Optimisation SQL

- Les articles -

Date de mise en ligne : vendredi 3 décembre 2004

Date de parution : 25 novembre 2004

Description :

Optimisation du code source, du modèle de donnée. Outils d'analyse.

Environnement iSeries

Introduction

- Les performances, assez médiocres en terme de temps d'exécution, ont longtemps freinées le développement d'application utilisant largement SQL.
- L'apparition de machine utilisant un (ou des) processeurs de type RISC, ainsi que l'amélioration des outils de diagnostic et d'analyse, ont permis au fil des versions, d'obtenir des temps d'exécution corrects (comparables à ceux d'un programme RPG).
- Un AS400 équipé d'un processeur RISC en V4R1 est un bon point de départ pour l'utilisation intensive de SQL.

L'optimiseur SQL

- L'optimiseur n'a qu'un seul but dans la vie : toujours faire le moins possible d'entrée/sorties (car ce qui ralentit les requêtes se sont les I/O et non la CPU.
- les colonnes de clé d'une table sont recommandées d'être contiguës (PRIMARY KEY = premières colonnes d'une table et dans l'ordre de la clé)
- les valeurs testées doivent être de même type pour qu'un index soit utilisable (si un index est défini sur une colonne TIMESTAMP, il ne fonctionnera pas sur une fonction DATE(TIMESTAMP))
- un index ne sera pas utilisé si une valeur n'est pas testée sur une égalité (SELECT * FROM table WHERE VALEUR<100 débouche sur un scan de la table. Il faut transformer la requête en VALEUR<=99)
- un index n'est efficace que s'il porte sur un nombre discriminant de valeurs (valeur unique ou "petit" nombre de valeurs, sinon c'est un autre type d'index (EVI) à mettre en oeuvre (à partir de la v5R1).
- l'ordre de jointure des tables n'a pas (ou plus) d'importance : l'optimiseur réarrange automatiquement les tables pour commencer par celle la plus apropriée.
- l'optimiseur examine d'abord les "sélections locales" (IN, = de valeurs fixes), puis les zones MIN/MAX, puis les colonnes de jointures, ensuite les zones groupées, puis enfin les critères de tris. Ce n'est pas pour autant qu'un index répondant au mieux à cette définition sera choisi. Par contre l'optimiseur peut choisir de l'utiliser.

Optimisation d'un programme

L'optimisation d'un programme utilisant SQL peut se faire selon deux axes :

- Optimisation du code source.
- Optimisation du modèle de données.

Vous pouvez connaitre le temps d'exécution d'une requête sans l'exécuter. Pour cela utiliser la commande CHGQRYA (paramètre QRYTIMLMT à 0) vos requêtes seront alors refusées (message CPA4259) , l'optimiseur vous indiquera le temps prévu (DSPJOBLOG).

Optimisation du modèle de données, quelques règles

- Standardiser le format et le type de vos colonnes.
- Utiliser des clés composées de préférence d'une colonne unique.

- Évitez les colonnes dont la valeur peut être nulle surtout si ces colonnes doivent être calculées.
- Préférez les types fixes (CHAR au lieu de VARCHAR) pour les colonnes fréquemment sollicitées en recherche et jointure.
- Indexer l'essentiel, pas le superflu.

Les index

- SQL utilise des chemins d'accès pour les clauses WHERE, GROUP BY, ORDER BY.
- Si ces chemins d'accès existent déjà, SQL les utilise, sinon il les crée (d'où l'intérêt de créer des index définitifs, CREATE INDEX ou LF).
- Un index ne peut jamais être explicitement utilisé, c'est le DATA BASE MANAGER qui décide et optimise (dans certains cas un balayage complet du fichier sera moins pénalisant).

Les index EVI

- Version 4.30 de l'OS/400 minimum.
- Les index EVI (encoded vector index) sont des index apparus à la version 4.30 ; il s'agit d' un concept propre à l'AS/400. [1]
- Ces index sont moins encombrants sur l'espace disque et permettent d'optimiser les temps d'accès. L'optimiseur SQL peut construire un chemin d'accès (pour un même fichier)à partir de plusieurs index EVI, ce qui est impossible avec un index classique.
- **ATTENTION :** les index EVI sont destinés uniquement à SQL (et QUERY, QM, ...).Ils ne sont donc utilisables qu' avec SQL et ne peuvent pas être utilisés en RPG (CHAIN, READ, etc.).

```
* Création d'un index EVI
 CREATE ENCODED VECTOR INDEX
 on fichier(clé1, clé2, ...)
 FOR x DICTINCT VALUES
```

Programmation Optimisation des ordres de sélection

- Limiter le nombre de colonnes à extraire :

```
mieux  SELECT ZON1, ZON2 FROM FICHIER
 que       SELECT * FROM FICHIER
```

Cette règle ne s'applique pas aux sous-requêtes ni à l'emploi de la fonction COUNT (COUNT(*) plus performant que COUNT(COL1)).

- Utilisez la clause OPTIMIZE FOR xxxx ROWS :

```
 SELECT ZON1 , ZON2 FROM FICHIER OPTIMIZE FOR 5 ROWS
```

Cette clause permet d'optimiser la récupération d'un nombre d'enregistrements donnés, ce qui s'avère très performant pour le traitement du chargement dynamique d'un sous-fichier.

- Utilisez BEETWEEN pour les plages de valeurs.

```
mieux SELECT NUM FROM FICHIER WHERE NUM BEETWEEN 10 AND 15
 que   SELECT NUM FROM FICHIER WHERE NUM <= 15 AND NUM >= 10
 pire  SELECT NUM FROM FICHIER WHERE IN(10, 11, 12, 13, 14)
```

- Préférez EXISTS à IN pour traiter les sous-requêtes.

```
mieux  SELECT A.ZON1A FROM FICHIER1 A WHERE NOT EXISTS(SELECT * FROM FICHIER2 B WHERE A.ZON1A = B.ZON1A AND B.ZON2B = 0)

 que  SELECT A.ZON1A FROM FICHIER1 A WHERE A.ZON1A NOT IN (SELECT       B.ZON1A       FROM FICHIER2 B WHERE B.ZON2B <> 0)
```

- N'utilisez pas de nombre avec ORDER BY. .

```
mieux       SELECT ZON1, ZON2 FROM FICHIER ORDER BY ZON1, ZON2
 que       SELECT ZON1, ZON2 FROM FICHIER ORDER BY 1, 2
```

Les outils d'analyses

- Messages de débogage de l'optimiseur de requête Les messages d'information (messages de type *INFO)sont consignés dans l'historique des travaux par l'optimiseur de requête lorsque vous exécutez votre requête en mode débogage (STRDBG).
- L'analyse des messages dans l'historique des travaux et la lecture des descriptifs de second niveau associés, vous donnera des indications pour identifier les modifications que vous pouvez apporter (par exemple, création d'un index) et obtenir ainsi de meilleures performances sur vos requêtes SQL.

Mise en oeuvre

```
STRDBG
 CALL xxxx
 DSPJOBLOG
```

Vous pouvez également accéder aux messages de débogage à partir de la fenêtre Exécution de scripts SQL et à l'aide de Visual Explain.

Depuis la V3R60 il est possible d'avoir ce type d'analyse en automatique, le résultat est écrit dans un fichier base de données (le fichier modèle est QAQQDBMN).

Moniteur de performances SQL

```
STRDBMON = démarre le moniteur de base de données, pour un job ou pour tous les jobs de la machine.
 ENDDBMON = arrêt du moniteur et écriture de l'analyse dans un fichier base de données.
```

Chaque requête est analysée en détail avec :

- liste des fichiers traités
- options utilisées (jonction, groupage, ...)
- liste des index examinés (raison du choix ou du refus)
- nombre d'enregistrements traités
- consommation CPU, ...

iSeries Navigator (V5R1)

- iSerie Navigator (anciennement Operations Navigator) possède un ensemble d'outils permettant d' ajuster des requêtes en analysant les données du moniteur de performances SQL directement ou via la représentation graphique fournie dans Visual Explain.

Astuces :

- SQE vous propose des suggestions d'index. Vous pouvez les consulter par exemple :

```
SELECT substr(table_nAME, 1, 10) as "TABLE_NAME" , TABLE_SCHEMA, KEY_COLUMNS_ADVISED
FROM QSYS2/SYSIXADV
where Table_SCHEMA <>  'QSYS'
```

Les liens

Voir l'article sur les performances

Voir l'article Optimisation SQL avec Visual Explain

---

[1] (Attention, serge cela n'est plus d'actualité),
- Oracle (et d'autres) possèdent désormais des mécanismes d'index similaire, Communément applelé index BITMAP Oracle possède en fait 4 types d'index (en version 9i)
- l'arbre équilibré (B-Tree), le plus connu, et le plus utilisé (option par défaut) : peut être défini sur 32 colonnes.
- inverse (reverse key), qui concerne le tables "clusterisées"
- chaine de bits (BITMAP) qui regroupe chaque valeur de la (ou des) colonne(s) indexée(s) sous la forme d'une chaine de bits. Ce type d'index peut être défini sur trente colonnes, mais n'est disponible qu'avec la version Entreprise Edition
- basés sur des calculs entre colonnes (function-based indexes)