Article
Lorenzo Scalese · Avr 15 26m de lecture

Open Exchange

APM - Surveillance des performances des requêtes SQL

Depuis Caché 2017, le moteur SQL comprend un nouvel ensemble de statistiques. Celles-ci enregistrent le nombre de fois qu'une requête est exécutée et le temps qu'elle prend pour s'exécuter.

C'est une mine d'or pour quiconque surveille et tente d'optimiser les performances d'une application qui comprend de nombreuses instructions SQL, mais il n'est pas aussi facile d'accéder aux données que certaines personnes le souhaitent.

Cet article et l'exemple de code associé expliquent comment utiliser ces informations et comment extraire de manière routinière un résumé des statistiques quotidiennes et conserver un historique des performances SQL de votre application.

Qu'est-ce qui est enregistré ?

Chaque fois qu'une instruction SQL est exécutée, le temps pris est enregistré. Ce système est très léger et vous ne pouvez pas le désactiver. Pour minimiser les coûts, les statistiques sont conservées en mémoire et écrites sur disque périodiquement. Les données comprennent le nombre de fois qu'une requête a été exécutée dans la journée et le temps moyen et total nécessaire.

Les données ne sont pas écrites sur le disque immédiatement, et après qu'elles aient été écrites, les statistiques sont mises à jour par une tâche "Update SQL query statistics" qui est généralement programmée pour s'exécuter une fois par heure. Cette tâche peut être lancée manuellement, mais si vous souhaitez voir les statistiques en temps réel tout en testant une requête, l'ensemble du processus requiert un peu de patience.

Avertissement : Dans InterSystems IRIS 2019 et les versions antérieures, ces statistiques ne sont pas collectées pour le embedded SQL dans **les classes ou les routines** qui ont été **déployées** à l'aide du mécanisme **%Studio.Project:Deploy** . Rien ne sera cassé avec l'exemple de code, mais il pourrait vous tromper (il m'a trompé) en pensant que tout était OK parce que rien n'est apparu comme coûteux.

Comment voyez-vous les informations ?

Vous pouvez voir la liste des requêtes dans le portail de gestion. Passez à la page SQL et cliquez sur l'onglet "SQL Statements". C'est une bonne chose pour une nouvelle requête que vous exécutez et regardez, mais si des milliers de requêtes sont exécutées, cela peut devenir ingérable.

L'alternative est d'utiliser SQL pour rechercher les requêtes. Les informations sont stockées dans des tableaux du schéma INFORMATION_SCHEMA. Ce schéma comporte un certain nombre de tableaux et j'ai inclus quelques exemples de requêtes SQL à la fin de cet article.

Quand les statistiques sont-elles retirées ?

Les données d'une requête sont supprimées chaque fois qu'elle est recompilée. Ainsi, pour les requêtes

dynamiques, cela peut signifier que les requêtes en cache sont purgées. Pour le embedded SQL, cela signifie que la classe ou la routine dans laquelle le embedded SQL se trouve est recompilée.

Sur un site actif, il est raisonnable de croire que les statistiques seront conservées pendant plus d'une journée, mais les tableaux contenant les statistiques ne peuvent pas être utilisés comme source de référence à long terme pour l'exécution de rapports ou d'analyses à long terme.

Comment pouvez-vous résumer l'information ?

Je recommande d'extraire les données chaque nuit dans des tableaux permanents avec lesquels il est plus facile de travailler pour générer des rapports de performance. Il est possible que certaines informations soient perdues si les classes sont compilées pendant la journée, mais il est peu probable que cela fasse une réelle différence dans votre analyse des requêtes lentes.

Le code ci-dessous est un exemple illustrant comment vous pourriez extraire les statistiques dans un résumé quotidien pour chaque requête. Il comprend trois classes courtes :

* Une tâche qui doit être exécutée chaque nuit.
* DRL.MonitorSQL est une classe principale qui extrait et stocke les données des tableaux INFORMATION_SCHEMA.

- La troisième classe DRL.MonitorSQLText est une optimisation qui permet de stocker le texte de la requête (potentiellement long) une seule fois et de ne stocker que le hashage de la requête dans les statistiques pour chaque jour.

Notes concernant l'exemple

La tâche extrait le jour précédent et doit donc être programmée peu après minuit.

Vous pouvez exporter plus de données historiques si elles existent. Pour extraire les 120 derniers jours

Faites ##class(DRL.MonitorSQL).Capture($h-120,$h-1)

Le code d'exemple lit directement le global ^rIndex car les premières versions des statistiques n'exposaient pas la Date à SQL.

La variante que j'ai incluse boucle sur tous les espaces de noms de l'instance, mais cela n'est pas toujours approprié.

Comment faire une requête sur les données extraites

Une fois les données extraites, vous pouvez trouver les requêtes les plus lourdes en exécutant

```
SELECT top 20
S.RunDate,S.RoutineName,S.TotalHits,S.SumpTIme,S.Hash,t.QueryText
from DRL.MonitorSQL S
left join DRL.MonitorSQLText T on S.Hash=T.Hash
where RunDate='08/25/2019'
order by SumpTime desc
```

De plus, si vous choisissez le hachage pour une requête coûteuse, vous pouvez voir l'historique de cette requête avec

```
SELECT S.RunDate,S.RoutineName,S.TotalHits,S.SumpTIme,S.Hash,t.QueryText
from DRL.MonitorSQL S
left join DRL.MonitorSQLText T on S.Hash=T.Hash
where S.Hash='CgOlfRw7pGL4tYbiijYznQ84kmQ='
order by RunDate
```

Au début de l'année, j'ai analysé les données d'un site en direct et j'ai examiné les requêtes les plus coûteuses. Une requête durait en moyenne moins de 6 secondes mais était appelée 14 000 fois par jour, ce qui représentait près de 24 heures de temps écoulé chaque jour. En fait, un noyau était entièrement occupé par cette seule requête. Pire encore, la deuxième requête qui prend une heure était une variation de la précédente requête.

RunDate	RoutineName	Nombre total de visites	Total Time	Hash	QueryText (en abrégé)
03/16/2019		14,576	85,094	5xDSguu4PvK04se2pPiOexeh6aE=	DECLARE C CURSOR FOR SELECT * INTO :%col(1) , :%col(2) , :%col(3) , :%col(4) "
03/16/2019		15,552	3,326	rCQX+CKPwFR9zOplmtMhxVnQxyw=	DECLARE C CURSOR FOR SELECT * INTO :%col(1) , :%col(2) , :%col(3) , :%col(4) , "
03/16/2019		16,892	597	yW3catzQzC0KE9euvlJ+o4mDwKc=	DECLARE C CURSOR FOR SELECT * INTO :%col(1) , :%col(2) , :%col(3) , :%col(4) , :%col(5) , :%col(6) , :%col(7) ,
03/16/2019		16,664	436	giShyiqNR3K6pZEt7RWAcen55rs=	DECLARE C CURSOR FOR SELECT * , TKGROUP INTO :%col(1) , :%col(2) , :%col(3) , ..
03/16/2019		74,550	342	4ZClMPqMfyje4m9Wed0NJzxz9qw=	DECLARE C CURSOR FOR SELECT "

Tableau 1: Résultats réels sur le site du client

Les tableaux du schéma INFORMATION_SCHEMA

En plus des statistiques, les tableaux de ce schéma gardent la trace de l'endroit où les requêtes, les colonnes, les indices, etc. sont utilisés. En général, l'instruction SQL est un tableau de départ et il est joint par quelque chose comme "Statements.Hash=OtherTable.Statement".

La requête équivalente pour accéder directement à ces tableaux afin de trouver les requêtes les plus coûteuses pour une journée serait...

```
SELECT DS.Day,Loc.Location,DS.StatCount,DS.StatTotal,S.Statement,S.Hash
FROM INFORMATION_SCHEMA.STATEMENT_DAILY_STATS DS
left join INFORMATION_SCHEMA.STATEMENTS  S
on S.Hash=DS.Statement
left join INFORMATION_SCHEMA.STATEMENT_LOCATIONS  Loc
on S.Hash=Loc.Statement
where Day='08/26/2019'
order by DS.stattotal desc
```

Que vous envisagiez ou non de mettre en place un processus plus systématique, je recommande à tous ceux qui ont une grande application utilisant SQL de lancer aujourd'hui cette requête.

Si une requête particulière apparaît comme coûteuse, vous pouvez obtenir l'historique en exécutant

```
SELECT DS.Day,Loc.Location,DS.StatCount,DS.StatTotal,S.Statement,S.Hash
FROM INFORMATION_SCHEMA.STATEMENT_DAILY_STATS DS
left join INFORMATION_SCHEMA.STATEMENTS  S
on S.Hash=DS.Statement
left join INFORMATION_SCHEMA.STATEMENT_LOCATIONS  Loc
on S.Hash=Loc.Statement
where S.Hash='jDqCKaksff/4up7Ob0UXlkT2xKY='
order by DS.Day
```

Exemple de code pour l'extraction quotidienne de statistiques

#Performances #SQL #Surveillance #Caché #InterSystems IRIS
Voir l'application sur InterSystems Open Exchange

URL de la source: https://fr.community.intersystems.com/post/apm-surveillance-des-performances-des-requ%C3%AAtes-sql