Édition numéro 6 pour la Romandie

DOAG SOUG News

Big Data

Article

Que gagneriez-vous en passant sur Exadata?

Geatano Bisaz
SOUG

Chers membres du SOUG,

Connaissez-vous le Raspberry Pi ou Arduino? Ces petits ordinateurs monocartes aux performances correctes, connectés au monde extérieur et avec un énorme écosystème d'accessoires.

Je me suis récemment commandé un Raspberry avec un module Sensor "Sense HAT". En plus d'un écran LED avec matrice 8x8, ce petit appareil dispose également de capteurs de température, d'accélération, d'humidité de l'air et de position. Le but réel du Raspberry pi est d'apprendre la programmation aux étudiants - en soi, un sujet vaste et fascinant. Le Sense HAT permet d'aller plus loin : l'idée de départ est de l'utiliser sur ISS avec une grande quantité de lignes de codes dans le cadre d'expériences scientifiques, sélectionnées par les étudiants eux mêmes lors d'un concours *(http://astro-pi.org)*.

Dès lors, nous pouvons légitimement nous poser cette question : qu'est-ce qu'un ordinateur bricolé et un capteur pour "jeunes chercheurs" ont à voir avec le professionnalisme intrinsèque aux bases de données et le caractère sérieux de nos requêtes SQL ? Jouer avec les nouvelles technologies, tout simplement. On peut planifier des Proofs of Concept avec In Memory, Container, Hadoop, Flash, SPARC, 12c et plus encore. La plus grande partie du temps est consacrée à la création de scénarios de tests et à la planification de flux de données, basées principalement sur l'expérience du passé et sur "ce qui fonctionnait". Cette approche est acceptable, mais devrait être accompagnée de ressources, de temps et de la possibilité de jouer avec les nouvelles technologies de manière irrationnelle et aléatoire.

Idéalement, ceci est possible en labo privé, mais aussi grâce aux nouvelles et multiples possibilités qu'offrent les Clouds matures, souvent même gratuitement. Et quand il s'agit de la connexion des mondes d'atomes et de bits, ces modules sont vraiment très bon marché et intéressants. Il ne m'était plus arrivé depuis longtemps de m'enthousiasmer à ce point pour de la technologie, dans le cas présent pour le "Sense HAT", en jouant. Bien longtemps après une amusante soirée "Python" éclosent encore les idées de ce qu'on peut en faire - y compris dans le cadre de l'entreprise.

Les nouveaux concepts et outils informatiques proposent plus que ce que les interfaces ne laissent supposer. Que nous apportent-ils vraiment, que peuvent-ils de plus que ce que nous ne savons déjà? La réponse en jouant!

Commentaires à nl@soug.ch

Gaetano Bisaz

Agenda

20.01.2016
SOUGbideLüt

02.3.2016
SIG 1 in Baden-Dättwil (ABB)

17.3.2016
SIG Primavera in Zürich

Que gagneriez-vous en passant sur Exadata?

Partie I – Mesurez l'activité éligible au SmartScan

Franck Pachot, dbi-services

Exadata vient avec beaucoup de fonctionnalités qui en font une machine très rapide pour différents types de charge: serveurs puissants, disques rapides, flash, infiniband,... Tout ça est intégré pour en faire la 'Database Machine' avec laquelle Oracle propose le Hardware et le Software dans une configuration déjà assemblée et testée. Mais en termes de performances, il y a une fonctionnalité qui n'existe que sur Exadata, c'est le SmartScan, et c'est donc l'argument final pour faire le choix. Cependant le SmartScan ne s'applique pas à tous les types de bases et d'applications. Développé au départ pour les datawarehouses, il est mis en avant maintenant aussi pour de l'OLTP. Qu'en est-il réellement ? Au-delà de la théorie et du marketing, voici comment évaluer ce que SmartScan peut vous apporter – sur votre base et vos applications.

SAN vs. Disques locaux, et la Database Machine

Les I/O – lectures et écritures sur disque – ne demandent rien de particulier au stockage à part écrire des blocs de manière persistante et hautement disponible, afin de pouvoir les relire plus tard. Certains layers de stockage peuvent faire du travail supplémentaire, compresser, ou mettre en cache par exemple, mais ce n'est pas nécessaire ni recommandé: Oracle a lui-même son cache, géré de manière plus intelligente puisque l'algorithme est adapté au type de données.

On n'attend donc pas d'autre intelligence du serveur de stockage. Et encore moins lorsqu'on est en ASM : tout ce qu'un SAN peut apporter comme mirroring, réorganisation online,... peut-être fait par ASM. L'accès aux disques par plusieurs serveurs, en cluster, et aussi géré par ASM.

L'idéal pour une base de données Oracle est donc de ne pas avoir un stockage trop intelligent qui va finalement rajouter un overhead. D'ailleurs, ASM avait été créé pour cela : ne faire que ce qui est nécessaire pour une base Oracle, de manière performante, et avec beaucoup plus de flexibilité que les anciens 'raw device'. Mettre des disques directement attachés au serveur de base de données et limiter les couches logicielles est ce qui donne les meilleures performances. C'est exactement ce que fait l'ODA par exemple.

Mais l'inconvénient, c'est l'évolutivité de la solution dans une entreprise où il y a plusieurs serveurs de bases de données. Gérer des serveurs avec leurs propres disques, garder les disques de spare pour chaque type de matériel, faire du capacity planning pour anticiper la taille de la base plusieurs années à l'avance, devient impossible. La solution rependue depuis 15 ans a alors été d'utiliser un SAN : le stockage est centralisé dans une baie de disques, mis à disposition sur un réseau dédié et rapide, et accédé par tous les serveurs. Ces serveurs voient des LUN comme si c'était des disques, mais on peut plus facilement les maintenir, les agrandir, les réorganiser, etc.

Mais pour augmenter cette flexibilité, on a rajouté un nouveau point de contention : la bande passante sur ce réseau SAN. Bien sûr, le matériel d'aujourd'hui permet d'avoir de très bons débits. Pour du transactionnel, où c'est plutôt la latence qui compte, on a de bonnes performances. Par contre pour du datawarehouse, aussi bien l'ETL que le reporting va saturer la bande passante de notre SAN.

On se retrouve donc à devoir choisir entre la maintenabilité et la performance.

iDB : Intelligent DataBase protocol

Alors toute l'idée d'Exadata, c'est d'avoir la flexibilité du SAN avec les performances des disques attachés. Dans une machine (c'est un 'Engineered System') on a des serveurs base de données tels qu'on les connait, des serveurs de stockage avec des disques (mécaniques ou flash) et un réseau InfiniBand pour les relier. On a la même évolutivité qu'un SAN : on peut rajouter des serveurs de stockage et/ou de bases de données, ce qui permet de partager les données sur tous les serveurs, d'assurer la haute disponibilité avec des niveaux de redondance, de réorganiser à chaud, etc.

La figure 1. Montre l'architecture d'une 'Exadata Database Machine' : les serveurs de base de donnée échangent les blocs avec les serveurs de stockage via le réseau Infin InfiniBand.

Mais pour garder de bonnes performances, il faut contrôler le volume de données qui circulent entre le stockage et la base de données, par le réseau SAN jusqu'à la CPU

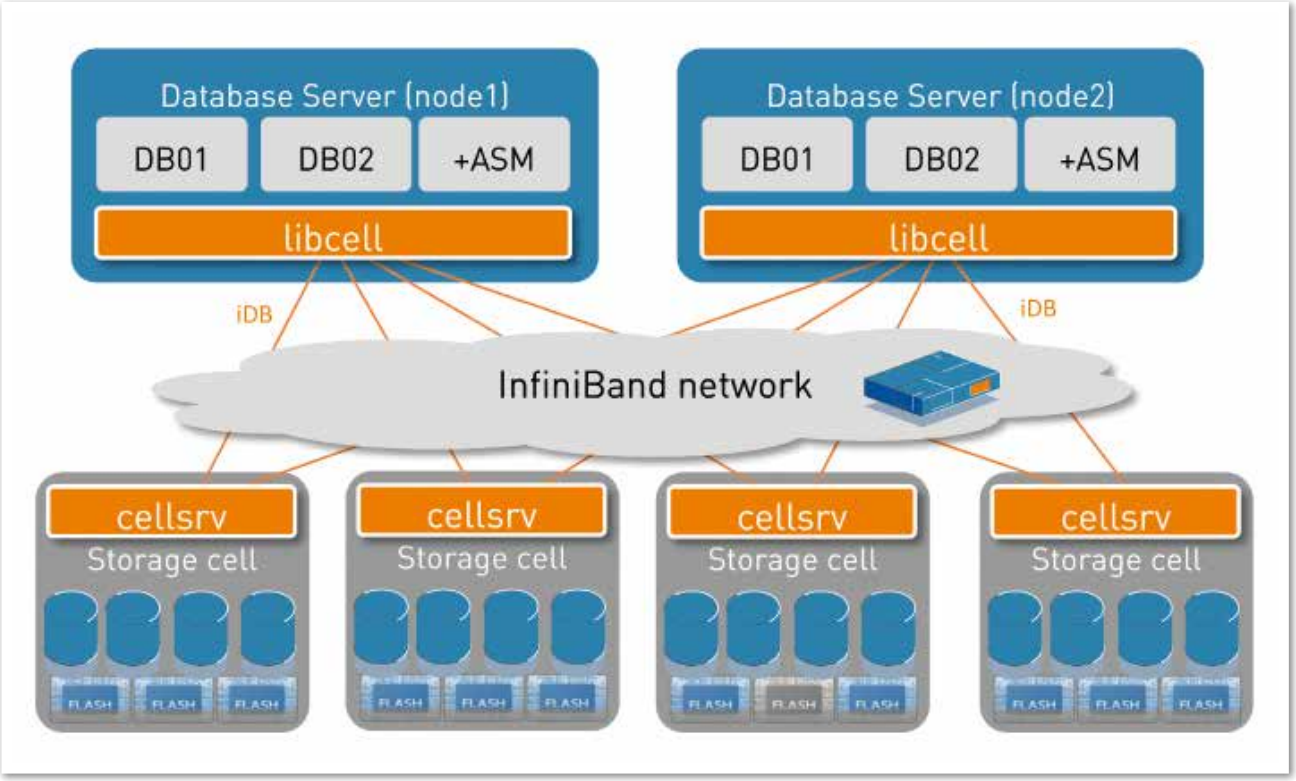

Fig 1. Exadata Database Machine

qui traite ces données. L'idée, c'est de filtrer en amont. Les requêtes qui font beaucoup de lecture, en BI par exemple, se retrouvent souvent à aller lire de grosses table – full table scan – pour finalement n'avoir besoin que d'un sous-ensemble des lignes (en fonction de la clause WHERE) et d'un sous-ensemble des colonnes (en fonction de la clause SELECT). Alors si on peut déjà filtrer dès le serveur de stockage, on a beaucoup moins d'information à envoyer sur le réseau SAN. Et voilà l'idée du '*predicate and projection offloading*' et du protocole 'iDB' – Intelligent DataBase protocol – qui était prévu pour être un protocole ouvert utilisable par différents systèmes de stockage. Ça a commencé avec HP pour l'Exadata V1 puis Oracle a racheté Sun et a décidé d'en faire quelque chose d'utilisable seulement sur leur propre hardware.

Nous avons donc un protocole qui envoi certaines informations en marge des appels I/O afin que le stockage puisse commencer à filtrer. L'intelligence du côté du stockage est appelé SmartScan, et c'est ce qu'on va détailler ici.

Direct path read

Si le stockage a déjà appliqué sur les blocs une partie des prédicats de la clause WHERE et de la projection de la clause SELECT, alors on se retrouve à récupérer des blocs incomplets, qui n'ont de sens que pour notre requête.

Cependant, on a plutôt l'habitude, lorsqu'on lit des blocs, de les partager dans le buffer cache : une session les a récupéré sur disque et les autres sessions peuvent les lires et les modifier tant que ça reste en cache. C'est bien pour les performances, mais c'est aussi indispensable pour la concurrence d'accès. En effet, le buffer cache étant partagé par toutes les sessions, on peut s'assurer que toutes les modifications se font à un seul endroit – la version courante. Et même en cluster – en RAC – le cache est vu comme un 'global cache' par toutes les instances. Le bloc courant va voyager entre les instances lorsqu'il doit être modifié.

La figure 2. montre les deux types de lectures. Vers le buffer cache, on doit lire des blocs complets car ils vont être utilisés par plusieurs requêtes. Il n'y a que lorsqu'on lit directement en PGA qu'on peut se permettre de récupérer des blocs qui ont déjà été filtrés par le serveur de stockage. Ne seront présentes que les lignes et les colonnes nécessaires à notre requête.

Le SmartScan, mécanisme de délégation des prédicats WHERE et projections SELECT, qui renvoit à la base des blocs incomplets, n'est donc pas possible pour les lectures vers le cache. Avant la 11g, la lecture directe vers la PGA ne se faisait qu'en Parallel Query. Il a été étendu pour être utilisable aussi en Serial (le mode par défaut, où une requête n'est traitée que par un seul process), et s'appelle '*serial direct read*'. Vous l'avez tous déjà vu passer, c'était une des surprises des migrations en 11g.

Il faut bien comprendre que le SmartScan, l'avantage principal de Exadata, n'est possible que pour les lectures directes. La première chose à faire si vous pensez passer sur Exadata, c'est donc de vérifier le temps que vous passez sur ces 'direct path read'.

Une précision quand même. J'ai parlé de blocs dans les deux cas pour simplifier, mais en réalité SmartScan n'opère pas sur des blocs, mais des Allocation Units ASM. Et ce n'est pas de blocs qu'il ne renvoie à la PGA, mais des 'tuples'.

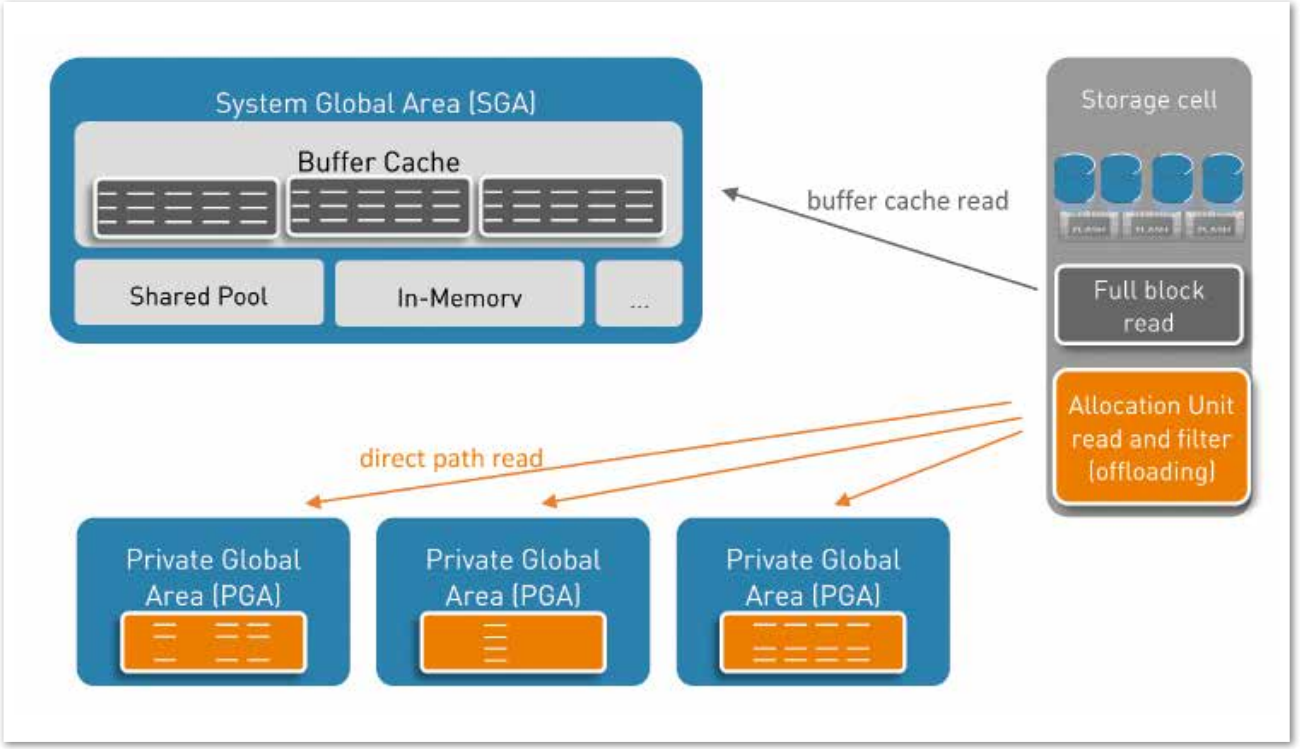

Fig 2. buffer cache read vs. direct path read

Statspack / AWR pour évaluer le gain apporté par Exadata

Donc sur votre système, si vous voulez voir ce que peut vous apporter Exadata SmartScan, il faut commencer par aller voir si vous faites beaucoup de lectures directes. Voici un rapport AWR sur 2 minutes d'activité d'une base ERP.

Top 10 Foreground Events by Total Wait Time

Ici je passe mon temps en CPU et en I/O '*db file scattered read*' et '*db file sequential read*'. Que va vous apporter Exadata là-dessus ? En CPU, Exadata a de très bons serveurs avec des CPU Intel. Ces processeurs ont un core facteur 0.5 et c'est donc le meilleur rapport performance/prix. Mais cela n'est pas particulier à Exadata.

Sur le temps passé en I/O, '*db file scattered read*' et '*db file sequential read*' sont des lectures vers le buffer cache – respectivement lecture d'un bloc ou de plusieurs. Il n'y aura pas de SmartScan là-dessus. La seule différence lorsque vous passerez à Exadata, c'est que les wait events s'appelleront respectivement '*cell single block read*' et '*cell multiblock read*'.

Mais ce n'est que le nom qui change. Le '*storage cell*' ne fera jamais de '*offloading*' là-dessus. Sur ces '*timed events*' typiques d'un ERP vous n'avez rien à attendre des fonctionnalités exclusives d'Exadata.

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		42		46.7	
db file sequential read	17,252	21.3	1.23	23.6	User I/O
db file scattered read	13,925	12.2	0.87	13.5	User I/O
log file sync	1,210	11.5	9.50	12.8	Commit
direct path write	3,670	5.5	1.51	6.2	User I/O
direct path sync	7	1.5	219.13	1.7	User I/O
enq: RO - fast object reuse	10	.3	32.71	.4	Application
SQL*Net more data to client	2,470	.3	0.11	.3	Network
direct path read	38	.2	4.72	.2	User I/O
db file parallel read	58	.2	3.04	.2	User I/O

tableau 1

Statistic	Total	per Second	per Trans
cell physical IO bytes eligible for predicate offload	23,412,736	261,177.52	18,291.20
cell physical IO interconnect bytes returned by smart scan	555,664	6,198.63	434.11
physical read bytes	26,942,472,192	164,853,317.21	15,755,831.69
physical read IO requests	67,342	412.05	39.38

tableau 2

Simulation SmartScan pour aller plus loin

Vous ne me croyez pas ? Et bien vous pouvez le vérifier par vous-même. Achetez ou louez un Exadata, faites tourner votre application dessus, et regarder les statistiques '*cell physical IO bytes eligible for predicate offload*' et '*cell physical IO interconnect bytes returned by smart scan*' et sur l'exemple précédent, vous verrez quelque chose comme ça *(confer tableau 2)*.

Ici, rien n'est éligible au SmartScan, et rien n'est retourné par le SmartScan. Rien, bien sûr c'est relatif: 23M seulement alors qu'on a lu en tout 26GB. C'est le même workload, mais sur Exadata. Il a tourné un peu plus longtemps: les '*physical read IO requests*' correspondent à la somme des '*db file sequential read*' et '*db file squattered read*' et ici, par rapport à l'exemple ci-dessus, j'en ai deux fois plus.

Mais vous n'avez pas besoin d'avoir un Exadata pour voir ça. Toute instance 11g ou 12c peut simuler le '*predicate and projection offloading*' et vous donner cette information. Il suffit de l'activer dans votre session ou pour l'instance *(confer listage 1)*.

Le paramètre '*cell_offload_plan_display*' vous affiche le plan d'exécution avec les informations SmartScan, par exemple :

Dans ce plan l'information 'STORAGE' donne simplement l'information que le FULL TABLE SCAN pourrait être sujet au SmartScan, et que le prédicat 'A=2' pourrait être exécuté sur le *storage cell*. Mais attention, ce n'est qu'une possibilité, et ça ne se fera que si l'exécution lit en '*serial direct read*'.

Le paramètre '*_rdbms_internal_fplib_enabled*', lui, active le code du SmartScan (*fplib* qui veut dire *filter processing library* est le même que ce qui tourne sur les *storage cells*) mais ici sur le nœud de base de données. Il n'y a donc pas de gain de performance (il y a même un certain overhead en CPU). Et il est instrumenté de la même manière, donc il nous permet de voir les statistiques du volume traité par SmartScan dans ce mode simulation.

Attention, c'est un paramètre non documenté. Il est préférable de ne pas l'utiliser directement sur la production, et il est préférable de l'activer seulement au niveau d'une ses-

```
SQL> alter session set cell_offload_plan_display=always "_rdbms_internal_fplib_enabled"=true;
```

listage 1

```
Execution Plan
----------------------------------------------------------
Plan hash value: 885065423

-------------------------------------------------------------------------------
| Id  | Operation                    | Name    | Rows  | Bytes | Cost (%CPU)|
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |         | 90909 |   175M| 15722   (1)|
|   1 |  NESTED LOOPS                |         | 90909 |   175M| 15722   (1)|
|   2 |   NESTED LOOPS               |         | 90909 |   175M| 15722   (1)|
|*  3 |    TABLE ACCESS STORAGE FULL | DEMO1   |   909 |   897K|   264   (0)|
|*  4 |    INDEX RANGE SCAN          | DEMO2PK |   100 |       |     2   (0)|
|   5 |   TABLE ACCESS BY INDEX ROWID| DEMO2   |   100 |    99K|    17   (0)|
-------------------------------------------------------------------------------

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

   3 - storage("DEMO1"."A"=2)
       filter("DEMO1"."A"=2)
   4 - access("DEMO1"."ID"="DEMO2"."ID")
```

listage 2

Statistic	Total	per Second	per Trans
cell simulated physical IO bytes eligible for predicate offload	11,706,368	71,627.93	6,845.83
cell simulated physical IO bytes returned by predicate offload	277,832	1,699.97	162.47
physical read bytes	10,767,089,664	120,110,768.98	8,411,788.80
physical read IO requests	31,654	353.11	24.73

tableau 3

sion plutôt que de l'instance. Cela dit, c'est ce paramètre qui est activé lorsque vous utilisez le Performance Analyzer du Tuning Pack, mais ici, vous n'avez pas besoin d'option.

En mode simulation, les statistiques sont :

- '*cell simulated physical IO bytes eligible for predicate offload*' à la place de '*cell physical IO bytes eligible for predicate offload*'
- '*cell simulated physical IO bytes returned by predicate offload*' à la place de '*cell physical IO interconnect bytes returned by smart scan*'

Voici un exemple sur mon rapport précédent *(confer tableau 3)*.

C'est le workload de 2 minutes sur ma base non-Exadata (les 31000 appels i/o correspondent à la somme des wait events '*db file ... read*'). Le mode simulation nous donne des stats identiques à Exadata - à l'exception de leur nom. Pour le '*offloading seulement*' car il n'y a pas de Storage Index ici.

C'est le moyen de savoir à l'avance le pourcentage de lecture qui va être traité en SmartScan lorsque vous serez sur Exadata. Mais pour ce cas pas, pas besoin d'aller jusque-là. La majorité des wait events ne sont pas des '*direct path read*' et ne seront jamais sujet au SmartScan. Donc le temps (DB Time) associé ne diminuera pas en passant à Exadata.

Buffer cache ou direct read : une question de checkpoint

Avant d'aller plus loin, la question est : pourquoi ne pas toujours faire des lectures directes lorsqu'on est en Exadata ? Et bien il y a un coût derrière, et le choix ne va se faire qu'au moment de l'exécution.

Une première chose est que cette lecture n'est efficace que s'il y a peu de blocs déjà en buffer cache. Sinon, c'est probablement mieux de passer par le buffer cache pour économiser la lecture de ces blocs. Le seuil qu'utilise Oracle, c'est que lorsqu'il y a plus de la moitié de la table en buffer cache, on ne fait pas de lecture directe.

La deuxième chose, c'est qu'avant de faire une lecture directe, il faut faire un checkpoint des modifications faites en buffer cache. Le seuil ici c'est que lorsque plus de 25% des blocs sont 'dirty', alors le checkpoint est considéré comme trop coûteux (et vous pouvez voir ce coût avec le wait event '*enq: RO - fast object reuse*').

Ce qu'il faut retenir, c'est que cette décision :

- N'est pas contrôlée par vous. C'est le comportement de l'application et des utilisateurs qui font que plus ou moins de blocs de la table sont en cache.
- Est indépendante d'Exadata. Ni l'optimiseur, ni le moteur d'exécution ne va faire un choix différent du fait qu'on est sur Exadata, et le fait que les '*direct path read*' sont plus efficaces sur cette plateforme n'est pas pris en compte.
- Même en Parallel Query, si vous êtes en '*parallel_degree_policy=auto*' la lecture peut se faire via buffer cache, et dans ce cas, sans aucun SmartScan.

Si vous voulez évaluer ce que peut-vous apporter Exadata SmartScan, vous ne pouvez pas le faire sans observer l'activité de votre base. Heureusement, on peut le voir dans un rapport AWR, et même avant de migrer.

A noter, c'est bien les '*direct path read*' qu'il faut regarder. Les '*direct path read temp*' lisent les tempfiles, qui ne sont pas sujets aux SmartScan.

A retenir : Exadata SmartScan n'accélère que les '*direct path read*'. Et le gain espéré ne peut pas aller au-delà du '% DB Time' associé à cet event.

Vous savez maintenant estimer si le SmartScan s'appliquerait sur l'activité de votre base de données. C'est intéressant d'avoir ces chiffres avant de réfléchir à tester ou à migrer vers Exadata. Dans la deuxième partie, nous verrons en détail ce qui peut être réellement filtré par SmartScan.

Franck Pachot
franck.pachot@dbi-services.com

Franck Pachot est consultant, formateur, et technology leader Oracle à dbi services, Oracle ACE et OCM 11g.

Conférence DOAG 2,015 - croissance régulière

La conférence DOAG continue de croître. Des présentations intéressantes et la légendaire soirée suisse ... Une source d'inspiration pour réserver dès maintenant la semaine de la 2016e édition (15. - 18. Nov.). Voici quelques impressions:

Merci à dbi services pour les Images.

Newbies CH

Pierre-Jean Giraud, Giraud Adequations
Daniel Meienberg, Diso AG
Ivan Korac, Altran
Mariusz Zuk, upc cablecom

Mentions légales

secrétariat SOUG:
Dornachstrasse 192, 4053 Basel
Tel.: 061 367 93 30, Fax: 061 367 93 31
sekretariat@soug.ch

rédaktion:
Geatano Bisaz
gaetano.bisaz@soug.ch

réalisation / DTP:
DOAG Dienstleistungen GmbH
Tempelhofer Weg 64, 12347 Berlin
office@doag.org

impression:
Druckerei Rindt GmbH & Co. KG
www.rindt-druck.de

rédaction Newsletter:
nl@soug.ch

inscription aux événements SOUG:
event@soug.ch

questions des membres:
sekretariat@soug.ch

site du web:
www.soug.ch

Photo couverture: © Mimi Potter / fotolia.com