

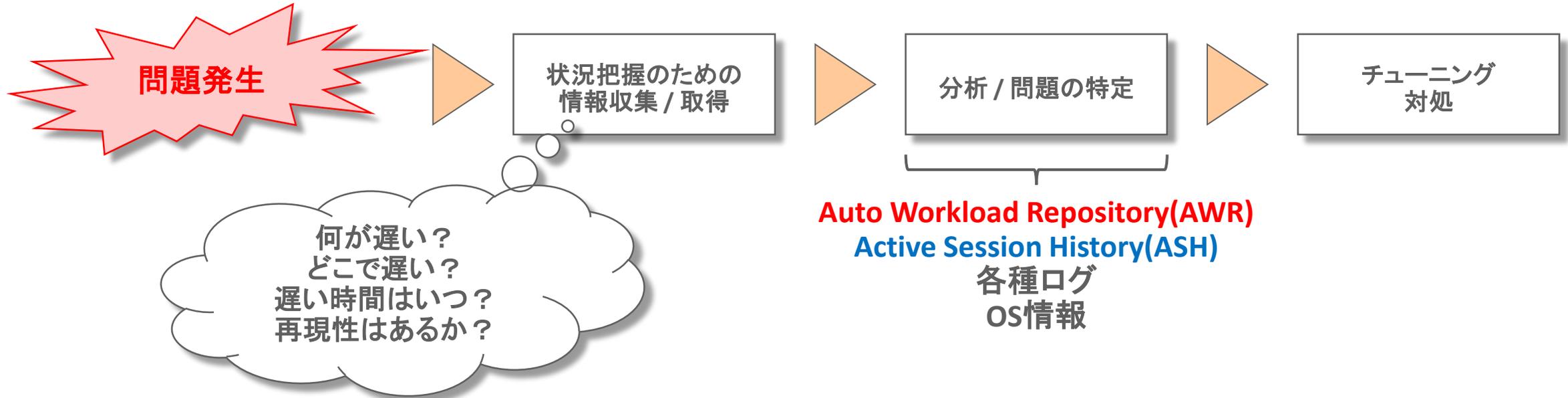
- 以下の事項は、弊社の一般的な製品の方向性に関する概要を説明するものです。また、情報提供を唯一の目的とするものであり、いかなる契約にも組み込むことはできません。以下の事項は、マテリアルやコード、機能を提供することをコミットメント(確約)するものではないため、購買決定を行う際の判断材料になさらないで下さい。オラクル製品に関して記載されている機能の開発、リリースおよび時期については、弊社の裁量により決定されます。

OracleとJavaは、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。文中の社名、商品名等は各社の商標または登録商標である場合があります。

アジェンダ

- 1 一般的なDBチューニングの流れ
- 2 AWR / ASHの概要
- 3 AWRの活用方法
- 4 ASHの活用方法
- 5 ケーススタディ

一般的なDBチューニングの流れ



アジェンダ

- 1 一般的なチューニングの流れ
- 2 AWR / ASHの概要**
- 3 AWRの活用方法
- 4 ASHの活用方法
- 5 ケーススタディ

AWR / ASHの概要

- 情報の特性、使用ケースから見たAWR と ASH

AWR(Automatic Workload Repository)

- 情報の特性

スナップショット取得タイミングにおけるDB稼働情報の集合
スナップショット間の差分を取得することで特定期間のDB稼働状態を確認可能

- 使用ケース

特定の期間内のインスタンス全体の状態を把握するために使用
スナップショット取得の負荷を考慮すると通常運用で1時間または30分、試験時でも10分程度が取得の最小間隔

ASH(Active Session History)

- 情報の特性

1秒または10秒間隔のアクティブなセッション状態情報の集合
アクティブ(= 処理中)なセッションの状態や実行処理を確認可能

- 使用ケース

一時的なパフォーマンスの問題を診断する際にセッションレベルの分析を行うために使用
1秒または10秒間隔のため、1秒未満の処理については情報が残っていない可能性がある

AWR / ASHの概要

• AWR / ASHを使用した調査

調査スコープ

発生期間	問題とその範囲	AWR	ASH
一定期間	DB全体の処理遅延	◎	△
	特定セッション・処理の遅延	△	○
一時的	DB全体の処理遅延	△	○
	特定セッション・処理の遅延	×	○

* Oracle Database 統計の詳細を参照するには、AWRを使用する必要があります

– ただし、AWRとASHの両方を使用して調査を進めるケースもあります。

エンキュー競合

AWRの全体的な傾向の分析にて待機イベントを特定

ASHでブロkkerセッションや対象SQLを特定

一時表領域書き出し

AWRの全体的な傾向の分析にて待機イベントを特定
AWRの全体的な傾向の分析にてPGAの状態を確認

ASHで特定のセッションやSQL実行時に確保する
PGAサイズや使用している一時表領域サイズを特定

AWR / ASH の概要

• AWRの概要

– インスタンスのメモリ上で保持する稼働情報をSYS_AUX内のテーブルに格納した情報



– AWR情報参照のためのDBA_HISTから始まる静的ビューが提供

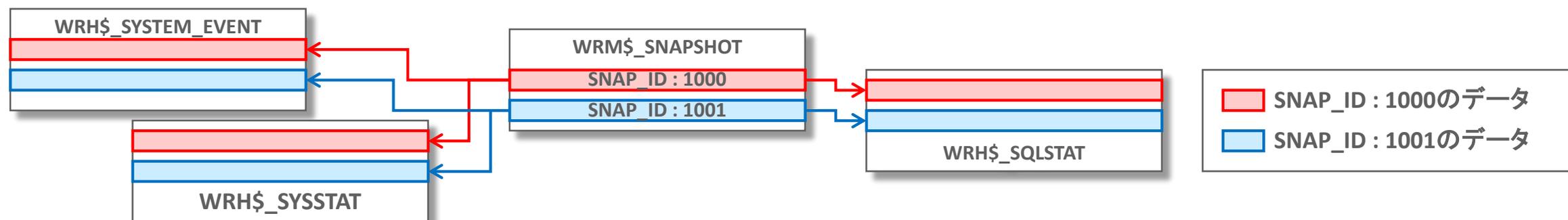
V\$ビュー	AWR実テーブル	AWRのディクショナリビュー
– (G)V\$SYSTEM_EVENT	→ WRH\$_SYSTEM_EVENT	→ DBA_HIST_SYSTEM_EVENT
– (G)V\$SYSSTAT	→ WRH\$_SYSSTAT	→ DBA_HIST_SYSSTAT
– (G)V\$SQL	→ WRH\$_SQLSTAT	→ DBA_HIST_SQLSTAT

* 上記の動的ビューは、AWRの各ビューと同様の情報を参照するためのビューです
実際の取り込み時の参照先はV\$ビューの基となる固定表等から取得されることがあります

AWR / ASH の概要

- AWRの概要

- スナップショットを取得するとその時点の各情報が対応するテーブルに格納される



スナップショットとはSNAP_IDで関連付け (= 同一タイミングで取得された) られ、
テーブルに分散して格納された稼働情報の集合体

AWR / ASH の概要

• AWRの概要

– 分析でAWR情報を使用する場合の注意点

- スナップショット内の情報は一部を除き、インスタンス起動からの累積値が格納される



特定の期間の分析を行う場合、各情報の終了時点と開始時点の対応する値にスナップショット情報の差分算出が必要

SNAP_ID	INSTANCE_NUMBER	STAT_NAME	VALUE
1085	1	redo size	7,580,255,456
1086	1	redo size	9,741,652,163

SNAP_ID 1085までに約7.5GBのREDOログを生成
SNAP_ID 1086までに約9.7GBのREDOログを生成
⇒ SNAP_ID 1085～1086の間に約2.2GBのREDOログを生成

- インスタンス再起動がすると累積値がリセットされ、スナップショット間の差分は取得できない



再起動直後の最初のスナップショットについては差分算出は不要(インスタンス起動からの値として分析)
再起動をまたいだ差分算出結果は値として意味がない



AWRの下記機能は、分析期間の中に再起動が入っている場合、使用できない

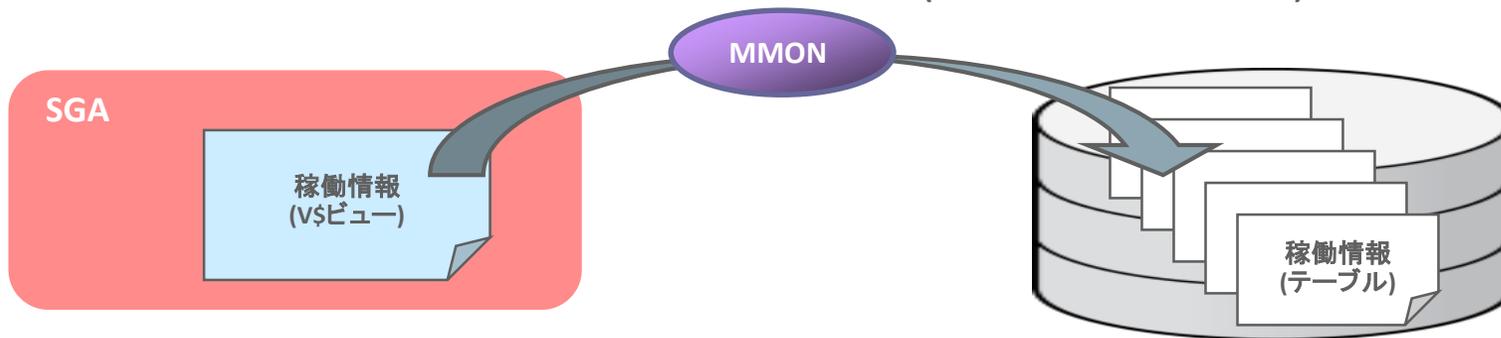
- AWRレポート生成
- AWRSQLレポート生成

AWR / ASH の概要

- AWRの概要

- スナップショットの取得

- MMONプロセスによる定期的な自動取得(デフォルト1時間)



- DBMS_WORKLOAD_REPOSITORYパッケージを使用した手動取得

```
SQL> EXECUTE DBMS_WORKLOAD_REPOSITORY.CREATE_SNAPSHOT;
```

- Oracle Enterprise Managerを使用した取得

AWR / ASH の概要

• AWRの概要

– AWRレポートの作成

- 指定した開始と終了スナップショット間の各情報の差分を算出し、レポートとして出力したファイル
 - 出力項目は、Oracle Database 側で予め定義された項目
 - 開始スナップショットと終了スナップショット間にインスタンス再起動がある場合は作成できない

- レポートの出力の形式としては、下記の2タイプが存在する

WORKLOAD REPOSITORY report for

DB Name	DB Id	Instance	Inst Num	Startup Time	Release	RAC
SHARED	784781254	sharedbl	1	30-7月 -14 14:10	11.2.0.4.0	YES

Host Name	Platform	CPUs	Cores	Sockets	Memory (GB)
consdb12n1.jp.oracle.com	Linux x86_64	2	1	1	3.91

Snap Id	Snap Time	Sessions	Curs/Sess	Instances
Begin Snap	1129 10-9月 -14 13:00:20	53	2.2	2
End Snap	1130 10-9月 -14 14:00:04	53	2.2	2
Elapsed	59.74 (mins)			
DB Time	0.18 (mins)			

Report Summary

Load Profile	Per Second	Per Transaction	Per Exec	Per Call
DB Time(s):	0.0	0.0	0.00	0.00
DB CPU(s):	0.0	0.0	0.00	0.00
Redo size (bytes):	1,021.6	2,038.9		
Logical read (blocks):	84.4	128.8		
Block changes:	4.2	8.3		
Physical read (blocks):	0.1	0.2		
Physical write (blocks):	0.4	0.7		
Read IO requests:	0.1	0.2		
Write IO requests:	0.3	0.5		
Read IO (MB):	0.0	0.0		
Write IO (MB):	0.0	0.0		
Global Cache blocks received:	0.1	0.2		
Global Cache blocks served:	0.2	0.4		
User calls:	1.8	3.6		
Parse (SQL):	0.9	1.7		
Hard parse (SQL):	0.0	0.0		
SQL Work Area (MB):	0.1	0.1		
Logons:	0.1	0.1		
Executes (SQL):	2.7	5.4		
Rollbacks:	0.0	0.0		
Transactions:	0.5	0.5		

HTML形式

テキスト形式

```
WORKLOAD REPOSITORY report for
+
DB Name      DB Id Instance  Inst Num Startup Time  Release  RAC+
+-----+-----+-----+-----+-----+-----+-----+
SHARED      784781254 sharedbl    1 30-7月 -14 14:10 11.2.0.4.0 YES+
+-----+-----+-----+-----+-----+-----+
Host Name    Platform                                CPUs Cores Sockets Memory(GB)+
+-----+-----+-----+-----+-----+-----+
consdb12n1.jp.o Linux x86_64-bit                          2 1 1 3.91+
+-----+-----+-----+-----+-----+-----+
Snap Id      Snap Time                               Sessions Curs/Sess Instances+
+-----+-----+-----+-----+-----+-----+
Begin Snap:  1129 10-9月 -14 13:00:20          53      2.2      2+
End Snap:    1130 10-9月 -14 14:00:04          53      2.2      2+
Elapsed:     59.74 (mins)+
DB Time:     0.18 (mins)+
+-----+-----+-----+-----+-----+-----+
Load Profile ...                          Per Second  Per Transaction  Per Exec  Per Call+
+-----+-----+-----+-----+-----+-----+
DB Time(s): 0.0                          0.0            0.00       0.00+
DB CPU(s):  0.0                          0.0            0.00       0.00+
Redo size (bytes): 1,021.6                2,038.9+
Logical read (blocks): 84.4                128.8+
Block changes: 4.2                        8.3+
Physical read (blocks): 0.1                0.2+
Physical write (blocks): 0.4                0.7+
Read IO requests: 0.1                    0.2+
Write IO requests: 0.3                    0.5+
Read IO (MB): 0.0                        0.0+
Write IO (MB): 0.0                        0.0+
RAC GC blocks received: 0.1                0.2+
RAC GC blocks served: 0.2                 0.4+
User calls: 1.8                          3.6+
Parse (SQL): 0.9                          1.7+
Hard parse (SQL): 0.0                     0.0+
SQL Work Area (MB): 0.1                   0.1+
Logons: 0.1                              0.1+
Executes (SQL): 2.7                       5.4+
Rollbacks: 0.0                           0.0+
Transactions: 0.5+
+-----+-----+-----+-----+-----+-----+
```

基本的にいずれの形式でも出力項目は同じ
(バージョンが同じ場合)

ただし、SQLテキスト全文は、**HTML形式のみ出力**される点に留意



AWR / ASH の概要

- AWRの概要

- AWRレポートの作成

- AWRレポートは下記の2つの方法で生成することができる
 - SQL*Plus で DBインスタンスに接続後、下記のスクリプトを実行

■ 対象インスタンスを選択しAWRレポートを生成する場合

```
SQL> @?/rdbms/admin/awrrpti.sql
```

■ 接続インスタンスのAWRレポートを生成する場合

```
SQL> @?/rdbms/admin/awrrpt.sql
```

*「?」は\$ORACLE_HOMEを示します

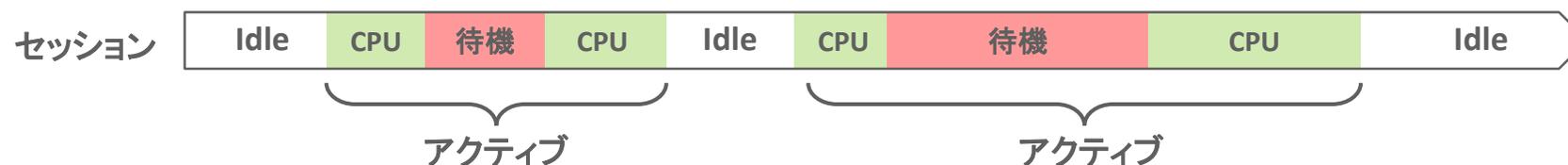
- Oracle Enterprise Manager 内の『パフォーマンス > AWR > AWRレポート』のセクションからの実行 (Cloud Control 12cの場合)

AWR / ASH の概要

• ASHの概要

– インスタンス内のセッションのうち、アクティブな状態のセッション情報の履歴を保持

▶ (G)V\$SESSIONで確認できるセッションのうち、CPU使用中または待機イベントクラスが Idle でない待機イベントで待機しているセッション



– ASH情報は下記の2つのビューで参照することができる

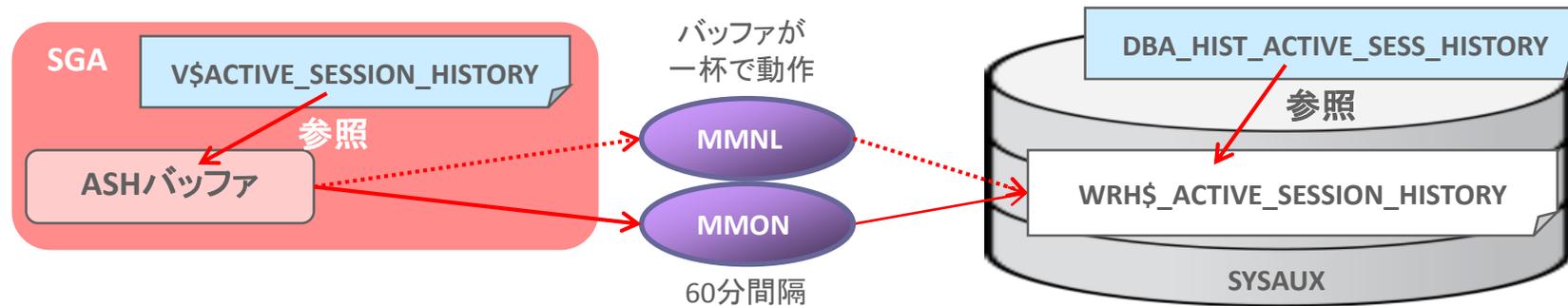
- (G)V\$ACTIVE_SESSION_HISTORY : SGA上のASHバッファに保持、**1秒間隔**
- DBA_HIST_ACTIVE_SESS_HISTORY : SYSAUX内のテーブルに保持、**10秒間隔**

* GV\$ACTIVE_SESSION_HISTORYを使用すると、RAC内の別のインスタンスのASH情報を参照することが可能です

AWR / ASH の概要

• ASHの概要

– 各ビューのASH情報は下記のように格納される



– 格納先毎の保存期間

- ASHバッファ上のASH情報の保存期間は下記に依存
 - 『アクティブ・セッション数』及び『ASHバッファのサイズ(共有プールサイズに依存)』
 - インスタンス停止(インスタンス停止でASHバッファ上の情報は消える)
- SYS_AUX内のASH情報の保存期間はデフォルト8日間(R11.2の場合)

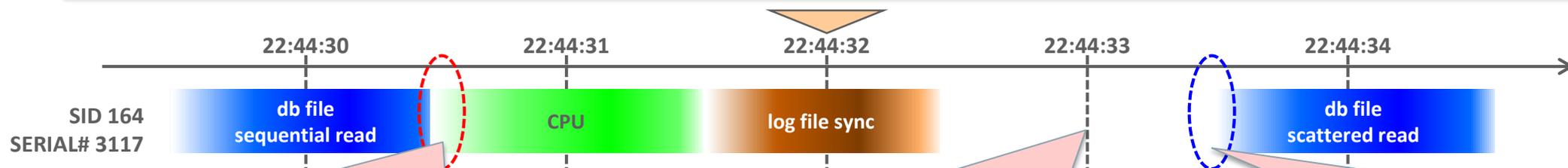
AWR / ASH の概要

• ASHの概要

– ASHに格納されている情報

- 対象セッション情報が記録された時間 : SAMPLE_TIME列で特定
- 対象セッションの情報 : SESSION_ID列及びSESSION_SERIAL#列で特定

SESSION_ID	SESSION_SERIAL#	SAMPLE_TIME	SQL_ID	EVENT	SESSION_STATE
164	3117	14-09-14 22:44:30.983	6m2ckkhmqctb	db file sequential read	
164	3117	14-09-14 22:44:31.871	6m2ckkhmqctb		ON CPU
164	3117	14-09-14 22:44:32.778		log file sync	
164	3117	14-09-14 22:44:34.678	13ffwur4e33gj	db file scattered read	



ASHは最小でも1秒間隔の情報のため、この間に別の待機が発生していた可能性はある

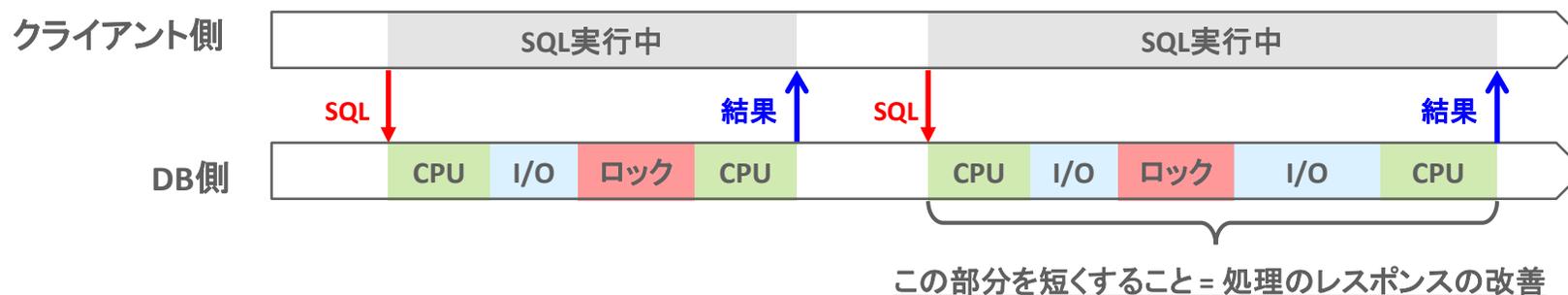
ASH情報が記録されていないため、セッションがアクティブな状態ではない

SQL_IDが変わっているため、別SQL実行時のセッション情報であると判断できる

AWR / ASH の概要

- AWR / ASHでの分析の範囲

- クライアント側、DB側で見た処理の流れ



- インスタンス全体で見たDB処理とAWR / ASHの分析範囲

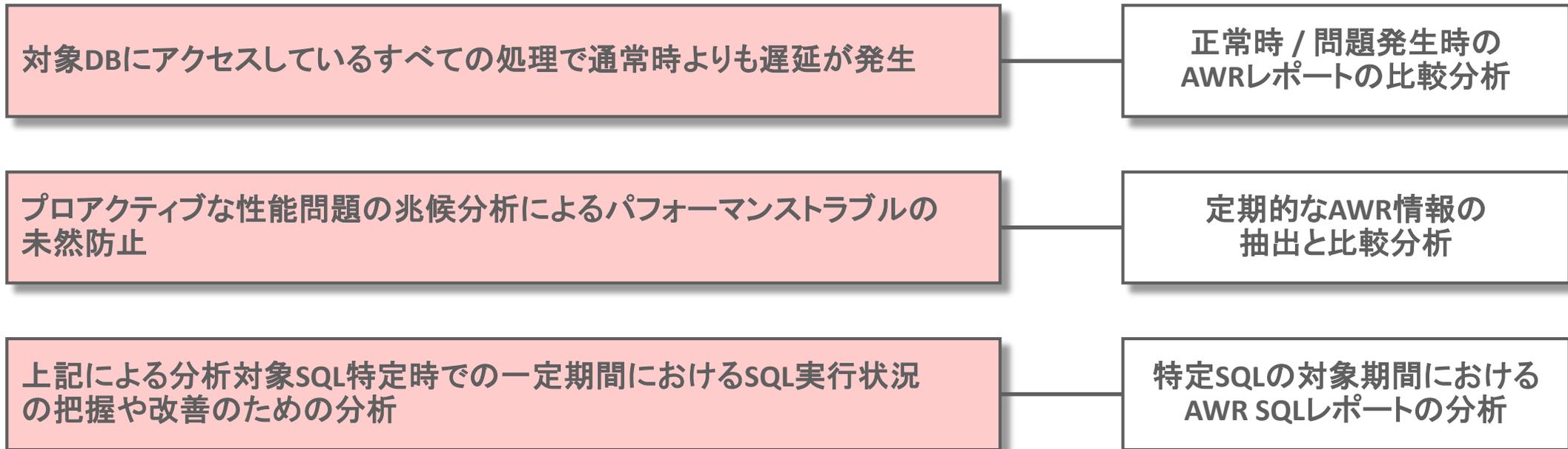


アジェンダ

- 1 一般的なチューニングの流れ
- 2 AWR / ASHの概要
- 3 AWRの活用方法**
- 4 ASHの活用方法
- 5 ケーススタディ

AWR活用方法

- 想定されるケースとAWR情報の活用方法



AWR活用方法

- 正常時 / 問題発生時のAWRレポートの比較分析の流れ

1. DBインスタンスの稼働状況を把握(負荷傾向の確認)



2. 待機イベントから問題発生時に発生しているボトルネックを特定(問題特定)



3. ボトルネックとなっている待機イベントの増加原因の分析と対策の策定(問題調査・対策検討)

AWR活用方法

- 正常時 / 問題発生時のAWRレポートの比較分析

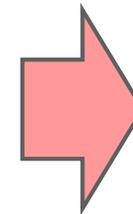
- DBインスタンスの稼働状況を把握(負荷傾向の確認)

正常時

Load Profile	Per Second	Per Transaction
DB Time(s) :	0.0	0.1
DB CPU(s) :	0.0	0.0
Redo size:	4,034,803.8	2,499.9
Logical reads:	215,587.1	164.6
Block changes:	6,833.0	9.5
Physical reads:	21.1	0.4
Physical writes:	899.5	0.4
User calls:	9,897.6	6.8
Parses:	26.3	6.8
Hard parses:	0.2	0.2
W/A MB processed:	75.2	0.6
Logons:	0.3	1.0
Executes:	3,011.2	1.7
Rollbacks:	805.0	0.0
Transactions:	2,035.3	

問題発生時

Load Profile	Per Second	Per Transaction
DB Time(s) :	24.5	0.0
DB CPU(s) :	6.8	0.0
Redo size:	16,605,841.7	2,578.4
Logical reads:	1,181,640.3	183.5
Block changes:	53,613.6	8.3
Physical reads:	109.4	0.0
Physical writes:	3,002.2	0.5
User calls:	28,759.9	4.5
Parses:	77.2	0.0
Hard parses:	0.3	0.0
W/A MB processed:	125.0	0.0
Logons:	0.3	0.0
Executes:	10,313.1	1.6
Rollbacks:	3,785.0	0.6
Transactions:	6,440.4	



正常時と比較して、インスタンスの処理の状況がどのように変化しているかを確認する

AWR活用方法

• 正常時 / 問題発生時のAWRレポートの比較分析

2. 待機イベントから問題発生時に発生しているボトルネックを特定(問題特定)

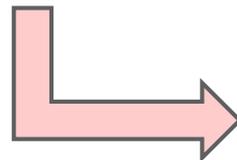
正常時

Top 10 Foreground Events by Total Wait Time			
Event	Waits	Total Time	Wait Avg (ms)
DB CPU		3483.6	90.5
log file sync	164,157	164.2	1
db file sequential read	78,151	156.3	2
gc cr block 2-way	25,156	25.2	1
gc current block 2-way	20,982	20.1	1
library cache pin	60	.5	0
control file sequential read	3,796	.5	1
SQL*Net more data to client	202,116	.4	0
IPC send completion sync	551	.2	0
enq: FB - contention	1,746	.0	0

正常時と比較して処理量の増加分等を考慮しても
待機時間が大幅に増加している待機イベント、
1待機あたりの平均待機時間が増加している待機イベントを特定

問題発生時

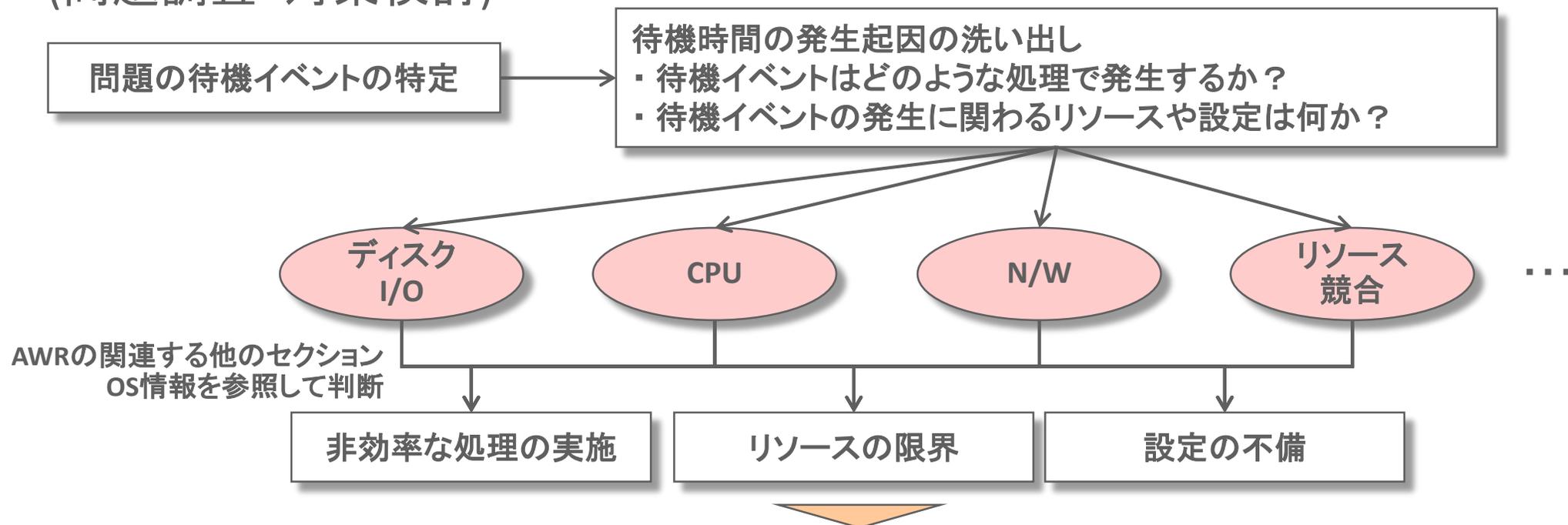
Top 10 Foreground Events by Total Wait Time				
Event	Waits	Total Time	Wait Avg (ms)	% DB time Wait Class
log file sync	100,157	2203.5	22	42.4 Commit
DB CPU		2067.6		39.8
db file sequential read	48,151	866.7	18	16.6 User I/O
gc cr block 2-way	25,156	25.2	1	.4 Cluster
gc current block 2-way	20,982	20.1	1	.4 Cluster
library cache pin	42	.4	0	.0 Concurrency
control file sequential read	1,717	.3	1	.0 System I/O
SQL*Net more data to client	104,116	.3	0	.0 Network
IPC send completion sync	451	.2	0	.0 Other
enq: FB - contention	1,625	.0	0	.0 Concurrency



AWR活用方法

• 正常時 / 問題発生時のAWRレポートの比較分析

3. ボトルネックとなっている待機イベントの増加原因の分析と対策の策定 (問題調査・対策検討)



関連するAWRの各セクションを参照し、問題となっている処理の抽出と原因に応じた対応を実施

AWR活用方法

- 定期的なAWR情報の抽出と比較分析の流れ

1. 一定期間の稼働情報のスナップショット間の差分情報取得とCSV形式で出力(情報採取)



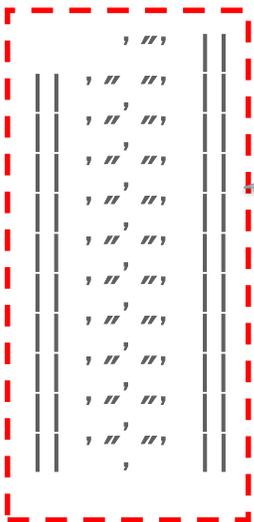
2. 取得したCSVデータのグラフ化と傾向分析(問題及び予兆のチェック)

AWR活用方法

- 定期的なAWR情報の抽出と比較分析

- 一定期間の稼働情報のスナップショット間の差分情報取得とCSV形式で出力

```
select
    es.DBID
  es.INSTANCE_NUMBER
  bs.SNAP_ID
  to_char (bs.END_INTERVAL_TIME, 'yyyy')
  to_char (bs.END_INTERVAL_TIME, 'mm')
  to_char (bs.END_INTERVAL_TIME, 'dd')
  to_char (bs.END_INTERVAL_TIME, 'hh24')
  to_char (bs.END_INTERVAL_TIME, 'mi')
  to_char (bs.END_INTERVAL_TIME, 'ss')
  es.SNAP_ID
...
from
  DBA_HIST_SNAPSHOT bs,
  DBA_HIST_SNAPSHOT es,
...
```



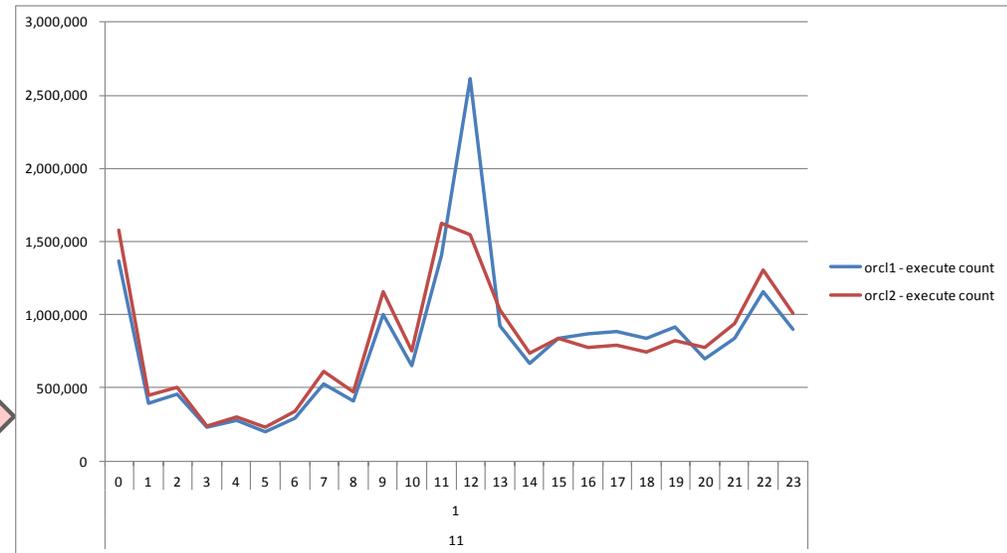
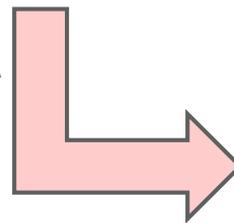
CSVで出力できるようにSQLを左記のような形で作成

AWR活用方法

- 定期的なAWR情報の抽出と比較分析
 - 取得したCSVデータのグラフ化と傾向分析

	A	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	INST_NUM	B_SNAP_ID	B_Y	B_MO	B_D	B_H	B_MI	B_S	E_SNAP_ID	E_Y	E_MO	E_D	E_H	E_MI	E_S	STAT_NAME	VALUE
2	1	1763	2014	11	1	0	0	1	1764	2014	11	1	0	30	1	execute count	746,871
3	1	1764	2014	11	1	0	30	1	1765	2014	11	1	1	0	0	execute count	618,106
4	1	1765	2014	11	1	1	0	0	1766	2014	11	1	1	30	1	execute count	251,483
5	1	1766	2014	11	1	1	30	1	1767	2014	11	1	2	0	0	execute count	129,718
6	1	1767	2014	11	1	2	0	0	1768	2014	11	1	2	30	1	execute count	268,604
7	1	1768	2014	11	1	2	30	1	1769	2014	11	1	3	0	0	execute count	185,290
8	1	1769	2014	11	1	3	0	0	1770	2014	11	1	3	30	0	execute count	136,889
9	1	1770	2014	11	1	3	30	0	1781	2014	11	1	4	0	0	execute count	94,845
10	1	1791	2014	11	1	4	0	0	1792	2014	11	1	4	30	1	execute count	174,060
11	1	1792	2014	11	1	4	30	1	1793	2014	11	1	5	0	0	execute count	105,038
12	1	1793	2014	11	1	5	0	6	1794	2014	11	1	5	30	1	execute count	96,990
13	1	1794	2014	11	1	5	30	1	1795	2014	11	1	6	0	0	execute count	103,663
14	1	1795	2014	11	1	6	0	0	1796	2014	11	1	6	30	0	execute count	178,438
15	1	1796	2014	11	1	6	30	0	1797	2014	11	1	7	0	0	execute count	117,797
16	1	1797	2014	11	1	7	0	0	1798	2014	11	1	7	30	0	execute count	210,367
17	1	1798	2014	11	1	7	30	0	1799	2014	11	1	8	0	0	execute count	318,798
18	1	1799	2014	11	1	8	0	0	1800	2014	11	1	8	30	0	execute count	192,544
19	1	1800	2014	11	1	8	30	0	1821	2014	11	1	9	0	1	execute count	215,224
20	1	1821	2014	11	1	9	0	1	1822	2014	11	1	9	30	0	execute count	598,950
21	1	1822	2014	11	1	9	30	0	1823	2014	11	1	10	0	1	execute count	404,876
22	1	1823	2014	11	1	10	0	1	1824	2014	11	1	10	30	1	execute count	297,427
23	1	1824	2014	11	1	10	30	1	1825	2014	11	1	11	0	0	execute count	352,987
24	1	1825	2014	11	1	11	0	0	1826	2014	11	1	11	30	0	execute count	738,443
25	1	1826	2014	11	1	11	30	0	1827	2014	11	1	12	0	1	execute count	675,522
26	1	1827	2014	11	1	12	0	1	1828	2014	11	1	12	30	1	execute count	1,453,795
27	1	1828	2014	11	1	12	30	1	1829	2014	11	1	13	0	1	execute count	1,159,773
28	1	1829	2014	11	1	13	0	1	1830	2014	11	1	13	30	0	execute count	483,906
29	1	1830	2014	11	1	13	30	0	1851	2014	11	1	14	0	1	execute count	438,055
30	1	1851	2014	11	1	14	0	1	1852	2014	11	1	14	30	0	execute count	318,230
31	1	1852	2014	11	1	14	30	0	1853	2014	11	1	15	0	0	execute count	349,569
32	1	1853	2014	11	1	15	0	0	1854	2014	11	1	15	30	0	execute count	374,766
33	1	1854	2014	11	1	15	30	0	1855	2014	11	1	16	0	0	execute count	465,795
34	1	1855	2014	11	1	16	0	0	1856	2014	11	1	16	30	1	execute count	443,050
35	1	1856	2014	11	1	16	30	1	1857	2014	11	1	17	0	0	execute count	426,851
36	1	1857	2014	11	1	17	0	0	1858	2014	11	1	17	30	0	execute count	384,533

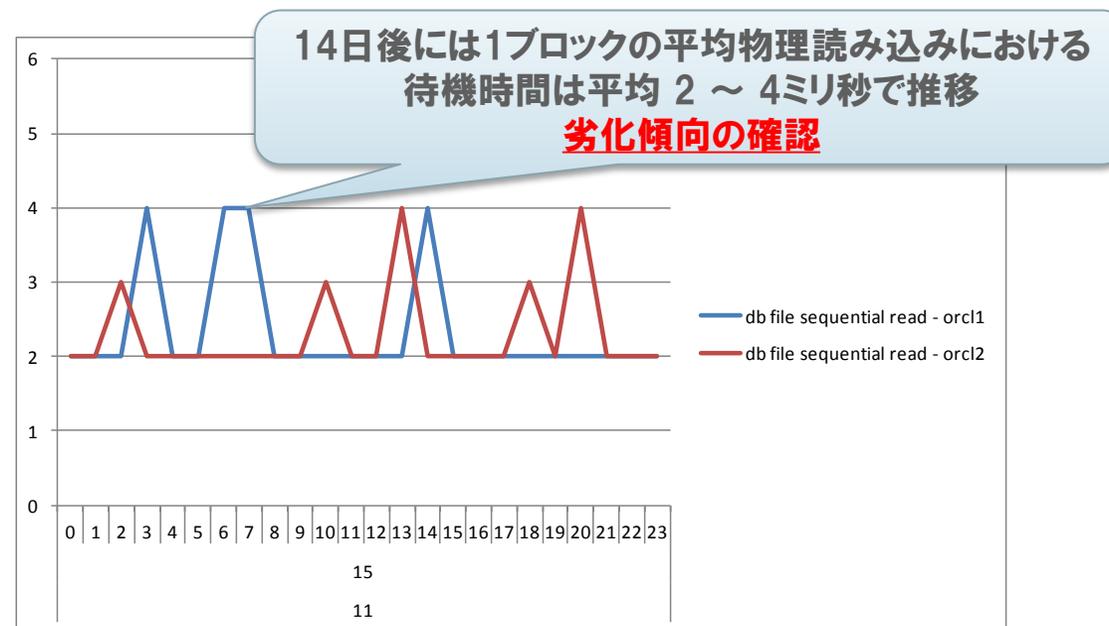
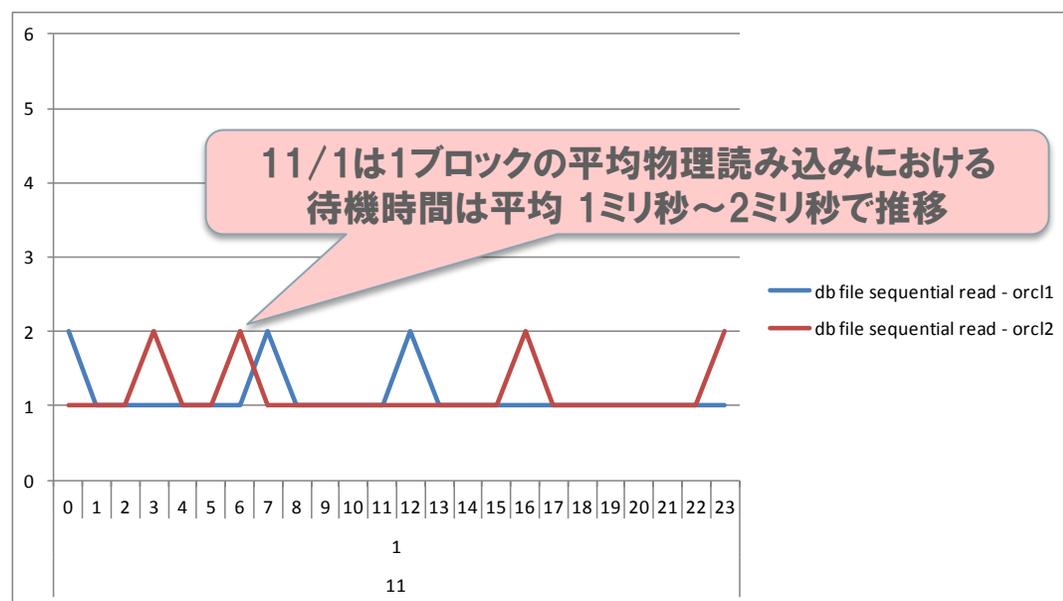
スプレッドシート機能を使用してグラフ化



AWR活用方法

- 定期的なAWR情報の抽出と比較分析
- 2. 取得したCSVデータのグラフ化と傾向分析

例) 待機イベント db file sequential read の1待機あたりの待機時間の推移

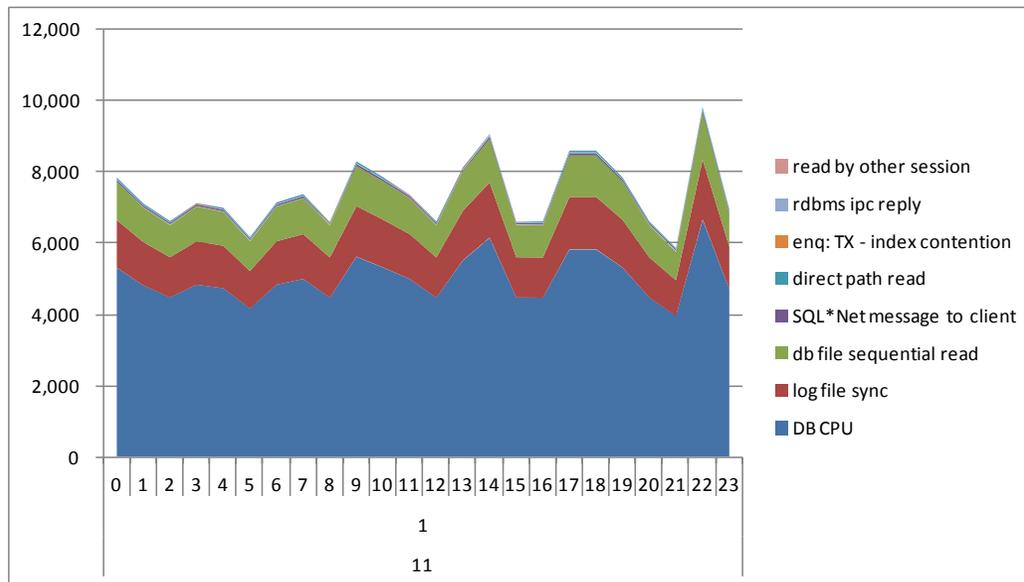


AWR活用方法

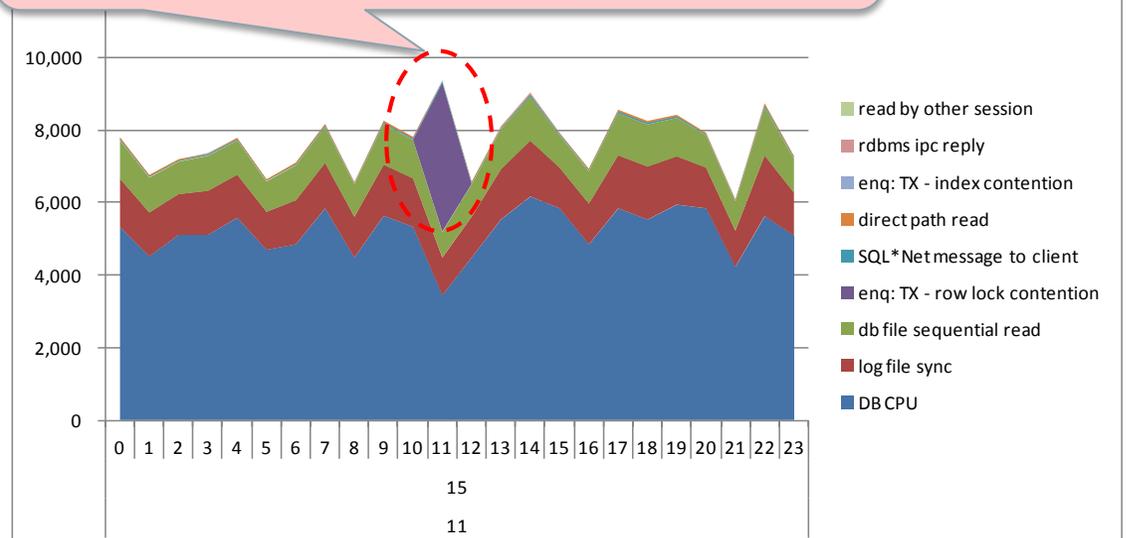
• 定期的なAWR情報の抽出と比較分析

2. 取得したCSVデータのグラフ化と傾向分析(問題及び予兆のチェック)

例) フォアグラウンドプロセスにて待機時間の多い上位待機イベントの待機時間の推移



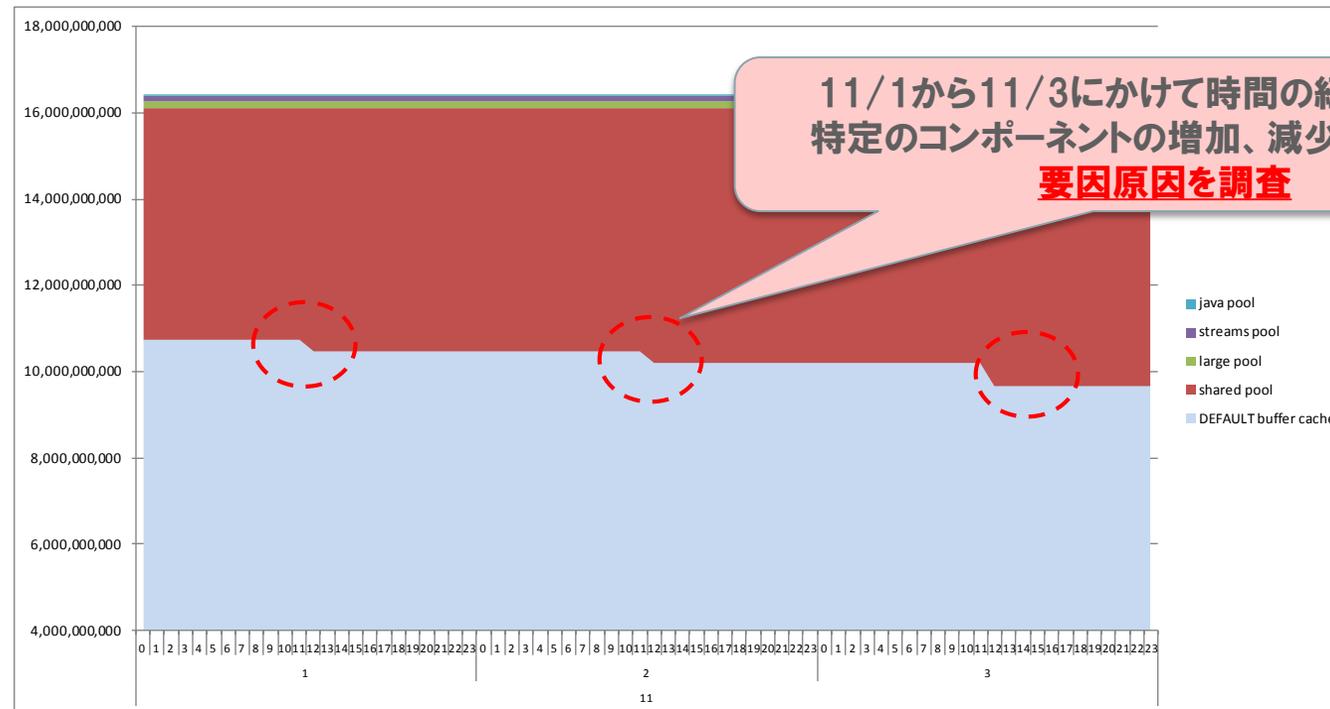
11/1には発生していなかった待機イベントが
待機時間の多い上位待機イベントとして発生
発生原因を調査



AWR活用方法

- 定期的なAWR情報の抽出と比較分析
 - 2. 取得したCSVデータのグラフ化と傾向分析(問題及び予兆のチェック)

例) SGA内の各コンポーネントサイズの推移



AWR活用方法

- 定期的なAWR情報の抽出と比較分析
 - インスタンス負荷傾向把握のための取得項目の例と使用するDBA_HISTビュー

	分析項目	DBA_HISTビュー	使用する列
差分	SQL実行回数	DBA_HIST_SYSSTAT (STAT_NAME = execute count)	①STAT_NAME ②VALUE
差分	REDO生成量	DBA_HIST_SYSSTAT (STAT_NAME = redo size)	
差分	論理読み込みブロック数	DBA_HIST_SYSSTAT (STAT_NAME in db block gets, consistent gets)	
差分	物理読み込みブロック数	DBA_HIST_SYSSTAT (STAT_NAME = physical reads)	

AWR活用方法

- 定期的なAWR情報の抽出と比較分析
 - インスタンスボトルネック傾向把握のための取得項目の例と使用するDBA_HISTビュー

分析項目	DBA_HISTビュー	使用する列
待機イベント・CPU時間	DBA_HIST_SYSTEM_EVENT	①EVENT ②TIME_WAITED_MICRO_FG
	DBA_HIST_SYS_TIME_MODEL (STAT_NAME = DB CPU)	①STAT_NAME ②VALUE

差分

AWR活用方法

- 定期的なAWR情報の抽出と比較分析

- インスタンスの性能傾向把握のための取得項目の例と使用するDBA_HISTビュー

	分析項目	DBA_HISTビュー	使用する列
差分	読み込みI/O性能	DBA_HIST_SYSTEM_EVENT (EVENT_NAME = db file sequential read)	①EVENT_NAME ②TIME_WAITED_MICRO_FG ③TOTAL_WAITS_FG
差分	書き込みI/O性能	DBA_HIST_SYSTEM_EVENT (EVENT_NAME = log file parallel write)	
差分	コミット性能	DBA_HIST_SYSTEM_EVENT (EVENT_NAME = log file sync)	1待機あたりの平均待機時間 = ② / ③

AWR活用方法

- 定期的なAWR情報の抽出と比較分析
 - インスタンスの性能傾向把握のための取得項目の例と使用するDBA_HISTビュー

分析項目	DBA_HISTビュー	使用する列
差分 キャッシュフュージョン性能 (Current ブロック)	DBA_HIST_SYSSTAT (STAT_NAME in (gc current blocks received gc current block receive time))	①STAT_NAME ②VALUE(~ receive time) ③VALUE(~ received)
差分 キャッシュフュージョン性能 (CR ブロック)	DBA_HIST_SYSSTAT (STAT_NAME in (gc cr blocks received gc cr block receive time))	1転送あたりの平均時間 = ② / ③

AWR活用方法

- 定期的なAWR情報の抽出と比較分析

- インスタンスのメモリ傾向把握のための取得項目の例と使用するDBA_HISTビュー

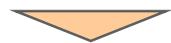
分析項目	DBA_HISTビュー	使用する列
SGAサイズ	DBA_HIST_MEM_DYNAMIC_COMP (COMPONENT in (DEFAULT buffer cache, java pool, large pool, shared pool, streams pool))	①COMPONENT ②CURRENT_SIZE
PGAサイズ (セッションあたりの確保可能サイズ)	DBA_HIST_PGASTAT (NAME = global memory bound)	①NAME ②VALUE
PGAサイズ (スナップ取得時の確保サイズ)	DBA_HIST_PGASTAT (NAME = total PGA allocated)	

* SGAサイズの確認では、32Kバッファ・キャッシュ等のコンポーネントを使用している場合は、適宜条件を変更してください

AWR活用方法

- 特定SQLの対象期間におけるAWR SQLレポートの分析の流れ

1. SQL実行統計の分析(SQLの改善点の確認)



2. SQL実行計画の分析(改善のための実行計画の検討)

– 分析対象SQLのSQL_IDを指定し、AWR情報からSQLレポートを生成

■ 対象インスタンスを選択しAWRレポートを生成する場合
SQL> @?/rdbms/admin/**awrsqrpi.sql**

■ 対象インスタンスを選択しAWRレポートを生成する場合
SQL> @?/rdbms/admin/**awrsqrpt.sql**

*「?」は\$ORACLE_HOMEを示します

AWR活用方法

- 特定SQLの対象期間におけるAWR SQLレポートの分析

1. SQL実行統計の分析

- AWR SQLレポート内のSQL実行統計から改善できるポイントを確認

Stat Name	Statement	Per Execution	% Snap
Elapsed Time (ms)	2,183,155	42.4	12.0
CPU Time (ms)	1,025,266	19.9	11.1
Executions	51,526	N/A	N/A
Buffer Gets	9,971,515	193.5	13.2
Disk Reads	8,781,526	170.4	53.5
Parse Calls	12	0.5	0.0
Rows	51,526	1	N/A
User I/O Wait Time (ms)	1,154,366	22.4	N/A
Cluster Wait Time (ms)	2,382	0.0	N/A
Application Wait Time (ms)	1,141	0.0	N/A
Concurrency Wait Time (ms)	0	N/A	N/A
Invalidations	0	N/A	N/A
Version Count	1	N/A	N/A
Sharable Mem(KB)	3,167	N/A	N/A

どの部分に時間を要しているか等
を確認する

バッファ読み込み数や物理ブロック読み込み数
の状況を確認する

AWR活用方法

- 特定SQLの対象期間におけるAWR SQLレポートの分析
 2. SQL実行計画の分析(改善のための実行計画の検討)

Execution Plan

Id	Operation	Name	Rows	Bytes	Cost	(%CPU)	Time
0	SELECT STATEMENT				2	(0)	
1	NESTED LOOPS		1	12	2	(0)	00:00:01
2	TABLE ACCESS BY INDEX ROWID	EMP	1		1	(0)	00:00:01
3	NESTED LOOPS		1	1	1	(0)	00:00:01
4	TABLE ACCESS BY INDEX ROWID	IDX1_DEPT	1	1	1	(0)	00:00:01
5	INDEX RANGE SCAN	DEPT	1	1	1	(0)	00:00:01
6	INDEX RANGE SCAN	IDX1_EMP	1	1	1	(0)	00:00:01

実行計画から実行統計で確認したポイント
の改善余地の有無を確認する
改善の余地があった場合は、SQLチューニングを実施

アジェンダ

- 1 一般的なチューニングの流れ
- 2 AWR / ASHの概要
- 3 AWRの活用方法
- 4 ASHの活用方法**
- 5 ケーススタディ

ASH活用方法

・パフォーマンス分析ケースごとのASH活用方法

分析対象となるセッションが特定できていないため、
確認できている入力情報からセッションを特定するケース

対象セッション
特定するためのSQL

分析対象となるセッションが特定できており、
該当セッションの状態を詳細に分析するケース

セッション単位の分析
を実施するためのSQL

分析対象SQLが特定できており、
該当SQLの実行セッションの状態を分析するケース

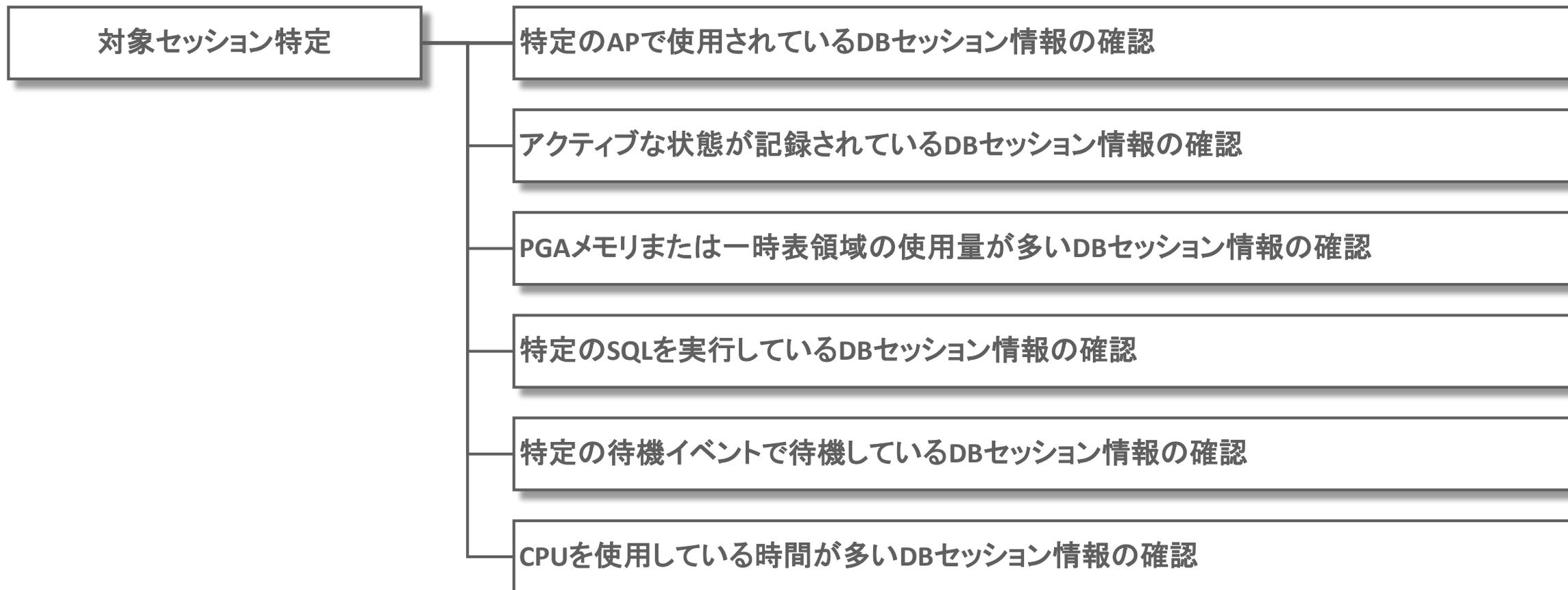
SQL単位の分析
を実施するためのSQL

分析対象待機イベントが特定できており、該当待機イベントでの
待機が発生しているセッションの状態を分析するケース

待機イベント単位の分析
を実施するためのSQL

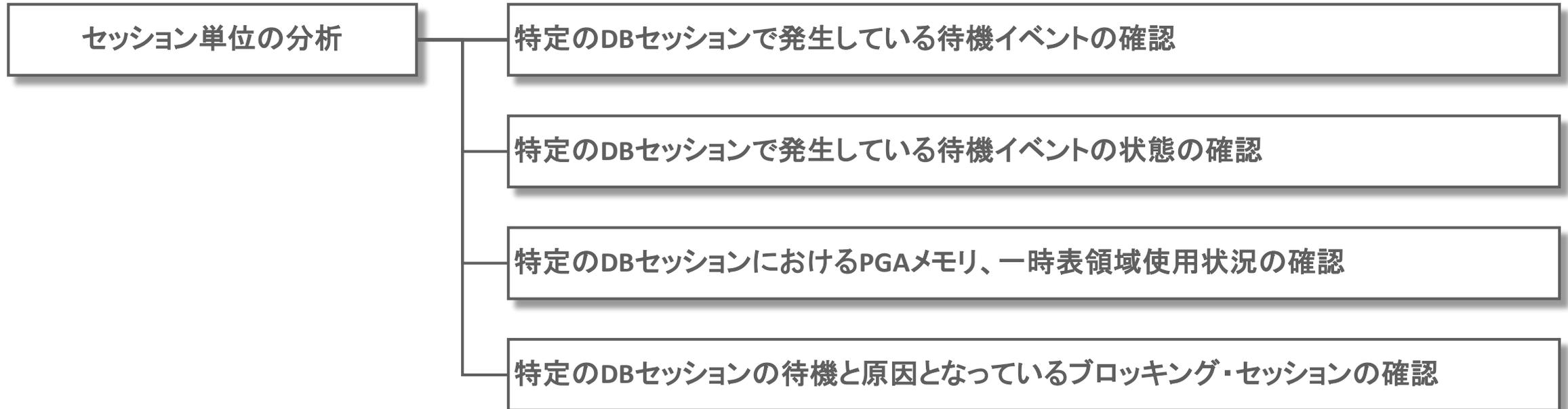
ASH活用方法

・パフォーマンス分析ケースごとのASH活用方法



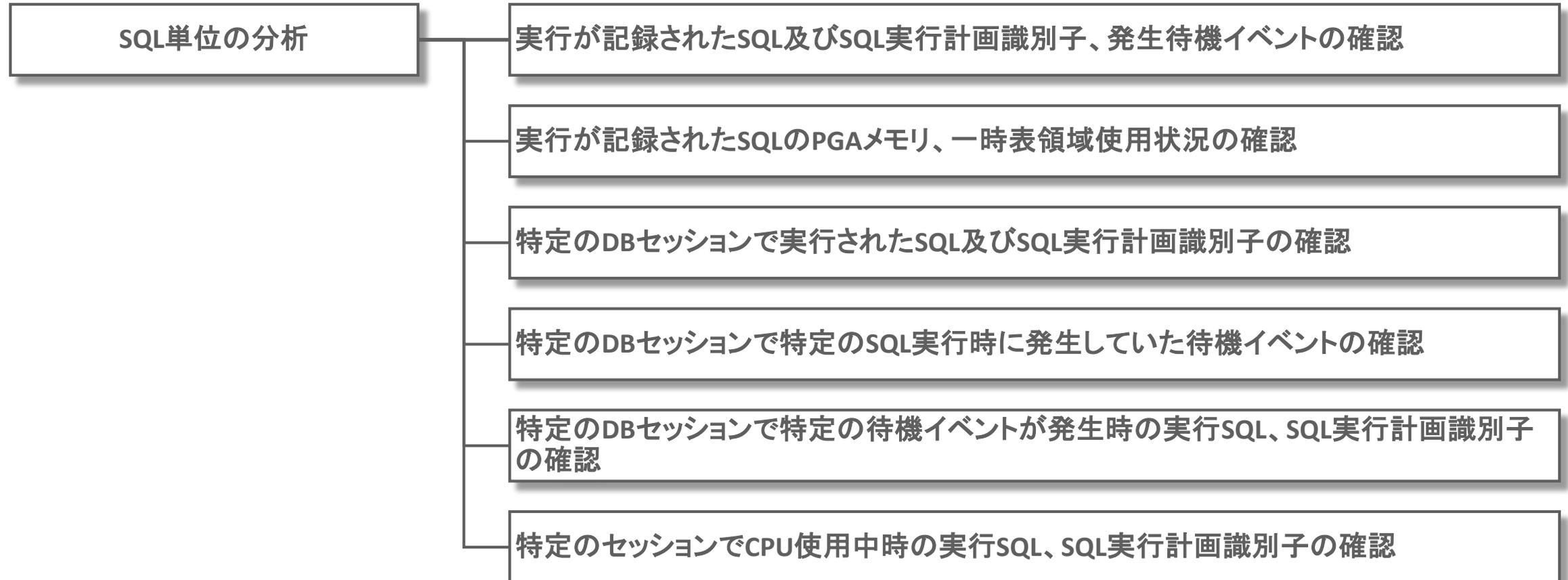
ASH活用方法

・パフォーマンス分析ケースごとのASH活用方法



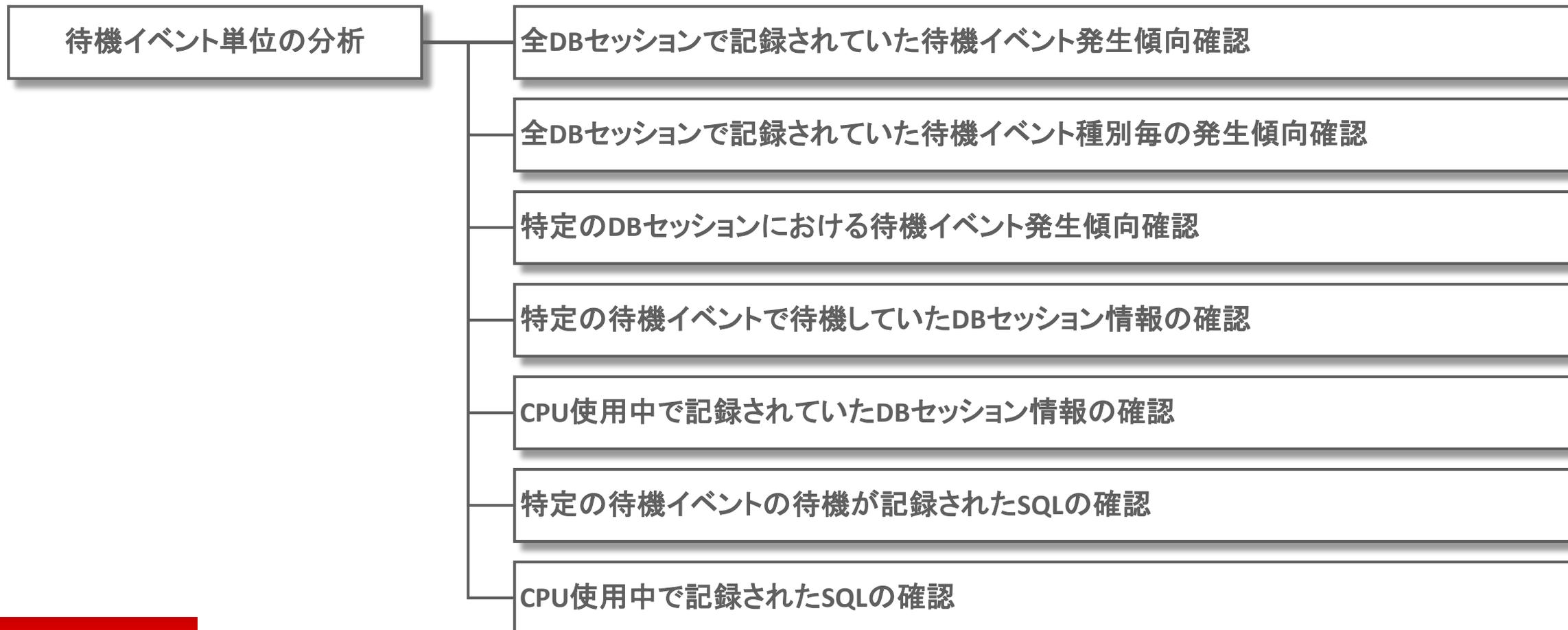
ASH活用方法

・パフォーマンス分析ケースごとのASH活用方法



ASH活用方法

・パフォーマンス分析ケースごとのASH活用方法



アジェンダ

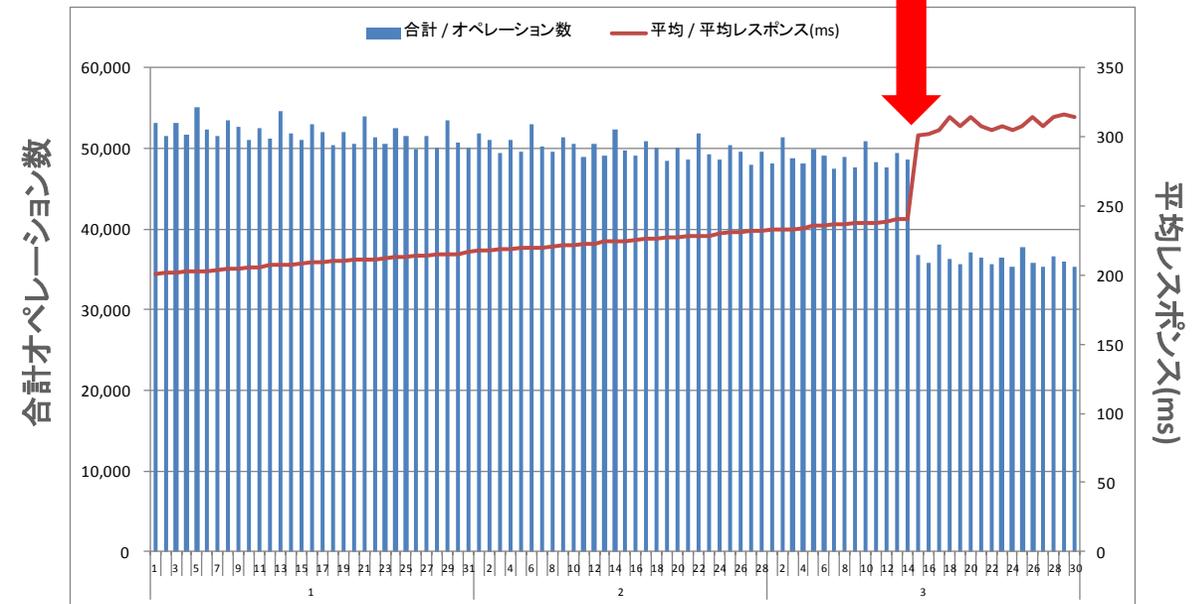
- 1 一般的なチューニングの流れ
- 2 AWR / ASHの概要
- 3 AWRの活用方法
- 4 ASHの活用方法
- 5 ケース・スタディ

【ケース1】定常的に実行される処理が急激に遅延

問題概要

- サービスインから一定期間経過後、突然ある日を境に特定の処理の性能劣化が発生
- 該当の処理は定常的に実施されている処理
- 対象の処理についてはユーザからのヒアリングの結果、徐々に遅くなっていた模様

大幅なレスポンス劣化
&
スループットダウン



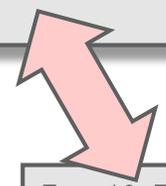
アプリケーション・ログから見たオペレーション数と平均レスポンス時間(ms)

【ケース1】定常的に実行される処理が急激に遅延 事象発生前後のインスタンス状況の分析

– AWRLレポートの上位待機イベントの比較を実施

正常時

Top 10 Foreground Events by Total Wait Time					
Event	Waits	Total Time	Wait Avg (ms)	% DB time	Wait Class
DB CPU		3483.6		90.5	
log file sync	164,157	164.2	1	4.3	Commit
db file sequential read	78,151	156.3	2	4.1	User I/O
...					



問題発生時

Top 10 Foreground Events by Total Wait Time					
Event	Waits	Total Time	Wait Avg (ms)	% DB time	Wait Class
DB CPU		2067.6		22.9	
direct path write temp	211,415	1268.8	6	13.8	USER I/O
direct path read temp	214,222	1071.1	5	11.6	USER I/O
...					

問題発生時の上位待機イベントから
一時表領域への読み込み、書き込みの待機が
大幅に増加していることを確認

ただし、この段階では対象の処理のSQLが原因かは断定できない



【ケース1】定常的に実行される処理が急激に遅延

特定した待機イベントの要因SQLの特定

– ASHから特定した待機イベント発生時の実行SQLを検索

```
select
  INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, EVENT, SESSION_STATE, count(1) "SESSION_COUNT"
from
  GV$ACTIVE_SESSION_HISTORY
where
  SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')
                 and to_date('<対象期間終了日時>','yyyymmddhh24miss')
and EVENT in ('direct path write temp', 'direct path read temp')
group by
  INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, EVENT, SESSION_STATE
order by 1,2,3,6;
```

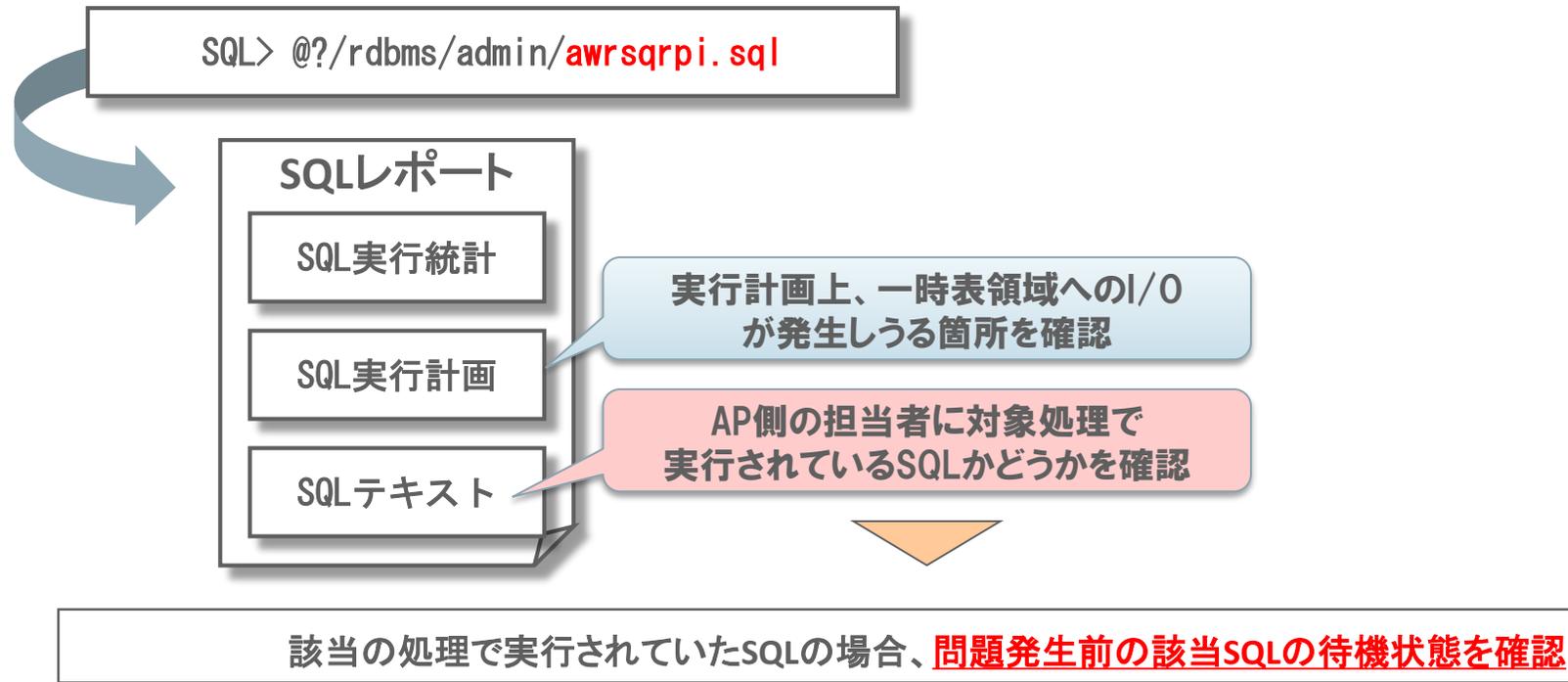
特定のSQLにて対象の
待機イベントが多数発生

INST_ID	SQL_ID	PLAN_HASH_VALUE	EVENT	SESSION_STATE	SESSION_COUNT
1	1fqg2t21f2gf4	3051237957	direct path write temp	WAITING	15116
1	1fqg2t21f2gf4	3051237957	direct path read temp	WAITING	14156
1	1gwhj4132ujr5	1021473	direct path write temp	WAITING	12
1	1nr3u4535gv3fw	241672552	direct path read temp	WAITING	6

【ケース1】定常的に実行される処理が急激に遅延

特定したSQLの実行計画 / 実行アプリケーション確認

– AWR情報から該当SQLのSQLレポートを抽出



【ケース1】定常的に実行される処理が急激に遅延

問題発生前に対象SQL実行時に発生していた待機イベントを確認

– ASH情報から問題発生前の該当SQLの待機イベントの状態を確認

```
select
  INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, EVENT, SESSION_STATE, count(1) "SESSION_COUNT"
from
  GV$ACTIVE_SESSION_HISTORY
where
  SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')
                 and to_date('<対象期間終了日時>','yyyymmddhh24miss')
and SQL_ID = '1fqg2t21f2gf4'
group by
  INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, EVENT, SESSION_STATE
order by 1,2,3,6;
```

大半がCPU使用であり、問題発生時に
確認された待機イベントの発生は見られない

INST_ID	SQL_ID	PLAN_HASH_VALUE	EVENT	SESSION_STATE	SESSION_COUNT
1	1fqg2t21f2gf4	3051237957		ON CPU	41116
1	1fqg2t21f2gf4	3051237957	db file sequential read	WAITING	50
1	1fqg2t21f2gf4	3051237957	SQL*Net message to client	WAITING	33

...

【ケース1】定常的に実行される処理が急激に遅延

該当SQLの問題発生前後の実行計画の状況の確認

– AWR情報から対象SQLの実行計画の変動有無を確認

```
select
  distinct INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE
from
  GV$ACTIVE_SESSION_HISTORY
where
  SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')
                 and to_date('<対象期間終了日時>','yyyymmddhh24miss')
and SQL_ID = '1fqg2t21f2f4';
```

正常時

INST_ID	SQL_ID	PLAN_HASH_VALUE
1	1fqg2t21f2gf4	3051237957

問題発生時

INST_ID	SQL_ID	PLAN_HASH_VALUE
1	1fqg2t21f2gf4	3051237957

SQL実行計画は問題発生前後
で変化なし

【ケース1】定常的に実行される処理が急激に遅延

該当SQL実行時のPGAメモリ及び一時表領域の状態の確認

- ASH情報から対象SQL実行時のPGAメモリ、一時表領域の確保サイズを確認

```
select
  INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, max(PGA_ALLOCATED) "MAX_PGA_ALLOCATED",
  max(TEMP_SPACE_ALLOCATED) "MAX_TEMP_ALLOCATED"
from
  GV$ACTIVE_SESSION_HISTORY
where
  SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')
                  and to_date('<対象期間終了日時>','yyyymmddhh24miss')
and SQL_ID = '1fqg2t21f2f4'
group by INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE,
order by 1,4,5,2,3;
```

正常時

INST_ID	SQL_ID	MAX_PGA_ALLOCATED	MAX_TEMP_ALLOCATED
1	1fqg2t21f2gf4	9045376	0

確保しているPGAサイズが若干増加し、
使用している一時表領域が発生

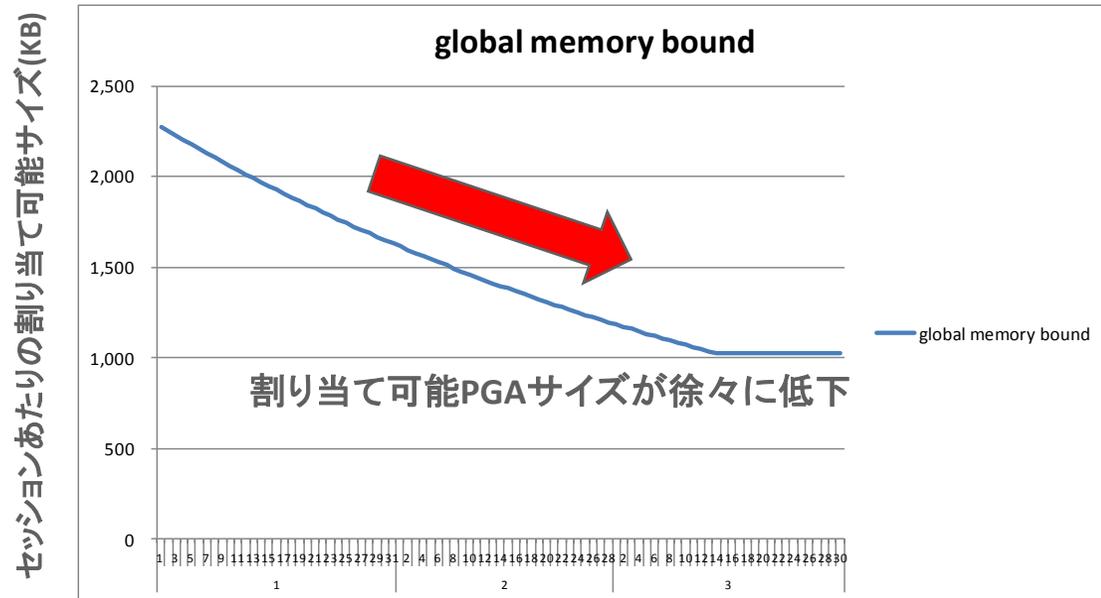
問題発生時

INST_ID	SQL_ID	MAX_PGA_ALLOCATED	MAX_TEMP_ALLOCATED
1	1fqg2t21f2gf4	9437184	4088768

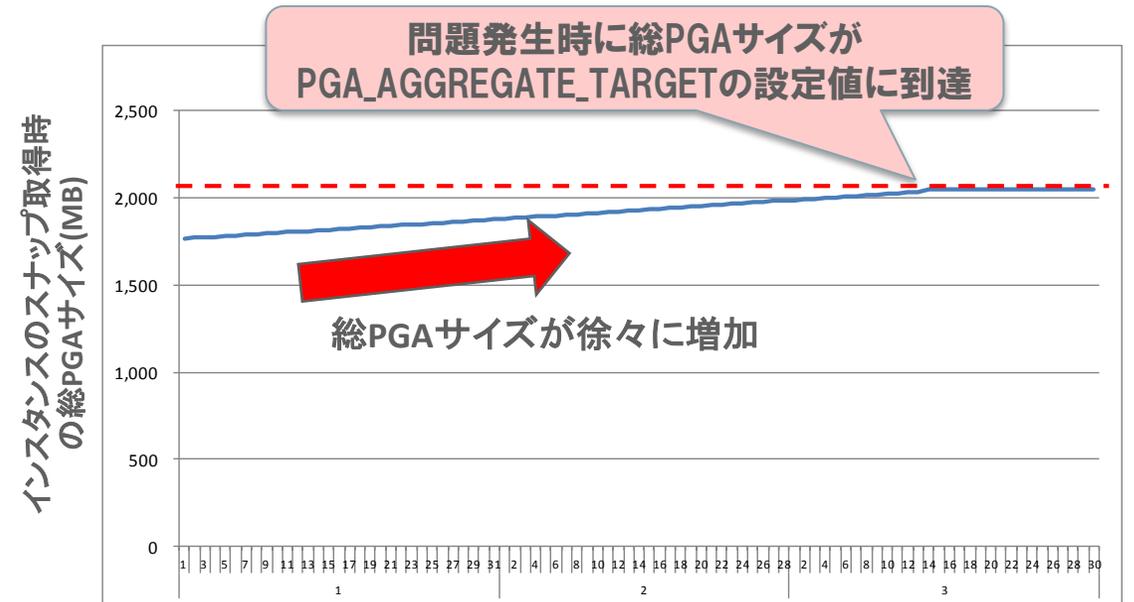
【ケース1】 定常的に実行される処理が急激に遅延

インスタンス及びセッションのPGAメモリの状況を確認

- AWR情報からPGAメモリの割り当て可能サイズの推移を確認



- AWR情報からPGAメモリの割り当てサイズの推移を確認



【ケース1】定常的に実行される処理が急激に遅延

原因と対処

- 対象の処理ではソートを実施しているが、対象の処理は3カ月の間はデータが蓄積し、1処理あたりの処理件数が増加する(= 必要なPGAサイズが増加)
- 徐々に1セッションあたりの確保PGAサイズが増加していった結果、インスタンスの総PGAサイズがPGA_AGGREGATE_TARGETに到達
- 上記の結果、処理で必要なPGA領域はさらに必要となるが、割り当てが行われず、一時表領域への読み込み、書き出しが行われ、性能が劣化
- 3ヶ月間のデータ増加を考慮したPGA_AGGREGATE_TARGETのサイズ設定により、1セッションあたりのPGAサイズを必要な分確保させることで対処

【ケース1】定常的に実行される処理が急激に遅延

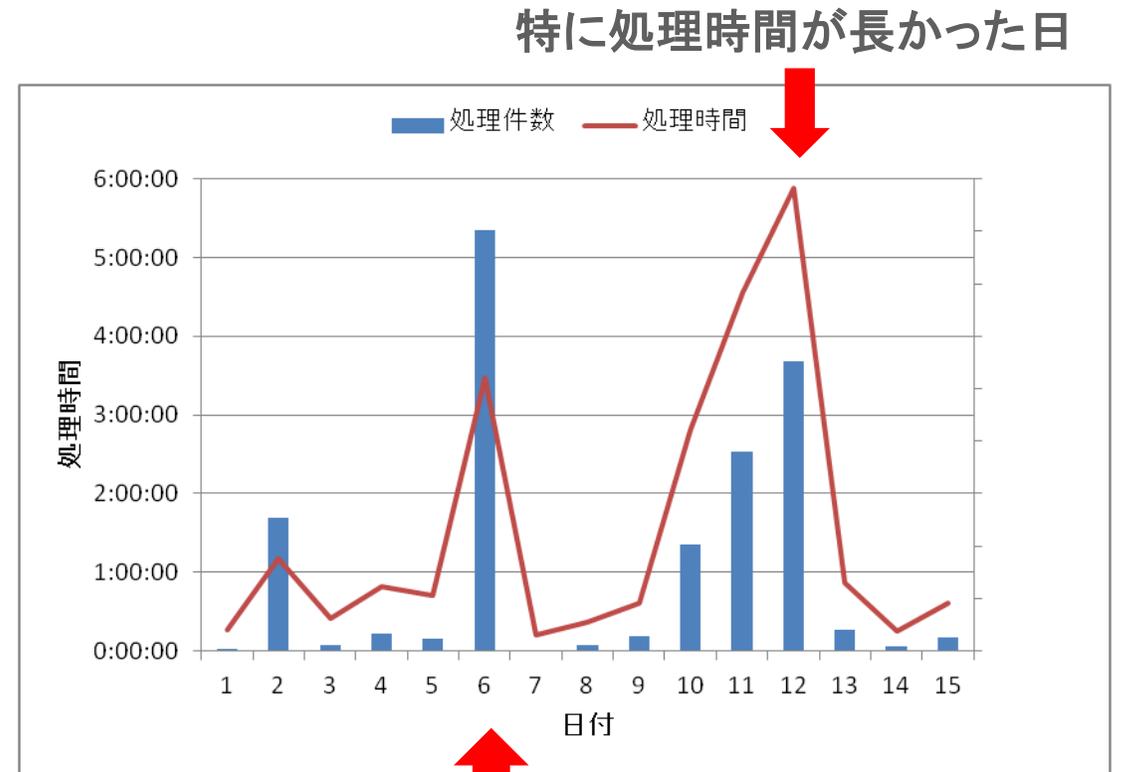
まとめ

項目	内容
問題概要	サービスインから2ヶ月半経過後、処理のレスポンスが劣化。 対象の処理は定常的に実施されており、徐々にレスポンスが劣化する傾向にあった。
AWR/ASHの活用	[AWR] Top 10 Foreground Events by Total Wait Timeで待機イベントの状況比較 [ASH] 発生していた待機イベントが記録されていたSQLの特定 [ASH] 特定したSQLの問題発生前の記録されていた待機イベントの確認 [ASH] 特定したSQLの実行計画の変動を有無を確認 [ASH] 特定したSQL実行時に確保されるPGA及び一時表領域サイズの確認 [AWR] DBA_HIST_PGASTATから1セッションでの確保可能PGAサイズの確認 [AWR] DBA_HIST_PGASTATからスナップ取得時の確保総PGAサイズの確認
原因と対処	PGA_AGGREGATE_TARGETが適切な設定ではなく、一時表領域への書き出しが発生。 PGA_AGGREGATE_TARGET のサイズを適正サイズに設定し、対処。

【ケース2】バッチ処理の長時間走行

問題概要

- 長くても3時間程度で終わっていたバッチ処理が5時間以上かかった
- バッチ処理内で実行されているSQLは“INSERT INTO 表 SELECT * FROM 表”によるデータのローディング
- 処理件数が増えると処理時間も増える傾向はあるものの、処理件数が特に多かった6日より、12日の方が処理時間が長い



処理件数が特に多かった日

【ケース2】バッチ処理の長時間走行

正常時と問題発生時のAWRLレポートを比較

- 両日ともにdb file sequential read が待機の上位に出ている
- 負荷傾向に大きな差異はない。
(バッチ処理が長かった日が特に負荷が高いということはない)

正常時

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Time	Wait Avg(ms)	% DB time	Wait Class
db file sequential read	727,180	5,551	8	53.1	User I/O
DB CPU	1,673	1656		15.9	
db file scattered read	610,294	1147	2	10.9	User I/O
log file sync	38,906	504	13	4.8	Commit
direct path read	728,362	422	1	4.0	User I/O ...

問題発生時

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Time	Wait Avg(ms)	% DB time	Wait Class
db file sequential read	742,021	5,608	8	53.7	User I/O
DB CPU	1,673	1656		16.0	
db file scattered read	622,749	1,159	2	11.0	User I/O
log file sync	39,701	510	13	4.8	Commit
direct path read	743,227	427	1	4.1	User I/O
...					

【ケース2】バッチ処理の長時間走行

処理中のAWRLレポートの待機イベントを確認

問題発生時(処理開始直後の30分)

Top 10 Foreground Events by Total Wait Time					
Event	Waits	Total Time	Wait Avg(ms)	% DB time	Wait Class
db file sequential read	742,021	5,608	8	53.7	User I/O
DB CPU		1,673		16.0	
db file scattered read	622,749	1,159	2	11.0	User I/O
log file sync	39,701	510	13	4.8	Commit
direct path read	743,227	427	1	4.1	User I/O
...					

問題発生時(処理終了前の30分)

Top 10 Foreground Events by Total Wait Time					
Event	Waits	Total Time	Wait Avg(ms)	% DB time	Wait Class
db file sequential read	802,667	5,600	7	50.4	User I/O
db file scattered read	521,189	2,111	4	19.0	User I/O
DB CPU		1,251		11.3	
log file sync	186,940	1,134	6	10.2	Commit
log file parallel write	185,912	714	4	6.4	User I/O
...					

バッチ処理時間が長かった日はI/O系の待機イベントが延々続いている

【ケース2】バッチ処理の長時間走行

特定のDBセッションにおける待機イベント発生傾向確認

- ASHからバッチ処理実行セッションに絞って待機イベントを集計すると db file sequential read が多い

```
select INSTANCE_NUMBER, SESSION_ID, SESSION_SERIAL#, nvl(EVENT, SESSION_STATE) "EVENT", count(1) "SESSION_COUNT"  
from   DBA_HIST_ACTIVE_SESS_HISTORY  
where  SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')  
        and to_date('<対象期間終了日時>','yyyymmddhh24miss')  
and    INSTANCE_NUMBER = <INSTANCE_NUMBER> and SESSION_ID = <SESSION_ID> and SESSION_SERIAL# = <SESSION_SERIAL#>  
group by INSTANCE_NUMBER, SESSION_ID, SESSION_SERIAL#, nvl(EVENT, SESSION_STATE)  
order by 1,5,4,2,3;
```

INSTANCE_NUMBER	SESSION_ID	SESSION_SERIAL#	EVENT	SESSION_COUNT
1	1501	14317	log file switch completion	1
1	1501	14317	direct path read	3
1	1501	14317	direct path read temp	3
1	1501	14317	direct path write temp	4
1	1501	14317	db file parallel read	6
1	1501	14317	log buffer space	11
1	1501	14317	db file scattered read	145
1	1501	14317	db file sequential read	3965

【ケース2】バッチ処理の長時間走行

特定のDBセッションにおける待機イベントの詳細情報の確認

- db file sequential read 待機の場合、CURRENT_OBJ# 列からオブジェクト番号が特定できるので対象オブジェクトを確認

```
select SAMPLE_TIME,INST_ID,SESSION_ID,SESSION_SERIAL#,EVENT,P1TEXT,P1,P2TEXT,P2,P3TEXT,P3,CURRENT_OBJ#  
from GV$ACTIVE_SESSION_HISTORY  
where SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')  
                        and to_date('<対象期間終了日時>','yyyymmddhh24miss')  
and EVENT= 'db file sequential read'  
and INST_ID = <INST_ID> and SESSION_ID = <SESSION_ID> and SESSION_SERIAL# = <SESSION_SERIAL#>  
order by 1;
```

SAMPLE_TIME	INST_ID	SESSION_ID	SESSION_SERIAL#	EVENT	P1TEXT	P1	P2TEXT	P2	P3TEXT	P3	CURRENT_OBJ#
01:28:23.870	1	1501	14317	db file sequential read	file#	152	block# 28646	blocks	1	1	100055
01:28:32.864	1	1501	14317	db file sequential read	file#	153	block# 18645	blocks	1	1	100055
01:28:33.864	1	1501	14317	db file sequential read	file#	152	block# 28648	blocks	1	1	100055
01:28:34.793	1	1501	14317	db file sequential read	file#	156	block# 9068	block	1	1	100055

INSERT先の表の索引と判明

【ケース2】バッチ処理の長時間走行

[参考] ASHの対象オブジェクトの特定方法

- ASHでは以下の条件の時に CURRENT_OBJ# 列に対象オブジェクト番号が入ります
 - セッションが待機状態
 - 待機タイプがApplication,Cluster, Concurrency, User I/O の場合

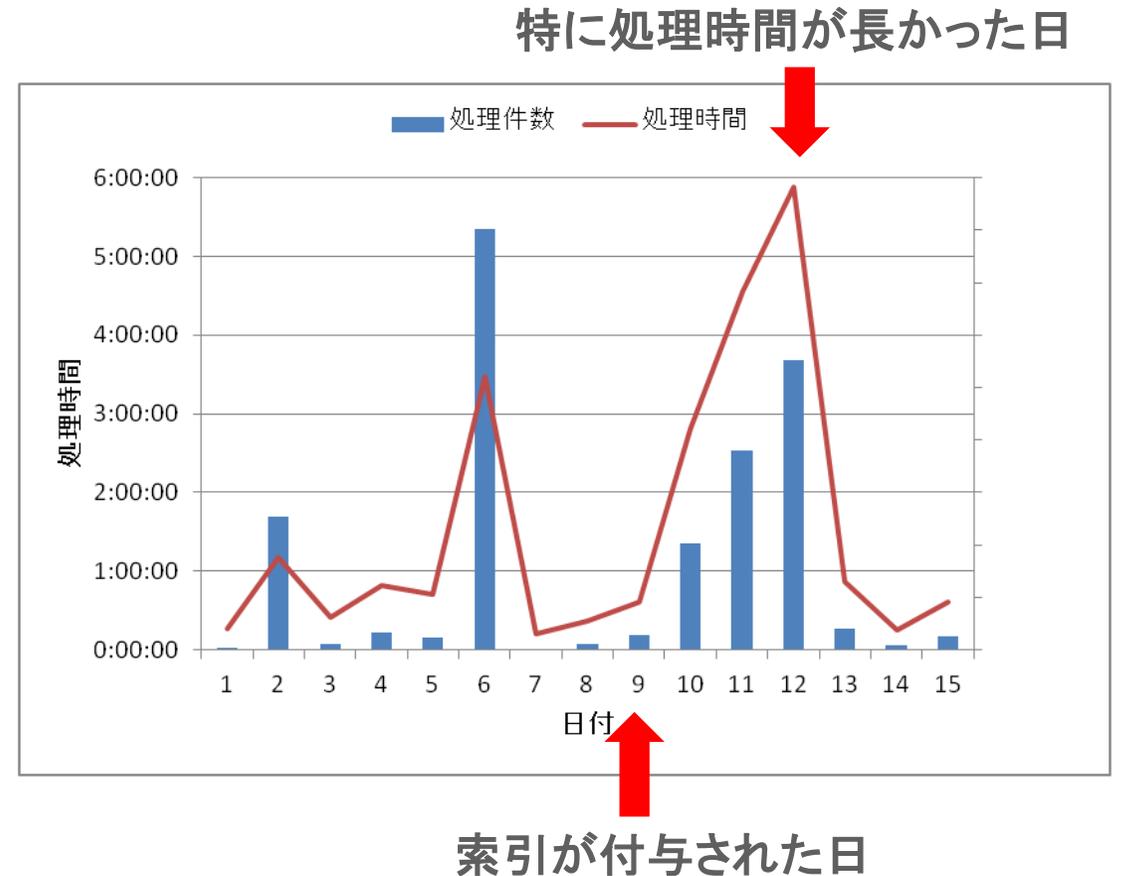
オブジェクト番号からオブジェクト名を特定する方法

```
select OWNER, OBJECT_NAME, SUBOBJECT_NAME
from   DBA_OBJECTS
where  OBJECT_ID = <オブジェクト番号>
```

【ケース2】バッチ処理の長時間走行

原因と対処

- db file sequential read が待機が多く見られた索引は 遅延が発生した数日前に付与されたものと判明
- 索引のサイズが大きく、SQL実行中に索引ブロックの読み込みとデータベースバッファキャッシュからのキャッシュアウトが繰り返し替えされていたため、頻繁に読み込みが生じていた。
- 対象の索引を削除してバッチ処理後に索引を再作成することで回避



【ケース2】バッチ処理の長時間走行

まとめ

項目	内容
問題概要	バッチ処理の遅延 バッチ処理内で実施されている SQL は “INSERT INTO 表 SELECT * FROM 表”
AWR/ASHの活用	[AWR] 問題発生時の待機イベントの全体傾向確認 [ASH] 特定のDBセッションにおける待機イベント発生傾向確認 [ASH] 特定のDBセッションにおける待機イベントの詳細情報の確認
原因と対処	INSERT対象表に付与された索引が大きく、SQL実行中にI/Oとキャッシュアウトが繰り返されたため。 バッチ処理前に索引を削除して再作成することで回避。

【ケース3】I/O負荷高騰

問題概要

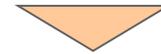
- 問題なく稼働していたDBに接続するアプリケーションのレスポンスが急に悪くなった。
- I/O系の待機イベントの平均待機時間 (Avg wait)が大きくなっている

例えば単一ブロック読み込みの待機イベントである **db file sequential read** が **30ミリ秒と遅くなっている**

正常時

Top 10 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
direct path read	41,432	141	3	42.4	User I/O
DB CPU		138		41.6	
db file sequential read	24,520	44	2	13.3	User I/O
gc cr block 2-way	26,116	4	0	1.1	Cluster
gc current block 2-way	14,740	2	0	.7	Cluster



問題発生時

Top 10 Timed Foreground Events

Event	Waits	Time(s)	Avg wait (ms)	% DB time	Wait Class
db file scattered read	72,239	2,500	36	27.1	User I/O
direct path write temp	7,000	1,404	199	14.9	User I/O
DB CPU		1,096		11.9	
db file sequential read	32,600	1,037	30	11.3	User I/O
direct path read	10,360	700	56	6.9	User I/O
...					

【ケース3】I/O負荷高騰

データベースのI/Oが増えたかどうか確認

– AWR の Instance Activity Stats から Database から発行されたI/O量を確認できる

Instance Activity Stats

Instance Activity Stats DB/Inst: TEST01/TEST011 Snaps: 12345-12346
-> Ordered by statistic name

Statistic	Total	per Second	per Trans
...			
physical read total IO requests	380,004	212.8	61.7
physical read total bytes	1,256,560,253,152	697,680,806.7	2.0150100E+08
physical write total IO requests	283,193	150.2	45.1
physical write total bytes	267,419,868,772	148,479,732.1	42,883,237.0
...			

physical read total IO requests : DISK読み取りの要求数
physical write total IO requests : DISK書き込みの要求数
Physical read total bytes : DISK読み取りの合計サイズ
Physical write total bytes : DISK書き込みの合計サイズ

このケースでは、正常時とI/O負荷高騰時で顕著なI/O回数、サイズの増減がなかった

【ケース3】I/O負荷高騰

[参考] I/O量が増えた場合に有効な情報

- 正常時とI/O量増加時を比較して何でI/Oが増えているのかを特定することができる。
- 複合した IOStat by Function/Filetype というセクションもあります

IOStat by Function summary (機能ごとのI/O情報)

Function Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Waits: Count	Avg Tm(ms)
Buffer Cache Re	53.9G	1384.6	30.8165	0M	0.0	0M	2293.5K	0.3
DBWR	0M	0.0	0M	4G	237.5	2.27843	0	N/A
LGWR	0M	0.0	0M	1.8G	173.7	1.03067	284K	0.1
Others	769M	30.8	.429124	181M	9.9	.101003	60.5K	1.8
Direct Reads	383M	26.3	.213725	0M	0.0	0M	0	N/A
Direct Writes	0M	0.0	0M	34M	1.4	.018972	0	N/A
TOTAL:	55.1G	1441.8	31.4594	6G	422.5	3.42908	2637.9K	0.3

IOStat by Filetype summary(ファイルタイプごとのI/O情報)

Function Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Waits: Count	Avg Tm(ms)
Data File	120.6G	1532.2	68.9372	4.1G	245.6	2.34706	0.2	13.2
Log File	0M	0.0	0M	1.8G	173.7	1.02788	N/A	N/A
Control File	676M	24.1	.377227	82M	3.0	.045758	0.1	N/A
Temp File	15M	0.1	.008370	16M	0.1	.008928	0.9	N/A
TOTAL:	121.3G	1556.5	69.3228	6G	422.3	3.42964	0.2	13.2

【ケース3】I/O負荷高騰

I/O系の待機イベントのヒストグラムを確認する

Wait Event Histogram

Wait Event Histogram		DB/Inst: TEST01/TEST011 Snaps: 12345-12346								
...		% of Waits								
Event	Total Waits	<1ms	<2ms	<4ms	<8ms	<16ms	<32ms	<=1s	>1s	
...										
db file scattered read	70.0K	70.6	.9	13.9	2.1	7.1	2.4	2.9	.1	
db file sequential read	34.8K	88.0	1.0	2.3	5.3	2.1	.5	.8	.1	
...										
Event	Waits to 64ms	<32ms	<64ms	<1/8s	<1/4s	<1/2s	<1s	<2s	>=2s	
db file scattered read	2144	96.9	1.1	.8	.6	.3	.1	.0	.1	
db file sequential read	280	99.1	.3	.2	.2	.1	.0	.0	.1	
...										

平均的に30ミリ秒程度レスポンスに時間がかかっているのではなく、一部のI/Oだけが非常に遅く平均レスポンス時間を引き上げている傾向

システム全体のI/O量が増加した結果、ストレージ負荷が限界になっているのではないと考える

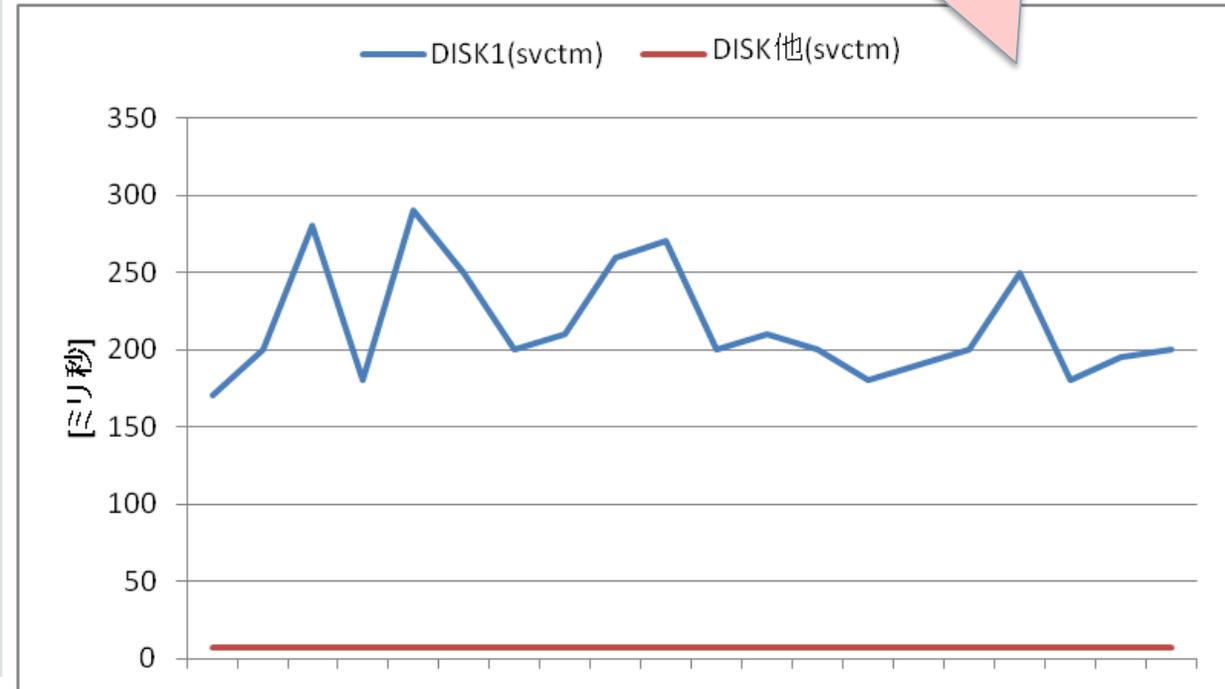
【ケース3】I/O負荷高騰

原因と対処

- OSのI/O統計(iostat) からsvctm(レスポンス時間)が非常に遅い DISKが1本あることを確認
- この DISKへのアクセスが全体のI/Oの Avg Wait を引き上げていた
- DISK故障と判断し、問題のDISKを ASM DISKGROUP からDROPIしたところ問題が解消(後日DISKを交換)

ほとんどのDISKの svctm が数ミリ秒の中、svctm が 200ミリ秒近くに高騰していたDISKがあった

OSの I/O統計(iostat より)



【ケース3】I/O負荷高騰

まとめ

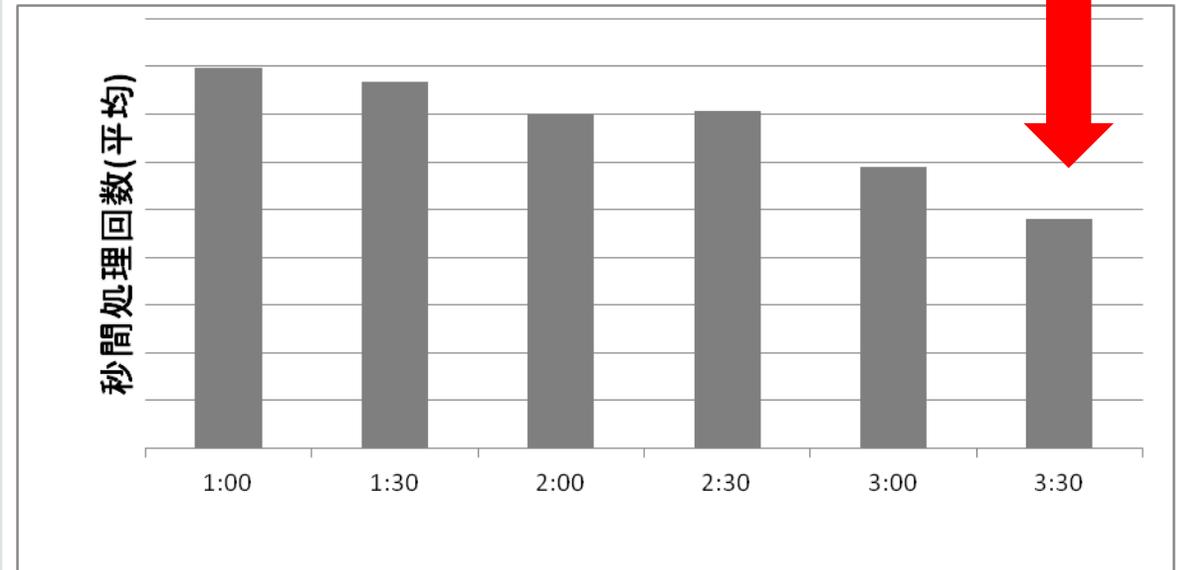
項目	内容
問題概要	I/O系の待機イベントの平均待機時間増加による性能低下
AWR/ASHの活用	[AWR] データベースのI/Oが増えたかどうか確認 [AWR] I/O系の待機イベントのヒストグラムを確認する
原因と対処	DISK故障。問題DISKを使用しないようにする。

【ケース4】実行時間が1秒以下のSQLの調査

問題概要

- 30分に1回、1分程度かけて実行されるアプリケーションで秒間実行回数を取得しているが、頻度は多くないものの秒間実行回数が低下する時がある。
- 該当の処理で実行しているSQLはINSERT文のみでSQL_IDも特定できているが、AWRレポートの『SQL Ordered by xxx』セクションには対象のSQLは出力されていない

秒間実行回数が低い時間

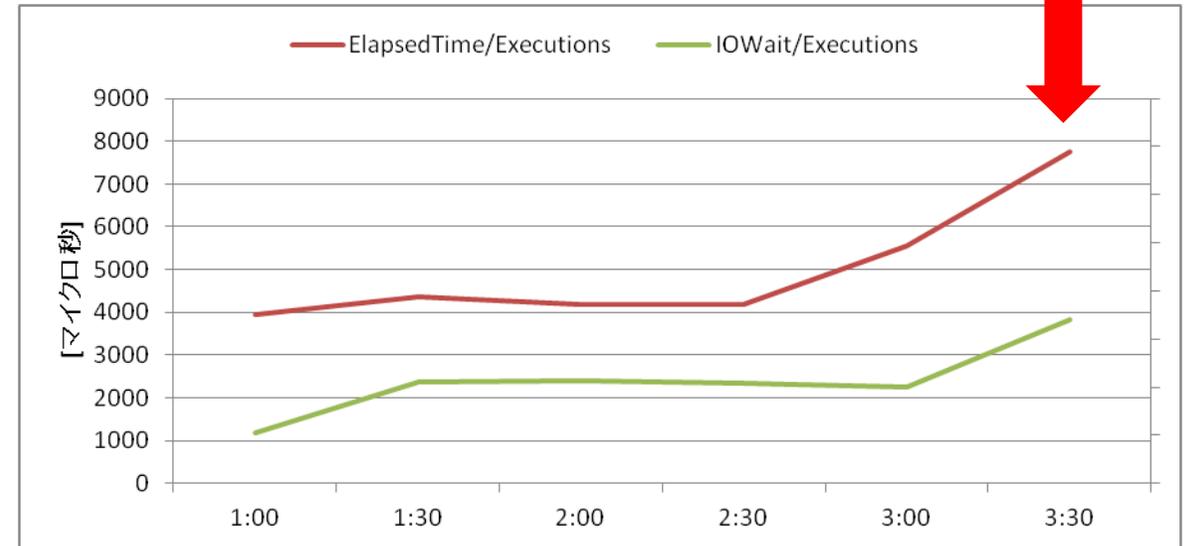


【ケース4】実行時間が1秒以下のSQLの調査

DBA_HIST_SQLSTATからの解析

- AWRレポートの『SQL Ordered by xxx』セクションに表示されていないSQLでもDBA_HIST_SQLSTATに統計が保存されている場合がある
- 対象のアプリケーションから実行されているSQLのSQL_IDの統計がDBA_HIST_SQLSTATに残っていたため分析を実施
- 秒間実行回数が低下している時は、1実行あたりのElapsedTimeが長くなり、IOWAITも増加していた

秒間実行回数が低い時間



DBA_HIST_SQLSTAT の列の意味

Executions : 実行回数

ELAPSED_TIME_DELTA : SQL実行時間 (単位マイクロ秒)

IOWAIT_DELTA : I/O待機時間(単位マイクロ秒)

【ケース4】実行時間が1秒以下のSQLの調査

[参考] DBA_HIST_SQLSTAT ビューで時間を確認できる列

列名

CPU_TIME_DELTA	このカーソルによって、解析、実行またはフェッチのために使用されたCPU時間(マイクロ秒)のデルタ値
ELAPSED_TIME_DELTA	このカーソルによって、解析、実行またはフェッチのために使用された経過時間(マイクロ秒)のデルタ値
IOWAIT_DELTA	ユーザーI/O待機時間のデルタ値(マイクロ秒)
CLWAIT_DELTA	クラスタ待機時間のデルタ値(マイクロ秒)
APWAIT_DELTA	アプリケーション待機時間のデルタ値(マイクロ秒)
CCWAIT_DELTA	同時実行性待機時間のデルタ値(マイクロ秒)
PLSEEXEC_TIME_DELTA	PL/SQL実行時間のデルタ値(マイクロ秒)
JAVEXEC_TIME_DELTA	Java実行時間のデルタ値(マイクロ秒)

【ケース4】実行時間が1秒以下のSQLの調査

対象モジュールのセッションにおける待機イベント種別毎の発生傾向確認

- ASH から特定の時間帯の WAIT_CLASS の傾向を確認。
- 普段はV\$ACTIVE_SESSION_HISTORY のエントリーには待機があまり出ていないが待機イベントが捕捉される回数が増えている。(特に User I/O)

```
select INST_ID, nvl(WAIT_CLASS, SESSION_STATE) "WAIT_CLASS", count(1) "SESSION_COUNT"
from   GV$ACTIVE_SESSION_HISTORY
where  SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss')
        and to_date('<対象期間終了日時>','yyyymmddhh24miss')
and    MODULE like 'MODULE10%'
group by INST_ID, nvl(WAIT_CLASS, SESSION_STATE)
order by 1,3,2;
```

正常時

INST_ID	WAIT_CLASS	SESSION_COUNT
1	User I/O	6
1	Cluster	12
1	Other	16

秒間実行回数低下時

INST_ID	WAIT_CLASS	SESSION_COUNT
1	Application	4
1	Other	13
1	Cluster	18
1	User I/O	30

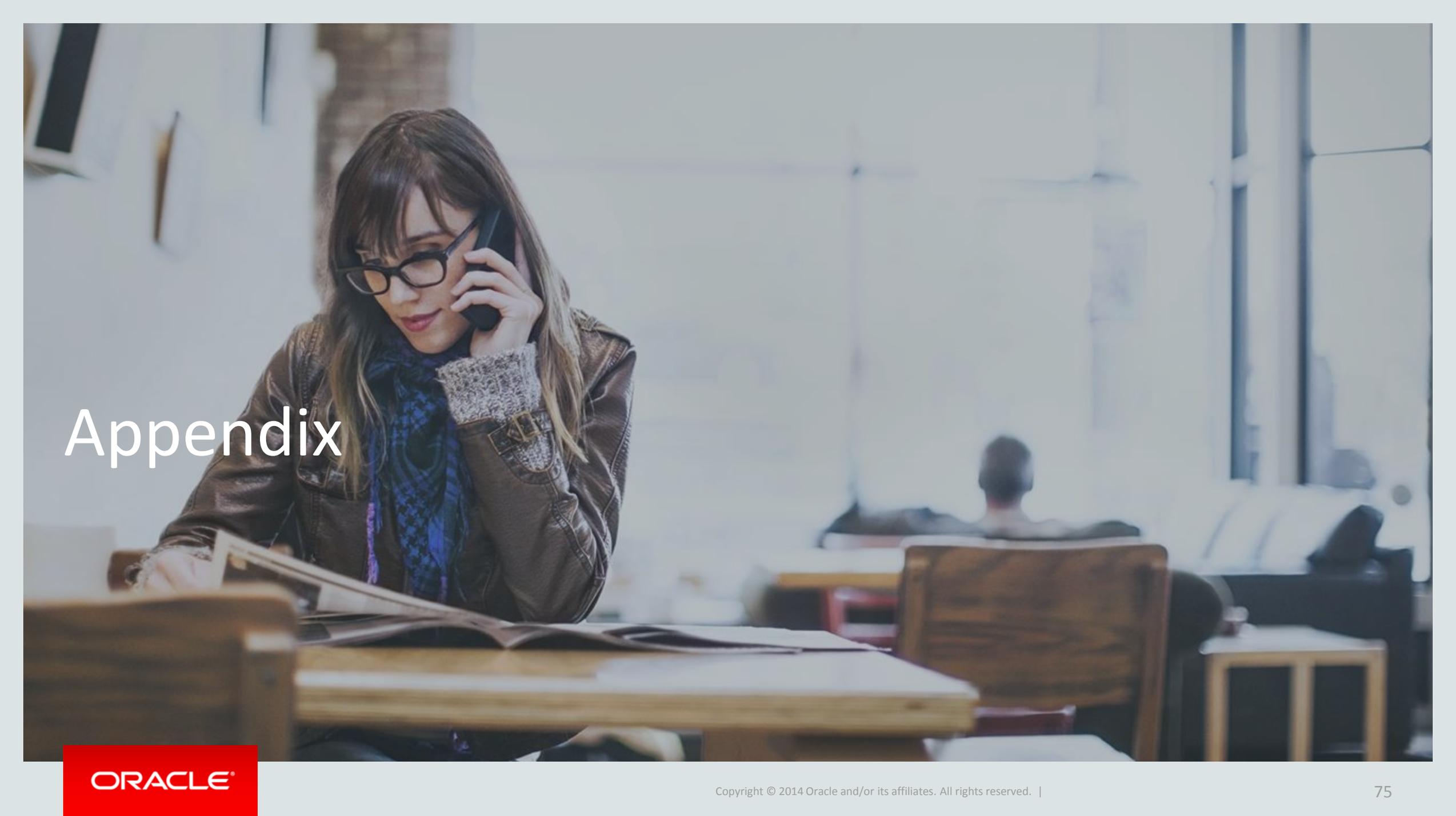
【ケース4】実行時間が1秒以下のSQLの調査

原因と対処

- AWR(DBA_HIST_SQLSTAT) や ASH からI/Oに問題がある可能性が高いと判断
- その後 iostat 等の調査から問題発生時は I/Oレスポンス時間が大きくなっていることを確認
- あるSQLの初回実行時に I/O回数(IOPS)が増加している傾向があったため、事前に実行してバッファキャッシュに乗せておく対処を実施

【ケース4】実行時間が1秒以下のSQLの調査 まとめ

項目	内容
問題概要	アプリケーションの特定処理の秒間実行回数が低下した。 SQLの1回あたりの実行時間が数ミリ秒増加していた
AWR/ASHの活用	[AWR] DBA_HIST_SQLSTATからの解析 [ASH] 全DBセッションで記録されていた待機イベント種別毎の発生傾向確認
原因と対処	AWR/ASH から I/O に問題がある点を絞り込み あるSQLの初回実行時にI/O回数が増加し、I/Oレスポンス時間が長くなる傾向が見られたため、事前実行することで回避

A woman with long brown hair and glasses is sitting at a wooden table in a cafe. She is wearing a brown leather jacket over a blue patterned scarf. She is holding a black smartphone to her ear with her left hand and looking down at a newspaper or magazine on the table with her right hand. The background is a bright, modern cafe with large windows and other people sitting at tables.

Appendix

DBA_HISTビューからの差分情報取得

- スナップ間差分を抽出するための考慮事項

例) 11/17 ~ 11/18の1時間毎のSQL実行回数の差分を取得するSQL

```
select
  bsn.SNAP_ID           || ' "' ||
  to_char (bsn.END_INTERVAL_TIME, 'yyyy') || ' "' ||
  ...                  || ' "' ||
  esn.SNAP_ID           || ' "' ||
  ...                  || ' "' ||
  esy.STAT_NAME         || ' "' ||
  (esy.VALUE - bsy.VALUE) || ' "'
```

DBA_HIST_SNAPSHOT bsn : 差分比較時の前スナップ情報取得
DBA_HIST_SNAPSHOT esn : 差分比較時の後スナップ情報取得
DBA_HIST_SYSSTAT bsy : 差分比較時の前スナップ内データ取得
DBA_HIST_SYSSTAT esy : 差分比較時の後スナップ内データ取得

```
from
  DBA_HIST_SNAPSHOT bsn, DBA_HIST_SNAPSHOT esn, DBA_HIST_SYSSTAT bsy, DBA_HIST_SYSSTAT esy
where bsn.DBID = esn.DBID and bsn.INSTANCE_NUMBER = esn.INSTANCE_NUMBER and bsn.END_INTERVAL_TIME = esn.BEGIN_INTERVAL_TIME
and bsn.STARTUP_TIME = esn.STARTUP_TIME
and bsn.DBID = bsy.DBID and bsn.INSTANCE_NUMBER = bsy.INSTANCE_NUMBER and bsn.SNAP_ID = bsy.SNAP_ID
and esn.DBID = esy.DBID and esn.INSTANCE_NUMBER = esy.INSTANCE_NUMBER and esn.SNAP_ID = esy.SNAP_ID
and bsy.STAT_NAME = esy.STAT_NAME and bsy.STAT_NAME = 'execute count'
and bsn.END_INTERVAL_TIME >= to_date('201411170000', 'yyyymmddhh24mi')
and esn.END_INTERVAL_TIME <= to_date('201411180002', 'yyyymmddhh24mi');
```

DBA_HISTビューからの差分情報取得

- スナップ間差分を抽出するための考慮事項

例) 11/17 ~ 11/18の1時間毎のSQL実行回数の差分を取得するSQL

```
select
  bsn.SNAP_ID || ' ' || to_char (bsn.END_INTERVAL_TIME, 'yyyy') || ' ' ||
  ...
  esn.SNAP_ID || ' ' ||
  ...
  esy.STAT_NAME || ' ' ||
  (esy.VALUE - bsy.VALUE) || ' ' ||
from
  DBA_HIST_SNAPSHOT bsn, DBA_HIST_SNAPSHOT esn, DBA_HIST_SYSS
where bsn.DBID = esn.DBID and bsn.INSTANCE_NUMBER = esn.INSTANCE_NUMBER
and bsn.STARTUP_TIME = esn.STARTUP_TIME
and bsn.DBID = bsy.DBID and bsn.INSTANCE_NUMBER = bsy.INSTANCE_NUMBER and bsn.SNAP_ID = bsy.SNAP_ID
and esn.DBID = esy.DBID and esn.INSTANCE_NUMBER = esy.INSTANCE_NUMBER and esn.SNAP_ID = esy.SNAP_ID
and bsy.STAT_NAME = esy.STAT_NAME and bsy.STAT_NAME = 'execute count'
and bsn.END_INTERVAL_TIME >= to_date('201411170000', 'yyyymmddhh24mi')
and esn.END_INTERVAL_TIME <= to_date('201411180002', 'yyyymmddhh24mi');
```

DBA_HIST_SNAPSHOT.END_INTERVAL_TIME が
スナップショットの取得時間となるため、
分析対象期間の開始、終了日時をそれぞれ指定

* スナップショットの取得時間は、数秒のズレが発生するため、終了日時の設定の際には1分~2分程度プラスした時間とします

DBA_HISTビューからの差分情報取得

- スナップ間差分を抽出するための考慮事項

例) 11/17 ~ 11/18の1時間毎のSQL実行回数の差分を取得するSQL

```
select
  bsn.SNAP_ID || ' ' || to_char (bsn.END_INTERVAL_TIME, 'yyyy') || ' ' || esn.SNAP_ID || ' ' || esy.STAT_NAME
  (esy.VALUE - bsy.VALUE)
from
  DBA_HIST_SNAPSHOT bsn, DBA_HIST_SNAPSHOT esn, DBA_HIST_SYSSTAT bsy, DBA_HIST_SYSSTAT esy
where bsn.DBID = esn.DBID and bsn.INSTANCE_NUMBER = esn.INSTANCE_NUMBER and bsn.END_INTERVAL_TIME = esn.BEGIN_INTERVAL_TIME
and bsn.STARTUP_TIME = esn.STARTUP_TIME
and bsn.DBID = bsy.DBID and bsn.INSTANCE_NUMBER = bsy.INSTANCE_NUMBER and bsn.SNAP_ID = bsy.SNAP_ID
and esn.DBID = esy.DBID and esn.INSTANCE_NUMBER = esy.INSTANCE_NUMBER and esn.SNAP_ID = esy.SNAP_ID
and bsy.STAT_NAME = esy.STAT_NAME and bsy.STAT_NAME = 'execute count'
and bsn.END_INTERVAL_TIME >= to_date('201411170000', 'yyyymmddhh24mi')
and esn.END_INTERVAL_TIME <= to_date('201411180002', 'yyyymmddhh24mi');
```

BEGIN_INTERVAL_TIMEは1つ前のスナップショット取得時間となる
差分比較時の前スナップの**END_INTERVAL_TIME**と結合
これにより前後のスナップ内のデータを結合できる

ASH調査用SQLサンプル

- パフォーマンス分析ケース毎のASH活用方法
 - 特定のAPで使用されているDBセッション情報の確認

目的	指定した期間に特定のAPに使用されているDBセッションを特定するための情報を取得する
入力情報	<ul style="list-style-type: none"> ①調査対象となる期間が特定できていること ②クライアントプログラム名、クライアントモジュール名、クライアント識別子、クライアントマシンのいずれかが特定できていること
SQL例	<pre> select distinct INST_ID, SESSION_ID, SESSION_SERIAL#, PROGRAM, MODULE, CLIENT_ID, MACHINE from GV\$ACTIVE_SESSION_HISTORY where SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss') and to_date('<対象期間終了日時>','yyyymmddhh24miss') and (PROGRAM = '<PROGRAM>' or MODULE = '<MODULE>' or CLIENT_ID = '<CLIENT_ID>' or MACHINE = '<MACHINE>') order by 1,2,3; </pre>

ASH調査用SQLサンプル

- パフォーマンス分析ケース毎のASH活用方法
 - 特定のDBセッションの待機と原因となっているブロッキング・セッションの確認

目的	指定した期間に特定のセッションを待機させていたブロッキング・セッションを特定する
入力情報	<ul style="list-style-type: none"> ①調査対象となる期間が特定できていること ②該当セッションの存在するインスタンス、セッションID、セッション・シリアルが特定できていること
SQL例	<pre>select SAMPLE_TIME, INST_ID, SESSION_ID, SESSION_SERIAL#, BLOCKING_INST_ID, BLOCKING_SESSION, BLOCKING_SESSION_SERIAL#, BLOCKING_SESSION_STATUS from GV\$ACTIVE_SESSION_HISTORY where SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss') and to_date('<対象期間終了日時>','yyyymmddhh24miss') and INST_ID = <INST_ID> and SESSION_ID = <SESSION_ID> and SESSION_SERIAL# = <SESSION_SERIAL#> order by 1;</pre> <p>* 上記をブロッキング・セッションが存在しなくなるまで繰り返し実施することで大元のブロッキング・セッションを特定</p>

ASH調査用SQLサンプル

- パフォーマンス分析ケース毎のASH活用方法
 - 実行が記録されたSQLのPGAメモリ、一時表領域使用状況の確認

目的	指定した期間に実行が記録されたSQL毎に確保されたPGAメモリ、一時表領域の最大サイズを確認する
入力情報	①調査対象となる期間が特定できていること
SQL例	<pre> select INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, max(PGA_ALLOCATED) "MAX_PGA_ALLOCATED", max(TEMP_SPACE_ALLOCATED) "MAX_TEMP_ALLOCATED" from GV\$ACTIVE_SESSION_HISTORY where SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss') and to_date('<対象期間終了日時>','yyyymmddhh24miss') and SQL_ID is not null group by INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE order by 1,4,5,2,3; </pre>

ASH調査用SQLサンプル

- パフォーマンス分析ケース毎のASH活用方法
 - 特定の待機イベントの待機が記録されたSQLの確認

目的	指定した期間に特定の待機イベントでの待機の記録回数が多いSQLと記録回数を確認する
入力情報	<ul style="list-style-type: none"> ①調査対象となる期間が特定できていること ②該当待機イベント名が特定できていること
SQL例	<pre> select INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, EVENT, count(1) "SESSION_COUNT" from GV\$ACTIVE_SESSION_HISTORY where SAMPLE_TIME between to_date('<対象期間開始日時>','yyyymmddhh24miss') and to_date('<対象期間終了日時>','yyyymmddhh24miss') and EVENT = '<EVENT>' and SQL_ID is not null group by INST_ID, SQL_ID, SQL_PLAN_HASH_VALUE, EVENT order by 1,5,4,2,3; </pre>

Oracle Database 12c おすすめ研修コース

Oracle Database 12c: SQL チューニング ワークショップ

概要	このコースでは、OracleのSQL文のチューニングや、Oracle Databaseに合わせて適切にチューニングされたSQL文を記述する方法を説明します。SQLトレース機能の使い方、実行計画の取得方法、オプティマイザ機能の活用方法などを、実機演習を通して習得することができます。	
学習項目	<ul style="list-style-type: none">■ Database Vaultの概要■ Database Vaultの構成■ 権限の分析 (12c 新機能)■ レルムの構成■ ルール・セットの定義	<ul style="list-style-type: none">■ コマンド・ルールの構成■ ルール・セットの拡張■ セキュア・アプリケーション・ロールの構成■ Database Vaultレポートによる監査■ ベスト・プラクティスの実装
コース日数	3 日間 【トレーニングキャンパス赤坂】 2014/12/3-5	

Oracle Database 12c: パフォーマンス・チューニング

概要	このコースでは、OracleのSQL文のチューニングや、Oracle Databaseに合わせて適切にチューニングされたSQL文を記述する方法を説明します。SQLトレース機能の使い方、実行計画の取得方法、オプティマイザ機能の活用方法などを、実機演習を通して習得することができます。	
学習項目	<ul style="list-style-type: none">■ 基本チューニング診断■ 自動ワークロード・リポジトリの使用■ パフォーマンス問題の範囲の定義■ メトリックとアラートの使用■ ベースラインの使用■ AWRベースのツールの使用■ リアルタイム・データベース操作監視■ アプリケーションの監視■ 問題のあるSQL文の識別■ オプティマイザへの影響	<ul style="list-style-type: none">■ SQL操作のコストの削減■ SQLパフォーマンス・アナライザの使用■ SQLパフォーマンスの管理■ データベース・リプレイの使用■ 共有プールのチューニング■ バッファ・キャッシュのチューニング■ PGAおよび一時領域のチューニング■ 自動メモリー管理の使用■ パフォーマンス・チューニングのまとめ
コース日数	5 日間 【トレーニングキャンパス赤坂】 2015/1/19-23	

詳細は [Oracle University Webサイト](#) にてご確認ください。

Hardware and Software Engineered to Work Together

VISION 2020

#1 CLOUD

ORACLE JAPAN

ORACLE®