

tech.E2SN knowledge base

Casa

Conferências virtuais E2SN

Leitura Adicional
Otimização Sistêmática do Oracle SQL na Vida Real

Oracle Living Books

- Oracle Exadata
- Oracle Internals and Architecture
- Oracle Performance
- Solução de problemas Oracle SQL

Oracle Performance Tuning, SQL Tuning and Troubleshooting Training

Scripts e ferramentas Oracle

Session Snapper

Mapa do site

Atividade recente do site

Atividade recente do site

Oracle Performance Tuning, SQL Tuning and Troubleshooting Training

editado por Tanel Poder

Treinamento de ajuste de desempenho e solução de problemas da Oracle

editado por Tanel Poder

Resolução de problemas de contenção de travas

editado por Tanel Poder

Treinamento de solução de problemas de desempenho Oracle

editado por Tanel Poder

ORADEBUG DOC

editado por Tanel Poder

Ver tudo

direito autoral

© 2009-2010 E2SN Pte Ltd.
Todos os direitos reservados.

[Scripts e ferramentas Oracle](#) >

Session Snapper

Se você deseja apenas fazer o download do Snapper, pode obtê-lo aqui:

- <http://blog.tanelpoder.com/files/scripts/snapper.sql>
(clique com o botão direito no arquivo e use Salvar como ... em vez de copiar e colar o conteúdo, pois alguns editores e emuladores de terminal bagunçam o código quando colar grande conteúdo - você acabaria com erros ORA-06550!)

No entanto, recomendo que você folheie o artigo para entender melhor as capacidades e limitações do Snapper!

Conteúdo

- 1 Introdução
- 2 Usando o Snapper
- 3 modo Snapper ASH
 - 3.1 Executar o Snapper no modo ASH em uma única sessão
 - 3.2 Execução do Snapper no modo ASH em todas as sessões (a instância inteira)
- 4 Modo de contador de desempenho / estatística do Snapper V \$
- 4.1 Executar o snapper em um subconjunto de sessões de instâncias
- 5 Baixe o Snapper
- 6 Leituras Adicionais
- 7 Feedback

Introdução

Sintaxe e exemplos do Oracle Session Snapper de Tanel Poder

O Oracle Session Snapper v3 tem algumas melhorias importantes em comparação com as versões anteriores (v2 e v1). Além de tirar instantâneos e relatar deltas de vários contadores de desempenho V \$ e X \$, ele também mostra os detalhes da atividade da sessão de V \$ SESSION. Isso é muito parecido com o que o Active Session History da Oracle faz (ASH é essencialmente apenas um histórico de exemplos de V \$ SESSION com alguns truques adicionais). No entanto, usar o ASH requer que você tenha licenças adicionais do Pacote de Diagnóstico, enquanto a consulta V \$ SESSION (que é como o Snapper funciona) não. A partir da versão 3.52, o Snapper também oferece suporte à amostragem no estilo ASH no Oracle 9.2 (nas versões anteriores, ele exigia o Oracle 10g +).

Graças à amostra de estilo ASH do V \$ SESSION, o Snapper agora pode relatar as sessões TOP, eventos de espera TOP, SQL_IDs TOP, procedimentos PLSQL TOP que causam a atividade do banco de dados e muito mais.

NB! Embora eu fale sobre a amostragem de estilo ASH no Snapper, não é o ASH da Oracle (licenciado separadamente). É apenas o mesmo conceito, mas uma ferramenta diferente. Tudo o que o Snapper faz é uma amostra da visão V \$ SESSION usando o antigo PL / SQL, de modo que vem "de graça" com o Oracle.

O Snapper foi criado para ser uma ferramenta de solução de problemas do desempenho ad-hoc rápida e fácil para os DBAs de campo que pr [Traduzir](#)

as mãos na massa sempre que um problema de banco de dados acontece (e consertar o problema rapidamente!). Ele se destina a ser uma **ferramenta flexível de solução de problemas de desempenho de primeira rodada**, uma ferramenta de ponto de entrada para solução de problemas, algo que você pode executar facilmente em alguns segundos em vez de ter que recorrer imediatamente a operações mais pesadas, como rastreamento SQL.

Observe que o Snapper não faz nenhuma mágica para você. Ele não faz nenhuma recomendação de desempenho inteligente nem oferece nenhum conselho de ajuste. Tudo o que faz (e faz bem) é apresentar os fatos. Ele tirará instantâneos de visualizações como V \$ SESSTAT e mais alguns e mostrará o quanto algum contador de desempenho aumentou para uma sessão durante o período de instantâneo. Além disso, o Snapper v3 mostrará um relatório TOP de amostras ativas da V \$ SESSION obtidas durante o período do instantâneo, assim como o ASH.

Aqui estão algumas coisas importantes sobre o Snapper:

1. O Snapper não cria nenhum objeto no banco de dados
2. Snapper é apenas um bloco PL / SQL anônimo, analisado e compilado em tempo real
3. O Snapper não requer nenhuma alteração no esquema ou nas configurações do banco de dados!

Observe que esta página que você está lendo agora não é um guia de solução de problemas de desempenho sistemático completo, mas apenas uma página de ferramenta que ilustra os recursos do Session Snapper.

Aqui estão alguns exemplos do que o novo Snapper 3 pode fazer:

Usando Snapper

Nos primeiros exemplos, estou supondo que você conseguiu descobrir o SID da sessão com mau comportamento (ou trabalho lento). Digamos que seja 144, então você pode executar o Snapper assim:

```
SQL> @snapper ash 5 1 144
```

Se você ainda não está familiarizado com o Snapper, ele usa 4 parâmetros:

1. O parâmetro 1 (cinza) indica **que tipo de medições fazer** :
 - No Snapper v3, você pode usar "ASH" para amostragem de estilo de histórico de sessão ativa
 - Em todas as versões do snapper, você pode usar "STATS" para tirar instantâneos de V \$ SESSTAT e outros contadores de desempenho
2. O parâmetro 2 (5) indica **a duração do período de instantâneo**
 - Neste exemplo, isso significa que o Snapper mede a atividade da sessão por 5 segundos e depois imprime o relatório
3. O parâmetro 3 (1) indica **quantas vezes obter o instantâneo de desempenho e o relatório** .
 - Para a solução de problemas na primeira rodada, geralmente uso apenas 1 instantâneo / relatório, mas às vezes uso um número maior para obter vários instantâneos e relatórios ao longo do tempo
4. O parâmetro 4 (144) é o SID do **SID da sessão** de interesse
 - Você pode especificar apenas um SID como no exemplo - 144
 - Você pode especificar vários SIDs separados por vírgulas - 144.145.200

- Você pode especificar "todos" para significar todas as sessões na instância - *todas*
- Você pode usar opções especiais como *user = SYSTEM* ou *program = sqlplus* ou *module = HR* para incluir sessões onde a coluna correspondente em V \$ SESSION corresponde à string dada (procure no script snapper para sintaxe completa)
- Finalmente, você pode usar qualquer subconsulta para especificar qualquer conjunto de SIDs como, "selecione sid de v \$ sessão onde username = 'SYSTEM' e programa diferente de 'sqlplus%'"

Veja os exemplos abaixo.

Modo Snapper ASH

Executando o Snapper no modo ASH em uma única sessão

Primeiro, vamos supor que identificamos o SID de uma sessão de trabalho em lote com mau comportamento é 144, vamos executar o snapper no modo ASH, com 5 segundos de duração do instantâneo, uma vez no SID 144:

```
SQL> @snapper ash 5 1 144
Sampling...

-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----+-----+-----+-----+
Active% | SQL_ID           | EVENT                | WAIT_...
-----+-----+-----+-----+
100%   | 5htvg1rhy5dfv    | enq: TM - contention | Application
-----+-----+-----+-----+
-- End of ASH snap 1, end=2010-03-22 19:54:09, seconds=5, sample=1

PL/SQL procedure successfully completed.
```

Aqui está, a saída é bem simples neste caso. Vamos ver o que significa:

- **A% ativa** é a métrica real medida - a atividade de uma sessão. Essa é a coluna que diz quanto do tempo de resposta durante a execução do Snapper nesta sessão foi gasto na operação / atividade listada naquela linha de saída.
- Portanto, 100% significa aqui que esta sessão (144) estava executando o SQL_ID **5htvg1rhy5dfv** 100% do seu tempo!
- Durante a execução desse SQL, também aconteceu de estar esperando por **enq: TM** - evento de espera de **contenção** durante todo o tempo total (os 5 segundos que gastamos amostrando com o Snapper).

Como a funcionalidade ash do Snapper (assim como o próprio ASH) funciona regularmente por meio de amostragem do V \$ SESSION (meio que tirando um instantâneo dele), ele não captura cada coisa que a sessão faz. No entanto, ele captura tudo o que é *significativo*.

Por exemplo, não podemos ter certeza absoluta de que esta sessão realmente estava 100% fazendo a operação mencionada acima, talvez ela estivesse ocupada executando aquele SQLID e esperando por aquele evento apenas 99% de seu tempo de resposta e fez outra coisa para 1 % de tempo. Snapper poderia ter perdido esse outro 1% devido à amostragem, pois essa "outra" coisa aconteceu tão rápido entre 2 amostras, então não foi capturada. No entanto, quando algo acontece com pouca frequência e muito rápido de modo que nem mesmo seja registrado em uma das várias amostras coletadas, então não pode ser muito significativo! Se algo consumir 50% do tempo de resposta, então certamente haverá algumas amostras (aproximadamente metade do total coletado) mostrando que "algo" está acontecendo.

O rodapé da saída do relatório snapper mostra quando o período de instantâneo terminou, durante quantos segundos a amostragem foi feita e quantas amostras V \$ SESSION foram feitas durante esse período. Acima, vemos que o Snapper coletou 42 amostras de ASH durante 5 segundos (mais de 8 amostras por segundo). O Snapper é escrito de forma que faz a amostragem muito rápido quando o período de instantâneo é curto (até 10 segundos), mas reduz a frequência de amostragem quando o período de instantâneo é mais longo, para reduzir a sobrecarga de medição. Durante longos períodos, a frequência de amostragem será de 1 Hz, uma amostragem por segundo, exatamente como ASH.

Continuando com o exemplo acima, identificamos que a sessão 144 estava esperando por um bloqueio de enfileiramento o tempo todo. Provavelmente queremos saber agora quem está nos bloqueando e que tipo de recurso estamos tentando bloquear. No caso de esperas de bloqueio de enfileiramento, as colunas de detalhes adicionais do evento de espera (Parâmetro2,3) nos darão informações adicionais sobre o objeto / recurso que estávamos tentando bloquear (procure no tipo V \$ LOCK para seus significados). Então, queremos amostrar as colunas **P2, P3** de V \$ SESSION para obter informações mais detalhadas sobre essa espera. Além disso, em caso de espera de enfileiramento, podemos experimentar V \$ SESSION. Coluna **BLOCKING_SESSION** (disponível a partir da 10g) para ver qual sessão estava nos bloqueando.

Portanto, em vez de confiar no agrupamento de atividades ASH TOP padrão, posso especificar essas colunas V \$ SESSION que desejo, usando a sintaxe **ASH =**. Observe que nem todas as colunas V \$ SESSION estão disponíveis no Snapper, mas as mais importantes estão. Posso apenas especificar a lista de colunas que desejo obter, separadas pelo sinal + :

```
SQL> @snapper ash=sql_id+event+wait_class+blocking_session+p2+
Sampling...
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----+-----+-----+-----+
Active% | SQL_ID          | EVENT           | WAIT_|
-----+-----+-----+-----+
100%   | 5htvg1rhy5dfv    | enq: TM - contention | Appli
-----+-----+-----+-----+
-- End of ASH snap 1, end=2010-03-22 19:55:31, seconds=5, sample_time=1000000000000000000

PL/SQL procedure successfully completed.
```

Agora vemos os detalhes adicionais imediatamente. O SID do nosso bloqueador é 162 (e posso executar o snapper nessa sessão para ver o que ele está fazendo) e o ID do objeto bloqueado que estamos esperando é 78452.

O exemplo acima realmente não mostrou a capacidade de criação de perfil do Snapper, pois a sessão estava paralisada executando a mesma instrução, esperando pela mesma coisa.

Vamos ver outro exemplo:

```
SQL> @snapper ash 5 1 156
Sampling...
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----+-----+-----+-----+
Active% | SQL_ID          | EVENT           | WAIT_|
-----+-----+-----+-----+
36%    | 3jbwa65aqmkvm    | read by other session | User
33%    | 3jbwa65aqmkvm    | direct path read  | User
28%    | 3jbwa65aqmkvm    | ON CPU          | ON CPI
3%     | 3jbwa65aqmkvm    | db file sequential read | User
```

```
-- End of ASH snap 1, end=2010-03-22 19:50:11, seconds=5, sa
```

Aparentemente, neste caso, a sessão não está presa esperando por um único evento de espera apenas, parece estar executando o mesmo SQL, pois todos os IDs de SQL no relatório de atividade da sessão TOP são os mesmos, mas durante a execução desse SQL a sessão aparentemente espera por eventos de espera diferentes e também passa algum tempo na CPU. Veja que o uso da CPU dessa sessão foi de 28% apenas durante a execução do Snapper e 72% do tempo de resposta restante é todo gasto em eventos de espera físicos relacionados ao IO (36% + 33% + 3%).

Portanto, ao solucionar o problema do desempenho dessa sessão, eu já saberia que 100% do tempo de resposta (durante a execução do snapper, pelo menos) foi gasto executando o SQL com ID **3jbwa65aqmkvm** e o próprio processo de execução esperou 72% do seu tempo por IO, então agora você sabe exatamente em qual instrução SQL procurar.

Vamos ver mais um exemplo:

```
SQL> @snapper ash 5 1 156
Sampling...
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----+
Active% | SQL_ID          | EVENT                                | WAIT_% |
-----+
30%    | gvgdv2v90wfa7    | db file sequential read             | User   |
7%     | 0bzhqhhj9mpaa   | db file sequential read             | User   |
5%     | 75621g9y3xmvd   | db file sequential read             | User   |
5%     | 05s4vdwsf5802   | db file sequential read             | User   |
5%     | 5raw2bxz227wp   | db file sequential read             | User   |
2%     | 0yas01u2p9ch4   | db file sequential read             | User   |
-----+
-- End of ASH snap 1, end=2010-03-22 19:50:29, seconds=5, sa
```

Nesse caso, parece que há muitas instruções diferentes executadas durante a execução do snapper de 5 segundos. O TOP parece ser aquele que ficou ativo pelo menos 30% do seu tempo (e aguardando a leitura sequencial do arquivo db).

Como há várias instruções diferentes relatadas, todas levando uma quantidade notável de tempo de resposta - e tudo por causa dos eventos de espera de E / S do usuário, este pode ser um caso para investigar problemas do subsistema de IO (de repente, todas as instruções SQL relatam IO como suas principais esperas)

NB! Há mais uma coisa interessante e importante a ver - a soma das amostras de% ativa é de apenas 54%! Isso significa que esta sessão foi ativa apenas (aproximadamente) 54% durante o tempo de amostragem! Portanto, o resto do tempo ele estava ocioso, aguardando a próxima solicitação de entrada do aplicativo, pela rede. Nos casos em que você vê a sessão do banco de dados sendo ativa apenas por uma pequena minoria de tempo, então não faz sentido ajustar nada no banco de dados - você precisará ver por que o aplicativo não está enviando novas solicitações ao banco de dados rapidamente o suficiente.

Os principais motivos para a sessão ficar inativa são simples:

- Tempo de reflexão do usuário. O usuário foi tomar café ou o PC está tão lento que não dá para fazer muito com o aplicativo :-)
- Tempo de reflexão do aplicativo. Isso geralmente acontece quando os servidores de aplicativos estão sobrecarregados ou contêm algum código incorreto. O aplicativo está tão ocupado ou preso que não é capaz de

- enviar as solicitações ao banco de dados e processar os dados resultantes muito rápido.
- Por último, latência e taxa de transferência da rede. Eu deixei isso deliberadamente como o último, pois faz sentido ver o que os usuários e aplicativos (de níveis mais elevados) estão realmente fazendo, em vez de assumir imediatamente que o problema deve estar na rede.

Observe que, como eu disse antes, o Snapper é uma ferramenta para fornecer a você os fatos, números de desempenho. Ele não faz recomendações de desempenho para você, você ainda precisa seguir uma abordagem sistemática para solução de problemas e ajustes, o Snapper é a ferramenta que dá suporte a isso. Escreverei mais sobre ajuste sistemático e abordagem de solução de problemas no meu [livro](#) atual sobre Oracle Performance (ou você pode apenas assistir a um de [meus seminários](#) ;-)

Portanto, trata-se de instantâneos de sessão única, mas ei, o Snapper pode fazer mais!

Às vezes, você tem várias sessões que deseja medir, às vezes nem conhece os SIDs com antecedência e às vezes deseja ver toda a atividade da instância. O Snapper também pode ajudar aqui!

Executar o Snapper no modo ASH em todas as sessões (a instância inteira)

Tornar o snapper para medir a atividade de todas as sessões é fácil, basta especificar **todas em** vez do SID:

```
SQL> @snapper ash 5 1 all
Sampling...

-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----+-----+-----+-----+
Active% | SQL_ID           | EVENT             | WAIT_...
-----+-----+-----+-----+
 69%  | 3h1z39qtgwc5h    | db file scattered read | User
 29%  | fy8n9175jyj7s    | db file scattered read | User
   9%  |                      | log file parallel write | System
   2%  | fy8n9175jyj7s    | ON CPU             | ON CPI
   2%  |                      | control file parallel wri | System

-- End of ASH snap 1, end=2010-03-22 17:33:17, seconds=5, sample=1000
PL/SQL procedure successfully completed.
```

Observe que a% ativa ainda mostra o tempo de resposta de uma (1) sessão! Se digamos que 2 sessões estão esperando por um bloqueio em todo o seu tempo de resposta, você verá um tempo de espera de 200% para isso.

Digamos que eu queira detalhar quantas sessões individuais estão esperando por algum evento, posso adicionar a coluna **sid** ao parâmetro ASH (como visto abaixo), agora o group by for TOP report é feito por SID, event e wait_class:

```
SQL> @snapper ash=sid+event+wait_class 5 1 all
Sampling...

-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----+-----+-----+-----+
Active% | SID    | EVENT             | WAIT_CLASS
-----+-----+-----+-----+
 95%  | 133    | db file scattered read | User I/O
  8%  | 165    | control file parallel wri | System I/O
   5%  | 133    | db file sequential read | User I/O

-- End of ASH snap 1, end=2010-03-22 17:33:53, seconds=5, sample=1000
PL/SQL procedure successfully completed.
```

Veja como o SID 133 esperou ~ 95% de seu tempo de resposta para *leitura dispersa de arquivo db* e 5% restante para *leitura sequencial de arquivo db*. O uso ocasional da CPU entre as operações de IO era tão curto que a amostragem V \$ SESSION do Snapper nem percebeu, em outras palavras, esta sessão não usou CPU significativamente durante a execução do Snapper.

Obtendo vários relatórios de atividade de sessão com o Snapper

Às vezes, apenas um relatório TOP como visto acima pode não ser suficiente. Você pode querer examinar os dados de atividade da sessão / instância de vários ângulos diferentes (agrupar as amostras de ASH por campos diferentes). É por isso que adicionei os **parâmetros ash1, ash2 e ash3** à sintaxe do Snapper, você pode mostrar até 4 análises de atividades ASH TOP em uma execução do Snapper. Por exemplo, digamos que eu queira quebrar a atividade da sessão também pelo object_id do pacote PLSQL e pelo ID do procedimento nele, para ver se há algum pacote PL / SQL causando a maior parte do trabalho (essas colunas estão disponíveis a partir de 10.2.0.3 em V \$ SESSÃO):

```
SQL> @snapper ash=sid+event+wait_class,ash1=plsql_object_id+plsql_subprogram...
```

```
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )
```

Active%	SID	EVENT	WAIT_CLASS
70%	133	db file scattered read	User I/O
25%	133	db file sequential read	User I/O
9%	165	control file parallel write	System I/O
7%	166	log file parallel write	System I/O
5%	133	ON CPU	ON CPU
Active%	PLSQL_OBJE	PLSQL_SUBP	SQL_ID
43%			dv59rkngpa8m1
30%			b8qywubug00u3
23%			fgkm2nvqhyyqh
16%			82hxvr8kxuzjq
2%	5357	135	1gu8t96d0bdmu
2%	4345	105	

```
-- End of ASH snap 1, end=2010-03-22 17:34:51, seconds=5, sample_time=100ms
```

Digamos que eu queira incluir mais um relatório, que me mostra o programa, módulo e ação TOP dos exemplos de atividade da sessão:

```
SQL> @snapper ash=sid+event+wait_class,ash1=plsql_object_id+plsql_subprogram...
```

```
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )
```

Active%	SID	EVENT	WAIT_CLASS
100%	133	db file scattered read	User I/O
5%	165	control file parallel write	System I/O
2%	162	ON CPU	ON CPU
2%	167	db file parallel write	System I/O
2%	166	log file parallel write	System I/O
Active%	PLSQL_OBJE	PLSQL_SUBP	SQL_ID
77%			a5xyjp9gt796s
23%			4g4u44bk830ms
12%			

Active%	PROGRAM	MODULE
100%	sqlplus@mac01 (TNS V1-V3)	sqlplus@mac01 (TNS V1-V:
5%	oracle@solaris02 (CKPT)	
2%	oracle@solaris02 (DBW0)	
2%	oracle@solaris02 (CJQ0)	
2%	oracle@solaris02 (LGWR)	

-- End of ASH snap 1, end=2010-03-22 17:35:06, seconds=5, sai

Da mesma forma, você pode usar o parâmetro **ash3**. Observe que se você apenas usar os parâmetros **ash** sem = e lista de colunas, ele usará alguns agrupamentos TOP padrão, que são configuráveis - procure CONFIG no arquivo snapper.sql.

Todo o material acima está disponível no Snapper v3. No entanto, é apenas uma parte dos recursos completos do Snapper. Às vezes, saber o TOP SQL_ID e o evento de espera não é suficiente. Pense nos casos em que uma sessão está 100% na CPU e não espera por nada e não há nada obviamente errado com o plano de execução do SQL. É quando você quer saber exatamente o que a sessão está fazendo. É aqui que os contadores de desempenho dinâmico do Oracle entram em ação - estou falando principalmente das estatísticas V \$ SESSTAT:

Modo de contador de desempenho / estatística do Snapper V \$

O Snapper tem o modo Estatísticas desde a versão 1. É por isso que escrevi o Snapper inicialmente, para me ajudar na solução de problemas de desempenho, especialmente nos casos em que as ferramentas convencionais como SQL trace ou ferramentas de desempenho de nível de instância como AWR / Statspack não apareceram qualquer coisa útil. A partir da v3, o Snapper não mostra automaticamente as estatísticas do V \$ SESSTAT, já que o Snapper deve ser usado como uma ferramenta de solução de problemas em toda a instância agora (e consultar o V \$ SESSTAT para todas as sessões pode ser caro quando você tem milhares de sessões em sua instância).

Então, se você quiser detalhar as amostras V \$ SESSTAT, você precisará usar o parâmetro de **estatísticas**, que diz ao Snapper para tirar instantâneos de V \$ SESSTAT e outras visualizações V \$ (ou em vez disso, você pode usar **tudo** que ativa tanto o **ash** como **estatísticas**)

Além disso, você pode especificar que tipo de estatísticas capturar, na opção de **coleta**. As opções são as seguintes (retiradas diretamente da seção de documentação no cabeçalho do script do Snapper):

- **Estatísticas no nível da sessão:**
 - s - Estatísticas da sessão de v \$ sesstat
 - t - Informação do modelo de tempo de sessão de v \$ sess_time_model
 - w - Sessão estatísticas de espera de v \$ Session_event e v \$ session_wait
- **Estatísticas em nível de instância:**
 - L - instância Latch obtém estatísticas (obtém + imediato_gets) de v \$ latch
 - e - bloqueio de enfileiramento de instância obtém estatísticas de v \$ enqueue_stat
 - b - buffer get Where statistics from x \$ kcbsw - útil em versões até 10.2.x
 - a - Tudo acima

Se a opção de **coleta** for omitida (mas as **estatísticas** estiverem habilitadas), o Snapper coletará apenas as estatísticas do nível da sessão (s, t, w).

Segue-se um exemplo, estou tirando instantâneos de atividades de todas as sessões, mas também peço ao Snapper que reúna as estatísticas do modelo de tempo (t) e V \$ SESSTAT, mas inclui apenas estatísticas do modelo de tempo (tinclude) que têm string % CPU% no nome e inclui apenas estas estatísticas V \$ SESSTAT (sinclude) que contêm a palavra% parse%:

```
SQL> @snapper ash=event+wait_class,stats,gather=ts,tinclude=CI
Sampling...
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----  

          SID, USERNAME , TYPE, STATISTIC  

-----  

    119, SYS      , STAT, parse count (total)  

    133, SYS      , STAT, parse time cpu  

    133, SYS      , STAT, parse time elapsed  

    133, SYS      , STAT, parse count (total)  

    133, SYS      , STAT, parse count (hard)  

   159, (J000)    , TIME, DB CPU  

   161, (MMON)    , STAT, parse count (total)  

   161, (MMON)    , STAT, parse count (hard)  

   161, (MMON)    , TIME, background cpu time  

   162, (CJQ0)    , STAT, parse count (total)  

   162, (CJQ0)    , TIME, background cpu time  

   167, (DBW0)    , TIME, background cpu time  

   170, (PMON)    , TIME, background cpu time  

-- End of Stats snap 1, end=2010-03-22 18:02:28, seconds=5

-----  

          Active% | EVENT           | WAIT_CLASS  

-----  

  105% | ON CPU             | ON CPU  

  17% | db file sequential read | User I/O  

   2% | log file parallel write | System I/O  

-- End of ASH snap 1, end=2010-03-22 18:02:28, seconds=5, sample_id=1

PL/SQL procedure successfully completed.
```

Na parte superior da saída acima, veja como o modo de estatísticas do Snapper facilmente revelará uma sessão 133 que está fazendo 2,25 mil análises difíceis por segundo (o que é muito!). Eu poderia usar qualquer uma das outras estatísticas disponíveis no V \$ SESSTAT (mais de 600 estatísticas diferentes para cada sessão no Oracle 11.2). Por exemplo, se eu tivesse usado "redo size" no parâmetro **sinclude**, teria facilmente visto qual sessão gera mais refazer.

Outra coisa a observar é que, embora a seção ASH abaixo diga que houve atividade da CPU equivalente a 105% do tempo de resposta de uma única sessão (o que significa que deve ter havido mais de uma sessão usando essa CPU), as estatísticas do modelo de tempo acima (estatísticas com type = TIME) não relatam uso significativo da CPU do banco de dados. O problema aqui é a granularidade da medição. As estatísticas do modelo de tempo são atualizadas nas visualizações V \$ apenas no final da chamada do banco de dados, mas se a chamada do banco de dados for de longa duração (como no meu caso de teste), as estatísticas em V \$ SESS_TIME_MODEL são atualizadas aproximadamente a cada 5 segundos por padrão. Graças ao meu curto tempo de execução do Snapper (5 segundos), o script aparentemente terminou antes que a atualização do array na memória V \$ SESS_TIME_MODEL ocorresse naquela sessão com a longa chamada do banco de dados. Se você executar o snapper com maior tempo de amostragem,

Até agora, cobrimos como executar o snapper em uma sessão ou em toda a instância. Às vezes, você deseja monitorar todas as sessões de um usuário, serviço ou programa específico. Isso também é possível no Snapper:

Executando o snapper em um subconjunto de sessões de instâncias

A partir da v3, o Snapper tem maneiras convenientes de especificar sessões pertencentes a um usuário, aplicativo, serviço específico etc. Por exemplo, em vez de especificar o SID como o quarto parâmetro, você pode simplesmente escrever user = XYZ (ou username = XYZ):

```
SQL> @snapper ash=sid+event+wait_class,ash1=sid+sqlid+module,:Sampling...
-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.net )

-----  

          SID, USERNAME , TYPE, STATISTIC  

-----  

      9, SOE      , TIME, DB CPU  

     17, SOE      , TIME, DB CPU  

    21, SOE      , TIME, DB CPU  

   144, SOE      , STAT, parse count (total)  

  144, SOE      , TIME, DB CPU  

  156, SOE      , TIME, DB CPU  

-- End of Stats snap 1, end=2010-03-22 18:34:09, seconds=5

-----  

Active% | SID | EVENT                                | WAIT_CLASS  

-----  

100%   | 21  | read by other session                      | User I/O  

93%    | 144 | read by other session                      | User I/O  

83%    | 156 | read by other session                      | User I/O  

73%    | 9   | read by other session                      | User I/O  

54%    | 17  | db file scattered read                     | User I/O  

27%    | 9   | db file scattered read                     | User I/O  

27%    | 17  | read by other session                      | User I/O  

17%    | 156 | db file scattered read                     | User I/O  

17%    | 17  | direct path read                          | User I/O  

  7%   | 144 | ON CPU                                     | ON CPU

-----  

Active% | SID | SQL_ID                               | MODULE  

-----  

100%   | 21  | 3jbwa65aqmkvm                         | Process Orders  

100%   | 9   | 3jbwa65aqmkvm                         | Process Orders  

100%   | 144 | 3jbwa65aqmkvm                         | Process Orders  

100%   | 156 | 3jbwa65aqmkvm                         | Process Orders  

100%   | 17  | 3jbwa65aqmkvm                         | Process Orders

-- End of ASH snap 1, end=2010-03-22 18:34:09, seconds=5, sample_time=5
```

Graças aos parâmetros especificados acima (ash e ash1), obtemos análises diferentes dos mesmos dados de atividade de sessão e medimos apenas as sessões pertencentes ao usuário SOE (em outras palavras, onde username = 'SOE' em V \$ SESSION).

Além do **usuário**, você pode usar outros parâmetros (autoexplicativos):

1. nome de usuário (igual ao usuário)
2. Sid
3. spid (igual a pid e ospid)
4. programa
5. máquina
6. osuser
7. módulo
8. ação
9. ID do Cliente

Você não pode combinar esses parâmetros em qualquer condição AND ou OR, você só pode passar um parâmetro por vez usando a sintaxe conveniente acima.

No entanto, se você quiser ser muito preciso sobre quais sessões o Snapper monitora (se quiser adicionar várias e / ou condições para selecionar sessões de interesse), você pode fazer a seleção de sessão da maneira antiga (compatível com Snapper v1 e v2), verifique o texto em vermelho:

```
SQL> @snapper ash=sid+event+wait_class,ash1=sid+sqlid+module,
Sampling...

-- Session Snapper v3.10 by Tanel Poder @ E2SN ( http://tech.e2sn.com )

-----  

SID, USERNAME , TYPE, STATISTIC  

-----  

9, SOE      , TIME, DB CPU  

17, SOE      , TIME, DB CPU  

21, SOE      , TIME, DB CPU  

144, SOE     , TIME, DB CPU  

156, SOE     , TIME, DB CPU  

-- End of Stats snap 1, end=2010-03-22 18:34:32, seconds=5

-----  

Active% | SID | EVENT | WAIT_CLASS  

-----  

97%   | 9   | read by other session | User I/O  

74%   | 144 | direct path read    | User I/O  

72%   | 156 | db file scattered read | User I/O  

72%   | 21  | direct path read    | User I/O  

72%   | 17  | direct path read    | User I/O  

28%   | 156 | read by other session | User I/O  

26%   | 17  | db file scattered read | User I/O  

26%   | 21  | read by other session | User I/O  

26%   | 144 | read by other session | User I/O  

3%    | 9   | db file scattered read | User I/O

-----  

Active% | SID | SQL_ID | MODULE  

-----  

100%  | 21  | 3jbwa65aqmkvm | Process Orders  

100%  | 9   | 3jbwa65aqmkvm | Process Orders  

100%  | 144 | 3jbwa65aqmkvm | Process Orders  

100%  | 156 | 3jbwa65aqmkvm | Process Orders  

100%  | 17  | 3jbwa65aqmkvm | Process Orders

-- End of ASH snap 1, end=2010-03-22 18:34:32, seconds=5, sample=5

PL/SQL procedure successfully completed.

SQL>
```

Você pode escrever qualquer subconsulta entre aspas duplas como o parâmetro SID, desde que caiba na linha de comando e retorne uma coluna de número único com uma lista de SIDs nela!

No meu caso, eu "seleciono SID de V \$ SESSION onde", mas esta consulta não precisa nem mesmo consultar V \$ SESSION, mas pode consultar, digamos, alguma tabela de agendamento de jobs batch E-Business Suite, PeopleSoft ou SAP (com SIDs de trabalhos em lote em execução) em vez disso!

Baixe o Snapper

Esta foi apenas uma introdução aos recursos do Snapper! Ele pode reunir e mostrar muito mais dados de desempenho e até mesmo persisti-los em um arquivo de **rastreamento** se você usar a opção de **rastreamento** !

Você pode baixar a última versão do Snapper aqui:

- Snapper v3.5 - <http://blog.tanelpoder.com/files/scripts/snapper.sql>

Como o snapper v3.5 agora oferece suporte ao Oracle 9.2, não há mais necessidade de usar o antigo Snapper v2. No entanto, se você precisar por algum motivo, aqui está:

- Snapper v2 - http://blog.tanelpoder.com/files/scripts/snapper_v2.sql

NB! Clique com o botão direito no link do arquivo e use Salvar como ... em vez de copiar e colar o conteúdo, pois alguns editores e emuladores de terminal bagunçam o código ao colar um conteúdo grande - você acabaria com erros ORA-06550!

Leitura Adicional

Já escrevi vários artigos do Snapper anteriormente em meu blog (sobre as versões 1 e 2), então você também pode navegar por eles (observe que há algumas alterações de sintaxe no Snapper v3. Você pode ver todos os artigos relacionados ao Snapper em meu blog aqui:

- <http://blog.tanelpoder.com/?s=snapper>

Claro, se você quiser ficar realmente bom no uso do script Snapper para ajuste de desempenho prático e solução de problemas, então verifique minha [oferta de seminários](#) (seminários online em breve! ;-)

Comentários

Escrevi o Snapper original há vários anos, mas ainda não tenho uma imagem clara de quantas pessoas o estão realmente usando e como o estão usando, portanto, se o Snapper o ajudou no passado (ou o está ajudando agora :) então, por favor, mande-me um e-mail com algumas linhas de como você o usou!

-
Tanel Poder
tanel@tanelpoder.com

Comentários

Você não tem permissão para adicionar comentários.