値が A,1、A,2、A,3、B,1、B,2、C,1、C,2、C,3 のローが含まれ、各ページには、図 2-2 に示す 4 つのローがあります。

図 2-2: 挿入前のクラスタード・テーブル



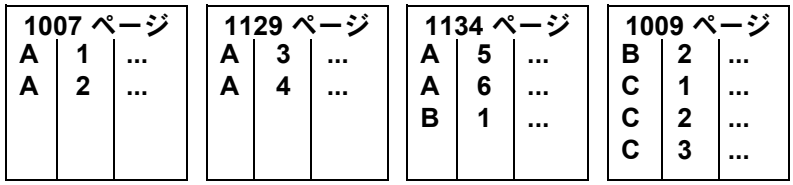
通常のページ分割メカニズムを使用して“A,4”を挿入すると、図 2-3 に示すように、新しいページの割り付けが発生してローの半分が新しいページに移動され、新しいローが所定の位置に挿入されます。

図 2-3: 挿入によって発生したページ分割



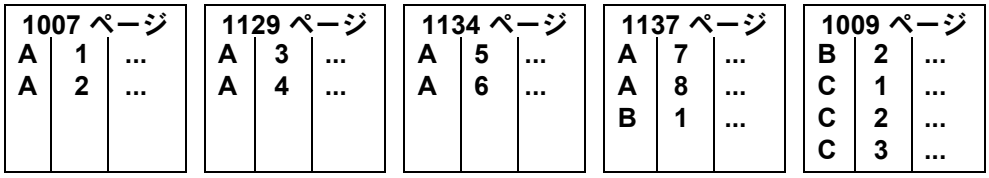
“A,5”が挿入された場合はページ分割は必要ありませんが、“A,6”が挿入された場合は、図 2-4 に示す別の分割が実行されます。

図 2-4: 別の挿入によって発生した別のページ分割



A,7 と A,8 を追加すると、さらに図 2-5 に示す別のページ分割が発生します。

図 2-5: ページ分割の続き

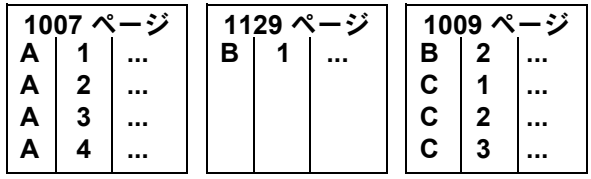


データ・ページ分割の詳細については、『パフォーマンス＆チューニング・シリーズ：基本』の「第 12 章 インデックスの働き」を参照してください。

昇順挿入の効果

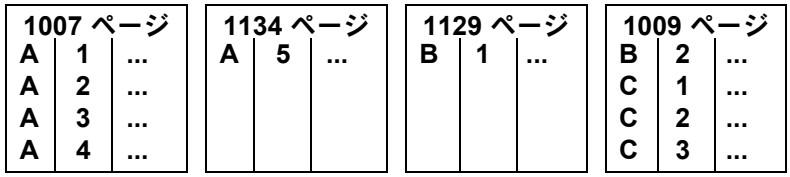
テーブルについて昇順挿入モードを設定し、ページの途中ではなく、挿入されたローでページを分割できます。図 2-2 (83 ページ) に示される元のテーブルに“A,4”を挿入すると、挿入ポイントで分割が発生し、図 2-6 に示すように、挿入したページにある残りのローが新しく割り付けたページへ移動されます。

図 2-6: 昇順挿入モードを使った最初の挿入



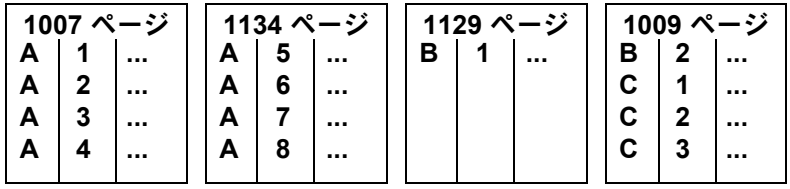
“A,5”を挿入すると、図 2-7 に示すように新しいページ割り付けが発生します。

図 2-7: 昇順挿入の追加によって発生したページ割り付け



“A,6”、“A,7”、“A,8”を追加すると、図 2-8 に示すように新しいページが満杯になります。

図 2-8: 追加の挿入による新しいページへの書き込み



テーブルに対する昇順挿入モードの設定

次のコマンドを使用すると、sales テーブルの昇順挿入モードがオンになります。

```
dbcc tune (ascinserts, 1, "sales")
```

昇順挿入モードをオフにするには、次のコマンドを使用します。

```
dbcc tune (ascinserts, 0, "sales")
```

これらのコマンドは、status2 の sysindexes ビットを更新します。

テーブルに対してランダムな挿入が実行される場合や、バッチ・ジョブ中に順序に従って挿入が実行される場合は、バッチ・ジョブの実行中にだけ dbcc tune (ascinserts) を有効にすることをおすすめします。

リトライとデッドロック

デッドロックの原因となったインデックス・ページの分割と縮小の数。Adaptive Server には、インデックス・ページのデッドロックによってトランザクションがロールバックされないようにする「デッドロック・リトライ」と呼ばれるメカニズムがあります。“Retries”では、Adaptive Server がこのメカニズムを使った回数がレポートされます。

インデックス・ページでのデッドロックは、2 つあるトランザクションのそれぞれが、互いにもう 1 つのトランザクションが保持しているロックを取得する必要がある場合に発生します。データ・ページでは、デッドロックが発生すると、1 つのプロセス (累積 CPU 時間が最も少ないプロセス) がデッドロック・ビクティムになるように選択され、プロセスがロールバックされます。

インデックスのデッドロックが発生するまでに、トランザクションはデータ・ページを更新し、データ・ページのロックを保持しているため、トランザクションがロールバックされると、オーバヘッドが発生します。

ページの分割や縮小によって発生したインデックスのデッドロックの大部分は、どちらのトランザクションも、1 組のインデックスのロックを断念し、インデックスのスキャンを再び開始すれば正常に実行されます。どちらか一方のプロセスによるインデックスのロックが解放され (データ・ページ上のロックは保持されます)、Adaptive Server は、インデックスのルート・ページからインデックスをたどって、再びインデックスをスキャンします。

通常は、スキャンによって分割が必要なインデックス・ページが見つかるまでに、もう一方のトランザクションが完了し、デッドロックは発生しません。デフォルトでは、ページの分割や縮小が原因となるインデックスのデッドロックのすべてについて、トランザクションがデッドロックしているとみなされてロールバックされるまでに 5 回のリトライが実行されます。

sp_configure "deadlock retries" を使用したデッドロック・リトライ数のデフォルト値を変更する方法については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

デッドロック・リトライのメカニズムは、データ・ページ上で通常よりも少しだけ時間が長いロックを発生させ、ロックとオーバヘッドの増大を引き起こします。しかし、デッドロック・リトライによって、デッドロックが原因でロールバックされるトランザクションの数が減ります。デフォルトの設定は、データ・ページのロックをより長時間保持するオーバヘッドと、再発行の必要があるトランザクションのロールバックの両方を、うまく調整します。

インデックスのデッドロックの数値が大きい場合は、インデックス B ツリーの小さな領域で多くの競合が発生しています。

作成したアプリケーションのデッドロックの数値が大きい場合は、インデックスを再作成するときに fillfactor を使用し、ページ分割を減らします。

詳細については、『パフォーマンス＆チューニング・シリーズ：ロックと同時実行制御』の「第 5 章 インデックス」を参照してください。

インデックス・レベルの追加

新しいインデックス・レベルが追加された回数。インデックス・レベルの追加が発生する頻度は低いので、ほとんどの場合、この値は 0 になっているはずです。サンプル間隔中に、空のテーブル、またはインデックスを持つサイズの小さなテーブルへの挿入が実行された場合は、この値が 1 または 2 になる可能性があります。

ページの縮小

インデックス・ローの削除によってインデックス・ページが縮小された回数。縮小は、インデックスのロックが原因でオーバヘッドを発生させ、隣接したページ上のポインタを更新することが必要になります。“count”で、繰り返されている値が 0 より大きい場合は、インデックス内に多くのページがあり、削除または更新オペレーションによって各ページに含まれるローの数が相当に少なくなっていることを示しています。縮小の数値が大きい場合は、インデックスの再構築を検討してください。

インデックス・スキャン

ロック・スキームによる前方スキャンおよび後方スキャンをレポートします。

- Ascending Scans － 全ページロック・テーブルで前方スキャンが実行された回数がレポートされる。
- DOL Ascending Scans － データオンリーロック・テーブルで前方スキャンが実行された回数がレポートされる。
- Descending Scans － 全ページロック・テーブルで後方スキャンが実行された回数がレポートされる。
- DOL Descending Scans － データオンリーロック・テーブルで後方スキャンが実行された回数がレポートされる。

前方スキャンと後方スキャンの詳細については、『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 5 章 インデックス」を参照してください。

メタデータ・キャッシュ管理

3 種類のメタデータ・キャッシュ (オブジェクト、インデックス、データベース) についての情報が格納される、メタデータ・キャッシュの使用状況をレポートします。さらにこのセクションでは、サンプル間隔中にアクティブだったオブジェクト記述子、インデックス記述子、およびデータベース記述子の数、およびサーバが最後に起動されてから使用された記述子の最大数もレポートされます。また、オブジェクト・メタデータ・キャッシュとインデックス・メタデータ・キャッシュのスピンロック競合についてもレポートされます。

出力例

```
Metadata Cache Management
-------------------------
  Metadata Cache Summary            per sec   per xact     count    % of total
  --------------------------       ---------  --------  --------    ---------
  Open Object Usage
    Active                              n/a       n/a      4604           n/a
    Max Ever Used Since Boot            n/a       n/a      4612           n/a
    Free                                n/a       n/a     25396           n/a
    Reuse Requests
      Succeeded                         n/a       n/a         0           n/a
      Failed                            n/a       n/a         0           n/a

  Open Index Usage
    Active                              n/a       n/a      2747           n/a
    Max Ever Used Since Boot            n/a       n/a      2753           n/a
    Free                                n/a       n/a      2253           n/a
    Reuse Requests
      Succeeded                         n/a       n/a         0           n/a
      Failed                            n/a       n/a         0           n/a

  Open Partition Usage
    Active                              n/a       n/a      2747           n/a
    Max Ever Used Since Boot            n/a       n/a      2753           n/a
    Free                                n/a       n/a      2253           n/a
    Reuse Requests
      Succeeded                         n/a       n/a         0           n/a
      Failed                            n/a       n/a         0           n/a

  Open Database Usage
    Active                              n/a       n/a        32           n/a
    Max Ever Used Since Boot            n/a       n/a        32           n/a
    Free                                n/a       n/a        18           n/a
    Reuse Requests
      Succeeded                         n/a       n/a         0           n/a
      Failed                            n/a       n/a         0           n/a

  Object Manager Spinlock Contention    n/a       n/a       n/a          1.3 %

  Object Spinlock Contention            n/a       n/a       n/a          0.0
  Index Spinlock Contention             n/a       n/a       n/a          0.0 %

  Index Hash Spinlock Contention        n/a       n/a       n/a          0.0
  Partition Spinlock Contention         n/a       n/a       n/a          0.0 %

  Partition Hash Spinlock Contention    n/a       n/a       n/a          0.0 %
```

オープンなオブジェクト、インデックス、データベースの使用率

これらのセクションは、それぞれ 3 種類のメタデータ・キャッシュについての同じ情報から構成されます。次の情報が出力されます。

- “Active”では、サンプル間隔中にアクティブだったオブジェクト、インデックス、またはデータベースの数がレポートされる。
- “Max Ever Used Since Boot”では、Adaptive Server が最後に再起動されてから使用された記述子の最大数がレポートされる。
- “Free”では、キャッシュ内の解放された記述子の数がレポートされる。
- “Reuse Requests”では、再利用できる記述子をキャッシュ内で検索する必要があった回数がレポートされる。
    - “Failed”は、キャッシュ内のすべての記述子が使用され、要求を発行したクライアントがエラー・メッセージを受信したことを意味する。
    - “Succeeded”は、要求が、再利用できる記述子をキャッシュ内に発見したことを意味する。“Succeeded”はクライアントがエラー・メッセージを受信しなかったことを意味するが、Adaptive Server は、再利用できる記述子をキャッシュ内で探し、ディスクからメタデータ情報を読み込むための特別な作業を実行している。

この情報を使用して、表 2-1 に示すように、設定パラメータ number of open indexes、number of open objects、number of open databases を設定できます。

表 2-1: メタデータ・キャッシュの使用率についての情報をもとにした対処方法

sp_sysmon の出力	対処法
“Free”記述子の数が多い	パラメータの設定値を小さくする
“Free”記述子の数が非常に少ない	パラメータの設定値を大きくする
“Reuse Requests Succeeded”が 0 ではない	パラメータの設定値を大きくする
“Reuse Requests Failed”が 0 ではない	パラメータの設定値を大きくする

すぐに破棄された記述子

Adaptive Server がすぐに破棄したすべての記述子の数。設定パラメータの数によって、Adaptive Server が使用する記述子の数が決まります (o/s file descriptors や txn to pss ratio など)。『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

オブジェクト・マネージャのスピンロック競合

これは、オブジェクト記述子の内部状態を管理するために使用されるサーバワイドなスピンロックです。

このスピンロックに対する競合が 10% を超える場合は、dbcc tune(des_bind) を使用して、競合を減らし、サーバのスケーラビリティを向上します。『ASE リファレンス・マニュアル』の dbcc tune(des_bind) を参照してください。

オブジェクトとインデックスのスピンロック競合

オブジェクト記述子のキャッシュとインデックス記述子のキャッシュ上で発生したスピンロック競合をレポートします。この情報を使用して、設定パラメータの open object spinlock ratio と open index spinlock ratio をチューニングできます。レポートされた競合が 3% 以上の場合は、該当するパラメータの値を減らし、シングル・スピンロックによって保護されるオブジェクトやインデックスの数よりも少なくなるようにしてください。

インデックス・ハッシュのスピンロック競合

インデックス・メタデータ・キャッシュのハッシュ・テーブル上でのスピンロックの競合をレポートします。この情報を使用して、open index hash spinlock ratio 設定パラメータをチューニングできます。レポートされた競合が 3% よりも多い場合は、このパラメータの値を減らしてください。

sp_monitorconfig による、メタデータ・キャッシュの使用状況情報の検索

sp_monitorconfig は、特定の共有サーバ・リソース上でのメタデータ・キャッシュの使用状況情報を表示します。この情報には、次のような統計値が含まれます。

- 同時にオープンできるデータベース、オブジェクト、インデックスの数
- 参照整合性クエリが使用する補助スキャン記述子の数
- 未使用の記述子とアクティブな記述子の数
- アクティブな記述子の割合
- サーバが起動されてから使用された記述子の最大値
- プロシージャ・キャッシュの現在のサイズと、実際に使用された量

たとえば、number of open indexes 設定パラメータが 500 に設定されている場合、ピーク時に sp_monitorconfig を実行すると、インデックス記述子に対する実際のメタデータ・キャッシュ使用状況を正確に測定したデータが表示されます。

```
1> sp_monitorconfig "number of open indexes"

Usage information at date and time: Apr 22 2002  2:49PM.
Name                      Num_free    Num_active  Pct_act  Max_Used
Reuse_cnt   Instance_Name
------------------------   ----------  ----------  -------  -----------
----------  --------------------
number of open                     217         283    56.60          300
         0                  NULL
```

このレポートでは、Adaptive Server のオープン・インデックス数が 500 に設定されているにもかかわらず、サーバが前回起動されてから使用されたオープン・インデックスの最大値が 300 と報告されています。したがって、number of open indexes 設定パラメータを 330 に再設定できます。これは、最大使用インデックス記述子数 300 に 10% を加えた数です。

また、sp_monitorconfig 'procedure cache size' を使用して、プロシージャ・キャッシュの現在のサイズを調べることもできます。これは、プロシージャ・キャッシュの領域の大きさと、これまでにプロシージャ・キャッシュが実際に使用した最大量を表します。たとえば、次のサーバでは、プロシージャ・キャッシュが 20,000 ページに設定されています。

```
1> sp_configure "procedure cache size"

option_name                    config_value run_value
------------------------------ ------------ ---------
procedure cache size                  20000      3271
```

しかし、sp_montorconfig "procedure cache size" を実行すると、次のように、プロシージャ・キャッシュがこれまで使用した最大ページ数は 14241 ページであることが判明しました。したがって、プロシージャ・キャッシュの実行値を現在より低く設定して、メモリを節約できます。

```
1> sp_monitorconfig "procedure cache size"

Usage information at date and time: Apr 22 2002  2:49PM.
Name                      Num_free    Num_active  Pct_act  Max_Used
Reuse_cnt   Instance_Name
------------------------  ----------  ----------  -------  -----------
----------  --------------------
procedure cache                 5878       14122    70.61        14241
          0                   NULL
```

パーティションのスピンロック競合

パーティションのスピンロック競合の量を示します。キャッシュを別のパーティションに分けることで、パーティションのスピンロック競合を減らします。『システム管理ガイド 第 2 巻』の「第 5 章 メモリの使い方とパフォーマンス」と『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 2 章 ロックの設定とチューニング」を参照してください。

パーティションのハッシュ・スピンロック競合

パーティションのハッシュ・スピンロック競合の量を示します。『システム管理ガイド 第 2 巻』の「第 5 章 メモリの使い方とパフォーマンス」と『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 2 章 ロックの設定とチューニング」を参照してください。

ロック管理

ロック、デッドロック、ロックの拡大、ロックの競合をレポートします。

出力例

```
Lock Management
---------------

Lock Summary                 per sec      per xact       count   % of total
-------------------------   ------------   -----------   ---------   ---------
Total Lock Requests          279236.5        6878.6   167541893         n/a
Avg Lock Contention               1.2           0.0         691       0.0 %
Cluster Locks Retained            0.0           0.0           0       0.0 %
Deadlock Percentage               0.0           0.0           0       0.0 %

Lock Detail                  per sec      per xact       count   % of total
-------------------------   ------------   -----------   ---------   ---------

Table Lock Hashtable
  Lookups                     9571.4         235.8     5742818         n/a
  Avg Chain Length               n/a           n/a     0.00981         n/a
  Spinlock Contention            n/a           n/a         n/a        0.2 %

Exclusive Table
  Granted                       14.3           0.4        8580      100.0 %
  Waited                         0.0           0.0           0        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total EX-Table Requests          14.3           0.4        8580        0.0 %

Shared Table
  Total SH-Table Requests         0.0           0.0           0         n/a

Exclusive Intent
  Granted                       139.7           3.4       83793      100.0 %
  Waited                          0.0           0.0           0        0.%
-------------------------  ------------  ------------  ----------  ----------
Total EX-Intent Requests         139.7           3.4       83793        0.1 %

Shared Intent
  Granted                      9412.9         231.9     5647759      100.0 %
  Waited                          0.0           0.0           0        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total SH-Intent Requests        9412.9         231.9     5647759        3.4 %

Page & Row Lock HashTable
  Lookups                    173637.2        4277.3   104182335         n/a
  Avg Chain Length               n/a           n/a     0.01387         n/a
  Spinlock Contention            n/a           n/a         n/a        0.4 %
```

```
Exclusive Page
  Granted                      306.5           7.6      183911      100.0 %
  Waited                         0.0           0.0           2        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total EX-Page Requests         306.5           7.6      183913        0.1 %

Update Page
  Granted                    19052.6         469.3    11431583      100.0 %
  Waited                         0.1           0.0          60        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total UP-Page Requests       19052.7         469.3    11431643        6.8 %

Shared Page
  Granted                    63587.2        1566.4    38152347      100.0 %
  Waited                         0.0           0.0           0        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total SH-Page Requests       63587.2        1566.4    38152347       22.8 %

Exclusive Row
  Granted                       33.9           0.8       20326      100.0 %
  Waited                         0.0           0.0           0        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total EX-Row Requests            33.9           0.8       20326        0.0 %

Update Row
  Granted                       10.1           0.2        6030       98.2 %
  Waited                         0.2           0.0         108        1.8 %
-------------------------  ------------  ------------  ----------  ----------
Total UP-Row Requests            10.2           0.3        6138        0.0 %


Shared Row
  Granted                    88431.3        2178.4    53058781      100.0 %
  Waited                         0.0           0.0           0        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total SH-Row Requests        88431.3        2178.4    53058781       31.7 %

Next-Key
  Total Next-Key Requests        0.0           0.0           0         n/a

Address Lock Hashtable
  Lookups                    98247.7        2420.2    58948613         n/a
  Avg Chain Length               n/a           n/a     0.00019         n/a
  Spinlock Contention            n/a           n/a         n/a        0.5 %

Exclusive Address
  Granted                     2758.9          68.0     1655359      100.0 %
  Waited                         0.6           0.0         380        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total EX-Address Requests     2759.6          68.0     1655739        1.0 %
```

```
Shared Address
  Granted                       95487.9        2352.2    57292733      100.0 %
  Waited                            0.2           0.0         141        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total SH-Address Requests       95488.1        2352.2    57292874       34.2 %

Last Page Locks on Heaps
  Granted                         799.4          19.7      479637      100.0 %
  Waited                            0.0           0.0           0        0.0 %
-------------------------  ------------  ------------  ----------  ----------
Total Last Pg Locks               799.4          19.7      479637      100.0 %

Deadlocks by Lock Type           per sec      per xact       count   % of total
-----------------------   ------------   -----------   ---------   ---------
Total Deadlocks                   0.0           0.0           0          n/a

Deadlock Detection
  Deadlock Searches               0.0           0.0          19          n/a
  Searches Skipped                0.0           0.0           0        0.0 %
  Avg Deadlocks per Search          n/a           n/a     0.00000          n/a

Lock Promotions
  Total Lock Promotions           0.0           0.0           0          n/a
Lock Timeouts by Lock Type       per sec      per xact       count   % of total
-----------------------   ------------   -----------   ---------   ---------
Total Timeouts                    0.0           0.0           0          n/a



Cluster Lock Summary             per sec      per xact       count   % of total
-----------------------   ------------   -----------   ---------   ---------

Physical Locks Summary           per sec      per xact       count   % of total
-----------------------   ------------   -----------   ---------   ---------
No physical locks are acquired

Logical Locks Summary            per sec      per xact       count   % of total
-----------------------   ------------   -----------   ---------   ---------

Object Locks Summary             per sec      per xact       count   % of total
-----------------------   ------------   -----------   ---------   ---------
```

サンプル間隔中にロックの拡大が発生しなかった場合は、ディテール・ローをレポートします (たとえば、上記の出力例の“Deadlocks by Lock Type”)。

ロックの概要

サンプル間隔中に発生したロック・アクティビティについての大まかな情報を出力します。

- “Total Lock Requests”では、ロック要求の総数がレポートされる。
- “Avg Lock Contention”では、ロック要求の総数に対するロック競合が発生した平均回数を示す割合が、パーセンテージでレポートされる。

  値が不十分であるためにロック競合の重大度を調べることができない場合は、monSysWaits、monProcessWaits、monLocks モニタリング・テーブルを問い合わせるか、sp_object_stats を使用すると、ロック競合の重大度を特定できます。詳細については、『パフォーマンス＆チューニング・シリーズ：モニタリング・テーブル』を参照してください。sp_object_stats の詳細については、『Adaptive Serverリファレンス・マニュアル：プロシージャ』を参照してください。ロックの動作をチューニングする方法の詳細については、『パフォーマンス＆チューニング・シリーズ：ロックと同時実行制御』を参照してください。
- “Cluster Locks Retained”では、保持されるクラスタ・ロックの数がレポートされる。値が大きい場合はアプリケーションが分割され、ほとんどのクラスタ・ロックが現在のインスタンスに既に付与されていることを示す。値が小さい場合は、オブジェクトのスキーマと作業負荷パターンを再評価し、複数のインスタンス間でのデータ・アクセスの競合を減少させる。

  monCLMobjectActivity には、各プロジェクトのクラスタレベルのアクティビティの値が含まれます。これらの値は、オブジェクト・レベルでロックについて決定する上で役立ちます。
- “Deadlock Percentage”では、ロック要求の総数に対するデッドロック数の割合が、パーセンテージでレポートされる。この値が大きい場合は、「ロックのタイプごとのデッドロック」(98 ページ) を参照。

ロックの詳細

アプリケーションがロック競合またはデッドロックに関連する問題の原因となっているかどうかを調べることができます。

ロック競合は、Adaptive Server のパフォーマンスに大きな影響を与える可能性があります。テーブル上に排他テーブル・ロックがある間は、別のタスクからはそのテーブルにアクセスできないので、テーブル・ロックでは、ページ・ロックあるいはロー・ロックよりも多くのロック競合が発生します。あるタスクが排他テーブル・ロックを必要とする場合は、共有ロックが解放されるまで待機する必要があります。ロック競合の値が大きい場合は、sp_object_stats を実行して関連するテーブルを特定します。

“Lock Detail”では、次の種類のロックがレポートされます。

- Granted － 各ロック･タイプがアプリケーションにすぐに付与された回数。
- Waited － 各ロック・タイプが特定のロック・タイプを待機した回数。
- Lookups －
- Avg Hash Chain Length － サンプル間隔中のハッシュ・バケットごとの平均ロック回数。
- Spinlock Contention － エンジンでスピンロック競合が発生した回数。

“Lock Detail”は、以下のロック・タイプの `Granted` と `Waited` の値についてレポートします。

- 排他テーブル
- 意図的排他
- 意図的共有
- 排他ページ
- 排他ロー
- 更新ロー
- 共有ロー
- 共有アドレス
- ヒープ上での最終ページ・ロック

“Lock Detail”は、以下のロック・タイプの Avg Hash Chain Length、Lookups、Spinlock Contention についてレポートします。

- 共有テーブル
- ページ/ロー・ロック・ハッシュテーブル
- 更新ページ
- 共有ページ
- ネクスト・キー
- アドレス・ロックのハッシュ・テーブル

ロック・ハッシュ・テーブルのサイズは、lock hashtable size を使用して設定できます。ハッシュ・チェーンごとのロックの平均回数が 4 回よりも多い場合は、ハッシュ・テーブルのサイズを大きくすることを検討する。

“% of total”カラムには、付与されたロックまたは待機しなければならなかったロックのタイプの割合が、ロック要求の総数に対するパーセンテージで表示されます。

ただし、バルク・コピーを使った大量の挿入には、このガイドラインはあてはまらない。大規模なバルク・コピーでは、ハッシュ・チェーンのロックが長くなることがある。

詳細については、『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 2 章 ロックの設定とチューニング」と『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

排他アドレスと共有アドレス

アドレス・ロックがすぐに付与された回数、またはタスクがアドレス・ロックを待機しなければならなかった回数。アドレス・ロックは、全ページロック・テーブルのインデックス・ページ上で保持されます。インデックス・ページでロックが保持されていると、そのインデックス・ページが示しているすべてのデータ・ページへのアクセスがブロックされるので、アドレス・ロックは重大な影響を与える可能性があります。

ヒープ上での最終ページ・ロック

分割されたヒープ・テーブルまたは分割されていないヒープ・テーブルの、最終ページに対して試みられたロックをレポートします。レポートされるのは、全ページロック・テーブルについてだけです。

この情報からは、データオンリー・ロックの使用、分割、あるいは分割数の増加によるメリットを利用できるテーブルがシステム内にあるかどうかがわかります。複数のデータ・ページにまたがって挿入をランダムに分散させるクラスタード・インデックスの追加も役に立ちます。1 つまたは複数のテーブルに最終ページの競合についての問題が発生していることがわかっている場合は、モニタリング・テーブルを使用して、どのテーブルに問題が発生しているかを調べることができます。『パフォーマンス&チューニング・シリーズ：モニタリング・テーブル』を参照してください。

分割されていないヒープ・テーブル上での最終ページ・ロックの問題をパーティションを利用して解決する方法の詳細については、『パフォーマンス&チューニング・シリーズ：物理データベースのチューニング』の「第 1 章 データの物理的配置の制御」を参照してください。

テーブル・ロック・ハッシュテーブル

ロック・ハッシュ・テーブルでページ、ロー、またはテーブルのロックが検索された回数。“Table Lock Hashtable”がレポートする対象は以下のとおりです。

- Lookups
- Avg Chain Length
- Spinlock Contention

ロック・ハッシュ・テーブルのサイズは、設定パラメータ lock hashtable size を使用して設定できます。ハッシュ・チェーンごとのロックの平均回数が 4 回よりも多い場合は、ハッシュ・テーブルのサイズを大きくすることを検討する。

詳細については、『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 2 章 ロックの設定とチューニング」を参照してください。

ロックのタイプごとのデッドロック

特定タイプのデッドロックの数。“% of total”では、デッドロックの総数に対する各タイプのデッドロック数の割合が、パーセンテージでレポートされます。

デッドロックは、同一のデータベース内で多くのトランザクションが同時に実行された場合に発生する可能性があります。トランザクションどうしのロック競合が増えるに従って、デッドロックが発生しやすくなります。

このカテゴリでは、次のタイプのデッドロックについてのデータがレポートされます。

- 排他テーブル
- 共有テーブル
- 意図的排他
- 意図的共有
- 排他ページ
- 更新ページ
- 共有ページ
- 排他ロー
- 更新ロー
- 共有ロー
- 共有次キー
- 排他アドレス
- 共有アドレス
- その他のデッドロック

“Total Deadlocks”では、すべての種類のロックについてデータが集計されます。

このセクションの例のように、デッドロックがない場合は、sp_sysmon は詳しい情報を表示しないで、“Total Deadlocks”のローに 0 の値を出力するだけです。

デッドロックの発生場所を特定するには、次の方法をどちらか、または両方使用します。

- sp_object_stats を使用する。

  詳細については、『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 2 章 ロックの設定とチューニング」を参照してください。

- デッドロック情報の詳細をログに出力する。

  詳細については、『パフォーマンス&チューニング・シリーズ：ロックと同時実行制御』の「第 3 章 ロックのレポート」を参照してください。

デッドロックの検出

デッドロックを発見したデッドロック検索の数と、サンプル間隔中に省略されたデッドロック検索の数。

詳細については、『パフォーマンス＆チューニング・シリーズ：ロックと同時実行制御』の「第 3 章 ロックのレポート」を参照してください。

デッドロック検索

Adaptive Server によってサンプル間隔中にデッドロック検索が開始された回数。デッドロックが発生しないアプリケーションや、発生するレベルが非常に低いアプリケーションにとっては、デッドロックのチェックは時間がかかるオーバヘッドになります。このデータを「デッドロック検索当たりのデッドロックの平均数」(99 ページ)と合わせて使用し、Adaptive Server がデッドロックをチェックする頻度が高すぎないかどうかを調べることができます。

省略された検索

デッドロック・チェックを実行するために開始されたタスクが、すでに実行中のデッドロック・チェックを検出したために、チェックを省略した回数。“% of total”では、検索の総数に対する省略されたデッドロック検索数の割合が、パーセンテージでレポートされます。

プロセスがロック競合によってブロックされると、そのプロセスは設定パラメータ deadlock checking period を使って設定された時間だけ待機します。この時間が経過すると、プロセスはデッドロック・チェックを開始します。デッドロック検索がすでに処理中の場合は、プロセスは検索を省略します。

複数のデッドロック検索が省略されていても、一部のデッドロック検索でデッドロックが検出されている場合は、パラメータの値を少しだけ増やしてください。多くのデッドロック検索が省略されているときに、デッドロックが発生していないかまたは非常に少ない場合は、パラメータの値を大幅に増やします。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

デッドロック検索当たりのデッドロックの平均数

1 回のデッドロック検索で検出されたデッドロックの平均値がレポートされます。

このカテゴリでは、Adaptive Server によるチェックの頻度が大きすぎないかどうかが測定されます。使用しているアプリケーションでデッドロックがほとんど発生しない場合は、deadlock checking period 設定パラメータの値を増やし、タスクがデッドロックを検索する頻度を調整します。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

ロックの拡大

次の拡大が発生した回数。

- “Ex-Page to Ex-Table”－ 排他ページから排他テーブルへの拡大
- “Sh-Page to Sh-Table”－ 共有ページから共有テーブルへの拡大
- “Ex-Row to Ex-Table”－ 排他ローから排他テーブルへの拡大
- “Sh-Row to Sh-Table”－ 共有ローから共有テーブルへの拡大
- “Sh-Next-Key to Sh-Table”－ 共有次キーから共有テーブルへの拡大

“Total Lock Promotions”では、1 秒当たりとトランザクション当たりのロックの拡大の平均値が、ロック拡大のタイプごとにレポートされます。

サンプル間隔中にロックの拡大が発生しなかった場合は、総数のローだけが出力されます。

ロックの拡大が発生しなかった場合は、sp_sysmon では詳細な情報は出力されません。

“Lock Promotions”のデータを使用して、次のようなことができます。

- 作成したアプリケーション内でのロックの拡大が、ロック競合やデッドロックの原因にならないかどうかを検出する。
- ロックの拡大に関連する変数をチューニングする前後に使用し、値の効果を調べる。

“Granted”と“Waited”のデータを調べ、競合の徴候を探してください。ロック競合の値が高く、ロックの拡大が頻繁に発生している場合は、関連するテーブルについて、ロック・プロモーション・スレッショルドの変更を検討します。

ロック・プロモーション・スレッショルドは、サーバワイドでも、テーブルごとでも設定できます。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

ロック・タイムアウトの情報

セッション・レベルあるいはサーバ・レベルのロック・タイムアウトのために、タスクがロックを待機しトランザクションがロールバックされた回数。ロックの種類を示すディテール・ローは、サンプル間隔中にロック・タイムアウトが発生した場合にのみ出力されます。ロック・タイムアウトが発生しない場合は、“Total Lock Time-outs”はすべての値が 0 として表示されます。

ロック・タイムアウトの詳細については、『パフォーマンス＆チューニング・シリーズ：ロックと同時実行制御』の「第 4 章 ロック・コマンドの使用」を参照してください。

クラスタ・ロックの概要

sp_sysmon では、クラスタ・ロックについて、次のアクティビティがレポートされます。

- Lock Garbage Collection － ロック・ガーベジ・コレクションを実行した回数。1 秒当たりのガーベジ・コレクションが大量にある場合は、クラスタでロックが不足している。
- Targeted Collection Success － ロック・ガーベジ・コレクションが目標のロック数を正常に再利用することができた回数。Cluster Edition は、ターゲット・ロック数は内部で決定し、設定可能なパラメータではない。1 秒当たりのガーベジ・コレクションの量が少ない場合は、クラスタでロックが不足しているか、利用できる CIPC メッセージ領域が不足している。
- Cluster Lock Requests － 保持ロックによって許可できず、クラスタ通信を必要とするユーザの物理、論理、またはオブジェクト・ロック要求の数。値が大きい場合は、クラスタ・トラフィック量が大きい。
- Local Master － ローカル・マスタを持つ物理、論理、またはオブジェクト・ロックの数。値が大きいほどロックとしては理想的で、クラスタ・ロック要求の処理中にオーバーヘッドが少なく済む場合がある。

  Cluster Edition でローカル・マスタが見つからない場合は、以下を実行します。

  - アプリケーション分割を再評価しなければならない場合がある。
  - 単一のインスタンスの負荷が大きすぎるため、リモート・マスタへのマスタの負荷分散が発生する。作業負荷のパターンとスキーマを変更する (データの分割など)。
- Lock Granted － 競合するタスク所有権のないクラスタ・ロックを所有するリモート・インスタンスを待機するクラスタ・ロック要求の数。`Lock Granted` の値が大きい場合は、クラスタ内の複数のインスタンスからのロック要求が競合している。アプリケーション分割を再評価する。
- Lock Waited － 競合するタスク所有権を持つクラスタ・ロックを所有するリモート・インスタンスを待機するクラスタ・ロック要求の数。`Lock Waited` の値が大きい場合は、ロック・レベルの競合が発生している。つまり、異なるインスタンス上の 2 つのタスクが、同じロックを取得しようとしているために競合している。問題を修正するには、アプリケーション分割を再評価する。
- Downgrade Req Recv － ローカル・インスタンスが所有するロックを求める場合に、リモート・インスタンスが送信したダウングレード要求の数を表示する。

データ・キャッシュ管理

sp_sysmon では、すべてのキャッシュについてまとめた情報がレポートされ、そのあとに各名前付きキャッシュの統計情報がレポートされます。

sp_sysmon では、デフォルトのデータ・キャッシュと各名前付きキャッシュについて、次のアクティビティがレポートされます。

- スピンロック競合
- 使用率
- ヒットおよびミスを含むキャッシュ検索
- 設定したすべてのプールに対するプールのターンオーバ
- クリーンな状態で超えたバッファ、すでに I/O 処理中のバッファ、ダーティ・バッファ・ウォッシングなどのバッファ・ウォッシングの動作
- 実行されたプリフェッチ要求と拒否されたプリフェッチ要求
- ダーティ・リード・ページ要求

sp_cacheconfig と sp_helpcache の出力を使用して、レポートのこのセクションのデータ分析に役立てることができます。sp_cacheconfig ではキャッシュとプールについての情報が、sp_helpcache ではキャッシュにバインドされたオブジェクトについての情報が提供されます。

これらのシステム・プロシージャの使用方法については、『システム管理ガイド 第2巻』の「第4章 データ・キャッシュの設定」を参照してください。

パフォーマンスの問題と名前付きキャッシュの詳細については、『パフォーマンス&チューニング・シリーズ：基本』の「第5章 メモリの使い方とパフォーマンス」を参照してください。

出力例

次の例は、sp_sysmon の“Data Cache Management”カテゴリの出力を示します。データの最初のブロック“Cache Statistics Summary”では、すべてのキャッシュについての情報が出力されます。sp_sysmon では、キャッシュごとにデータのブロックを分けてレポートされます。これらのブロックは、キャッシュ名で識別されます。ここに示す出力例にはデフォルト・データ・キャッシュのブロックしかありませんが、名前付きキャッシュを設定していればそのキャッシュのブロックも出力されます。

```
Data Cache Management
---------------------

Cache Statistics Summary (All Caches)
-------------------------------------
                              per sec       per xact       count   % of total
                           ----------     -----------    -------    --------
```

```
Cache Search Summary
  Total Cache Hits            39472.3        185026.5   11841696     99.9 %
  Total Cache Misses             33.8           158.4      10139      0.1 %
----------------------   ------------   ------------   --------
Total Cache Searches          39506.1        185184.9   11851835

Cache Turnover
  Buffers Grabbed               177.0           829.6      53097        n/a
  Buffers Grabbed Dirty           0.0             0.0          0      0.0 %

Cache Strategy Summary
  Cached (LRU) Buffers        35931.1        168427.1   10779332     98.6 %
  Discarded (MRU) Buffers       511.7          2398.4     153500      1.4 %

Large I/O Usage
  Large I/Os Performed           46.9           219.9      14071     94.5 %

Large I/Os Denied due to
  Pool < Prefetch Size            2.5            11.6        745      5.0 %
  Pages Requested
  Reside in Another
  Buffer Pool                     0.3             1.3         80      0.5 %
----------------------   ------------   ------------   --------
Total Large I/O Requests        49.7           232.8      14896

Large I/O Effectiveness
  Pages by Lrg I/O Cached       171.3           803.1      51398        n/a
  Pages by Lrg I/O Used         567.2          2658.7     170157    331.1 %

Asynchronous Prefetch Activity
  APFs Issued                     9.6            45.0       2878      1.2 %
  APFs Denied Due To
    APF I/O Overloads             0.0             0.0          0      0.0 %
    APF Limit Overloads           0.0             0.0          0      0.0 %
    APF Reused Overloads          0.0             0.0          0      0.0 %
  APF Buffers Found in Cache
      With Spinlock Hel           1.7             7.8        501      0.2 %
      W/o Spinlock Held         787.1          3689.5     236128     98.6 %
----------------------   ------------   ------------   --------
Total APFs Requested            798.4          3742.3     239507

  Other Asynchronous Prefetch Statistics
    APFs Used                     9.4            43.9       2808        n/a
    APF Waits for I/O             6.8            31.8       2034        n/a
    APF Discards                  0.0             0.0          0        n/a


Dirty Read Behavior
    Page Requests                 0.0             0.0          0        n/a
```

```
-------------------------------------------------------------------------------
Cache: default data cache
                              per sec        per xact      count   % of total
                            ----------     -----------    -------    --------
  Spinlock Contention             n/a             n/a        n/a        2.8 %

  Utilization                     n/a             n/a        n/a       82.9 %

  Cache Searches
    Cache Hits                32731.6        153429.6    9819492     100.0 %
      Found in Wash             288.4          1351.7      86508       0.9 %
    Cache Misses                 10.9            50.9       3257       0.0 %
-----------------------   ------------    ------------   --------
Total Cache Searches          32742.5        153480.5       9822749

Pool Turnover
  2  Kb Pool
    LRU Buffer Grab             123.0           576.7         36908    92.5 %
      Grabbed Dirty               0.0             0.0             0       0.0 %
  16 Kb Pool
    LRU Buffer Grab              10.0            47.1          3012     7.5 %
      Grabbed Dirty               0.0             0.0             0     0.0 %
-----------------------   ------------    ------------    ----------
Total Cache Turnover            133.1           623.8         39920

Buffer Wash Behavior
  Statistics Not Available - No Buffers Entered Wash Section Yet

Cache Strategy
  Cached (LRU) Buffers        30012.1        140681.9       9003640    98.3 %
  Discarded (MRU) Buffers       511.7          2398.4        153500     1.7 %

Large I/O Usage
  Large I/Os Performed           10.0            47.1          3012    80.2 %

  Large I/Os Denied due to
    Pool < Prefetch Size          0.8             3.9           247     6.6 %
    Pages Requested
    Reside in Another
    Buffer Pool                   1.7             7.8           498    13.3 %
-----------------------   ------------    ------------    ----------
Total Large I/O Requests         12.5            58.7          3757



Large I/O Detail
  4   Kb Pool
    Pages Cached                  0.0             0.0             0       n/a
    Pages Used                    0.0             0.0             0       n/a
  16  Kb Pool
    Pages Cached                 80.3           376.5         24096       n/a
```

```
    Pages Used                      76.1           356.5         22817    94.7 %



Dirty Read Behavior
      Page Requests                  0.0             0.0             0       n/a


------------------------------------------------------------------------------
Cache: pub_log_cache
                               per sec        per xact      count  % of total
                             ----------     -----------    ----------  -------
  Spinlock Contention              n/a             n/a           n/a    0.2 %

  Utilization                      n/a             n/a           n/a    9.1 %

  Cache Searches
    Cache Hits                  3591.9         16837.1       1077574  100.0 %
      Found in Wash                0.0             0.0             0   0.0 %
    Cache Misses                   0.0             0.0             0   0.0 %
-----------------------   ------------    ------------    ----------
Total Cache Searches            3591.9         16837.1       1077574

Pool Turnover
  2  Kb Pool
    LRU Buffer Grab                0.3             1.3            80    0.8 %
      Grabbed Dirty                0.0             0.0             0     0.0 %
  4  Kb Pool
    LRU Buffer Grab               33.9           158.9         10171   99.2 %
      Grabbed Dirty                0.0             0.0             0    0.0 %
-----------------------   ------------    ------------    ----------
Total Cache Turnover              34.2           160.2         10251

Buffer Wash Behavior
  Statistics Not Available - No Buffers Entered Wash Section Yet

Cache Strategy
  Cached (LRU) Buffers          2872.7         13465.9        861817  100.0 %


  Discarded (MRU) Buffers          0.0             0.0             0    0.0 %

Large I/O Usage
Large I/Os Performed              33.9           158.9         10171   99.2 %

Large I/Os Denied due to
  Pool < Prefetch Size             0.0             0.0             0    0.0 %
  Pages Requested
  Reside in Another
  Buffer Pool                      0.3             1.3            80    0.8 %
```

```
-----------------------   ------------    ------------    ----------
Total Large I/O Requests          34.2           160.2         10251

Large I/O Detail
  4   Kb Pool
    Pages Cached                  67.8           317.8         20342       n/a
    Pages Used                    67.8           317.8         20342   100.0 %

Dirty Read Behavior
    Page Requests                  0.0             0.0             0      n/a

--------------------------------------------------------------------------------
Cache: tempdb_data_cache
                              per sec        per xact      count  % of total
                            ----------     -----------    ----------  -------
  Spinlock Contention              n/a             n/a          n/a      0.0 %

  Utilization                      n/a             n/a          n/a      8.0 %

Cache Searches
  Cache Hits                  3148.8         14759.8       944630     99.3 %
    Found in Wash                7.9            37.0         2365      0.3 %
Cache Misses                    22.9           107.5         6882      0.7 %
-----------------------   ------------    ------------   ----------
Total Cache Searches          3171.7         14867.4       951512

Pool Turnover
  2  Kb Pool
    LRU Buffer Grab               6.8            31.8         2038     69.7 %
      Grabbed Dirty               0.0             0.0            0      0.0 %
  4  Kb Pool
    LRU Buffer Grab               0.1             0.4           24      0.8 %
      Grabbed Dirty               0.0             0.0            0      0.0 %
  16 Kb Pool
    LRU Buffer Grab               2.9            13.5          864     29.5 %
      Grabbed Dirty               0.0             0.0            0       0.0%


-----------------------   ------------    ------------   ----------
Total Cache Turnover              9.8            45.7         2926

Buffer Wash Behavior
  Statistics Not Available - No Buffers Entered Wash Section Yet


Cache Strategy
  Cached (LRU) Buffers        3046.3         14279.3       913875    100.0 %
  Discarded (MRU) Buffers        0.0             0.0            0      0.0 %

Large I/O Usage
  Large I/Os Performed            3.0            13.9          888    100.0 %
```

```
  Large I/Os Denied due to
    Pool < Prefetch Size           0.0             0.0            0       0.0 %
    Pages Requested
    Reside in Another
    Buffer Pool                    0.0             0.0            0       0.0 %
-----------------------   -------------    -------------   -----------
Total Large I/O Requests           3.0            13.9          888

  Large I/O Detail
    4   Kb Pool
      Pages Cached                 0.2             0.8           48         n/a
      Pages Used                   0.2             0.7           47      97.9 %
    16  Kb Pool
      Pages Cached                23.0           108.0         6912         n/a
      Pages Used                   2.9            13.5          864      12.5 %

Dirty Read Behavior
      Page Requests                 0.0             0.0            0       n/a
```

キャッシュ情報の概要 (すべてのキャッシュ)

“Cache Statistics Summary”のセクションには、デフォルトのデータ・キャッシュとすべての名前付きキャッシュの動作が 1 つにまとめられています。対応する情報は、データ・キャッシュごとに出力されます。「キャッシュごとのキャッシュ管理」(112 ページ) を参照してください。

キャッシュ検索の概要

キャッシュのヒットとミスについてまとめた情報を出力します。このデータを使用して、効果的なキャッシュ設計の概要を知ることができます。キャッシュ・ミスの数が多い場合は、各キャッシュの情報を調べてください。

- “Total Cache Hits”では、必要なページがキャッシュ内に見つかった回数がレポートされる。“% of total”では、キャッシュ検索の総数に対するキャッシュ・ヒット数の割合が、パーセンテージでレポートされる。
- “Total Cache Misses”では、必要なページがキャッシュ内に見つからなかったので、ディスクから読み込んだ回数がレポートされる。“% of total”では、キャッシュ検索の総数に対して、バッファがキャッシュに見つからなかった回数の割合が、パーセンテージでレポートされる。
- “Total Cache Searches”では、すべてのキャッシュのヒットとミスを合わせた、キャッシュ検索の総数がレポートされる。

キャッシュのターンオーバ

キャッシュのターンオーバの概要を示します。

- “Buffers Grabbed”では、すべてのキャッシュ内で置き換えられたバッファの数がレポートされる。“count”カラムには、Adaptive Server がキャッシュの LRU 側の終端からバッファをフェッチし、データベース・ページを置き換えた回数が示される。サーバが再起動されたばかりでバッファが空の場合は、空のバッファへのページの読み込みはここにはカウントされない。
- “Buffers Grabbed Dirty”では、バッファのフェッチによってキャッシュの LRU 側の終端にダーティ・ページが見つかり、バッファがディスクに書き込まれるまでバッファのフェッチを待機しなければならなかった回数がレポートされる。この値が 0 以外の場合は、どのキャッシュが影響を受け、どのディスク I/O が適切に実行されているかの確認作業を行い、プール内のウォッシュ・サイズを大きくする。その結果は、パフォーマンスに対する重大な影響を表す。

キャッシュ方式の概要

使用されたキャッシュ方式についてまとめた情報を出力します。

- “Cached (LRU) Buffers”では、すべてのキャッシュの MRU (最も最近に使用された) 方式と LRU (最も長い間使用されていない) 方式のチェーンの先頭に配置されたバッファの総数がレポートされる。
- “Discarded (MRU) Buffers”では、使い捨て方式に従った、すべてのキャッシュ内のバッファ、つまりウォッシュ・マーカに配置されたバッファの総数がレポートされる。

大容量 I/O の使用率

すべてのキャッシュでの大容量 I/O 要求についてまとめた情報を出力します。“Large I/Os Denied”の値が大きい場合は、個々のキャッシュを調査し、原因を調べてください。

- “Large I/Os Performed”では、要求された大容量 I/O が実行された回数が測定される。“% of total”では、実行された I/O 要求の総数に対して、実行された大容量 I/O 要求数の割合が、パーセンテージで示される。
- “Large I/Os Denied”では、大容量 I/O を実行できなかった回数がレポートされます。“% of total”では、実行された要求の総数に対して、拒否された大容量 I/O 要求数の割合が、パーセンテージで示される。
- “Total Large I/O Requests”では、すべてのキャッシュについて、大容量 I/O が要求された総数 (付与された要求と拒否された要求の両方) がレポートされる。

大容量 I/O の効果

パフォーマンスに対する大容量 I/O のメリットを調べる場合に役立ちます。このセクションでは、大容量 I/O によってキャッシュに取り込まれたページ数が、キャッシュ内で実際に参照されたページ数と比較されます。“Pages by Lrg I/O Used”のパーセンテージが低い場合は、キャッシュに取り込まれたページのほとんどがクエリからアクセスされていません。個々のキャッシュを調査し、問題の原因を調べてください。また、optdiag を使用して、テーブルとインデックスごとに“Large I/O Efficiency”の値をチェックしてください。

- “Pages by Lrg I/O Cached”では、サンプル間隔中に実行されたすべての大容量 I/O 操作によって、すべてのキャッシュに取り込まれたページ数がレポートされる。パーセンテージが低い場合は、次のいずれかを示している可能性がある。
  - テーブルの記憶領域における割り付けの断片化
  - 不適切なキャッシュ方式
- “Pages by Lrg I/O Used”では、大容量 I/O によってキャッシュに取り込まれてから使用されたページの総数がレポートされる。sp_sysmon では、“Pages by Lrg I/O Cached”がない場合、このカテゴリは出力されない。

非同期プリフェッチ・アクティビティに関するレポート

すべてのキャッシュについて、非同期プリフェッチ・アクティビティをレポートします。

各データベース・デバイスでの非同期プリフェッチの詳細については、「ディスク I/O 管理」(125 ページ) を参照してください。非同期プリフェッチの詳細については、『パフォーマンス&チューニング・シリーズ：基本』の「第 6 章 非同期プリフェッチのチューニング」を参照してください。

“Total APFs Requested”では、プリフェッチするのに適切なページの総数、つまり、サンプル間隔中に発行されたすべてのクエリの予備セットの合計サイズがレポートされます。“Asynchronous Prefetch Activity”のそのほかのローには、次の 3 つのカテゴリごとに詳細な情報が出力されます。

- プリフェッチされたページ (“APFs Issued”)
- プリフェッチが拒否された原因
- キャッシュ内で見つかったページ

発行された APF

サンプル間隔中にシステムによって発行された非同期プリフェッチ要求の数。

APF が拒否された原因

APF が発行されなかった原因が次のようにレポートされます。

- “APF I/O Overloads”では、ディスク I/O の構造体の不足またはディスク・セマフォの競合が原因で APF の使用が拒否された回数がレポートされる。

  この数値が大きい場合は、“Disk I/O Management”に出力される次の情報をチェックする。

  - disk i/o structures 設定パラメータの値をチェックする。「ディスク I/O 構造体」(127 ページ) を参照。
  - 各データベース・デバイスのデバイス・セマフォの競合に対応する値をチェックし、問題の原因を調べる。「付与されたミラー・セマフォと待機したミラー・セマフォ」(130 ページ) を参照。

  ディスク I/O 構造体の不足が問題の原因である場合は、設定パラメータの値を高くし、もう一度テストする。ディスク・セマフォの競合が大きいことが問題の原因である場合は、I/O の回数が多いオブジェクトの物理的な配置を調べる。

- “APF Limit Overloads”は、非同期プリフェッチに使用できるバッファ・プールのパーセンテージを上回っていることを示す。この制限は、global async prefetch limit 設定パラメータによってサーバ全体に設定される。sp_poolconfig を使用してプールごとにチューニングする。

- “APF Reused Overloads”は、ページ・チェーンのねじれ、または APF によって取り込まれたバッファがアクセスされる前に押し出されたことが原因で、APF の使用が拒否されたことを示す。

  連続したページに連続したローが含まれているのが理想的である。ただし、ページ・チェーンが少数のページで現在の位置から遠く離れているディスクの一部分に移動した後、元の位置に戻った場合、ページは「ねじれた」とみなされる。たとえば、ページ 1、2、396、397、3、4 を使用するページ・チェーンはねじれている。

キャッシュ内に見つかった APF バッファ

サンプル間隔中に、データ・キャッシュ内に見つかった APF の予備セットから取り込まれるバッファの数。非同期プリフェッチは、キャッシュのスピンロックを保持しないクイック・スキャンを使用して、読み込むページをデータ・キャッシュ内で見つけようとします。その処理が失敗した場合は、スピンロックを保持するスルー・スキャンが実行されます。

非同期プリフェッチに関するその他の情報

非同期プリフェッチについて、さらに 3 つの情報をレポートします。

- “APFs Used”では、非同期プリフェッチによってキャッシュに取り込まれ、サンプル間隔中に使用されたページ数がレポートされる。このレポート用にカウントされるページは、サンプル間隔中にキャッシュに取り込まれたか、またはサンプル間隔が開始される前に発行された非同期プリフェッチ要求によってキャッシュに取り込まれたページである。
- “APF Waits for I/O”では、プロセスが非同期プリフェッチの完了を待機しなければならなかった回数がレポートされる。これは、プリフェッチの発行が遅れたために、クエリがページを必要とする前にページをキャッシュに取り込めなかったことを示す。“APF Waits”の値は、通常は数パーセントと予想される。タスクが待機しなければならない場合、次のような理由がある。
  - クエリの最初の非同期プリフェッチ要求は、通常“APF Waits”に含まれる。
  - 逐次スキャンが新しいアロケーション・ユニットに移動され、プリフェッチ要求を発行するたびに、クエリは最初の I/O が完了するまで待機しなければならない。
  - ノンクラスタード・インデックス・スキャンによって条件に合うローが見つかり、そのページに対するプリフェッチ要求が発行されるたびに、この要求は最初のページが返されるのを待機しなければならない。

  “APF Waits for I/O”に影響を与える可能性があるそのほかの要因としては、各ページ上で実行される必要がある処理の量と、I/O サブシステムの速度がある。
- “APF Discards”は、非同期プリフェッチによって読み込まれ、使用される前に廃棄されたページ数を示す。“APFs Discards”の値が大きい場合は、バッファ・プールのサイズを増やすとパフォーマンスが向上すること、または APF がクエリによって必要とされないキャッシュにページを取り込んでいることを示す。

ダーティ・リードの動作

ダーティ・リード (独立性レベルが 0 の読み込み) が、システムに影響を及ぼす程度を分析する場合に役立つ情報を出力します。

ページ要求

独立性レベルが 0 のときに要求されたページの平均値をレポートします。“% of total”カラムでは、ページ読み込みの総数に対するダーティ・リードの割合がパーセンテージでレポートされます。

ダーティ・リードの再起動が多く発生するとダーティ・リードによるページ要求のオーバヘッドが大きくなります。

ダーティ・リードの再起動

ダーティ・リードによる再起動の数。このカテゴリは、サーバ全体についてのみレポートされ、個々のキャッシュについてはレポートされません。sp_sysmon では、“Dirty Read Page Requests” がない場合はこのカテゴリは出力されません。

ダーティ・リードがページ上でアクティブな場合に、別のプロセスがページを変更してページの割り付けを解除すると、ダーティ・リードによる再起動が実行されます。レベル 0 のスキャンも、もう一度開始してください。

“% of total”では、ページ読み込みの総数に対して、独立性レベル 0 の状態におけるダーティ・リードによる再起動回数の割合が、パーセンテージで示されます。

上記の値が大きい場合は、アプリケーションに変更を加えて値を低くするという処置をとります。これは、ダーティ・リードとそれに起因する再起動のオーバヘッドによるコストが非常に高いためです。ほとんどのアプリケーションでは、発生するオーバヘッドが大きくなるので、再起動を避けてください。

キャッシュごとのキャッシュ管理

サーバ上のアクティブなキャッシュごとに、キャッシュの使用率をレポートします。出力例は、デフォルト・データ・キャッシュの結果を示します。

キャッシュのスピンロック競合

キャッシュに対するスピンロック要求の総数に対して、キャッシュ上でスピンロック競合が発生してエンジンが待機しなければならなかった回数の割合 (パーセンテージで表示)。この値は SMP 環境でだけ意味を持ちます。

ユーザ・タスクがキャッシュに対して何らかの変更を行っている間、スピンロックは、ほかのすべてのタスクによるそのキャッシュへのアクセスを拒否します。スピンロックが保持されるのは非常に短い時間ですが、トランザクション・レートが高いマルチプロセッサ・システムでは、スピンロックによってパフォーマンスが低下する可能性があります。スピンロック競合が 10% を超える場合は、名前付きキャッシュの使用か、またはキャッシュ・パーティションの追加を検討してください。

キャッシュの追加については、『パフォーマンス&チューニング・シリーズ：基本』の「第 5 章 メモリの使い方とパフォーマンス」を参照してください。

使用率

キャッシュ全体の検索数に対する、あるキャッシュを使用した検索数の割合を、パーセンテージとしてレポートします。各キャッシュに対応するこの値を比較し、使用率の高いキャッシュや使用率の低いキャッシュがないかどうかを調べることができます。キャッシュがうまく使用されていない場合は、次のように対処できます。

- キャッシュのバインドを変更し、使用率のバランスをとる。詳細については、『パフォーマンス&チューニング・シリーズ：基本』の「第 5 章 メモリの使い方とパフォーマンス」を参照。
- 使用率がより適切な値になるまでキャッシュのサイズを変更する。

  詳細については、『システム管理ガイド：第 2 巻』の「第 4 章 データ・キャッシュの設定」を参照してください。

キャッシュの検索、ヒット、およびミスに関する情報

ヒットとミスの数、およびこのキャッシュに対する検索の総数。キャッシュ・ヒットは、statistics io によってレポートされる論理読み込みの値とほぼ一致します。キャッシュ・ミスは、物理読み込みの値とほぼ一致します。sp_sysmon では、システム・テーブル、ログ・ページ、OAM ページおよびそのほかのシステム・オーバヘッドの I/O もレポートされるので、statistics io が示す値よりも大きい値が常にレポートされます。

キャッシュ・ヒットについてのデータを解釈するには、アプリケーションが各キャッシュを使用する方法について理解しておく必要があります。インデックスやルックアップ・テーブルなどの特定のオブジェクトを保持するために作成されたキャッシュでは、キャッシュ・ヒット率が 100% に達する場合があります。非常に大きなテーブルでランダムに実行されるクエリに使用されるキャッシュでは、キャッシュ・ヒット率がきわめて低くなりますが、キャッシュを効果的に使用していることが示されます。

キャッシュのヒットとミスは、メモリの追加によってパフォーマンスが向上するかどうかを調べる場合にも役立ちます。たとえば、“Cache Hits”の値が大きい場合は、メモリを追加してもあまり大きな効果は得られません。

キャッシュのヒット

必要なページがデータ・キャッシュ内で見つかった回数。“% of total”では、キャッシュ検索の総数に対するキャッシュ・ヒットの割合が、パーセンテージでレポートされます。

ウォッシュ内での発見

必要なページがキャッシュのウォッシュ・セクションで見つかった回数。“% of total”では、バッファがウォッシュ・エリアで見つかった回数の割合が、ヒットの総数に対するパーセンテージとしてレポートされます。データによって、ウォッシュ・セクションで見つかったキャッシュ・ヒットのパーセンテージが大きいことが示される場合は、ウォッシュ・エリアが大きすぎます。ただし、読み込み専用のキャッシュや書き込みの回数が少ないキャッシュでは問題ありません。

ウォッシュ・セクションが大きいと、Adaptive Server はすべてのダーティ・ページでの書き込みをウォッシュ・マーカをまたいで開始するため、物理 I/O が増加します。ウォッシュ・エリアにあるページがディスクに書き込まれ、次に更新されると、I/O が浪費されます。ウォッシュ・マーカをまたぐことにより、数多くのバッファが書き込みを開始していないかどうかチェックしてください。

詳細については、「バッファ・ウォッシュの動作」(116 ページ) を参照してください。

キャッシュ内のテーブル上のクエリが、非同期プリフェッチでない I/O について「使い捨て」方式を使用する場合は、ページに対応する最初のキャッシュ・ヒットはウォッシュ内で見つかります。バッファは、チェーンの MRU 側の終端に移動されるので、最初のキャッシュ・ヒットの直後に発生する 2 回めのキャッシュ・ヒットでは、バッファがまだウォッシュ・エリアの外側にあることがわかります。

詳細については、「キャッシュ方式」(117 ページ) を参照してください。

ウォッシュのサイズを変更できます。ウォッシュのサイズを小さくした場合は、すべての負荷をかけた環境で sp_sysmon を再度実行し、“Grabbed Dirty” の値が 0 より大きいかどうかをチェックしてください。

「キャッシュのターンオーバ」(108 ページ) を参照してください。

キャッシュのミス

必要なページがキャッシュ内に見つからずにディスクから読み込んだ回数。“% of total” では、検索の総数に対するバッファがキャッシュ内に見つかった回数の割合が、パーセンテージとしてレポートされます。

キャッシュ検索の総数

キャッシュ検索アクティビティについてまとめた情報を出力します。“Found in Wash” のデータは “Cache Hits” の数のサブカテゴリであり、集計には使用されません。

プールのターンオーバー

バッファがキャッシュ内の各プールから置き換えられた回数。各キャッシュには 2K、4K、8K、16K までの I/O サイズに対応する 4 つのプールがあります。“Pool Turnover” が発生している場合、sp_sysmon では、設定された各プールについての “LRU Buffer Grab” の情報と “Grabbed Dirty” の情報、およびキャッシュ全体に対するターンオーバの総数が出力されます。“Pool Turnover” が発生していない場合、sp_sysmon では、“Total Cache Turnover” に 0 のローだけが出力されます。

この情報は、プールとキャッシュのサイズが適切かどうかを調べる場合に役立ちます。

LRU バッファの取り込み

“LRU Buffer Grab”は、ページが別のページと置換されたときにだけ増加します。Adaptive Server を再起動した直後、またはオブジェクトやデータベースのバインドを解除して再度キャッシュへバインドした場合、空のバッファへのページの読み込みはターンオーバにはカウントされません。

スループットと比較してメモリ・プールが小さすぎる場合は、プールのターンオーバが多く発生し、キャッシュ・ヒット率が低下し、I/O 率が高くなります。一部のプールでターンオーバの値が大きく、そのほかのプールでは小さい場合は、アクティブであることの少ないプールからよりアクティブなプールへ領域を移す必要があります。特に、そうすることでキャッシュのヒット率が向上する場合は、この処理を行ってください。

プールに 1000 個のバッファがあるときに、Adaptive Server が 1 秒ごとに 100 個のバッファを置換する場合は、バッファの 10% が 1 秒ごとにターンオーバされます。これは、バッファがキャッシュに残される時間が不十分なために、オブジェクトがそのキャッシュを使用できない可能性があることを示します。

取り込まれたダーティ・バッファ

ディスクに書き込まれる前に LRU に到達したダーティ・バッファの数についての情報を出力します。Adaptive Server は、ディスクからページをフェッチするためにキャッシュの LRU 側の終端からバッファを取り込む必要があり、クリーン・バッファではなくダーティ・バッファを見つけた場合、ダーティ・バッファでの I/O が完了するのを待機する必要があります。“% of total”では、取り込まれたバッファの総数に対する取り込まれたダーティ・バッファ数の割合が、パーセンテージでレポートされます。

“Grabbed Dirty”の値が 0 以外の場合は、プール内のスループットと比較してプールのウォッシュ・エリアが小さすぎます。プールの設定や使用方法に従って、次のように対処してください。

- プールが非常に小さく、ターンオーバの値が大きい場合は、プールとウォッシュ・エリアのサイズを増やすことを検討する。
- プールが大きく、多くのデータ変更オペレーションによってプールが使用されている場合は、ウォッシュ・エリアのサイズを増やす。
- 複数のオブジェクトがキャッシュを使用している場合は、一部のオブジェクトを別のキャッシュに移動することで対処できる可能性がある。
- キャッシュが create index によって使用されている場合は、I/O 率が高いためにダーティ・バッファ取り込みを発生させている可能性がある。特に、小さな 16K プールの場合にこれが当てはまる。このような場合は、プールのウォッシュ・エリアのサイズを、できるだけ大きく設定する (プール内のバッファの 80% まで)。
- キャッシュが分割されている場合は、分割の数を減らす。

- クエリ・プラン、およびキャッシュを使用するオブジェクトの I/O についての情報をチェックし、クエリがプール内で多くの物理 I/O を実行していないかどうかを調べる。可能な場合には、インデックスを追加してクエリをチューニングする。

「バッファ・ウォッシュの動作」(116 ページ) の“Buffers Already in I/O”と“Buffers Washed Dirty”の“per second”の値をチェックしてください。LRU に到達する前に I/O を完了できるように、ウォッシュ・エリアを十分な大きさにします。I/O の完了に必要な時間は、使用しているディスク・ドライブによって実行される 1 秒当たりの実際の物理書き込みの数に依存します。

「ディスク I/O 管理」(125 ページ) もチェックし、I/O の競合が原因でディスク書き込みが低速になっていないかどうかを調べてください。

さらに、housekeeper free write percent 設定パラメータの値を増やすことで、対処できる場合があります。『システム管理ガイド 第 1 巻』の第 5 章を参照してください。

キャッシュのターンオーバの総数

キャッシュ内のすべてのプールから取り込まれたバッファの総数をまとめた情報を出力します。

クラスタ・キャッシュの動作

sp_sysmon では、クラスタ環境でのバッファ用に取得したロックについて説明します。

バッファ・ウォッシュの動作

バッファがプールのウォッシュ・マーカに到達したときのバッファの状態に関する情報をレポートします。バッファがウォッシュ・マーカに到達すると、次の 3 つのうちのいずれかの状態になります。

- “Buffers Passed Clean”は、クリーンな状態でウォッシュ・マーカを超えたバッファの数をレポートする。バッファは、キャッシュ内にある間に変更されなかったか、変更されたがハウスキーピングまたはチェックポイントによってすでにディスクに書き込まれている。“% of total”では、ウォッシュ・マーカを超えたバッファの総数に対して、クリーンな状態でウォッシュ・マーカを超えたバッファ数の割合が、パーセンテージでレポートされる。
- “Buffers Already in I/O”では、ウォッシュ・エリアに入ったときに、すでに I/O がバッファ上でアクティブだった回数がレポートされる。キャッシュ内にある間、ページが変更された。ハウスキーピングまたはチェックポイントがページの I/O を開始したが、まだ完了していない。“% of total”では、すでに I/O 処理中であるバッファの割合が、ウォッシュ・エリアに入ったバッファの総数に対するパーセンテージでレポートされる。

- “Buffers Washed Dirty”では、バッファがダーティな状態でウォッシュ・エリアに入り、まだ I/O 処理中ではなかった回数がレポートされる。キャッシュ内にある間にバッファが変更され、ディスクには書き込まれなかった。ページがウォッシュ・マーカを超えると、非同期 I/O がそのページ上で開始される。“% of total”では、ウォッシュ・エリアに入ったバッファの総数に対して、ダーティな状態でウォッシュ・エリアに入ったバッファ数の割合が、パーセンテージでレポートされる。

サンプル間隔中にウォッシュ・マーカを超えたバッファがない場合は、sp_sysmon によって次のように出力されます。

```
Statistics Not Available - No Buffers Entered Wash Section Yet!
```

キャッシュ方式

MRU (使い捨て) キャッシュ方式または LRU (ノーマル) キャッシュ方式に従ってキャッシュに配置されたバッファの数。

- Cached(LRU) Buffers － ノーマル・キャッシュ方式を使用し、キャッシュの MRU 側の終端に配置されたバッファの数がレポートされる。これには、ディスクからダーティな状態で読み込まれ、MRU 側の終端に配置されたすべてのバッファ、およびキャッシュ内で見つかったすべてのバッファが含まれる。論理 I/O の終了時に、バッファはキャッシュの MRU 側の終端に配置される。
- Discarded (MRU) Buffers － 使い捨て方式を使用してウォッシュ・マーカに配置されたバッファの数がレポートされる。

  テーブル全体がキャッシュに格納されると予測できるにもかかわらず、“Discarded Buffers”に出力された値が大きい場合は、showplan を使用して、オプティマイザがノーマル・キャッシュ方式を使用すべきときに、使い捨て方式を生成していないかどうかを調べる。

  詳細については、『パフォーマンス＆チューニング・シリーズ：クエリ処理と抽象プラン』の「第 2 章 showplan の使用」を参照してください。

大容量 I/O の使用率

Adaptive Server による大容量 I/O のプリフェッチ要求についての情報を出力します。ここでは、実行された大容量 I/O 要求の数と、拒否された大容量 I/O 要求の数についての情報がレポートされます。

実行された大容量 I/O

要求された大容量 I/O が実行された回数を測定します。“% of total”では、行われた要求の総数に対する、実行された大容量 I/O 要求数の割合が、パーセンテージでレポートされます。

sp_sysmon では、Adaptive Server がキャッシュ内で大容量 I/O を実行した場合にのみ、このセクションを表示します。

拒否された大容量 I/O

大容量 I/O を実行できなかった回数。“% of total”では、実行された要求の総数に対して、拒否された大容量 I/O 要求数の割合が、パーセンテージで示される。

Adaptive Server では、次のような場合には大容量 I/O を実行できません。

- バッファ内のいずれかのページがすでに別のプール内に存在する場合。
- 要求されたプール内に使用できるバッファがない場合。
- アロケーション・ユニットの最初のエクステント。このエクステントは、アロケーション・ページで構成され、常に 2K プールに読み込まれる。

拒否された大容量 I/O のパーセンテージが大きい場合は、より大きなプールを使用しても効果はありません。キャッシュが大容量 I/O のプールから構成されており、クエリが同じオブジェクトで 2K と 16K の両方の I/O を実行する場合は、ページが 2K プールに存在するので、実行できない大容量 I/O が常に数パーセントは存在します。

半数を超える大容量 I/O が拒否された場合に 16K の I/O を使用しているときには、すべての領域を 16K プールから 8K プールに移動してみてください。テストを再実行し、I/O の総数が減ったかどうかを調べてください。16K の I/O が拒否される場合は、Adaptive Server では 8K または 4K のプールがチェックされず、2K プールが使用されます。

このカテゴリの情報と“Pool Turnover”の情報を使用して、プールのサイズが適切であるかどうかを判断します。

sp_sysmon では、Adaptive Server がキャッシュ内で大容量 I/O を実行した場合にのみ、このセクションを表示します。

大容量 I/O 要求の総数

実行された大容量 I/O と拒否された大容量 I/O についてまとめた情報を出力します。

大容量 I/O の詳細

各プールごとにまとめた情報を出力します。このセクションは、キャッシュに設定された 4K、8K、16K のプールごとにまとめた情報から構成されます。さらに、設定されている各 I/O サイズごとに、取り込まれたページ (“Pages Cached”) と参照されたページ (“Pages Used”) が出力されます。

たとえば、あるクエリによって 16K の I/O が実行されて 1 つのデータ・ページが読み込まれた場合、“Pages Cached”の値は 8、“Pages Used”の値は 1 になります。

- “Pages by Lrg I/O Cached”では、キャッシュに読み込まれたページの総数が出力される。
- “Pages by Lrg I/O Used”では、キャッシュ内にある間にクエリによって使用されたページ数がレポートされる。

ダーティ・リードの動作

独立性レベル 0 で要求されたページ数の平均値をレポートします。

“Dirty Read Page Requests”の“% of total”では、ページ読み込みの総数に対するダーティ・リードの割合がパーセンテージで示されます。

プロシージャ・キャッシュ管理

以下が表示されます。

- ストアド・プロシージャが再コンパイルされた回数。
- 実行時、再コンパイルなど、再コンパイルをトリガしたフェーズ。
- テーブルの欠落、パーミッションの変更など、再コンパイルの理由。

出力例

```
Procedure Cache Management       per sec      per xact       count   % of total
-------------------------  ------------   -----------  ---------   ---------
  Procedure Requests               14.2           0.3       8499         n/a
  Procedure Reads from Disk         0.0           0.0         27        0.6%
  Procedure Writes to Disk          0.0           0.0          0        0.0%
  Procedure Removals                0.0           0.0          0         n/a
  Procedure Recompilations          0.0           0.0          0         n/a

SQL Statement Cache:
  Statements Cached                 0.6           2.7        171         n/a
  Statements Found in Cache         2.0           9.5       1529         n/a
  Statements Not Found              0.6           2.7        171         n/a
  Statements Dropped                0.2           0.8         52         n/a
  Statements Restored               0.0           0.0          0         n/a
  Statements Not Cached             0.0           0.0          0         n/a
```

プロシージャ・キャッシュには、要求されたプロシージャに対する多くの実行プランが含まれています。この例では、プロシージャ・キャッシュはディスクから 0.6% だけを読み取ります。

Statements Found in Cache の値が 611 であるため、このシステムのステートメント・キャッシュは有効になります。この値は、Statements Not Found と Statements Cached の値をはるかに超えているため、ステートメントの再利用率は高くなります。

プロシージャ要求

ストアド・プロシージャが実行された回数。

プロシージャの実行時には、次のいずれかの動作を行います。

- クエリ・プランのアイドル・コピーがメモリ内にあるので、コピーされてから使用される。
- プロシージャのコピーがメモリ内にはないか、またはメモリ内にあるプランのコピーがすべて使用されているので、プロシージャをディスクから読み込む必要がある。

ディスクからのプロシージャの読み込み

アイドル・コピーを使用できず Adaptive Server がディスクからプロシージャを読み取らなければならなかった (つまり、Adaptive Server がキャッシュ内にすでにあったものをコピーできなかった) 回数。

“% of total”では、プロシージャ要求の総数に対して、ディスクから読み込まれたプロシージャ数の割合が、パーセンテージでレポートされます。この割合が比較的大きい場合は、プロシージャ・キャッシュが小さすぎる可能性があります。

ディスクへのプロシージャの書き込み

サンプル間隔中に作成されたプロシージャの数。アプリケーション・プログラムがストアド・プロシージャを生成する場合は、この値が非常に高くなる可能性があります。

プロシージャの削除

プロシージャがキャッシュに保持される期限を過ぎた回数。

プロシージャの再コンパイル

ストアド・プロシージャが再コンパイルされた回数。

SQL ステートメント・キャッシュ

インターバル中にキャッシュされた、キャッシュ内で見つかった、見つからなかった、削除された、リストアされた、およびキャッシュされなかった文の数。

出力例

```
SQL Statement Cache:
Statements Cached                0.6          2.7          171       n/a
Statements Found in Cache        2.0          9.5          611       n/a
Statements Not Found             0.6          2.7          171       n/a
Statements Dropped               0.2          0.8           52       n/a
Statements Restored              0.0          0.0            0       n/a
Statements Not Cached            0.0          0.0            0       n/a
```

- Statements Cached － サンプリング時間中にステートメント・キャッシュに追加された文の数。
- Statements Found in Cache － ステートメント・キャッシュ内で見つかった文の数。この値は、Adaptive Server が文の再コンパイルを回避するためにステートメント・キャッシュを使用した回数を表す。
- Statements Not Found － Adaptive Server がステートメント・キャッシュに含まれている文を検索したが見つからなかった回数。
- Statements Dropped － Adaptive Server がキャッシュする新しい文のための領域を確保するために、ステートメント・キャッシュ内にすでにある文を削除した回数。
- Statements Restored － 既存のプランが文によって参照されるオブジェクトに対するスキーマ変更によって無効になった後で、リストアされたクエリ・プランの数。
- Statements Not Cached － Adaptive Server がコンパイルしたが、ステートメント・キャッシュに入力しなかった文の数。statement cache size を 0 に設定した場合 (ステートメント・キャッシュを無効にした場合)、Statements Not Cached は Adaptive Server によってキャッシュ可能であった文の数を表す。

メモリ管理

サンプル間隔中に割り付けられたページ数と割り付けが解除されたページ数。

出力例

```
Memory Management              per sec       per xact       count   % of total
-----------------------   ------------   -----------    --------    --------
Pages Allocated                2565.7          63.2     1539423         n/a
Pages Released                 2565.7          63.2     1539423         n/a
```

割り付けられたページ

新しいページがメモリに割り付けられた回数。

解放されたページ

ページが解放された回数。

リカバリ管理

このデータは、ノーマル・チェックポイント・プロセスによって発生したチェックポイントの数、ハウスキーピング・タスクによって開始されたチェックポイントの数、種類ごとの平均時間を示します。この情報を使用すると、リカバリのパラメータとハウスキーピングのパラメータを正しく設定できます。

注意 Adaptive Server 12.5.3 以降を使用している場合、チェックポイント・タスクの実行速度が以前より最大 200% 高速化し、データベースのリカバリ速度が大幅に向上したことが社内ベンチマークによって示されています。この速度の向上はユーザが何もしなくても実現されます。

ただし、パフォーマンスの向上は、使用している I/O サブシステムの効率に左右されます。IO 帯域幅を使用できるまでは、パフォーマンスが向上しない場合もあります。

出力例

```
===============================================================================

Recovery Management
-------------------
Checkpoints                      per sec     per xact    count    % of total
-------------------------    -----------   ----------  -------      --------
# of Normal Checkpoints              0.2          0.0      106        100.0 %
# of Free Checkpoints                0.0          0.0        0          0.0 %
-------------------------  ------------   ----------  -------
Total Checkpoints                    0.2          0.0      106
Average Time per Normal Chkpt
```

チェックポイント

チェックポイントは、ダーティ・ページ (メモリ内で変更され、ディスクへは書き込まれなかったページ) をデータベース・デバイスへ書き込みます。Adaptive Server の自動 (ノーマル) チェックポイント・メカニズムが動作し、リカバリ間隔を最小に維持します。チェックポイントが最後に実行された時点以降のトランザクション・ログのログ・レコード数を追跡することで、トランザクションのリカバリに必要な時間がリカバリ間隔を超えるかどうかを予測できます。トランザクションのリカバリに必要な時間がリカバリ間隔を超える場合は、チェックポイント・プロセスがすべてのデータ・キャッシュをスキャンし、変更されたすべてのデータ・ページを書き込みます。

Adaptive Server に処理すべきユーザ・タスクがない場合は、ハウスキーピング・ウォッシュ・タスクがダーティ・バッファのディスクへの書き込みを開始します。これらの書き込みは、サーバのアイドル・サイクル中に行われるため、「フリー書き込み」と呼ばれます。この書き込みによって、CPU の使用率が向上し、トランザクション処理中にバッファ・ウォッシングの必要性が減少します。

ノーマル・チェックポイントの数

ノーマル・チェックポイント・プロセスによって実行されたチェックポイントの数。

ノーマル・チェックポイントが作業のほとんどを実行している場合、特に非常に長い時間を必要とする場合は、ハウスキーピング・ウォッシュ・タスクによって実行される書き込みの数を増やすと、場合によっては効果が期待できます。

設定パラメータの詳細については、『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

フリー・チェックポイントの数

ハウスキーピング・ウォッシュ・タスクによって実行されたチェックポイントの数。ハウスキーピング・ウォッシュ・タスクは、設定されたすべてのキャッシュからすべてのダーティ・ページをクリアするときにだけ、チェックポイントを実行します。

ハウスキーピング・ウォッシュ・タスクは、全キャッシュ内のダーティ・ページをすべてディスクに書き込む処理を終了すると、最後のチェックポイント以降のトランザクション・ログのロー数をチェックします。100 を超えるログ・レコードがある場合は、チェックポイントを発行します。このチェックポイントで必要なオーバヘッドは非常に少ないので、これを「フリー・チェックポイント」と呼びます。さらに、フリー・チェックポイントでは、ノーマル・チェックポイントで将来発生するオーバヘッドが減ります。

housekeeper free write percent パラメータを使用して、ハウスキーピング・ウォッシュ・タスクがデータベース書き込みを増加できる最大のパーセンテージを設定できます。『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

チェックポイントの総数

サンプル間隔中に発生したノーマル・チェックポイントとフリー・チェックポイントを合わせた数をレポートします。

ノーマル・チェックポイント当たりの平均時間

ノーマル・チェックポイントが継続した平均時間をレポートします。

フリー・チェックポイント当たりの平均時間

フリー (またはハウスキーピング) チェックポイントが継続した平均時間をレポートします。

ハウスキーピングのバッチ制限の増加

ハウスキーピング・ウォッシュ・タスクには、個々のデバイスについてディスク I/O の負荷が高くなりすぎるのを避けるための、バッチ制限値が組み込まれています。デフォルトでは、ハウスキーピングが書き込みを行うためのバッチ・サイズは 3 に設定されています。ハウスキーピングは、1 つのデバイスに対して 3 つの I/O が発行されたことを検出するとすぐに、現在のバッファ・プールでの処理を停止し、別のプール内のダーティ・ページのチェックを開始します。次のプールからの書き込みが同じデバイスに割り当てられる場合は、別のプールへ進みます。ハウスキーピングがすべてのプールのチェックを完了すると、ハウスキーピングが発行した最後の I/O が完了するまで待機し、再度サイクルを開始します。

デフォルトのバッチ制限値は、低速のディスクで優れたデバイス I/O 特性が発揮されるように設計されています。高速なディスク・デバイスでは、バッチ・サイズを増やすことで、より良いパフォーマンスを得ることができます。この制限値は、サーバ上のすべてのデバイスについてグローバルに設定したり、異なるスピードのディスクに対して異なる値を設定したりできます。Adaptive Server を再起動した場合は、毎回、制限値を再設定してください。

次のコマンドは、sysdevices から取得した仮想デバイス番号を使用して、1 つのデバイスのバッチ・サイズを 10 に設定します。

```
dbcc tune(deviochar, 8, "10")
```

デバイス番号を調べるには、sp_helpdevice または次のクエリを使用します。

```
select name, vdevno
from sysdevices
```

サーバ上のすべてのデバイスに対して、ハウスキーピングのバッチ・サイズを変更するには、次のように、デバイス番号の代わりに -1 を使用します。

```
dbcc tune(deviochar, -1, "5")
```

非常に高速なドライブでは、バッチ・サイズを 50 程度に設定すると、パフォーマンスが向上するケースもあります。

次の場合には、バッチ・サイズを大きくする必要があります。

- ノーマル・チェックポイントの平均時間の値が大きい場合。
- I/O 設定の制限を超えたり、デバイスのセマフォで競合が発生する問題がない場合。

ディスク I/O 管理

ディスク I/O をレポートします。サーバ全体のディスク I/O についての大まかな情報が出力され、各論理デバイスでの読み込み、書き込み、セマフォ競合がレポートされます。

出力例

```
Disk I/O Management
-------------------

Max Outstanding I/Os            per sec      per xact       count   % of total
------------------------    -----------   -----------   ---------   ---------
  Server                            n/a           n/a         527         n/a
  Engine 0                          n/a           n/a         429         n/a
  Engine 1                          n/a           n/a         448         n/a
  Engine 2                          n/a           n/a         457         n/a
  Engine 3                          n/a           n/a         508         n/a
  Engine 4                          n/a           n/a         526         n/a
  Engine 5                          n/a           n/a         416         n/a
  Engine 6                          n/a           n/a         425         n/a


I/Os Delayed by
  Disk I/O   Structures             n/a           n/a           0         n/a
  Server Config   Limit             n/a           n/a           0         n/a
  Engine Config Limit               n/a           n/a           0         n/a
  Operating System Limit            n/a           n/a           0         n/a
```

```
Total Requested Disk I/Os          547.2           13.5       328339

Completed Disk I/O's
Asynchronous I/O's
  Engine 0                         135.0            3.3        81017       24.7 %
  Engine 1                          70.3            1.7        42172       12.8 %
  Engine 2                          50.1            1.2        30083        9.2 %
  Engine 3                          68.0            1.7        40789       12.4 %
  Engine 4                          72.5            1.8        43473       13.2 %
  Engine 5                          81.2            2.0        48749       14.8 %
  Engine 6                          70.1            1.7        42070       12.8 %
Synchronous I/O's
  Total Completed I/Os               0.0            0.0            0          n/a
-----------------------   ------------   -----------   ---------
Total Completed I/Os               547.3           13.5       328353


Device Activity Detail
----------------------

Device:
  /dev/sybase/c1t0d0s0r
  master                        per sec      per xact       count   % of total
-------------------------    -----------   -----------   ---------   ---------
  Reads
    APF                             0.0            0.0           2        0.2 %
    Non-APF                         0.3            0.0         154       17.5 %
  Writes                            1.2            0.0         723       82.3 %
--------------------------    ----------   -----------   ---------   ---------
Total I/Os                          1.5            0.0         879        0.3 %


--------------------------------------------------------------------------------

Device:
  /dev/sybase/c1t0d0s1r
  dev01                         per sec      per xact       count   % of total
-------------------------    -----------   -----------   ---------   ---------
Reads
  APF                               1.0            0.0         584        0.5 %
    Non-APF                         8.5            0.2        5075        4.3 %
    Writes                        185.6            4.6      111355       95.2 %
--------------------------   -----------   -----------   ---------   ---------
Total I/Os                        195.0            4.8      117014       35.6 %


--------------------------------------------------------------------------------

Device:
```

```
  /dev/sybase/c1t0d0s3r
  dev02                          per sec      per xact       count   % of total
------------------------    -----------   -----------   ---------   ---------
  Reads
    APF                              0.0           0.0           1       0.5 %
    Non-APF                          0.0           0.0           5       2.4 %
  Writes                             0.3           0.0         204      97.1 %
-------------------------   -----------   -----------   ---------   ---------
Total I/Os                           0.4           0.0         210        0.1%
```

未処理のまま残っている I/O の最大数

サンプル間隔中の任意の時点で保留される I/O の最大数が、Adaptive Server 全体 (最初の行) および各 Adaptive Server エンジンについてレポートされます。

この情報は、“I/Os Delayed By”の値が 0 以外の場合に、サーバまたはオペレーティング・システム・レベルで I/O パラメータを設定する場合に役立ちます。

I/O 遅延の原因

システムが I/O 遅延の問題を見つけると、1 つ以上の Adaptive Server またはオペレーティング・システムの制限によって、I/O がブロックされる可能性があります。

ほとんどのオペレーティング・システムには、実行できる非同期 I/O の数を制限するカーネル・パラメータがあります。

ディスク I/O 構造体

ディスク I/O 構造体の制限値に達したために遅延した I/O の数。Adaptive Server が利用できるディスク I/O 制御ブロックの数を超えた場合、I/O が遅延されます。これは、Adaptive Server は、タスクがディスク I/O 制御ブロックを取得してから I/O 要求を開始することを要求するからです。

結果として 0 以外の値が出力された場合は、設定パラメータ disk i/o structures を増やすことによって、利用できるディスク I/O 制御ブロックの数を増やしてみてください。『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

サーバ設定の制限

Adaptive Server は、同時にその Adaptive Server 全体で未処理にできる、非同期ディスク I/O 要求の数に対する制限値を超えることができます。max async i/os per server 設定パラメータを使用してこの制限値を増やすことができます。『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

エンジン設定の制限

エンジンは、まだ処理されていない非同期ディスク I/O 要求に対する制限値を超えることができます。max async i/os per engine 設定パラメータを使用してこの制限値を変更できます。『システム管理ガイド 第 1 巻』の「第 5 章 設定パラメータ」を参照してください。

オペレーティング・システムの制限

サンプル間隔中に、まだ処理されていない非同期 I/O に対するオペレーティング・システムの制限値を超えた回数。オペレーティング・システムのカーネルは、プロセスまたはシステム全体が、任意の時点で保留できる非同期 I/O の最大数を制限します。『システム管理ガイド』を参照してください。また、オペレーティング・システムのマニュアルも参照してください。

要求されたディスク I/O と完了したディスク I/O

このデータは、要求されたディスク I/O の総数、および各 Adaptive Server エンジンによって完了された I/O の数とパーセンテージを示します。

“Total Requested Disk I/Os”と“Total Completed I/Os”は、同一の値または非常に近い値でなければなりません。飽和状態に達したために、要求された I/O が完了しない場合は、“Total Requested Disk I/Os”と“Total Completed I/Os”が非常に異なった値になります。

要求された I/O の値には、サンプル間隔中に開始されたすべての要求が含まれるため、これらの要求の一部は、サンプル間隔が終了してから完了する可能性があります。これらの I/O は“Total Completed I/Os”には含まれないので、パーセンテージが 100 より小さくなります。この場合は、飽和に関連する問題はありません。

逆の状況も当てはまります。サンプル間隔の開始前に I/O 要求が行われ、サンプル間隔中に完了した場合は、“Total Completed I/Os”の“% of Total”の値が 100% よりも大きくなります。

非常に多くのディスク I/O が要求され、完了したディスク I/O の数が要求された数よりも少ないことをデータが示している場合は、I/O を遅延させているオペレーティング・システムにボトルネックがある可能性があります。

要求されたディスク I/O の総数

Adaptive Server がディスク I/O を要求した回数。

完了したディスク I/O

各エンジンが I/O を完了した回数。“% of total”では、すべての Adaptive Server エンジンによって完了した I/O を合わせた総数に対して、各エンジンが I/O を完了した数の割合が、パーセンテージでレポートされます。

スレッド・モードでは、Adaptive Server はエンジン単位の完了したディスク I/O でなく、サーバワイドの概要を出力します。

```
Disk I/O Management
-------------------

 Max Outstanding I/Os            per sec      per xact       count  % of total
 -------------------------  ------------  ------------  ----------  ---------
  Server                            n/a           n/a          17        n/a
  Engine 0                          n/a           n/a           5        n/a
  Engine 1                          n/a           n/a           5        n/a
  Engine 2                          n/a           n/a          17        n/a

 I/Os Delayed by
  Disk I/O Structures               n/a           n/a           0        n/a
  Server Config Limit               n/a           n/a           0        n/a
  Engine Config Limit               n/a           n/a           0        n/a
  Operating System Limit            n/a           n/a          20        n/a

 Total Requested Disk I/Os         12.6         104.3         417

 Completed Disk I/O's
  Asynchronous I/O's
   Total Completed I/Os            12.6         104.3          41      100.0 %
  Synchronous I/O's
   Total Completed I/Os             0.0           0.0           0        n/a
 -------------------------  ------------  ------------  ----------
 Total Completed I/Os              12.6         104.3         417
```

この情報を使用して、オペレーティング・システムがすべてのエンジンによって行われるディスク I/O を処理できるかどうかを調べることができます。

デバイス・アクティビティの詳細

各論理デバイスでのアクティビティをレポートします。この情報は、データベース・デバイス全体の I/O のバランスがとれているかどうかをチェックする場合、および I/O を遅延させる可能性のあるデバイスを見つける場合に役立ちます。たとえば、“Task Context Switches Due To”のデータがデバイス競合の量が多いことを示している場合は、“Device Activity Detail”を使用して、どのデバイスが問題の原因となっているか判断できます。

このセクションでは、サーバ上の各データ・デバイスの I/O について、次の情報が出力されます。

- 論理デバイス名と物理デバイス名
- 読み込みと書き込みの数および I/O の総数
- デバイス上ですぐに付与されたデバイス・セマフォ要求の数と、プロセスがデバイス・セマフォを待機しなければならなかった回数。

読み込みと書き込み

デバイスへの読み込みまたは書き込みが実行された回数。“Reads”では、非同期プリフェッチによって読み込まれたページ数と、別の I/O アクティビティによってキャッシュに取り込まれたページ数がレポートされます。“% of total”カラムでは、デバイスに関する I/O の総数に対する読み込み数または書き込み数の割合を、パーセンテージでレポートします。

I/O の総数

あるデバイスに対する読み込みと書き込みを合計した数をレポートします。“% of total”カラムでは、すべてのデバイスに対して行われた読み込みと書き込みの総数に対して、指定した各デバイスでの読み込みと書き込みを合わせた数の割合を、パーセンテージでレポートします。

この情報を使用して、複数のディスクにわたる I/O の分散パターンをチェックしたり、複数のデバイスにまたがってディスク I/O のバランスがとれるように、オブジェクトの配置を決定したりできます。たとえば、このデータによって、一部のディスクを使用する頻度が他のディスクと比べて大きいことが示される場合があります。すべての I/O の大部分が、特定の名前を持つデバイスに対して行われている場合は、そのデバイスに存在するテーブルを調査して、問題の対処方法を決定できます。

『パフォーマンス＆チューニング・シリーズ：物理データベースのチューニング』の「第 1 章 データの物理的配置の制御」を参照してください。

付与されたミラー・セマフォと待機したミラー・セマフォ

ミラー・セマフォの要求がすぐに付与された回数と、セマフォがビジー状態にあったためにセマフォが解放されるまでタスクが待機しなければならなかった回数。“% of total”カラムでは、要求されたミラー・セマフォの総数に対して、デバイス・セマフォが付与された (またはタスクが待機しなければならなかった) 回数の割合が、パーセンテージで出力されます。このデータは、SMP 環境でだけ意味を持ちます。

Adaptive Server では、ディスク・ミラーリングが有効であるか、ディスク I/O が遅延された場合にのみ、セマフォを使用します。ディスク・ミラーリングは、設定パラメータをデフォルトの「無効」から変更する場合に有効になります。

待機した I/O 要求のパーセンテージが高くなっていることは、セマフォ競合の問題を示します。この場合は、物理デバイス上でデータを再分配する必要があります。

ネットワーク I/O 管理

各 Adaptive Server エンジンの次のネットワーク・アクティビティをレポートします。

- 要求されたネットワーク I/O の総数
- 遅延されたネットワーク I/O
- Tabular Data Stream (TDS) パケットの総数および送受信されたバイト数
- 送受信されたパケットの平均サイズ

各エンジンは独自のネットワーク I/O を行うため、このデータはエンジンごとに分割されています。アンバランスな状態が発生するのは、通常は、次のいずれかの場合です。

- タスクよりもエンジンの数が多いため、実行する作業がないエンジンから I/O がレポートされない。
- ほとんどのタスクは小さなパケットを送受信しているが、バルク・コピーなどの負荷の高い I/O を実行しているタスクもある。

出力例

```
Network I/O Management
----------------------

  Network I/O Requests             per sec      per xact       count % of total
  -------------------------  ------------   -----------    --------  --------
  Total Network I/O Requests       1384.2         938.4      166099        n/a
  Network I/Os Delayed                0.0           0.0           0      0.0 %

  Network Receive Activity        per sec      per xact       count
  -------------------------  ------------  ------------  ----------
  Total TDS Packets Rec'd          1380.7         936.1      165684
  Total Bytes Rec'd             2267752.3     1537459.1   272130277
  Avg Bytes Rec'd per Packet          n/a           n/a        1642

  Network Send Activity           per sec      per xact       count
  -------------------------  ------------   -----------  ----------
  Total TDS Packets Sent           1339.5         908.2      160743
  Total Bytes Sent                56194.2       38097.8     6743306
  Avg Bytes Sent per Packet           n/a           n/a          41
```

ネットワーク I/O 要求の総数

送受信されたパケットの総数をレポートします。

ネットワークが処理できる 1 秒当たりのパケット数を知っている場合は、Adaptive Server がネットワークの処理能力以上の I/O を発生させていないかどうかを調べることができます。

I/O がインバウンドまたはアウトバウンドのどちらであっても、問題は同じです。Adaptive Server はパケットのサイズよりも大きいコマンドを受信すると、コマンド全体を受信するまで処理を開始せずに待機します。したがって、複数のパケットを必要とするコマンドの実行は低速になり、I/O 要求の数が多くなります。

パケット当たりの平均バイト数が、サーバで設定されているデフォルトのパケット・サイズに近い場合は、いくつかの接続について、より大きなパケット・サイズを設定できます。すべての接続、または特定の接続がより大きなパケット・サイズを使用してログインできるように、ネットワーク・パケット・サイズを設定できます。

『パフォーマンス＆チューニング・シリーズ：基本』の「第 2 章 ネットワークとパフォーマンス」を参照してください。

遅延されたネットワーク I/O

I/O が遅延した回数。数値が常に 0 以外の場合は、ネットワーク管理者に問い合わせてください。

受信した TDS パケットの総数

各エンジンの受信した TDS パケットの数。“Total TDS Packets Rec’d” では、サンプル間隔中に受信されたパケットの数がレポートされます。

受信された総バイト数

各エンジンが受信したバイト数をレポートします。“Total Bytes Rec’d” では、サンプル間隔中に受信された総バイト数がレポートされます。

パケット当たりの平均受信バイト数

サンプル間隔中に受信されたすべてのパケットの平均バイト数。

送信された TDS パケットの総数

各エンジンが送信したパケットの数とサーバ全体での総数。

送信された総バイト数

サンプル間隔中に、各 Adaptive Server エンジンが送信したバイト数と、サーバ全体で送信されたバイト数。

パケット当たりの平均送信バイト数

サンプル間隔中に送信されたすべてのパケットの平均バイト数をレポートします。

Replication Agent

Replication Agent グループには、Replication Agent 用に設定されたすべてのデータベースに対する Replication Agent 固有のアクティビティが表示されます。

出力例

```
Replication Agent

----------------

Replication Agent:  primdb

Replication Server: MRP_REPSRVR1
                                  per sec      per xact       count % of total
                               -----------   -----------    --------  --------

 Log Scan Summary
   Log Records Scanned               n/a           n/a      710273       n/a
   Log Records Processed             n/a           n/a      127098       n/a
   Number of Log Scans               n/a           n/a           0       n/a
   Amount of Time for Log Scans (ms) n/a           n/a       19868       n/a
   Longest Time for Log Scan (ms)    n/a           n/a       19868       n/a
   Average Time per Log Scan (ms)    n/a           n/a         0.0       n/a

 Log Scan Activity
   Updates                           n/a           n/a           0       n/a
   Inserts                           n/a           n/a       66380       n/a
   Deletes                           n/a           n/a           0       n/a
   Store Procedures                  n/a           n/a           2       n/a
   DDL Log Records                   n/a           n/a           0       n/a
   Writetext Log Records             n/a           n/a           0       n/a
   Text/Image Log Records            n/a           n/a       60420       n/a
   CLRs                              n/a           n/a           0       n/a
   Checkpoints Processed             n/a           n/a           4       n/a
```

```
  SQL Statements Processed          n/a             n/a             0       n/a

Transaction Activity
  Opened                            n/a             n/a           140       n/a
  Commited                          n/a             n/a           141       n/a
  Aborted                           n/a             n/a             1       n/a
  Prepared                          n/a             n/a             1       n/a
  Delayed Commit                    n/a             n/a             0       n/a
  Maintenance User                  n/a             n/a             4       n/a

Log Extension Wait
  Count                             n/a             n/a             2       n/a
  Amount of time (ms)               n/a             n/a         14346       n/a
  Longest Wait (ms)                 n/a             n/a         14346       n/a
  Average Time (ms)                 n/a             n/a        7173.0       n/a

Schema Cache
 Usage
  Max Ever Used                     n/a             n/a             4       n/a
  Schemas reused                    n/a             n/a             0       n/a
 Forward Schema Lookups
  Count                             n/a             n/a             0       n/a


  Total Wait (ms)                   n/a             n/a             0       n/a
  Longest Wait (ms)                 n/a             n/a             0       n/a
  Average Time (ms)                 n/a             n/a           0.0       n/a

Backward Schema Lookups
  Count                             n/a             n/a             0       n/a
  Total Wait (ms)                   n/a             n/a             0       n/a
  Longest Wait (ms)                 n/a             n/a             0       n/a
  Average Time (ms)                 n/a             n/a           0.0       n/a

Truncation Point Movement
 Moved                              n/a             n/a             0       n/a
 Gotten from RS                     n/a             n/a             0       n/a

Connections to Replication Server
 Success                            n/a             n/a             0       n/a
 Failed                             n/a             n/a             0       n/a

Network Packet Information
  Packets Sent                      n/a             n/a         39429       n/a
  Full Packets Sent                 n/a             n/a          4381       n/a
  Largest Packet                    n/a             n/a             0       n/a
  Amount of Bytes Sent              n/a             n/a      73240036       n/a
  Average Packet                    n/a             n/a        1857.5       n/a

I/O Wait from RS
  Count                             n/a             n/a         48191       n/a
```

```
Amount of Time (ms                n/a             n/a         58995        n/a
Longest Wait (ms)                 n/a             n/a            17        n/a
Average Wait (ms)                 n/a             n/a           1.2        n/a
```

Log Scan Summary

Replication Server に対するログ・アクティビティの概要。Replication Agent は、実行中のデータベースのトランザクション・ログをスキャンし、スキャンした情報を Replication Server に送信します。

プライマリ・データベースのトランザクション・ログには Replication Agent が処理しない変更が含まれています。これは、これらの変更が複写対象としてマーク付けされていないオブジェクトに属するためです。

スキャンされたログ・レコード

Replication Agent がスキャンしたログ・レコードの総数をレポートします。

処理されたログ・レコード

複写対象としてマーク付けされているオブジェクトを含んでいたために、Replication Server に送信されたログ・レコードの総数をレポートします。

ログ・スキャンの数

Rep Agent がログをスキャンした総数。

ログ・スキャンの時間

Rep Agent がログのスキャンに費やした時間 (ミリ秒単位)。

ログ・スキャンの最長時間

Rep Agent がログ・スキャンに費やした最長時間 (ミリ秒単位)。

ログ・スキャン当たりの平均時間

Rep Agent がログ・スキャンに費やした平均時間 (ミリ秒単位)。

ログ・スキャン・アクティビティ

ログから次のアクティビティをレポートします。

- Updates － 処理された更新数。
- Inserts － 処理された挿入数。

- Deletes － 処理された削除数。
- Store Procedures － 複写対象としてマーク付けされているストアド・プロシージャの実行数。
- DDL Log Records － 複写されるデータ定義言語 (DDL) を含んでいるログ・レコードの数。この値はプライマリ・データベースで実行された DDL コマンド数に対応しないことがあります。これは、プライマリ・データベースで 1 つの DDL コマンドが実行されると、複数の DDL コマンド・ログ・レコードが生成される場合があるからです。
- Writetext Log Records － writetext によって生成された処理済みのログ・レコードの数。
- Text/Image Log Records － 複写対象としてマーク付けされた text または image データを持つテーブル用のデータ操作言語 (DML) ログ・レコードの数。
- CLRs － 処理された補正ログ・レコード (CLRs) の数。
- Checkpoints Processed － 取得されたチェックポイントの数。
- SQL Statements Processed － 処理された SQL 文の数。

トランザクション・アクティビティ

次のアクティビティをレポートします。

- Opened － プライマリ・ログで見つかった begin transaction コマンドの数。
- Committed － コミットされたトランザクションの数。
- Aborted － アボートされたトランザクションの数。
- Prepared －「準備」ステータスでのトランザクションの数。
- Delayed Commit － delayed commit トランザクションの数。
- Maintenance User － 管理ユーザがオープンしたトランザクションの数。

ログ拡張の待機

Replication Agent は通常の処理中にトランザクション・ログの終わりに達すると、Adaptive Server がログを拡張するまで待機する必要があります (この結果、プライマリ・データベースによってアクティビティが再開されます)。

“Log Extension Wait”では、次のアクティビティがレポートされます。

- Count － Adaptive Server がログを拡張するまで Replication Agent が待機した回数。
- Amount of time － Adaptive Server がログを拡張するまで Replication Agent が待機した時間 (ミリ秒単位)。

- Longest Wait － Adaptive Server がログを拡張するまで Replication Agent が待機しなければならなかった最長時間 (ミリ秒単位)。
- Average Time － Adaptive Server がログを拡張するまで Replication Agent が待機した平均時間 (ミリ秒単位)。

スキーマ・キャッシュ

- Max Ever Used － スキーマ・キャッシュ内のスキーマの最大数。
- Schemas Reused － 処理中に再利用されたスキーマの数。

“Schema Cache”では、次のアクティビティがレポートされます。

- Forward Schema － Replication Agent は、オブジェクト・スキーマの変更を検索するために、プライマリ・トランザクション・ログで前方スキャンを実行しなければならないことがあります。
- Count － Replication Agent が前方スキャンを実行した回数。
- Total Wait － Replication Agent が前方スキャンの実行に費やした時間 (ミリ秒単位)。
- Longest Wait － Replication Agent が前方スキャンの実行に費やした最長時間 (ミリ秒単位)。
- Average Time － Replication Agent が前方スキャンの実行に費やした平均時間 (ミリ秒単位)。

後方スキャン

Adaptive Server がトランザクション内で DDL を実行する場合、Replication Agent はプライマリ・トランザクション・ログで後方スキャンを実行する必要があります。“Backward Schema”では、次のアクティビティがレポートされます。

- Count － Replication Agent が後方スキャンを実行した回数。
- Total Wait － Replication Agent が後方スキャンの実行に費やした時間 (ミリ秒単位)。
- Longest Wait － Replication Agent が後方スキャンの実行に費やした最長時間 (ミリ秒単位)。
- Average Time － Replication Agent が後方スキャンの実行に費やした平均時間 (ミリ秒単位)。

トランケーション・ポイントの移動

次のアクティビティをレポートします。

- Moved － Replication Agent がプライマリ・データベースのセカンダリ・トランケーション・ポイントを移動した回数。
- Gotten from RS － Replication Agent が新しいセカンダリ・トランケーション・ポイントを取得するために、Replication Server にクエリを実行する回数。

Replication Server への接続

次のアクティビティをレポートします。

- Success － Replication Server への接続に成功した数。
- Failed － Replication Server への接続に失敗した数。

ネットワーク・パケット情報

次のアクティビティをレポートします。

- Packets Sent － Replication Server に送信されたパケット数。
- Full Packets Sent － Replication Server に送信されたフル・パケット数。
- Largest Packet － Replication Server に送信された最大パケット。
- Amount of Bytes Sent － Replication Server に送信されたバイト数。
- Average Packet － 平均パケット・サイズ。

RS からの I/O 待機

次のアクティビティをレポートします。

- Count － Replication Agent が Replication Server に ct_sendpassthru を発行した回数。
- Amount of Time － Replication Agent が Replication Server から取得した結果の処理に費やした時間 (ミリ秒単位)。
- Longest Wait － Replication Agent が結果の処理に費やした最長経過時間 (ミリ秒単位)。
- Average Wait － Replication Agent が結果の処理に費やした平均時間 (ミリ秒単位)。

索引

S

T

U

あ

い

え

お

か

き

く

こ

さ

し

す

せ

そ

た

ち

て

と

ね

の

は

ひ

ふ

へ

ま

み

め

も

ゆ

よ

り

る

ろ