

Veille sur les bases de données :

Article 1 : ClickHouse Cloud, BDD en cloud (06/10/2022)

<https://cloud-computing.developpez.com/actu/337357/ClickHouse-Cloud-annonce-de-la-version-beta-de-la-base-de-donnees-en-cloud-elle-ameliore-les-performances-et-attenuer-les-temps-de-latence-grace-a-la-mise-en-cache-multi-niveaux/>



ClickHouse vont lancer une base de données en cloud qui défierait la concurrence. Elle se baserait sur une technologie OpenSource et permettrait aux utilisateurs de tirer partie de ses atouts avec le moins de crainte de perte de donnée possible. Elle dissocie le stockage et le calcul, ClickHouse Cloud permet d'évoluer de manière automatique en y ajoutant de la puissance de calcul pour s'adapter à la

charge de travail demandé.

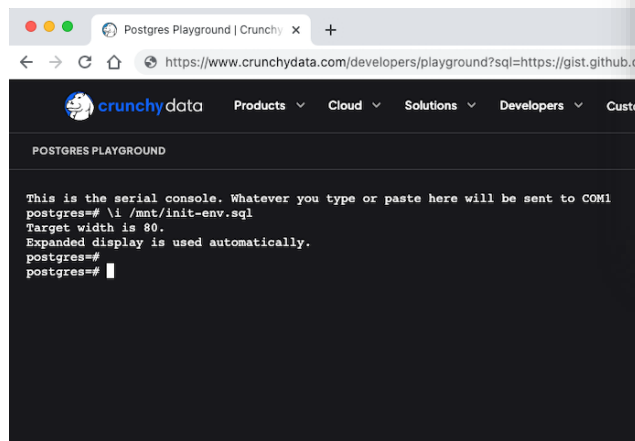
Améliore le rapport coût-efficacité.

GSB pourrait réfléchir quant à l'utilisation de cette solution pour les bases de données de l'entreprise.

Supabase met à disposition « Postgres-wasm », qui est un serveur PostgreSQL fonctionnant dans un navigateur.

Le populaire PostgreSQL peut dorénavant être accessible directement via une page WEB.

Cela permet aux administrateurs de bases de données de GSB d'accéder à la base de manière extrêmement simple depuis un navigateur même à l'extérieur du réseau de l'entreprise, et cela, depuis n'importe quel appareil.



Article 2 : SQL Server 2022 (17/11/2022)

<https://sqlserver.developpez.com/actu/338624/Microsoft-publie-SQL-Server-2022-avec-de-nouvelles-fonctionnalites-pour-Azure-et-des-ameliorations-de-performances-l-entreprise-affirme-que-cette-version-est-la-plus-compatible-avec-Azure-a-ce-jour/>

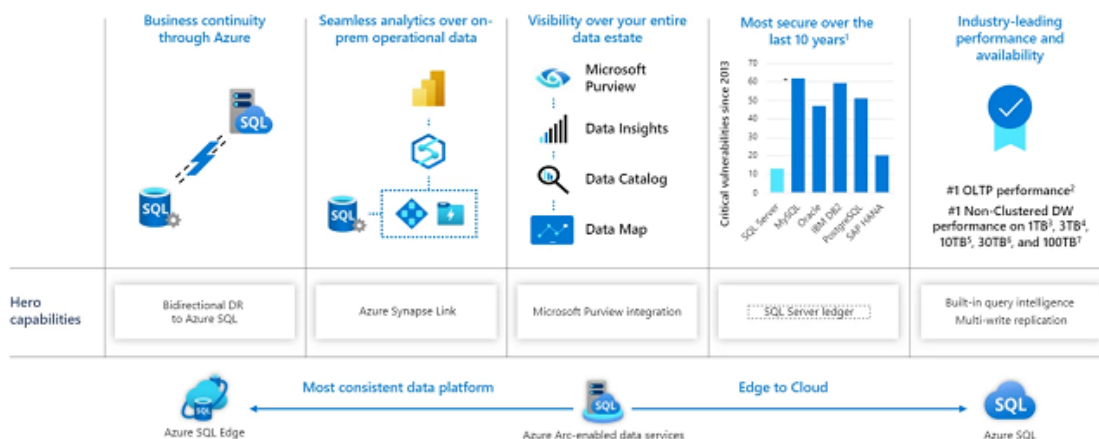
SQL Server 2022 cloud-connected capabilities



Microsoft a annoncé la disponibilité générale de SQL Server 2022, la dernière version de sa plateforme de base de données relationnelle. Cette version offre de nouvelles fonctionnalités telles qu'une intégration avec Azure Synapse pour le stockage et l'analyse de données, ainsi qu'avec Azure Purview pour la classification et la protection des informations. T-SQL, le langage de requête de SQL Server, a été amélioré avec de nouvelles fonctions JSON et la prise en charge de l'API AWS S3 pour la sauvegarde et la restauration de données. Cette version est considérée comme la plus orientée vers le cloud de SQL Server jamais publiée, avec une intégration transparente avec d'autres services cloud de Microsoft tels qu'Azure SQL Database Managed Instance, Azure Active Directory et Microsoft Purview.

SQL Server 2022

Azure-enabled with continued performance and security innovation



Bidirectional disaster recovery is available in limited public preview. General availability will occur at a future date.

TPC Claims as 11/2022

¹National Institute of Standards and Technology Comprehensive Vulnerability Database ²<https://www.tpc.org/9062> ³<https://www.tpc.org/3386> ⁴<https://www.tpc.org/3385> ⁵<https://www.tpc.org/3383> ⁶<https://www.tpc.org/3364> ⁷<https://www.tpc.org/3382>

Article 3 : HC-tree, concurrent à SQLite (23/01/2023)

<https://sgbd.developpez.com/actu/340658/HC-tree-un-dorsal-experimental-de-base-de-donnees-haute-concurrence-pour-SQLite-il-vise-a-developper-un-nouveau-module-de-base-de-donnees-qui-ameliore-SQLite/>

Le projet HC-tree est une tentative d'améliorer le module de base de données de SQLite, un moteur de base de données relationnelle léger.

SQLite est souvent utilisé comme noyau d'un système de base de données client/serveur, mais son module de base de données n'a pas été conçu pour cela.

Le projet HC-tree propose un nouveau module de base de données qui utilise des structures B-arbre pour améliorer les performances et la concurrence de SQLite. Les structures B-arbre permettent une exécution rapide des opérations d'insertion et de suppression en temps logarithmique. Le projet HC-tree améliore également la concurrence de SQLite en permettant plusieurs scripts simultanés grâce à un verrouillage optimisé au niveau de la page.

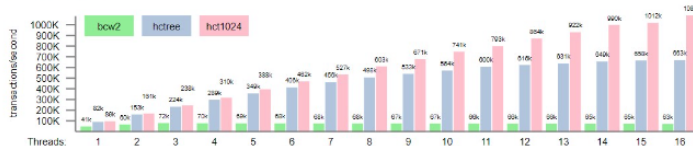
HC-tree propose trois principales améliorations vis-à-vis de SQLite :

- Amélioration de la concurrence
- Support de la réplication
- Suppression des limitations de taille des bases de données

Amélioration de la concurrence : le SQLite standard est limité à un seul script simultané. L'utilisation de l'extension begin-concurrent change cela de façon à ce que plusieurs scripts puissent fonctionner simultanément en utilisant le verrouillage optimisé au niveau de la page. Cela améliore quelque peu la concurrence, mais le verrouillage au niveau de la page peut détecter des conflits entre des transactions logiquement indépendantes, et les opérations COMMIT doivent toujours être sérialisées.

Hctree utilise un verrouillage optimisé au niveau des lignes et est conçu pour supporter des dizaines de scripts simultanés fonctionnant à pleine vitesse. Les résultats des tests obtenus à partir du prototype montrent que cela est possible. Tous les tests ont été effectués sur un processeur AMD 5950X à 16 cœurs avec 32G de mémoire et utilisant tmpfs comme système de fichiers. Les fonctions avancées de gestion de la fréquence du CPU du BIOS comme "Core Boost", "AMD Cool & Quiet" et *Simultaneous Multi-Threading* sont toutes désactivées.

Test case update1 - nUpdate=1, nScan=0



Support de la réplication : SQLite standard supporte l'extension sessions, qui permet au contenu d'une transaction validée d'être sérialisé pour être transmis et appliqué à une seconde base de données.

L'extension de session fournit un mécanisme permettant d'enregistrer les modifications apportées à certaines ou à toutes les rowid des tables d'une base de données SQLite, et de regrouper ces modifications dans un fichier *changeset* ou *patchset* qui peut être utilisé ultérieurement pour appliquer le même ensemble de modifications à une autre base de données ayant le même schéma et des données compatibles de départ. Un *changeset* peut également être inversé et utilisé pour "annuler" une session.

HC-tree intègre cette extension dans le backend de la base de données, et ajoute la prise en charge de l'application de telles transactions aux bases de données subordonnées dans des configurations *leader-follower*. Dans ce cas, les transactions reçues d'une base de données *leader* peuvent être appliquées plus rapidement et avec une plus grande simultanéité que celle avec laquelle elles ont été initialement appliquées à la base de données *leader*, car aucune validation de transaction n'est requise.

Lorsqu'une transaction doit être validée dans un système de concurrence optimisé, elle doit d'abord être validée. La validation d'une transaction consiste à vérifier que, considérée de concert avec toutes les autres transactions passées et actuelles, la transaction aurait pu être validée dans un système qui sérialise toutes les transactions sans changer le résultat en termes d'état final de la base de données ou de résultats de requête renvoyés à un client. En pratique, il existe deux façons de valider une transaction Hc-tree :

1. Si aucun autre client n'a engagé une transaction dans la base de données depuis l'ouverture, la transaction doit être valide ;
2. Si aucun autre client n'a modifié une entrée ou une plage de B-tree à laquelle la transaction en cours d'exécution a accédé par une plage de requête, alors la transaction est valide.

Suppression des limitations de taille des bases de données : le SQLite standard utilise des numéros de page de 32 bits. En utilisant la taille de page par défaut de 4KiB, cela conduit à une taille maximale de base de données de 2^{32} octets, ou 16TiB. HC-tree utilise des numéros de page de 48 bits, permettant des bases de données de 2^{48} octets, ou 1EiB. En gros, un million de TiB.

Ce projet contient un fork du projet SQLite qui a été modifié pour inclure un prototype du dorsal de la base de données Hc-tree. Il peut être construit et utilisé de la même manière que le code de la bibliothèque SQLite. Le prototype est suffisamment avancé pour être utilisé à titre expérimental et pour effectuer des tests de performance multi-threads, mais il est incomplet.

Article 4 : ChatGPT trouve des bogues dans les bases de données (13/03/2023)

<https://intelligence-artificielle.developpez.com/actu/342409/ChatGPT-aurait-reussi-a-trouver-des-bogues-dans-les-bases-de-donnees-apres-avoir-ete-entraine-par-l-equipe-de-StarRocks-pour-effectuer-des-tests-automatises/>

Les utilisateurs de ChatGPT, un outil d'intelligence artificielle, ont exploré de nouveaux cas d'utilisation pour faciliter et accélérer leurs tâches quotidiennes.

En particulier, l'équipe du projet StarRocks a utilisé ChatGPT pour tester les performances et la robustesse de bases de données en comparaison avec SQLsmith, un outil de tests automatisés de base de données établi.

Les développeurs de chez StarRocks ont demandé à ChatGPT de générer une requête SQL complexe à l'aide de fonctions StarRocks JSON, même si ChatGPT avait initialement indiqué que ces fonctions n'étaient pas prises en charge. Cependant, après avoir fourni des documents JSON de StarRocks, ChatGPT a pu apprendre et générer une requête avec les fonctions JSON de StarRocks.

Les développeurs ont conclu que ChatGPT pourrait potentiellement remplacer SQLsmith, mais qu'il n'était pas encore certain qu'il puisse le remplacer complètement dans toutes les tâches.

xi please generate the select statements with three table join ,aggregate, sort, limit, window function, multi subqueries, PARSE_JSON and JSON_QUERY function for test_basic tables by in Starrocks syntax



Sure, here's an example of a SELECT statement with a three table join, aggregate, sort, limit, window function, and multi subqueries, using the PARSE_JSON and JSON_QUERY functions:

```
sql Copy code

SELECT
  tb1.id_int,
  SUM(tb2.id_bigint) AS total_sum,
  RANK() OVER (ORDER BY tb1.id_int DESC) AS rank,
  JSON_QUERY(tb3.json_col, '$.info.name') AS name,
  (SELECT COUNT(*) FROM tb4 WHERE tb4.id_int = tb1.id_int) AS count_subquery,
  (SELECT MAX(id_datetime) FROM tb5 WHERE tb5.id_int = tb1.id_int) AS max_su
FROM
  test_basic tb1
JOIN test_basic2 tb2 ON tb1.id_int = tb2.id_int
JOIN test_basic3 tb3 ON tb1.id_int = tb3.id_int
WHERE
  tb1.id_int BETWEEN 1 AND 100
GROUP BY
  tb1.id_int,
  tb3.json_col
ORDER BY
  total_sum DESC,
  tb1.id_int ASC
LIMIT 10;
```