Édition numéro 5 pour la Romandie

DOAG SOUG News

Article

Les bases BI sont-elles différentes?

SOUG SIG

Les adhérents du SOUG se sont réunis le 10 septembre 2015

Geatano Bisaz
SOUG

Chers membres du SOUG,

En informatique, la notion d' "Analyse" s'est, à l'origine, principalement concentrée sur la définition de processus métier qui furent ensuite retranscrits en programmes utilisant des bases de données. Par la suite l' "Analyse" s'est rapprochée du code, restant ainsi liée à la programmation à proprement parler. Aujourd'hui, la notion ne se restreint plus au développement informatique et s'utilise largement dans le domaine de l'informatique décisionnelle (business intelligence, BI). Mais, me direz-vous, la business intelligence, n'est-ce pas ce monde de "costume-cravate" dont "nous", informaticiens, nous méfions depuis toujours? Cela signifie donc que le développement de bases de données ne fait pas qu'améliorer les processus existants, mais fourni également des données pour l'ensemble des activités de l'entreprise, en considérant d'autres sources de données, qui seront représentées par des tableaux de bord … Assistons-nous à l'émergence d'un autre monde? L'informatique décisionnelle a cependant commencé à se répandre dès le milieu des années 1990. L'article de Franck Pachot dans cette édition nous présente les spécificités des environnements à bases de données décisionnelles – chose qui nous sera fort utile pour votre prochain projet en informatique décisionnelle.

Pour vous donner un avant-goût de la chose, nous vous recommandons les huit sympathiques petits clips présentant les avantages de l'informatique décisionnelle élaborés par Oracle, disponibles sur notre page Youtube: *http://tinyurl.com/pp6dotn*. Personnellement, c'est le clip avec Jim qui me plaît le plus.

Ces clips vous auront-ils inspirés, au point de vous précipiter au marketing et lancer une production vous-même? La réussite du projet pourrait très bien être présentée au moyen d'un tableau de bord décisionnel. N'hésitez pas à nous transmettre votre clip et votre tableau de bord, pour que nous puissions les présenter dans une des prochaines éditions, ou bien votre expérience en matière de BI de manière générale. nl@soug.ch

Gaetano Bisaz

Agenda

10.11.2015
SOUG SIG Romande
BI et des licences Oracle
sekretariat@soug.ch

17.-20.11.2015
DOAG 2015 Conférence + exposition

18.11.2015
Soirée Suisse à la conférence DOAG

Les bases BI sont-elles différentes?

Franck Pachot, dbi-services

Il y a souvent des difficultés dans la communication entre les équipes d'exploitation et les équipes de développement. Les enjeux ne sont pas les mêmes: les uns ont pour mission de stabiliser le système, les autres au contraire de le faire évoluer. Ces incompréhensions sont encore plus fortes avec les équipes BI, car les bases BI ont des besoins très différents des applications traditionnelles. En expliquant ces différences, j'espère amener à une meilleure compréhension entre les équipes. C'est aussi l'occasion de parler des technologies récentes qui adressent les besoins BI: Exadata, In-Memory, réplication temps réel,...

Business Intelligence

C'est le nom qu'on utilise aujourd'hui, et il peut paraître pompeux. Les premières bases BI sur lesquelles j'ai travaillé – en 1996 – nous les appelions modestement 'infocentre'. Il s'agissait simplement de récupérer des informations venant des différentes bases opérationnelles. Ces bases étaient interrogées par un outil permettant à l'utilisateur de construire des requêtes à la demande (Business Objects), il fallait donc charger les données dans un modèle adapté à cela. Dès qu'on a ouvert cette possibilité aux utilisateurs, les besoins se sont multipliés: besoins de plus de données, de plus d'historique, etc. Les besoins de reporting adressés d'habitude par les applications ont aussi été transférés sur ces projets, du fait de la simplicité des outils.

On a très vite appelé ça 'entrepôt de données' (datawarehouse) vu la consolidation de toutes les données de l'entreprise. Puis 'systèmes d'aide à la décision' (DSS – Decision Support System) ou décisionnel, vu les nouveaux groupes d'utilisateurs. Lorsque les outils ont intégré de plus en plus cette aide à la décision, on a commencé à parler de Business Intelligence. On parle aussi de requêtes analytiques.

Volumes de données

Ce qui caractérise en premier lieu les bases datawarehouse, c'est leur volume. Les informations viennent de tout le système d'information, sont historisées, sont souvent stockées plusieurs fois pour différents besoins de reporting, sont dénormalisées pour éviter de refaire plusieurs fois les mêmes jointures. En 8i, Oracle a introduit dans sa documentation un volume 'Data Warehousing Guide' qui a très vite été réaménagé pour se scinder en deux, les fonctionnalités liées au volume (vues matérialisées, parallel query, partitioning) sont décrites dans le VLDB – Very Large Database – Guide.

Beaucoup de ces notions sont devenues courantes aujourd'hui. On partitionne les tables, si on est en Enterprise Edition avec option partitioning. On met en place des technologies de backup adaptées à ces volumes (snapshots offerts par le stockage). Les DBA sont habitués à administrer des bases de plusieurs TB (terabytes).

Tablespace TEMP

Mais il y a d'autres points qui sont plus difficiles à accepter pour les DBA. Lorsque l'équipe BI demande d'agrandir le tablespace TEMP, on négocie la taille en GB. Mais pour un datawarehouse, c'est trop faible. Le tablespace temporaire sert à faire plusieurs passes pour les tris et jointures qui ne logent pas en mémoire. Si on a seulement 200 MB de données à trier, on n'a pas besoin de tempfiles pour ça. On doit pouvoir le faire en PGA. En datawarehouse, on manipule de dizaines de GB. Lorsqu'on fait un gros chargement de données, on va reconstruire les index. Il faudra pouvoir trier les données indexées. On va compter en dizaines de GB, voire centaine.

Quelle taille de TEMP est raisonnable? Regardez le plus gros index, regardez la plus grosse table de dimension, regardez la plus grosse partition des tables de faits, et ça vous donnera une idée. Vous aurez besoin de faire des tris, jointure par table de hachage, de mettre en buffer, sur ces volumes, et vous voulez le faire en une passe.

Evidemment, si vous partitionnez les tables et n'avez que des 'local index', vous limitez le volume nécessaire. Pensez à partitionner sur les mêmes colonnes les tables de faits qui vont participer à une jointure. Le 'partition wise join' permet de limiter la taille du hash area nécessaire.

Par contre, les chargements du datawarehouse, et peut-être certains reportings, vont faire du parallel query. N'oubliez pas que vous aurez besoin de beaucoup plus de TEMP pour deux raisons : plus de session en parallèle et plus de buffering entre opérations.

Cependant, n'allez pas augmenter le tablespace TEMP dès qu'il y a une requête qui se plante. C'est parfois le symptôme d'un mauvais plan d'exécution. Lorsqu'une requête se plan-

te pour cette raison, il faut aller voir son plan d'exécution. Si vous avez Tuning Pack, alors SQL Monitoring permet de voir le volume de TEMP en temps réel, directement dans le plan d'exécution. Sinon, c'est les V$SQL_WORKAREA.

PGA aggregate target

Bien sûr, avant de remplir le TEMP, il faut utiliser au maximum la mémoire. En datawarehouse, on n'a pas besoin d'une grosse SGA. On ne s'attend pas à garder les tables en buffer cache, sauf peut-être les petites dimensions, et les index bitmap. On n'a pas besoin d'une grande shared pool, sauf si on a un ETL écrit en pl/sql.

Donc il faut affecter un maximum de mémoire aux sessions pour la PGA. Si vous êtes en AMM (MEMORY_TARGET) ces zones vont s'adapter en fonction du besoin. Mais lorsque la mémoire se compte en dizaine de GB, il faut préférer l'utilisation des hugepages, et on doit définir explicitement SGA_TARGET et PGA_AGGREGATE_TARGET

De toute façon, la gestion automatique des workareas, avec PGA_AGGREGATE TARGET, est une fonctionnalité faite pour l'OLTP. Il s'agit de répartir la mémoire aux différentes sessions. Le chargement datawarehouse (inserts en direct-path, rebuild d'index, etc) va souvent se faire en une seule session. Alors si vous n'êtes pas en parallel query, les tailles de workarea calculées par PGA_AGGREGATE TARGET vont être trop petites.

La solution, c'est ALTER SESSION SET WORKAREA_SIZE_POLICY=MANUAL et définir manuellement une SORT_AREA_SIZE assez grande. Non, ce n'est pas un retour à la préhistoire. C'est juste que si vous avez une seule session, et que vous êtes le seul sur la machine, alors vous voulez utiliser toutes les ressources.

DDL

Et oui, les bonnes pratiques de l'OLTP ne s'appliquent pas au datawarehouse. Si faire du DDL en OLTP (drop/create/alter) doit être prohibé, il est tout à fait normal d'en voir passer lors du chargement quotidien du datawarehouse. Là encore, il faut garder à l'esprit que lors du chargement du datawarehouse il n'y a pas d'autres utilisateurs. Pas de problème si tous les curseurs sont invalidés. Ce qu'on veut c'est optimiser ce chargement. Si cette idée vous gêne, alors vous devez considérer le chargement quotidien comme si c'était une release applicative, avec une reprise de données. La base n'est pas ouverte aux utilisateurs et on essaie de leur mettre en place au plus vite la nouvelle version.

Donc les opérations réservées à l'administration de la base deviennent des opérations courantes en datawarehouse, et automatiques. Il va falloir l'accepter, et donner les privilèges nécessaires à l'équipe BI. En général la base est dédiée, et on conseille de modulariser les tables en plusieurs schemas, alors on peut même devoir accepter des droits 'ANY'.

Et c'est votre intérêt en tant que DBA. Vous ne voulez probablement pas voir passer des milliards de delete/insert sur la base toutes les nuits? Alors il faut donner les moyens de travailler sur des gros volumes de données. RMAN duplicate, transportable tablespace, snapshot standby,... il est tout à fait possible de les programmer toutes les nuits pour alimenter l'ODS. Les ETL sont fait pour gérer les centaines de flux et règles métiers. Pas pour le chargement massif.

Et attention, cette automatisation, ce n'est pas de simples scripts qui déroulent les commandes. La disponibilité et la cohérence des données dépendent de leur bon fonctionnement. Cette automatisation est un réel développement: versioning, tests, monitoring, évolution. Alors qui doit le faire? Très souvent, ce sont les DBA qui font un script vite fait et qu'ils fournissent à l'équipe BI. Et plus personne ne veut le maintenir ensuite. Les DBA n'ont pas le temps ni les outils pour faire du développement. L'équipe BI n'a pas les droits, environnements de test, ni les connaissances des fonctionnalités DBA.

Quelle que soit l'équipe à laquelle il est rattaché, il faut avoir un DBA applicatif pour faire et maintenir ce travail. Avec toutes les connaissances de DBA et la possibilité de travailler en mode projet, guidé par le besoin métier.

Bind variables

Puisqu'on en est à oublier les bonnes pratiques du transactionnel, soyons clairs: pas de bind variables en datawarehouse. On ne veut pas partager les curseurs. Au contraire, on veut que chaque requête s'exécute avec le plan d'exécution idéal pour les paramètres spécifiques. Un reporting couvrant une journée d'activité ne doit pas partager le même plan que le rapport annuel. Peu importe si on passe quelques dizaines de millisecondes de plus à chercher le meilleur plan d'exécution : on ne lance pas 1000 requêtes par seconde comme en transactionnel. Si ces millisecondes permettent d'avoir en rapport en 5 secondes au lieu de 20 minutes, alors c'est le mode optimal.

Ce qui veut aussi dire qu'il faut fournir un maximum de statistiques à l'optimiseur : histogrammes, extended statistics, global statistics, etc. Les plans d'exécution sont beaucoup plus complexes dès qu'on fait du reporting, de l'analytique. Il y a plus de tables, plus de jointures, des selectivités très différentes. Il faut que les estimations de l'optimiseur soient correctes sans quoi il partira sur une plan d'exécution qui peut durer des heures.

Et puisqu'on parle de statistiques: vous devez arrêter le job de collecte automatique qui va tourner à 22 heures si vous avez gardé le paramétrage par défaut. A 22 heures, vous allez tomber sur des tables vides en cours de chargement. Avoir une estimation de cardinalité à zéro ligne est bien pire que de ne pas avoir de statistiques. Si vous voyez un plan d'exécution avec cardinalité estimée à 1, alors il y a probablement ce genre de problème. Les tables vidées et rechargées, on collecte les statistiques à la fin du chargement, avec dbms_stats. Pour être sûr que le job automatique n'y touche pas, on les verrouille avec lock_table_stats, puis le gather_table_stats se fait avec force=>true.

Paramétrage

Et puisqu'on est dans l'optimiseur, le niveau de dynamic sampling doit être au moins 4 sur une base BI. On veut des estimations précises et dès qu'une clause where est un peu complexe, les statistiques statiques ne peuvent faire qu'une mauvaise approximation. Il n'y a que le dynamic sampling qui peut déterminer précisément la sélectivité d'un prédicat complexe. En 12c le choix du niveau de dynamic sampling peut être adaptif. C'est intéressant pour les quelques requêtes analytiques qu'on fait sur une base OLTP, mais en BI, on veut des performances dès la première exécution d'une requête. Et le Adaptive Dynamic Sampling de la 12c est encore nouveau et pose parfois quelques problèmes.

Enfin, on sait tous que sur une application transactionnelle les performances sont critiques. Mais si vous doutez de l'importance des temps de réponse en BI, n'oubliez pas que vos directeurs n'iront jamais faire des saisies de masse dans votre ERP. Par contre, s'ils doivent attendre une heure un tableau de bord, ils n'auront pas une bonne image de la performance de l'IT…

Environnements de développement

Comme pour toute application informatique, on a besoin d'environnements pour développer et tester avant de mettre en production. Mais il y a là encore une différence où les normes définies doivent être adaptées à la BI.

En BI, inutile d'avoir un environnement de développement avec un faible volume. On ne développe pas des transactions. On développe des données. Le modèle, le chargement, le contrôle, et l'interrogation doit être adaptée aux données. Donc on a besoin d'un environnement de même volume que la production, et avec les données de production, alimenté avec les données de production. C'est là qu'on met au point une nouvelle version de datawarehouse.

Ensuite, il y a peu d'intérêt d'avoir une copie de la prod. Si l'équipe BI doit vérifier quelque chose en production, alors autant accéder directement à la prod. En OLTP on a peur de se retrouver avec des requêtes qui prennent toutes les ressources. Mais en BI, la base est prévue pour ça, c'est l'activité normale. Au sujet de la sécurité des données, je ne vois pas comment l'équipe BI peut travailler sur une base sur laquelle on aura fait du data masking. Tout le travail dépend des données. Et de toute façon, on ne devrait pas trouver des données sensibles détaillées dans une base BI. On stocke des dimensions et des mesures, pas des numéros de carte de crédit.

Modélisation

Le modèle physique de données est complètement différent de celui d'une application OLTP. Une base OLTP est optimisée pour les transactions: normalisation, jointures, accès par index, optimisation du buffer cache.

Une base BI est optimisée pour la consultation. On ne va pas refaire les jointures à chaque interrogation. On la fait une seule fois lors du chargement et on essaie de stocker tous les faits dans une table. C'est l'idée du modèle dimensionnel, souvent appelé 'étoile' car on a table de faits qui référence un certain nombre de dimensions. La table de fait ayant beaucoup de lignes, on essaie d'optimiser la taille des lignes: normalisation, compression, etc. Les tables de dimensions sont souvent plus petites et peuvent donc avoir beaucoup de colonnes : dénormalisation, redondance, etc.

La particularité, c'est qu'on ne fait plus de mises à jour une fois que c'est chargé. On peut alors utiliser des structures qu'on évite à tout prix en OLTP mais qui ont tout leur sens ici: les bitmap index. C'est une approche colonne. Comme les requêtes vont plutôt s'intéresser à beaucoup de lignes filtrées sur quelques colonnes, alors on indexe chaque colonne. Et l'avantage du bitmap index, c'est qu'on peut les combiner ensuite pour n'aller voir que les lignes qui nous intéressent dans la table de fait.

C'est aussi la raison pour laquelle en terme de paramétrage, on va définir STAR_TRANSFORMATION_ENABLED=TRUE car cette transformation va permettre justement d'utiliser de manière optimale ces index, en faisant un range scan sur chaque colonne.

Parce qu'elles ne font que grossir, et qu'elles contiennent un historique, les tables de fait sont partitionnées par range sur la date la plus significative (le requêtes font en sorte de filtrer sur cette date). Si vous n'avez pas l'option partitionnement, il est possible de faire des tables différentes et les requêtes se feront sur une vue 'union all'.

Requêtes

Les requêtes qu'on voir passer sur une base BI sont monstrueuses, et ne sont clairement pas optimales. Ce n'est pas une raison pour évoquer une incompétence des développeurs: ce ne sont pas eux qui font les requêtes.

Les requêtes sont générées par des outils, à partir de ce que choisit l'utilisateur. On n'y peut rien. On ne peut pas demander à une personne de l'IT de faire une requête à la demande à chaque besoin de sortir un rapport. La seule marge de manœuvre, c'est le modèle de données. Le challenge, c'est d'offrir un modèle de données qui permet de faire toutes les requêtes possibles, en garantissant que toutes ces requêtes qu'on ne connaît pas d'avance seront performantes et donneront un résultat pertinent.

Je ne dis pas ça pour accepter n'importe quoi sur vos serveurs. Mais il est clair que certaines requêtes liront beaucoup plus de lignes que nécessaires. Et lorsque c'est le cas, l'optimisation ne consiste pas à ré-écrire une requêtes, mais à repenser complètement le design des quelques tables utilisées.

Et donc, comme on ne contrôle pas ces requêtes, le DBA doit mettre en place de quoi contrôler l'utilisation des ressources : resource manager, instance caging, statement queuing, etc.

Parallel query

Si vous êtes en Enterprise Edition, vous avez la possibilité de faire du parallel query. Et vous devez le faire pour du data-

warehouse. Combien avez-vous de CPU? Vous payez des licences Oracle pour tous les cores. Alors utilisez les tous. Le chargement quotidien doit utiliser toutes les ressources de la machine puisqu'il n'y a rien d'autre qui tourne en même temps. N'oubliez-pas que pour faire des inserts en parallèle, il faut l'activer au niveau de la session (ALTER SESSION ENABLE PARALLEL DML). Et si vous avez plus de cores que d'utilisateurs qui lancent des requêtes en même temps, alors pourquoi ne pas tout utiliser? En 12c le Auto DOP fonctionne mieux qu'en 11g. Il est difficile de prévoir combien de requêtes vont tourner en même temps, alors c'est bien de laisser l'instance décider du degré de parallélisme.

Enterprise Edition, Exadata, In-Memory

Si vous regardez les nouvelles fonctionnalités qui sont arrivées depuis Oracle 7 vous verrez que la plupart concernent le datawarehouse. Oracle 7 était déjà presque parfait pour le transactionnel. C'est l'arrivée des datawarehouse, de l'analytique sur des gros volumes (Big Data), qui ont motivé la plupart des évolutions. Avant il n'y avait que Teradata, avec ses serveurs massivement parallèles, qui pouvait traiter des volumes impressionnants. Sur Oracle sont venu le partitionnement, les transportable tablespaces, pour pouvoir continuer à administrer ces volumes qui augmentent. Regardez toutes les nouveautés de l'optimiseur depuis le CBO: c'est pour ces requêtes analytiques générées. Les requêtes qu'on écrit pour l'OLTP n'ont pas besoin d'un optimiseur intelligent et adaptif. Parlez-en aux éditeurs d'ERP qui regrettent le RBO et essaie de faire la même chose en recommandant des valeurs extrêmes pour OPTIMIZER_INDEX_COST_ADJ.

Exadata, même si aujourd'hui on peut le considérer comme une solution pour consolider toutes les bases de données, n'avait qu'un but au départ: enlever le bottleneck de la bande passante i/o des full table scans des datawarehouse. SmartScan applique la where clause de ces full scans en amont de la CPU du serveur de base de données.

L'option In-Memory aujourd'hui est faite pour les bases OLTP (pas besoin d'index bitmap ni de modèle en étoile pour les requêtes analytiques) pour faire du reporting dessus. C'est pour faire de la BI sans avoir une base BI.

On ne peut donc pas ignorer les besoins spécifiques de la BI car c'est ça qui évolue le plus aujourd'hui.

Réplication

Le besoin qui se fait de plus en plus sentir aujourd'hui, c'est le besoin de reporting temps-réel. Fini la base à J-1, on a besoin d'analyser et de prendre des décisions sur les données actuelles. Il faut alors trouver une alternative au chargement quotidien. C'est la réplication.

Il y a deux types de réplication : physique et logique.

La réplication physique temps réel, c'est possible avec une standby en real-time apply : la standby continue à appliquer le redo reçu de la primaire, mais elle est ouverte en read-only pour le reporting. Il faut être en Enterprise Edition avec l'option Active Data Guard pour ça. Le seul inconvénient de la réplication physique, c'est qu'on a une base identique à la primaire. On ne peut pas garder plus d'historique, on ne peut pas partitionner ou indexer différemment.

Ce n'est pas encore possible en 12.1 mais on l'attend pour la 12.2 : la possibilité d'activer In-Memory sur une standby en Active Data Guard. Financièrement, c'est un cumul d'options, mais la fonctionnalité n'a pas d'équivalent : avec In-Memory, plus besoin de rajouter des index dédiés au reporting. On a une base complètement isolée de la production, avec les mêmes données, et disponible pour notre BI. Comme on ne doit plus développer et maintenir une base BI, les économies réalisées peuvent justifier l'achat de l'option.

L'autre solution, c'est la réplication logique. Oracle Golden Gate ou Dbvisit replicate. Dbvisit replicate est une solution simple à mettre en œuvre, et permet de travailler en mode incrémental (mode audit) utile lorsqu'on doit ensuite envoyer les données dans un modèle de données très différent (normalisé, compressé, etc). Oracle Golden Gate a l'avantage d'inclure Active Data Gard, donc on peut avoir à la fois une réplication physique et logique, pour couvrir les deux besoins différents.

Conclusion

L'idée de cet article est très ancienne, et me vient à l'esprit chaque fois que je fais l'intermédiaire entre les équipes BI et l'équipe DBA. Les besoins BI ont évolués très vite ces dernières années et nous devons faire évoluer l'administration de ces bases. Le risque si on ne le fait pas, alors la BI devra se tourner vers d'autres technologies, qui sont très prometteuses (Hadoop) mais n'ont pas encore la maturité de nos bases relationnelles.

Franck Pachot
franck.pachot@dbi-services.com

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

Les adhérents du SOUG se sont réunis le 10 septembre 2015

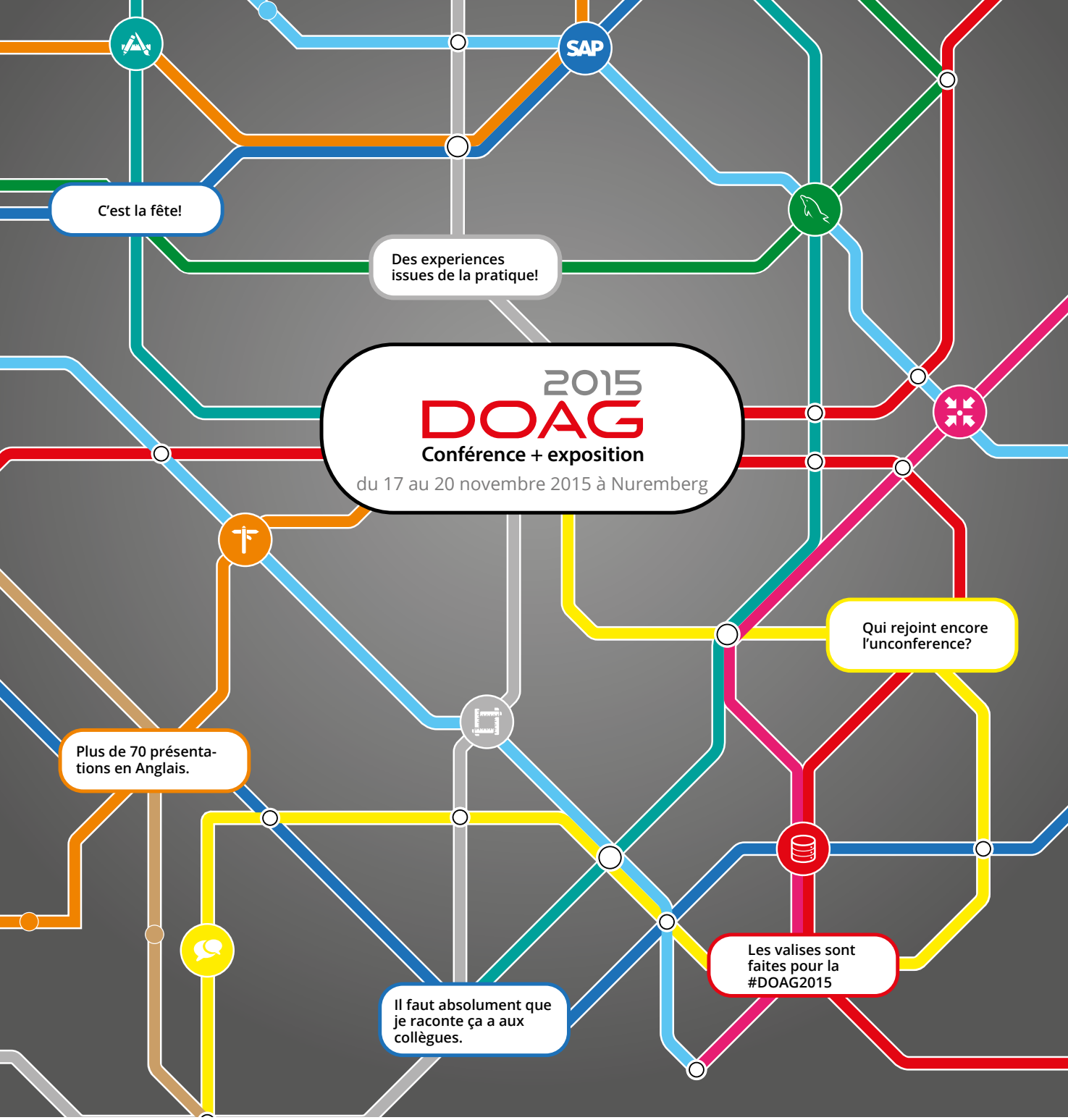
Les adhérents du SOUG se sont réunis le 10 septembre 2015, une troisième et dernière fois pour cette année. Les participants à cet évènement ont pu s'informer sur les thèmes suivants: "Développement et déploiement" ainsi que "SOA & mobile computing".

Les thèmes "Innovation dans le management de l'information avec les sémantiques" ou "XA & EAI Message Exchange Pattern" ont été traités pendant le stream "Développement et déploiement".

Au cours de l'exposé "APEX Interactive Reports", le référent a démontré, exemples à l'appui, l'élaboration de rapports APEX ainsi que les dernières innovations avec APEX 5.0. Le stream s'est fini par une présentation sur le déploiement via le Cloud Control.

Le stream "SOA & Mobile Computing" a démarré avec des présentations sur les thèmes "Microservices dans Oracle SOA Suite" et "Introduction à Java JMeter - stress tests pour des scénarios sous Internet Explorer". Après une courte pause café, était venu le moment d'aborder "Mobile Services" avec les thèmes suivants "Repenser le Mobile avec le Mobile Cloud Service d'Oracle" et "MAF - Une introduction au développement d'applications mobiles avec Oracle".

L'entreprise esentri AG a sponsorisé l'apéritif de clôture, ce dernier a permis des discussions sur les nouvelles et intéressantes informations abordées pendant la rencontre. Le réseautage entre collègues n'a bien entendu pas été négligé.



Newbies CH

Michael Krebs, esentri AG

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

mpression:
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: © 1xpert / fotolia.com