IBM Informix Warehouse Accelerator

Performance is everything

Keshava Murthy

Senior Technical Staff Member

IBM Informix Development

Translated to French by

Cyrille Deleruyelle and Frederic Delest

Imaginez que vos analystes de marché, vos responsables d'entreprise recevant leurs rapports en quelques minutes au lieu de quelques heures. Imaginez l'amélioration des capacitées d'analyse et de prévision. Imaginez les requêtes de vos « datawarehouse » s'exécutant plus rapidement sans avoir constamment à les surveiller et les optimiser. Imaginez faire tout cela sans avoir à construire de cubes, de tables de résumés, d'indexes, de statistiques ou d'implémenter des stratégies de partionnement. Après des tests exhaustifs de « Informix Warehouse Accelerator », Ashutosh Khunte, architecte sénior de base de données chez Sketchers affirme : " Avant l'utilisation de Informix Warehouse Accelerator, les requêtes complexes d'inventaire ou d'analyse de vente sur les magasins de l'entreprise contenant plus d'un milliard de rangées prenaient jusqu'à 45 minutes pour s'exécuter. Lorsque ces mêmes requêtes ont été passées sur Informix Warehouse Accelerator, elles se sont terminées en 2 à 4 secondes ! Ce qui signifie quelles ont été exécutées de 60 à 1400 fois plus rapidement, avec en moyenne un facteur multiplicateur de 450 ; le tout sans création d'indexes ou de cubes, sans optimiser les requêtes ou d'effectuer des changements dans l'application ! ".
Lester Knutsen de Advanced Data Tools, Informix Data Champion, a pris les données de son client le Département de l'Agriculture des Etats-Unis et a exécuté les traitements. Il affirme " Il [Informix Warehouse Accelerator] offre des performances très impressionnantes avec des requêtes s'exécutant 30 fois plus rapidement qu'avant. La technologie « columnar » permet d'économiser beaucoup de temps de traitement ; cela a permis de réduire notre charge de traitement de 9,5 heures à 15 minutes, sans faire l'optimisation de la base de données, ni augmenter la capacité de stockage.".
Thomas Gemesi, ATG IT consulting Gmbh affirme : "Informix Warehouse Accelerator (IWA) combine les capacités de datawarehousing et de l'OLTP (Online Transaction Processing) sur une même plateforme. Les requêtes peuvent être passées en quelques secondes sans avoir à investir dans des outils supplémentaires.".

L'évolution est permanente. Le besoin de rapidité est constant. Il est nécessaire de comprendre les schémas, prévoir les tendances, et de s'ajuster aux fluctuations du marché. L'analyse des données permet de comprendre les tendances et les évolutions du marché. Faire tout cela plus rapidement que les autres, en permanence et avec un TOC plus faible garantis un avantage sur la concurrence.

« Informix Warehouse Accelerator » fournit des performances exceptionnelles tout en supprimant la nécessité de l'optimisation requise pour le « datawarehouse » traditionnel. Il est conçu pour traiter de gros volumes de données en quelques secondes et pour mettre à disposition de l'entreprise les bonnes données au bon moment sans maintenance supplémentaire ou changement des infrastructures applicatives.

L'innovation est la clef. La différence de performance entre le processeur accédant à des données en mémoire en comparaison au processeur accédant à des données sur disque est en croissance. Les systèmes de base de données traditionnels choisissent des configurations mémoires faibles et optimisent les opérations pour minimiser les Entrées/Sorties sur le disque, en utilisant la bufférisassions pour conserver les données récemment accédée en mémoire. La chute des prix de la mémoire et l'apparition de système à plusieurs téraoctets de RAM à un prix abordable nécessite de repenser cette conception. IBM a des systèmes supportant 3TB et cette valeur est appelée à augmenter. L'accélérateur tire parti cette tendance en interceptant et en compressant toutes les données en mémoire, éliminant les Entrées/Sorties sur le disque. Les processeurs se voient doté de plus

de cœurs et de caches plus volumineux à chaque nouvelle version. L'accélérateur exploite cette tendance grâce à la parallélisation, en minimisant la synchronisation et en optimisant les algorithmes pour tirer partis des caches du processeur. Cela amène les performances vers de nouveau sommets grâce à des conceptions simples et élégantes.

La puissance de la simplicité

« La perfection est atteinte, non pas lorsqu'il n'y a plus rien à ajouter,
mais lorsqu'il n'y a plus rien à retirer. »
-- Antoine de Saint-Exupery

L'accélérateur est composé de 3 parties :

- Accélération sans optimisation manuelle pour chaque traitement
- Le Support des outils et applications existantes
- La capacité à travailler avec les infrastructures de « datawarehousing » existantes.

Si vous avez déjà de l'Informix dans votre environnement, l'accélérateur s'y insère. Pour les nouveaux déploiements, vous avez le choix d'une plateforme flexible et l'intégration ouverte avec un serveur de base de données Informix.

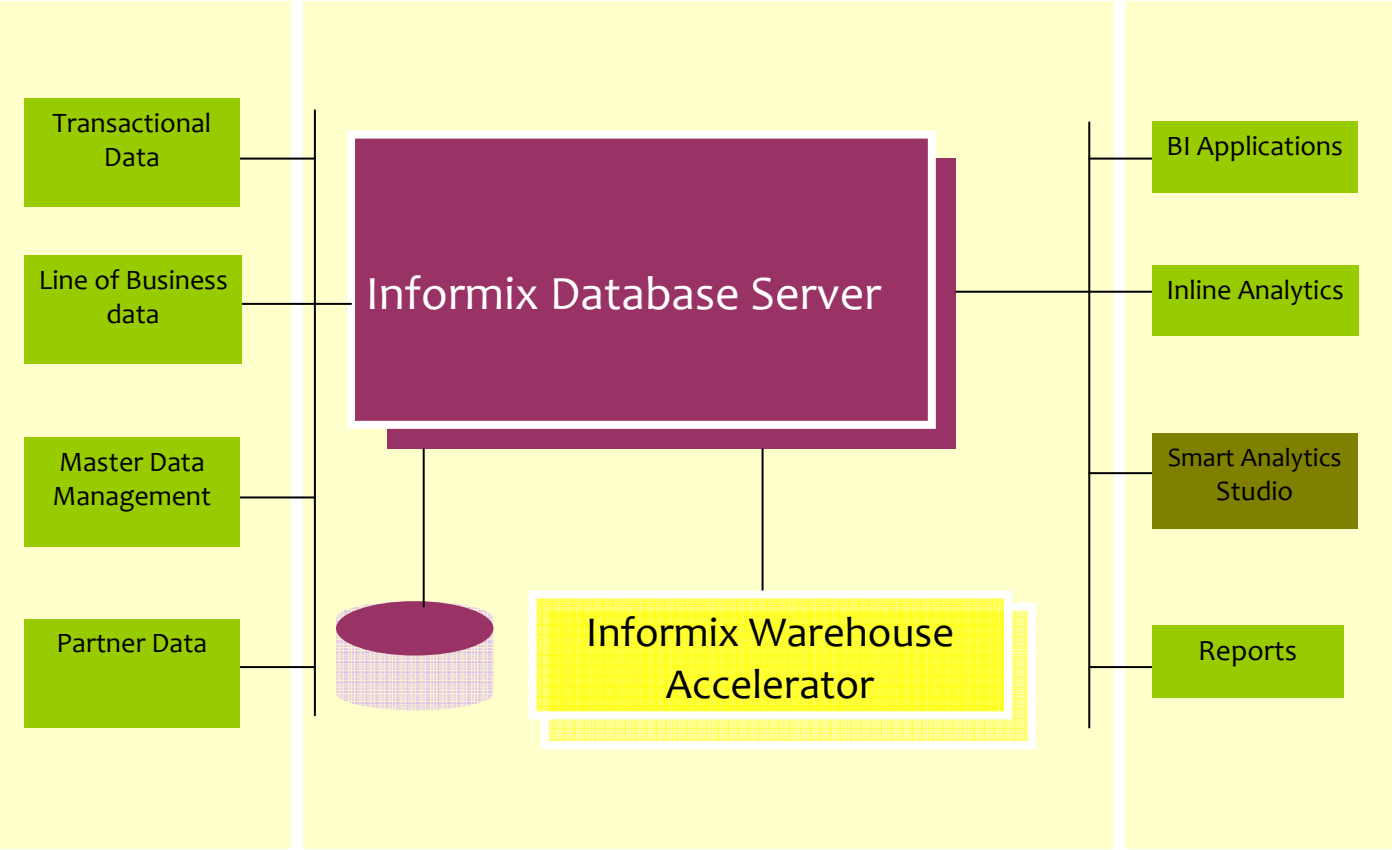
Il suffit juste de présenter les données à l'accélérateur ; il est déjà prêt à démontrer des performances de pointes.

Pas de création d'indexes, pas besoin « d'index advisor », pas de réorganisation d'index nécessaire. Le moteur d'exécution de requête de l'accélérateur lit des milliards de rangées en quelques secondes, voir millisecondes. La représentation en « deep columnar » des données, le traitement des requêtes sur des données compressées, les algorithmes innovants exploitant les fonctionnalités des processeurs récents, rendent caduques la nécessité des indexes.

Pas de statistiques à échantillonner. Les optimisateurs traditionnels dépendent de la collecte régulière de données statistiques pour créer de meilleurs plans d'accès. Avec l'accélérateur, l'ordre des jointures est déterminé automatiquement, les indexes en étoile sont utilisés de façon systématique. L'absence d'indexe réduiant la complexité et le temps de l'optimisation (explications plus complètes plus tard dans l'article) il n'y a plus besoin ni d'échantillonnage, ni d' « advisors ».

Pas de schéma de partitionnement à créer. Les données sont automatiquement partionnées à la fois verticalement et horizontalement. Les requêtes bénéficient aussi de l'élimination verticale et horizontale (aussi connu sous le terme d' « élimination de fragment ») grâce à un stockage en « deep columnar cells ». Ceci permet d'éliminer la planification de schéma de partitionnement .
Pas d'optimisation pour chaque requête ou montées en charges. Au cours de l'installation de l'accélérateur, vous fournissez les informations de base pour la configuration de la mémoire et du stockage. Des plans cohérents, l'élimination d'Entrée/Sortie disque, des jointures et des lectures rapides rendent caduques les optimisations à l'exécution.
Pas de gestion de stockage. Les données sont stockées en mémoire ; seule une copie de l'image en mémoire est faite sur disque. Il n'est pas nécessaire de planifier ni de créer des espaces de stockage pour les tables et les indexes.
Pas de matériel onéreux. L'accélérateur est conçu pour fonctionner sur du matériel courant et d'évoluer avec votre infrastructure. L'accélérateur tourne sur Linux/Intel. Il s'intègre avec un serveur Informix sur Linux/Intel, AIX/Power, HP-UX/Itanium, Solaris/Sparc.
Pas de changement sur la base de données. L'accélérateur utilise le schéma logique du « datawarehouse » existant.
Pas de changement applicatif. L'accélérateur s'intègre dans un serveur de base de données Informix en tant que ressource. Le serveur de base de données Informix connait les « data marts » qui sont stockés dans l'Accélérateur et y redirige automatiquement les requêtes éligibles. Ni vos applications ni vos outils ne requièrent de changements.
Pas de table de résumé ou de matérialisation de vue à créer. Les lectures et jointures de tables faites par l'accélérateur sont d'un ordre de grandeur plus rapide que ceux des serveurs de base de données traditionnels. Ceci supprime la nécessité de la création et de la maintenance de tables de résumés ainsi que l'utilisation des « advisors » correspondants.
Pas de taille de page ou de taille de block à configurer. La technologie « deep columnar » détermine et optimise automatiquement la taille des cellules mémoires.
Pas d'espace disque temporaire à allouer et à surveiller. Les résultats intermédiaires sont compressés et prennent moins d'espace en mémoire.
Pas d'indication d'optimisation pour les requêtes accélérée. L'optimisateur utilise systématiquement des plans à jointures en étoile. Le gestionnaire de requête de l'accélérateur ajuste les ordres de jointures en fonction des statistiques d'exécution.

L'architecture Informix Warehouse



Le serveur de base de données est scalable et hautement disponible. Il utilisés par des dizaines de milliers de clients pour assurer le fonctionnement critiques d'applications transactionnelles et analytiques. Le serveur de base de données Informix inclus des outils d'ETL, le support intégré de la gestion des données temporelles, des opérations à chaud, la compression en profondeur et les techniques de traitement et d'optimisation conçus pour les requêtes de décisionnelles complexes. Les clients Informix peuvent utiliser des outils de BI comme Cognos, Microstrategy, etc… , pour analyser et améliorer leurs marchés.

Traditionnellement, les requêtes de type décisionnel sont appelées complexes pour une bonne raison. Ces requêtes accèdent a des millions et des milliards de rangées ; les résultats intermédiaires sont volumineux ; les requêtes s'exécutent au mieux lorsqu'elles sont parallélisées. Le technique d'optimisation des jointures en étoile est conçue pour gérer cela au sein d'un système traditionnel. Seul un administrateur système expérimenté est nécessaire pour comprendre le traitement et optimiser les paramètres systèmes et les indexes pour accommoder la charge.

« Informix Warehouse Accelarator » permet une avancée des performances des requêtes décisionnelles, éliminant la nécessité d'optimiser la base de données ; tout cela sans qu'il soit nécessaire d'apporter des modifications aux applications existantes ou à l'environnement des outils. La

suite de ce document détaille la technologie de l'accélérateur, son utilisation et son intégration avec Informix.

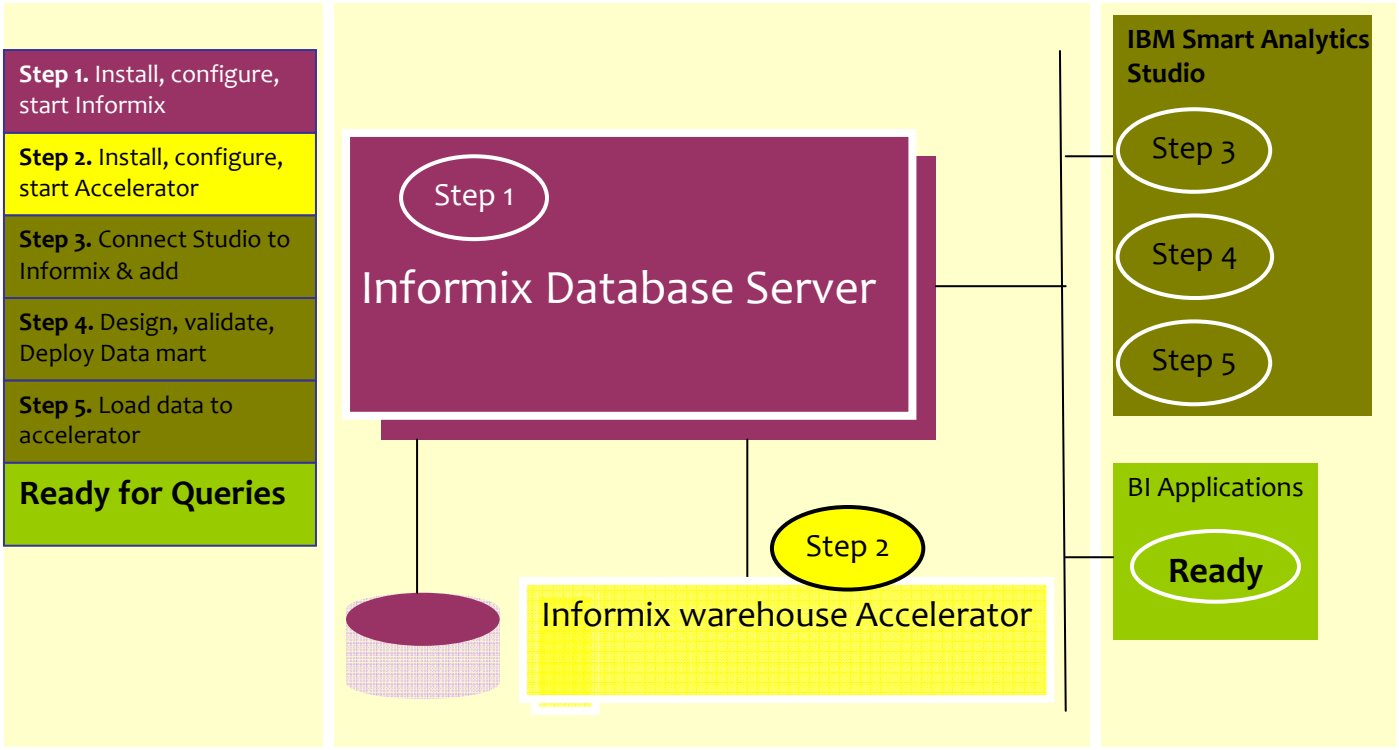
Déploiement d' « Informix Warehouse Accelerator »

L'accélérateur est conçu pour fonctionner sur un système linux de haute performance sur plateforme Intel. Le serveur de base de données Informix et l'accélérateur peuvent fonctionner sur un même serveur ou bien sur des serveurs différents. Le serveur Informix peut tourner sur Linux/Intel, AIX/Power, HP-UX/Itanium, Solaris/Sparc et l'Accélérateur tourne sur Linux/Intel.

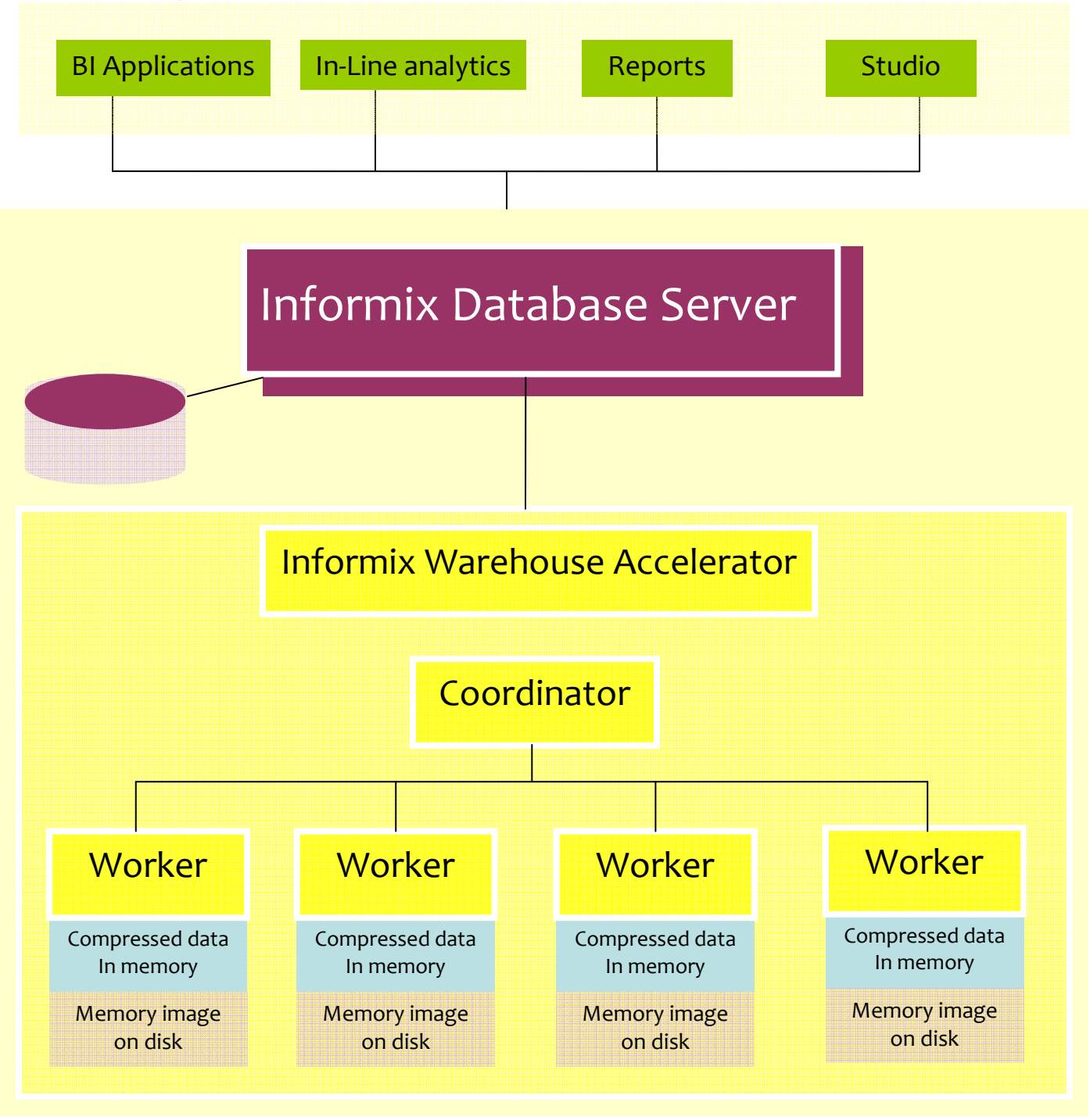
L'accélérateur est inclus dans Informix Ultimate Warehouse Edition (IUWE) qui comprend le serveur de base de données Informix (Ultimate Edition) et Informix Warehouse Accélérator. La distribution du produit inclus le binaire de l'accélérateur ainsi que l'IBM Smart Analytics Optimizer Studio, un outil conçus pour configurer et gérer l'accélérateur avec un serveur de base de données Informix. Ci-dessous se trouve la séquence d'opération à réaliser pour réussir un déploiement complet. Le produit inclut un guide de démarrage rapide et un guide de l'administration avec une documention fournie sur l'installation et la configuration d'Informix et de l'accélérateur. Il n'y a besoin que de quelques étapes simples de configuration : la localisation du système de fichier pour la sauvegarde des données, la quantité de mémoire et les ressources CPU. Les informations concernant la configuration seront présentées ultérieurement dans ce document ainsi que dans le guide de l'administration ?

L'IBM Smart Analytics Optimizer Studio est un outil basé sur Eclipse, utilisé pour la gestion de l'accélérateur, pour la définition et le déploiement des « data marts » d'Informix vers l'accélérateur. Le Studio est livré en 2 versions : une version pour Linux et une version pour Windows. Il est possible d'utiliser la version Linux sur la même machine que le serveur de base de données Informix, ou sur une machine différente. Pour utiliser la version Windows, il faut transférer le binaire auto-extractible vers une machine Windows et l'installer.

Les étapes pour le déploiement du serveur de base de données Informix et de l'accélérateur.


Step 1. Install, configure, start Informix
Step 2. Install, configure, start Accelerator
Step 3. Connect Studio to Informix & add
Step 4. Design, validate, Deploy Data mart
Step 5. Load data to accelerator
Ready for Queries
Step 1
Informix Database Server
Step 2
Informix warehouse Accelerator
IBM Smart Analytics Studio
Step 3
Step 4
Step 5
BI Applications
Ready

Les composants de l'accélérateur



L'accélérateur se connecte sur le serveur de base de données Informix au travers d'un réseau TCP/IP. Si l'accélérateur et le serveur de base de données se trouvent sur la même machine, ils communiquent au travers d'une connexion TCP/IP en « loopback ». L'accélérateur est composé de processus coordinateurs et sous-traitants. Ces 2 processus partagent la responsabilité du traitement de la requête. La configuration définie détermine le nombre de processus ainsi que la quantité de mémoire et de CPU utilisée.

L'ajout d'information se fait au travers de Studio après qu'Informix et l'accélérateur ont été démarrés (cf manuel pour les détails). Cette opération achevée, Informix ajoute les informations de connexion à son fichier SQLHOSTS. Les entrées de ce fichier ressembleront à l'exemple ci-dessous :

```
sales_acc      group   -- c=1,a=4b3f3f457d5f552b613b4c587551362d2776496f226e714d75217e22614742677b424224
sales_acc_1    dwsoctcp       127.0.0.1      21022   g=sales_acc
```

Ici, le nom de l'accélérateur est sales_acc. Informix créera un nouveau groupe avec ce nom. La valeur hexadécimale est le code d'authentification servant à s'assurer que seule la base de données Informix communique avec l'accélérateur sales_acc. Le nom du coordinateur est sales_acc_1. Le protocole dw est utilisé pour communiquer au-dessus TCP/IP. Ce protocole est très proche du protocole DRDA, optimisé pour les communications entre Informix et l'accélérateur. 127.0.0.1 (l'adresse TCP/IP de « loopback ») indique que l'accélérateur et le moteur de base de données fonctionnent sur la même machine. Avec un nombre important de sous-traitants, viennent de multiples coordinateurs pour gérer la reprise sur panne. Dans ces configurations, il y aura une entrée pour chaque coordinateur.

Le coordinateur a 3 fonctions importantes :

1- C'est le point principal de communication pour la base de données Informix. Le serveur de base de données se connecte au coordinateur pour lui envoyer des données ; c'est aussi le point contact principal pour envoyer la requête et récupérer le résultat.

2- Durant la phase de chargement des données (étape 8 [step 8]), il distribue les données parmi les différents sous-traitants. Le coordinateur recueille ensuite la totalité du dictionnaire de compression, l'intègre puis le distribue de telle sorte que tous utilisent le même dictionnaire de référence.

3- Pendant la phase de traitement des requêtes, le coordinateur reçoit des demandes en provenance du serveur de base de données, envoi la requête à chacun des sous-traitants, récupère les jeux de données intermédiaires, fusionne les groupes et ensuite décompresse les données si nécessaire avant d'envoyer les résultats finaux au serveur de base de données Informix.

Les processus sous-traitants ont deux fonctions importantes.

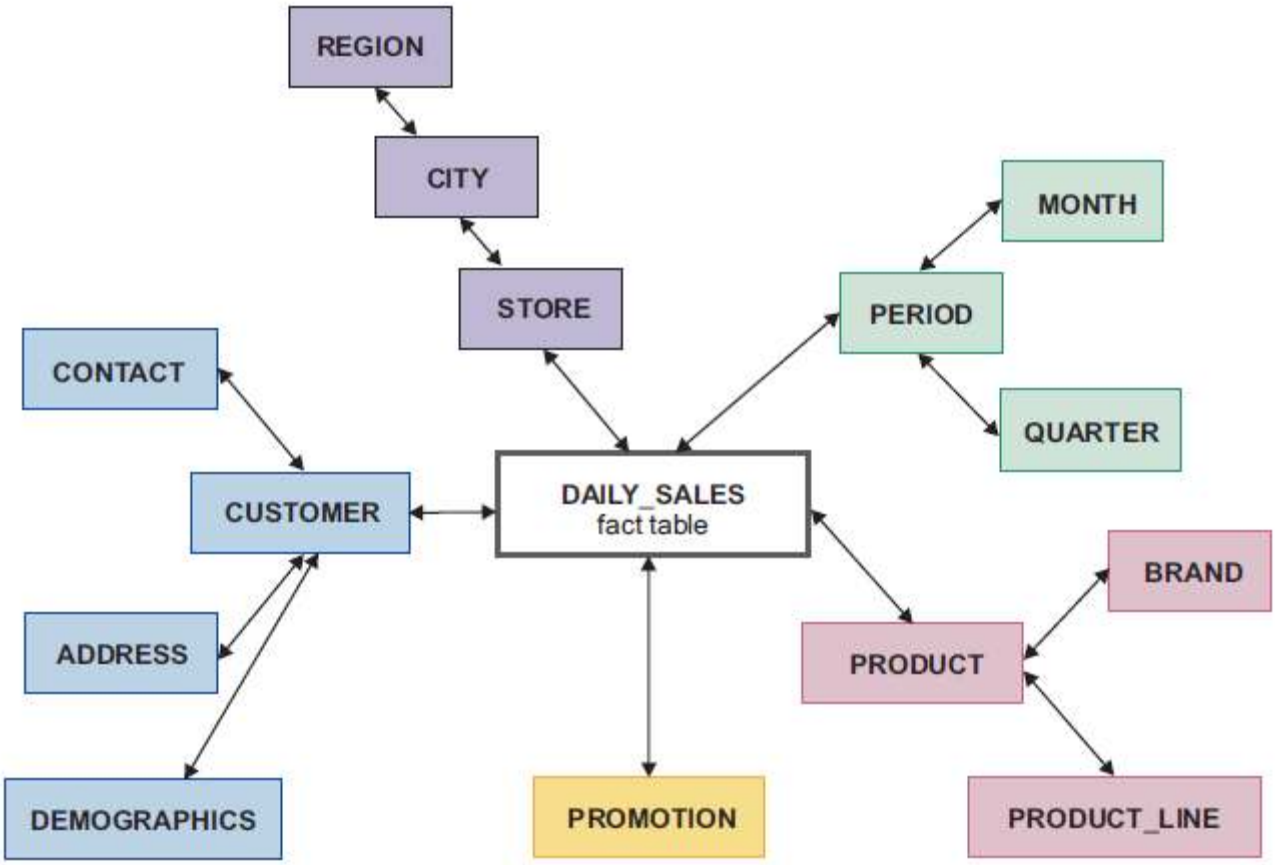
1- Au cours de la phase de chargement, chaque sous-traitant analyse les données arrivant en fonction du partitionnement basé sur les occurrences pour automatiquement répartir les données horizontalement et verticalement puis les compresser, grâce aux techniques « deep columnar ». Une fois que les données sont compressées, elles sont écrites sur disque afin

d'assurer la reprise sur panne. Nous reviendrons sur les techniques « deep columnar » ultérieurement dans cet article.

2- La 2ème fonction est le traitement de la requête sur des données compressées. Chaque sous-traitant maintient sa copie compressée des tables de dimension et une portion de la table de fait. Chaque sous-traitant renvoi les résultats intermédiaires au coordinateur. Le traitement de la requête est entièrement fait en mémoire.

Les coordinateurs et les sous-traitants travaillent ensemble en parallélisant l'exécution de toutes les requêtes pour fournir des résultats rapidement.

Wikipédia définie le « Data Mart » comme étant un sous ensemble d'un « datawarehouse », souvent ayant trait à une ligne commerciale ou un service de l'entreprise. Nous utiliserons cette définition dans l'article. Le « datawarehouse » de l'entreprise peut contenir toute sorte d'information, allant des données commerciales, de l'inventaire, du service client jusqu'aux données du marché. A partir ce cela, il est possible d'accélérer les « data marts » pour fournir une haute valeur ajoutée. Par exemple, le responsable des ventes peut vouloir analyser les ventes et l'inventaire pour comprendre les tendances et créer les promotions adaptées. Par conséquent, il y a seulement besoin d'accélérer les « data marts » correspondant aux tables de fait : ventes et inventaire. Chaque schéma en étoile est composé d'une table de fait et des tables de dimension correspondantes. Dans le schéma en étoile ci-dessous, DAILY_SALES est la table de fait et est en relation avec les dimensions qui décrivent les faits.



Les paragraphes suivants expliquent les étapes allant de la conception à l'utilisation de l'accélérateur. Ces étapes font référence au diagramme *« Les étapes pour le déploiement du serveur de base de données Informix et de l'accélérateur »*.

IBM Smart Analytics Optimizer Studio est utilisé pour la **conception** et le déploiement de « data marts ». Ils sont défini par une table de fait et des tables de dimension ainsi que leurs relations. Généralement, les tables de dimension contiennent beaucoup moins de rangées que les tables de fait ; ici, « product information » et « customer information ». Dans certains cas, les tables de dimension peuvent être relativement volumineuses, par exemple : « California residents ». Une fois identifié les tables nécessaires à la création d'un « data mart », il reste à définir les relations entre les tables de fait et de dimension.

L'étape de **validation** du « data mart » garantit que toutes les relations entre les tables sont définies. Il est nécessaire de traiter toutes les erreurs apparues pendant cette étape avant de **déployer** le « data mart ». Lors de l'étape de déploiement, IBM Smart Analytics Optimizer Studio envoit les définitions du « data mart » au format XML à l'accélérateur qui renvoit la définition en SQL. Elle est sauvegardée en tant que vue dans les catalogues Informix avec un indicateur spécifique ainsi que les informations correspondantes. Cette vue est une « Accelerated Query Table (AQT) ». Cette AQT est ensuite utilisée pour identifier les requêtes correspondantes et les envoyer vers l'accélérateur. Voici un exemple d'AQT avec « daily_sales » en tant que table de fait et ses dimensions : « product », « stores », « customer » et « promotion ». Elle est créée automatiquement et affichée ici à titre d'information. Ni les utilisateurs ni le DBA utilisent cette vue. Le SQL en provenance des applications ou des outils, fonctionne normalement en utilisant les tables définies dans le schéma.

```
create view "dwa"."aqt2dbca0d9-509d-434b-9cc9-4a12c6de6b3d"
("COL16","COL17","COL18","COL19","COL20","COL21","COL22","COL23","COL24","COL25","COL2
6","COL27","COL28","COL29","COL30","COL31","COL32","COL33","COL34","COL35","COL36","CO
L37","COL38","COL39","COL40","COL41","COL42","COL43","COL44","COL45","COL46","COL47","
COL07","COL08","COL09","COL10","COL11","COL12","COL13","COL14","COL15","COL48","COL49"
,"COL50","COL51","COL52","COL53","COL54","COL55","COL56","COL57","COL58","COL59","COL6
0","COL61","COL01","COL02","COL03","COL04","COL05","COL06","COL62","COL63","COL64","CO
L65","COL66","COL67","COL68","COL69","COL70","COL71","COL72","COL73","COL74","COL75","
COL76","COL77","COL78","COL79","COL80","COL81") as
  select x0.perkey ,x0.storekey ,x0.custkey ,x0.prodkey ,x0.promokey
    ,x0.quantity_sold ,x0.extended_price ,x0.extended_cost ,x0.shelf_location
    ,x0.shelf_number ,x0.start_shelf_date ,x0.shelf_height ,x0.shelf_width
    ,x0.shelf_depth ,x0.shelf_cost ,x0.shelf_cost_pct_of_sale
    ,x0.bin_number ,x0.product_per_bin ,x0.start_bin_date ,x0.bin_height
    ,x0.bin_width ,x0.bin_depth ,x0.bin_cost ,x0.bin_cost_pct_of_sale
    ,x0.trans_number ,x0.handling_charge ,x0.upc ,x0.shipping
    ,x0.tax ,x0.percent_discount ,x0.total_display_cost ,x0.total_discount
    ,x1.perkey ,x1.calendar_date ,x1.day_of_week ,x1.week ,x1.period
    ,x1."year" ,x1.holiday_flag ,x1.week_ending_date ,x1."month"
     ,x2.prodkey ,x2.upc_number ,x2.package_type ,x2.flavor ,
    x2.form ,x2.category ,x2.sub_category ,x2.case_pack ,x2.package_size
    ,x2.item_desc ,x2.p_price ,x2.category_desc ,x2.p_cost ,x2.sub_category_desc
    ,x3.storekey ,x3.store_number ,x3.city ,x3.state ,x3.district
```

```
    ,x3.region ,x4.custkey ,x4."name" ,x4."address" ,x4.c_city
    ,x4.c_state ,x4.zip ,x4.phone ,x4.age_level ,x4.age_level_desc
    ,x4.income_level ,x4.income_level_desc ,x4.marital_status
    ,x4.gender ,x4.discount ,x5.promokey ,x5.promotype ,x5.promodesc
    ,x5.promovalue ,x5.promovalue2 ,x5.promo_cost
from
 (((((("informix".daily_sales x0 left join "informix".period x1 on (x0.perkey
    = x1.perkey ) )left join "informix".product x2 on (x0.prodkey
    = x2.prodkey ) )left join "informix"."store" x3 on (x0.storekey
    = x3.storekey ) )left join "informix".customer x4 on (x0.custkey
    = x4.custkey ) )left join "informix".promotion x5 on (x0.promokey
    = x5.promokey ) );
```

Durant l'étape de chargement, une image des données des tables du serveur de base de données est envoyée vers l'accélérateur. C'est dans cette phase qu'il distribue les données à ses sous-traitants. Ceux-ci analysent les données à la recherche de valeur d'occurrence en fréquence, ainsi que des relations entre les colonnes puis procède au partitionnement vertical et horizontal. Lorsque celui-ci est établis, les données sont compressées en utilisant le processus « deep columnar » décrit ultérieurement dans l'article. Il est à noter que dans cette phase, les données sont compressées et conservées en mémoire, alors qu'une copie est écrite sur disque pour la persistance. Aucun indexe n'est créé, ni de tables de résumé ou de cubes. Les données peuvent être rafraîchies périodiquement (par exemple chaque nuit) à partir du serveur de base de données Informix.

Après le chargement, le « data mart » est prêt à être utilisé dans l'accélérateur. Il n'y a juste qu'à l'accélérateur avec les données et constater immédiatement des performances de pointe. La fonctionnalité Informix Warehouse Accelerator inclus un utilitaire en ligne de commande pour la conception, la validation, le déploiement et le chargement, ce qui est très utile pour l'automatisation à l'aide de programmation en script.

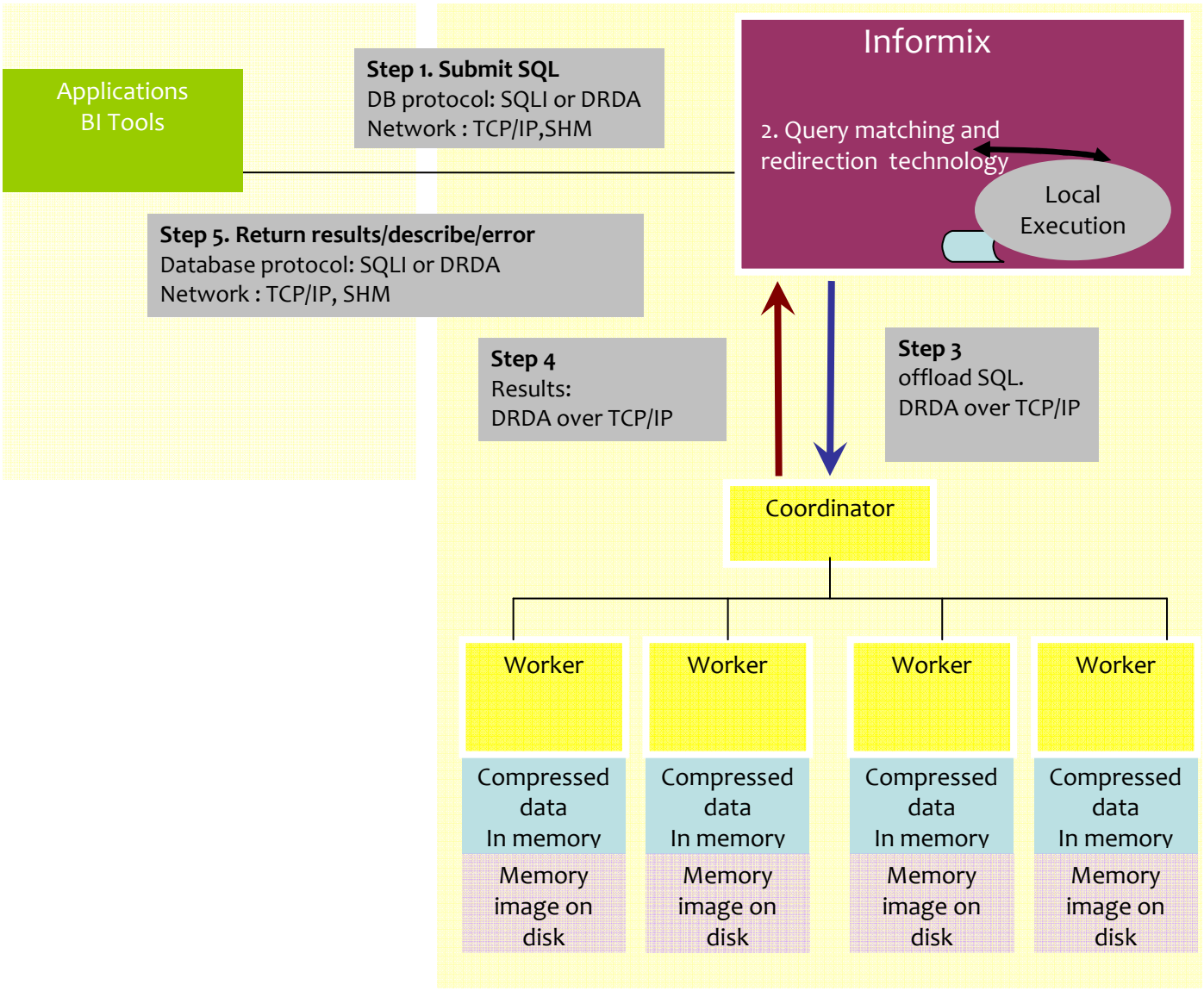
Typiquement, les requêtes SQL font la jointure entre la table de fait et une ou plusieurs tables de dimension puis repère des motifs spécifiques parmi les données. Dans l'exemple ci-dessous, une jointure est faite entre la table de fait web_sales et 4 autres tables de dimension. Si l'on a créé et déployé un « data mart » qui utilise ces tables, Informix fera la correspondance entre la requête et une définition d'une vue spécifique du « data mart » (AQT) et enverra donc la requête à l'accélérateur. Cela fonctionne comme une requête distribuée exécutée d'un serveur Informix sur un autre. Le résultat est retourné en utilisant la même connexion et est alors renvoyé à l'application cliente. Elle l'obtient ainsi de façon transparente mais aussi beaucoup plus rapidement.

```
select first 100 i_item_id,
        avg(ws_quantity) avg_quantity,
        avg(ws_list_price) avg_list_price,
        avg(ws_coupon_amt) avg_coupton_amt,
        sum(ws_sales_price) sum_sales_price
 from web_sales, customer_demographics, date_dim, item, promotion
 where ws_sold_date_sk = d_date_sk and
       ws_item_sk = i_item_sk and
```

```
        ws_bill_cdemo_sk = cd_demo_sk and
        ws_promo_sk = p_promo_sk and
        cd_gender = 'F' and
        cd_marital_status = 'M' and
        cd_education_status = 'College' and
        (p_channel_email = 'N' or p_channel_event = 'N') and
        d_year = 2001
 group by i_item_id;
 order by sum(ws_sales_price) desc;
```

L'accélérateur aura besoin que les requêtes SQL fassent la jointure entre la table de fait et zéro ou plusieurs tables de dimension et fasse la jointure en utilisant les clefs de jointure spécifiées lors de la définition des relations dans le « data mart ». Les jointures « inner join » et « left join » sont supportées avec la table de fait côté dominant. Le guide d'administration Informix donne plus de détails sur l'éligibilité des requêtes. L'accélérateur exécute les requêtes sans interruption, sur la base du premier arrivé, premier servi. Toutes les données et les résultats intermédiaires sont conservés en mémoire. Chaque sous-traitant a une copie de la table de dimension avec laquelle la jointure est faite. Chaque thread de jointure tente de mettre en cache la table de hachage dans le tampon L2 pour optimiser les accès à la mémoire. Les sous-traitants scannent les données et font les jointures indépendamment les uns des autres avec un minimum d'échanges d'information et de synchronisation avec les autres sous-traitants. On constate habituellement des temps d'exécution des requêtes de l'ordre de quelques secondes contre plusieurs minutes ou quelques heures avec un système traditionnel. Par conséquent, l'accélérateur exécute les requêtes de façon successive après les avoir mises en attente dans une file d'attente.

Remarques sur la configuration



Remarques sur la configuration

Lors de l'installation d'« Informix Warehouse Accelerator », vous aurez à configurer le nombre de nœud (coordinateurs, sous-traitants), la mémoire disponible pour les sous-traitants et celle disponible pour les coordinateurs. Le nombre de coordinateurs et de sous-traitants sera déterminé automatiquement. Par exemple, définir 5 nœuds créera automatiquement 1 coordinateur et 4 sous-traitants. Prenons l'exemple du déploiement d'un « data mart » composé d'une table de fait "sales" et des tables de dimension "customer", "store" et "time". Ces tables de dimension seront compressées et conservées dans une portion de mémoire privée de chaque sous-traitant. Il y aura donc 4 copies des tables de dimension ; les rangées de la table de fait "sales" seront distribuées uniformément parmi les 3 sous-

traitants. Chacun d'eux maintiendra à jour les tables de dimension "customer", "store" et "time" ainsi que 25% des rangées de la table de fait – "sales" dans notre cas – dans la mémoire principale.

Le débit de transfert des données augmente avec l'accroissement du nombre de sous-traitants, en supposant qu'il y ait assez de puissance CPU. Un plus grand nombre de sous-traitants permettra d'améliorer la rapidité d'exécution des requêtes, bien que cela ne soit pas spectaculaire. Ce gain de performance dépend aussi de la requête traitée. (*)

Ceci étant dit, combien de mémoire faut-il allouer pour chaque sous-traitant ? Quelles sont les besoins en mémoire du système ?

Comparativement, il a généralement été constaté un ratio de compression de 3 pour 1 sur les données Informix non compressées par rapport à celles compressées en mémoire d'Informix Warehouse Accelerator. En supposant que la taille totale de la table de fait "sales" et des tables de dimension "customer", "store", "time" soit équivalente à 100GB et que la majeure partie est consommée par la table "sales", 33GB de mémoire sera nécessaire pour le stockage les données. Il est facile de connaître la taille des tables en utilisant Open Admin Tool (OAT) ou en consultant directement les tables du catalogue.

Au cours de l'exécution, chaque sous-traitant a besoin de suffisamment de mémoire pour stocker les résultats intermédiaires. La mémoire est allouée et libérée dynamiquement à chaque requête. Cette gestion est similaire à celle de l'espace temporaire avec Informix. Comment évaluer le besoin pour le stockage des résultats intermédiaires et les tris ? Quelles sont les corrélations entre les données, combien de groupes seront nécessaires dans chaque catégorie ? Bien que cela soit difficile à dire précisément, nous avons constaté qu'une configuration allant d'un-cinquième à un-tiers de mémoire en plus de le taille des données est généralement suffisant.

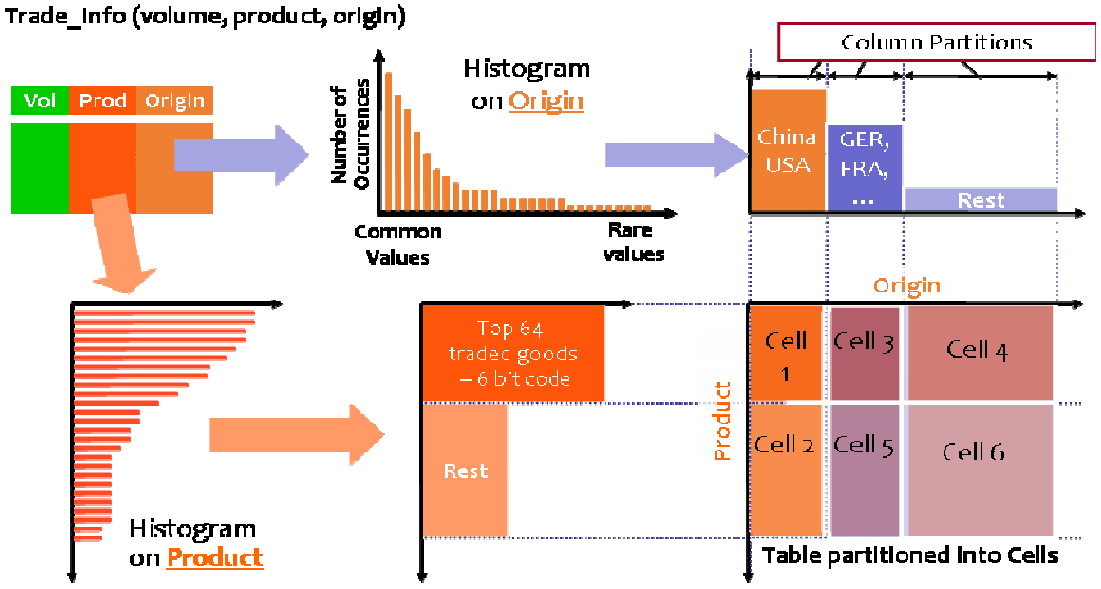
L'étape finale de l'exécution des requêtes par l'accélérateur est faite par le coordinateur. Ce dernier a besoin de mémoire pour la fusion, la décompression et le tri de résultats. Là également, la mémoire nécessaire dépend de la taille des résultats attendus. Il ne faut pas perdre de vue qu'en exécutant une requête avec une clause FIRST, par exemple, "SELECT FIRST 1000… ORDER BY sum(sales.amount)", le coordinateur d'abord devra trier toutes les rangées pour obtenir les 1000 premières. Evidemment, le coordinateur sera en mesure de remplacer ou de rejeter les nouveaux groupes une fois les 1000 premiers créés.

Des performances de pointe

Les performances extrêmes obtenues avec l'accélérateur proviennent de la somme des résultats de recherche technologique effectuée par le pôle recherche et développement d'IBM. Ce document présente une vue d'ensemble des technologies et techniques mises en œuvre dans l'accélérateur afin d'améliorer les performances et éliminer les tâches d'administration et de maintenance. Les documents cités en référence fournissent de plus amples détails sur ces techniques et théories. Ils ont été publiés dans des revues renommés spécialisés et présentés lors de conférences. IBM détient les brevets relatifs à ces techniques.

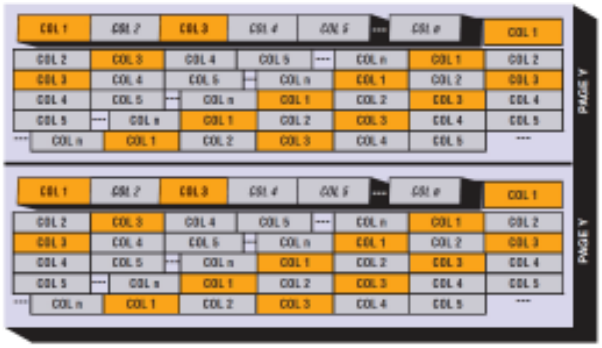
La technologie « deep colomnar » va au delà de celle du stockage traditionnel en colonne. La compression extrême et l'exécution des requêtes sur les données compressées élimine les Entrées/Sorties disques et permet de bénéficier de grand « datawarehouse » en mémoire. L'exploitation d'architecture multi-cœurs et de la technologie SIMD permet d'obtenir d'excellentes performances sans indexation ni tables de résumé. Regardons chaque technique plus en détail.

Partionnement par occurrence



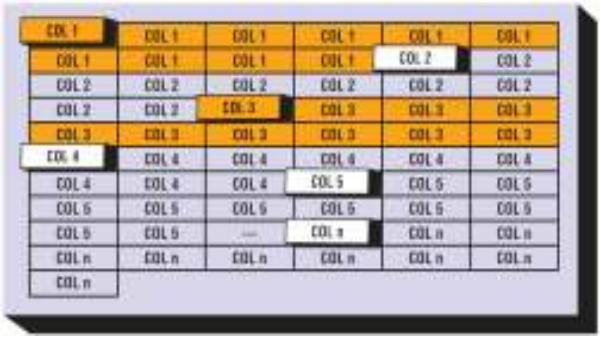
Chaque table du « data mart » est analysée à la recherche de valeurs récurrentes dans chaque colonne et dans les groupes de colonnes afin de déterminer les colonnes optimales à prendre en compte pour former un « tuplet ». Dans l'exemple ci-dessus, les colonnes "product" et "origin" sont corrélées et donc combinées pour former un « tuplet ». Un « tuplet » est une partie d'un n-uplet. Un n-uplet est un une rangée complète. L'avantage de l'encodage de Huffman est que les valeurs de forte occurrence sont codées avec un nombre de bit minimum. Dans le graphique ci-dessus, les 64 valeurs les plus représentées de la colonne "product" sont combinées avec les valeurs les plus représentées de la colonne "origin" (USA , China) pour former la plus petite cellule cell1 grâce à l'encodage de Huffman. Cette technique améliore l'efficacité de la compression et peut être utilisée pour évaluer les prédicats d'égalité et d'intervalle. L'exécution étant faite sur des données compressées, moins de bits sont manipulés, cela qui se traduit par une plus grande vitesse de traitement.

Le stockage « colomnar » dans l'accélérateur

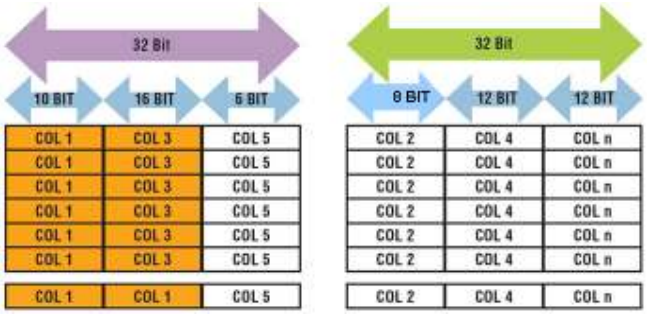


Les moteurs de base de données traditionnels basés sur la technologie de rangée (ou n-uplet) stockent une rangée complète dans une page, suivie d'une autre rangée. La conception est faite pour optimiser les Entrées/Sorties de rangées et suppose que la requête recherche la majorité des valeurs de la colonne. Si la donnée est stockée compressée, elles doivent être décompressées pour chaque lecture (fetch). Cela est efficace pour un travail transactionnel accédant à seulement quelques rangées par requête.

En ce qui concerne les requêtes analytiques, les lectures sont généralement faites sur des millions ou des milliards de rangées mais pour chaque requête, on analyse les relations entre les sous-ensembles des colonnes de la table de fait. Par exemple, si l'on est intéressé par la totalité des ventes des produits par région en 2010, la requête nécessite d'accéder seulement à trois des colonnes de la table de fait : "item", le montant "sales" et "location" et faire la jointure avec les tables de dimension. Dans ce cas, accéder puis décompresser chaque rangée n'est pas efficace.



Les systèmes de base de données « colomnar » stockent toutes les valeurs de colonne ensemble. A chaque insertion ou chargement de données, chaque rangée est décompressée pour séparer les valeurs de colonne et à chaque fois que la rangée est accédée, les valeurs de colonne sont récupérées séparément puis compressées pour former la rangée. C'est parce que les valeurs de colonne sont stockées ensemble que l'on obtient une meilleure compression. Dans l'exemple précédent, nous avons vu que l'intérêt pour les requêtes analytiques se trouve dans les sous-ensembles de rangée. Pour cette requête dans un système de base de données de type « colomnar », il n'est nécessaire de récupérer les pages contenant "item", "sales" et "location". Pour les requêtes accédant de grand sous-ensemble dans les rangées et utilisant l'accès séquentiel, le stockage « colomnar » améliore les performances.



Informix Warehouse Accelerator stocke les données par groupes de colonne, ou partitions verticales de la table appelés aussi « banks ». La partie d'une rangée (ou n-uplet) qui tient dans chaque « bank » est appelé un « tuplet ». L'assignation des colonnes aux « banks » est particulière à chaque cellule en raison de la variation de la taille des colonnes d'une cellule à l'autre.

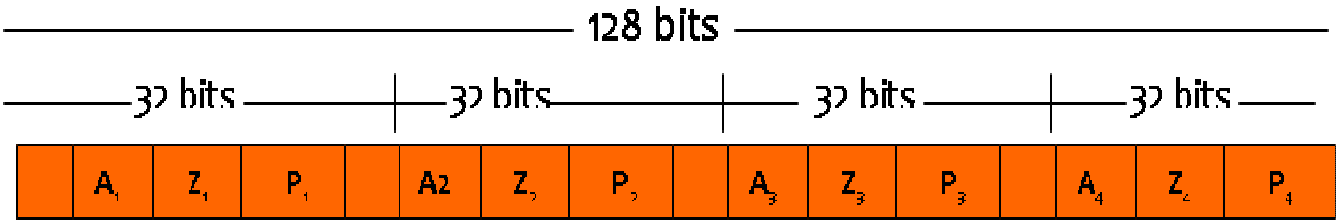
L'assignation utilise un algorithme « bin-packing » basé sur l'éligibilité d'une colonne à tenir dans un « bank », dont la largeur est une fraction d'un mot, plutôt que de se baser sur son utilisation dans un flux. Les accès sont faits uniquement sur les « banks » qui contiennent les colonnes référencées dans une requête donnée, évitant donc d'accéder les autres « banks » n'ayant aucune colonne référencée dans la requête. Cette projection est similaire à la façon dont les systèmes traditionnels en colonne arrivent à minimiser les Entrées/Sorties disques. L'accélérateur stockant tout en mémoire, il n'y a pas d'accès au disque. Cependant, grâce à cette technique, Informix Warehouse Accelerator réduit le nombre d'accès à la mémoire et permet un gain considérable de cycle CPU.

Parallélisme de traitement des données en une seule instruction

Considérons la requête suivante:

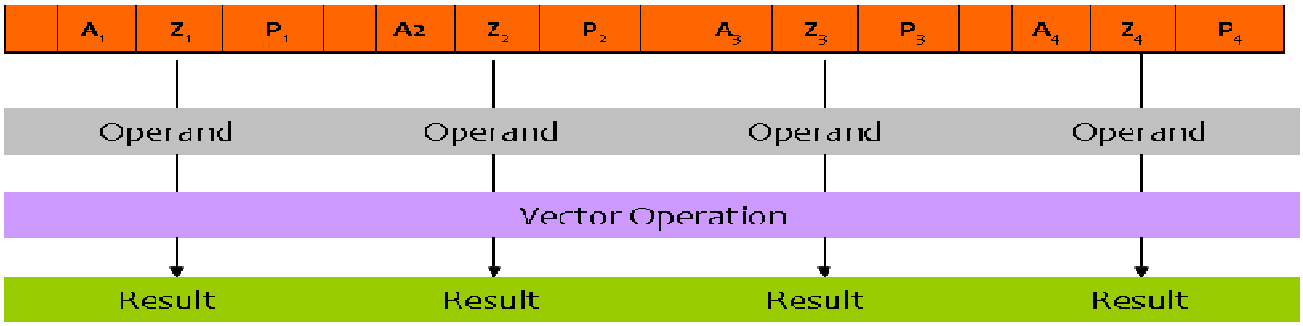
```
SELECT SUM(s.amount) FROM sales AS s WHERE s.prid = 100 GROUP BY s.zip;
```

Si les colonnes "amount" (A), "prid" (P), "zip" (Z) proviennent du même "bank", plusieurs de ces valeurs peuvent être chargées dans un registre CPU de 128 bits en une fois. Dans ce cas, il s'agit de 12 valeurs à la fois.



Les instructions SIMD des processeurs Intel Xeon travaillent sur des registres 128 bits. La technique de compression de l'accélérateur demande généralement peu de bits pour chaque colonne et on peut donc charger plusieurs champs dans chaque registre de 128 bits. Après chargement de plusieurs valeurs, l'application de prédicats se fait sur toutes les colonnes de façon simultanée. Lors de l'exécution de requête, ce type de chargement est réalisé sur tous les cœurs alloués ce qui permet une parallélisation à l'extrême.

Avec cette technique, on peut travailler sur les 12 valeurs simultanément en une seule instruction CPU ce qui garanti des gains de performance significatifs.

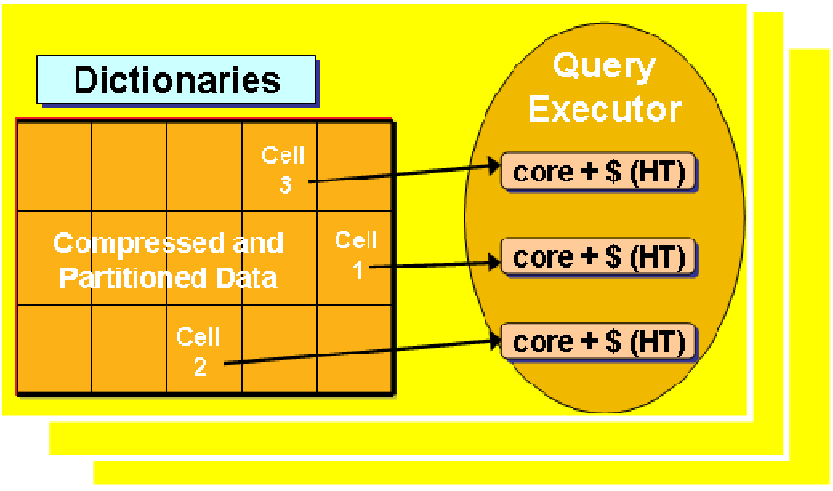


L'exécution de requête

Jusqu'à présent, nous avons présenté la façon dont les données étaient encodées, stockées et comment l'exécution était réalisée au niveau le plus bas. Etudions tout cela de plus haut.

Un balayage efficace des données est la base de l'exécution de requête.

Nous savons que les données sont organisées en « tuplets », « banks » et cellules. En ce qui concerne le balayage de données, l'unité de référence est la cellule. L'accélérateur crée dynamiquement des threads afin d'utiliser au mieux les ressources CPU configurées et assigne des cellules à chaque cœur. Ce balayage est fait sur les données compressées, en utilisant les instructions SIMD et tire avantage de l'encodage de Huffman. L'évaluation de prédicat tel que le GROUP BY est réalisé sur les données compressées mais le calcul des agrégats se fait lui sur les données décompressées. La technique d'encodage fonctionne aussi comme une fonction de hachage très dense permettant la mise en cache de la table de hachage dans le cache L2 du processeur et donc une lecture très rapide.



Cela permet de réaliser les GROUP BY très rapidement sur les données compressées. Une jointure entre deux tables de hachage avec un encodage dense peut amener à des groupes clairsemés de valeurs. L'accélérateur détecte ces situations et va changer d'algorithme dynamiquement pour l'exploration linéaire. C'est le fait de combiner toutes ces techniques qui permet l'exécution des requêtes sur les données compressées.

Nous savons que le « data mart » est un schéma en étoile ou en flocon contenant une large table de fait et un certain nombre de tables de dimension. Les requêtes vont faire leur jointure entre ces tables et la table de fait en utilisant le prédicat de jointure d'abord entre la table de fait et une table de dimension puis avec les autres tables de dimension. Pour chaque requête, chaque sous-traitant créera un modèle en flocon pour les tables. Le traitement va alors débuter par la partie extérieure de la branche puis progresser dans la table de fait. Chaque branche du modèle en flocon est traitée et le résultat joue alors le rôle de table dimension pour le niveau suivant. Tout d'abord, les prédicats locaux sont appliqués aux tables de dimension pour créer une liste de clefs correspondantes. Ces clefs sont utilisées pour la jointure avec la table de fait (ou les tables de dimension jouant ce rôle dans la branche du modèle en flocon) pour former le prochain niveau d'agrégations et de relations. Ce processus est appliqué de façon récursive jusqu'à ce que toutes les jointures aient été faites par tous les sous-traitants. Etant donné que chaque sous-traitant a une copie de la table de dimension, il y a très peu de données échangées entre les sous-traitants pendant l'exécution de la requête et chacun d'entre eux travaille de façon la plus performante possible. Les sous-traitants procèdent alors à l'envoi des résultats intermédiaires au coordinateur.

Sachant que les données ne sont représentées que d'une seule manière – table de type « colomnar » compressées – l'accélérateur exécute à chaque fois le même code pour chaque table. Toute l'efficacité de l'utilisation de cellules et de l'élimination de blocks provient de l'encodage de la compression. En fait, l'accélérateur n'utilise pas d'indexes et n'a pas à se préoccuper de tables de résumé. Il suit, à peu de choses près et à chaque fois le même processus d'exécution. Grâce à cela, l'accélérateur a des performances constantes.

Le nœud coordinateur récupère les résultats de chaque sous-traitants, fusionnent les résultats intermédiaires dans les groupes correspondants, décompresse les données, et termine par les ordres ORDER BY et HAVING avant d'envoyer les données au serveur Informix grâce au protocole DRDA. Le serveur Informix renvoie les résultats à l'application cliente.

Conclusion

L'approche innovante de l'exécution de requêtes complexes par l'accélérateur Informix Warehouse améliore la productivité en fournissant des réponses plus rapidement sans accroître votre charge de travail ou grever votre budget. Grâce à son étroite liaison avec le moteur de base de données Informix, vous pouvez exploiter votre environnement au mieux en divisant la charge de travail entre le moteur Informix et l'accélérateur.

De meilleurs temps de réponse signifient des réponses délivrées plus vite, un aperçu plus rapide et un environnement de travail qui s'adapte plus vite. Vous pouvez planifier l'accélération de la partie à haute valeur ajoutée de votre « datawarehouse » et faire évoluer dynamiquement l'infrastructure afin de l'adapter à vos besoins.

La prochaine étape

Pour en savoir plus sur l'accélérateur Informix Warehouse et l'édition Informix Ultimate Warehouse, vous pouvez contacter votre représentant marketing IBM ou un partenaire IBM, ou visiter les sites suivants :
http://www.ibm.com/informix
http://www.ibm.com/informix/warehouse

Remerciements

Ce produit a été développé en collaboration avec IBM Almaden Research, le laboratoire IBM Böblingen et l'équipe Informix. Merci à toute l'équipe Informix pour la revue et l'amélioration de ce document.

Références

- **VLDB 2008:** *"Main-Memory Scan Sharing for Multi-core CPUs",* Lin Qiao, Vijayshankar Raman, Frederick Reiss, Peter Haas, Guy Lohman
- **VLDB 2008:** *"Row-Wise Parallel Predicate Evaluation",* Ryan Johnson, Vijayshankar Raman, Richard Sidle, Garret Swart
- **VLDB 2006**: *"How to wring a table Dry: Entropy Compression of Relations and Querying Compressed Relations",* Vijayshankar Raman, Garret Swart
- **SIGMOD 2007**: *"How to barter bits for chronons: compression and bandwidth trade offs for database scans",* Allison L. Holloway, Vijayshankar Raman, Garret Swart, David J. DeWitt
- **ICDE 2008**: *"Constant-time Query Processing",* Vijayshankar Raman, Garret Swart, Lin Qiao, Frederick Reiss, Vijay Dialani, Donald Kossmann, Inderpal Narang, Richard Sidle
- **BTW 2009**: Bringing BLINK Closer to the Full Power of SQL. Knut Stolze, Vijayshankar Raman, Richard Sidle, Oliver Draese

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.
Produced in the United States of America
March 2011