Article
Kevin Koloska · Nov 30, 2022 7m de lecture

Open Exchange

Suite de tests d ' E/S PerfTools

Suite de tests d ' E/S PerfTools

Analytics #Caché #HealthShare #InterSystems IRIS #Open Exchange #TrakCare

But
Cette paire d ' outils (RanRead et RanWrite) est utilisée pour générer des événements de lecture et d ' écriture aléatoires dans une base de données (ou une paire de bases de données) afin de tester les opérations d ' entrée/sortie par seconde (IOPS). Ils peuvent être utilisés conjointement ou séparément pour tester la capacité matérielle des E/S, valider les IOPS cibles et garantir des temps de réponse disque acceptables. Les résultats recueillis à partir des tests d ' E/S varient d ' une configuration à l ' autre en fonction du sous-système d ' E/S. Avant d ' exécuter ces tests, assurez-vous que la surveillance correspondante du système d ' exploitation et du niveau de stockage est configurée pour capturer les mesures de performances des E/S en vue d ' une analyse ultérieure. La méthode suggérée consiste à exécuter l ' outil Performance du système fourni avec IRIS. Veuillez noter qu ' il s ' agit d ' une mise à jour d ' une version précédente, qui peut être trouvée ici.
Installation
Téléchargez les outils PerfTools.RanRead.xml et PerfTools.RanWrite.xml depuis GitHub ici.
Importez des outils dans l ' espace de noms USER.
UTILISATEUR> faire $system. OBJ. Load(« /tmp/PerfTools.RanRead.xml »,"ckf »)
UTILISATEUR> faire $system. OBJ. Load(« /tmp/PerfTools.RanWrite.xml »,"ckf »)
Exécutez la méthode Help pour afficher tous les points d ' entrée. Toutes les commandes sont exécutées dans USER.
USER> do ##class(PerfTools.RanRead). Aide()
• do ##class(PerfTools.RanRead). Setup(Répertoire,DatabaseName,SizeGB,LogLevel)
Crée une base de données et un espace de noms portant le même nom. Le niveau logarithmique doit être compris entre 0 et 3, où 0 correspond à « aucun » et 3 à « verbeux ».
• do ##class(PerfTools.RanRead). Exécuter(Répertoire,Processus,Dénombrement,Mode)
Exécutez le test d ' E/S en lecture aléatoire. Le mode est 1 (par défaut) pour le temps en secondes, 2 pour les itérations, en référence au paramètre Count précédent
• do ##class(PerfTools.RanRead). Stop()
Termine toutes les tâches en arrière-plan.
• do ##class(PerfTools.RanRead). Reset()
Supprime les statistiques des exécutions précédentes. Ceci est important pour l ' exécution entre les tests, sinon les statistiques des exécutions précédentes sont moyennées dans la version actuelle.
• do ##class(PerfTools.RanRead). Purger(Répertoire)
Supprime l ' espace de noms et la base de données du même nom.
• do ##class(PerfTools.RanRead). Export(Répertoire)
Exporte un résumé de tous les tests de lecture aléatoire dans un fichier texte délimité par des virgules.
USER> do ##class(PerfTools.RanWrite). Aide()
• do ##class(PerfTools.RanWrite). Setup(Répertoire,NomBase de données)
Crée une base de données et un espace de noms portant le même nom.
• do ##class(PerfTools.RanWrite). Run(Répertoire,NumProcs,RunTime,HangTime,HangVariationPct,Longueur du nom global,Profondeur globale du nœud,Longueur globale du sous-nœud)
Exécutez le test d ' E/S en écriture aléatoire. Tous les paramètres autres que le répertoire ont des valeurs par défaut.
• do ##class(PerfTools.RanWrite). Stop()

Termine toutes les tâches en arrière-plan.
• do ##class(PerfTools.RanWrite). Reset()
Supprime les statistiques des exécutions précédentes.
• do ##class(PerfTools.RanWrite). Purger(Répertoire)
Supprime l ' espace de noms et la base de données du même nom.
• do ##class(PerfTools.RanWrite). Export(Directory)
Exporte un résumé de tout l ' historique des tests d ' écriture aléatoire dans un fichier texte délimité par des virgules.
Coup monté
Créez une base de données vide (pré-développée) appelée RAN au moins deux fois la taille de la mémoire de l ' hôte physique à tester. Assurez-vous que la taille du cache du contrôleur de stockage de la base de données vide est au moins quatre fois supérieure à celle du contrôleur de stockage. La base de données doit être plus grande que la mémoire pour garantir que les lectures ne sont pas mises en cache dans le cache du système de fichiers. Vous pouvez créer manuellement ou utiliser la méthode suivante pour créer automatiquement un espace de noms et une base de données.
USER> do ##class(PerfTools.RanRead). Configuration(« /ISC/tests/TMP »,"RAN »,200,1)
Répertoire créé /ISC/tests/TMP/
Création d ' une base de données de 200 Go dans /ISC/tests/TMP/
Base de données créée dans /ISC/tests/TMP/
Remarque : On peut utiliser la même base de données pour RanRead et RanWrite, ou utiliser des bases séparées si l ' intention de tester plusieurs disques à la fois ou à des fins spécifiques. Le code RanRead permet de spécifier la taille de la base de données, mais pas le code RanWrite, il est donc probablement préférable d ' utiliser la commande RanRead Setup pour créer des bases de données prédimensionnées souhaitées, même si l ' on utilisera la base de données avec RanWrite.
Méthodologie
Commencez avec un petit nombre de processus et des temps d ' exécution de 30 à 60 secondes. Ensuite, augmentez le nombre de processus, par exemple commencez à 10 tâches et augmentez de 10, 20, 40, etc. Continuez à exécuter les tests individuels jusqu ' à ce que le temps de réponse soit constamment supérieur à 10 ms ou que les IOPS calculées n ' augmentent plus de manière linéaire.
L ' outil utilise la commande ObjectScript VIEW qui lit les blocs de base de données en mémoire , donc si vous n ' obtenez pas les résultats attendus, tous les blocs de base de données sont peut-être déjà en mémoire.
À titre indicatif, les temps de réponse suivants pour les lectures aléatoires de base de données de 8 Ko et 64 Ko (non mises en cache) sont généralement acceptables pour les baies entièrement flash :
• Moyenne <= 2ms
• Ne pas dépasser <= 5ms
Courir
Pour RanRead, exécutez la méthode Run en augmentant le nombre de processus et en prenant note du temps de réponse au fur et à mesure. Le principal moteur des IOPS pour RanRead est le nombre de processus.
USER> do ##class(PerfTools.RanRead). Exécuter(« /ISC/tests/TMP »,5,60)
Outil de performance d ' E/S à lecture aléatoire InterSystems
Processus RanRead 11742 créant 5 processus de travail en arrière-plan.
RanReadJob préparé 11768 pour le parent 11742 RanReadJob préparé 11769 pour le parent 11742 RanReadJob préparé 11770 pour le parent 11742 RanReadJob préparé 11771 pour le parent 11742

RanReadJob préparé 11772 pour le parent 11742
Démarrer 5 processus pour le numéro de tâche RanRead 11742 maintenant!
Pour terminer l ' exécution : faites ##class(PerfTools.RanRead). Stop()
En attente de finir.....
Tâches en arrière-plan de lecture aléatoire terminées pour le parent 11742 Travail RanRead
5 processus de 11742 (62,856814 secondes) Temps de réponse moyen = 1,23 ms
IOPS calculé pour la tâche RanRead 11742 = 4065
Le paramètre Mode de la commande Run utilise par défaut le mode 1, qui utilise le paramètre Count (60 dans l ' exemple ci-dessus) comme secondes. Si vous définissez Mode sur 2, le paramètre Count est défini sur un nombre d ' itérations par processus, donc s ' il est défini sur 100 000, chacune des 5 tâches lira 100 000 fois dans la base de données. C ' est le mode utilisé à l ' origine par ce logiciel, mais les exécutions chronométrées permettent une coordination plus précise avec des outils de surveillance tels que System Performance, qui sont également chronométrés, et l ' outil RanWrite .
Pour RanWrite, exécutez la méthode Run en diminuant le paramètre Hangtime. Ce paramètre indique le temps d ' attente entre les écritures en secondes et constitue le principal moteur d ' IOPS pour RanWrite. On peut également augmenter le nombre de processus en tant que moteur pour IOPS.

USER> do ##class(PerfTools.RanWrite). Exécuter(« /ISC/tests/TMP »,1,60,.001)
Processus RanWrite 11742 créant 1 processus de travail en arrière-plan.
Préparé RanWriteJob 12100 pour parent 11742
Démarrer 1 processus pour le numéro de tâche RanWrite 11742 maintenant!
Pour terminer l ' exécution : faites ##class(PerfTools.RanWrite). Stop()
En attente de finir.....
Travaux d ' écriture aléatoire en arrière-plan terminés pour le parent 11742 Les processus 1 de la tâche RanWrite 11742
(60 secondes) avaient un temps de réponse moyen = 0,912 ms
IOPS calculé pour la tâche RanWrite 11742 = 1096
Les autres paramètres de RanWrite peuvent généralement être laissés à leurs valeurs par défaut en dehors de circonstances inhabituelles. Ces paramètres sont les suivants : - HangVariationPct : variance sur le paramètre hangtime, utilisée pour imiter l ' incertitude ; c ' est un pourcentage du paramètre précédent
- Longueur globale du nom : RanWrite choisit aléatoirement un nom global, et c ' est la longueur de ce nom.
Par exemple, s ' il est défini sur 6, le Global peut ressembler à Xr6opg-
Profondeur du nœud global et Longueur du sous-nœud global : le Global supérieur n ' est pas celui qui est rempli. Ce qui est réellement rempli, ce sont des sous-nœuds, donc définir ces valeurs sur 2 et 4 donnerait une commande comme « set ^Xr6opg(« drb7 »,"xt8v ») = [value] ». Le but de ces deux paramètres et de la longueur du nom global est de s ' assurer que le même global n ' est pas défini encore et encore, ce qui entraînerait un minimum d ' événements d ' E/S
Pour exécuter RanRead et RanWrite ensemble, au lieu de « do », utilisez la commande « job » pour les deux afin de les exécuter en arrière-plan.
Résultats
Pour obtenir les résultats simples pour chaque exécution enregistrés dans USER dans la table SQL PerfTools.RanRead et PerfTools.RanWrite, utilisez la commande Export pour chaque outil comme suit.
Pour exporter le jeu de résultats dans un fichier texte délimité par des virgules (csv), exécutez la commande suivante :
USER> do ##class(PerfTools.RanRead). Exporter(« /ISC/tests/TMP/ «)
Exportation du résumé de toutes les statistiques de lecture aléatoire vers
/usr/iris/db/zranread/PerfToolsRanRead20221023-1408.txt
Terminé.
Analyse
Il est recommandé d ' utiliser l ' outil SystemPerformance intégré pour acquérir une véritable compréhension du système analysé. Les commandes de SystemPerformance doivent être exécutées dans l ' espace de noms %SYS. Pour passer à cela, utilisez la commande ZN:
UTILISATEUR> ZN « %SYS »
Pour trouver des détails sur les goulots d ' étranglement dans un système, ou si l ' on a besoin de plus de détails sur la façon dont il fonctionne à ses IOPS cibles, il faut créer un profil SystemPerformance avec une acquisition de données à cadence élevée :
%SYS> set rc=$$addprofile^SystemPerformance(« 5minhighdef »,"Un échantillonnage d ' exécution de 5 minutes toutes les secondes »,1 300)
Ensuite, exécutez ce profil (à partir de %SYS) et revenez immédiatement à USER et démarrez RanRead et/ou RanWrite en utilisant « job » plutôt que « do »:
%SYS>set runid=$$run^SystemPerformance(« 5minhighdef »)
%SYS> ZN « UTILISATEUR »
USER> job ##class(PerfTools.RanRead). Exécuter(« /ISC/tests/TMP »,5,60)
USER> job ##class(PerfTools.RanWrite). Exécuter(« /ISC/tests/TMP »,1,60,.001)
On peut alors attendre la fin du travail SystemPerformance et analyser le fichier html résultant à l ' aide d ' outils tels que yaspe.
Nettoyer
Une fois l ' exécution terminée des tests, supprimez l ' historique en exécutant :
%SYS> do ##class(PerfTools. RanRead). Reset()
Vérifiez l ' application associée sur InterSystems Open Exchange

#Analytique #Caché #HealthShare #InterSystems IRIS #Open Exchange #TrakCare
Voir l'application sur InterSystems Open Exchange

URL de la source:https://fr.community.intersystems.com/post/suite-de-tests-d%E2%80%99es-perftools