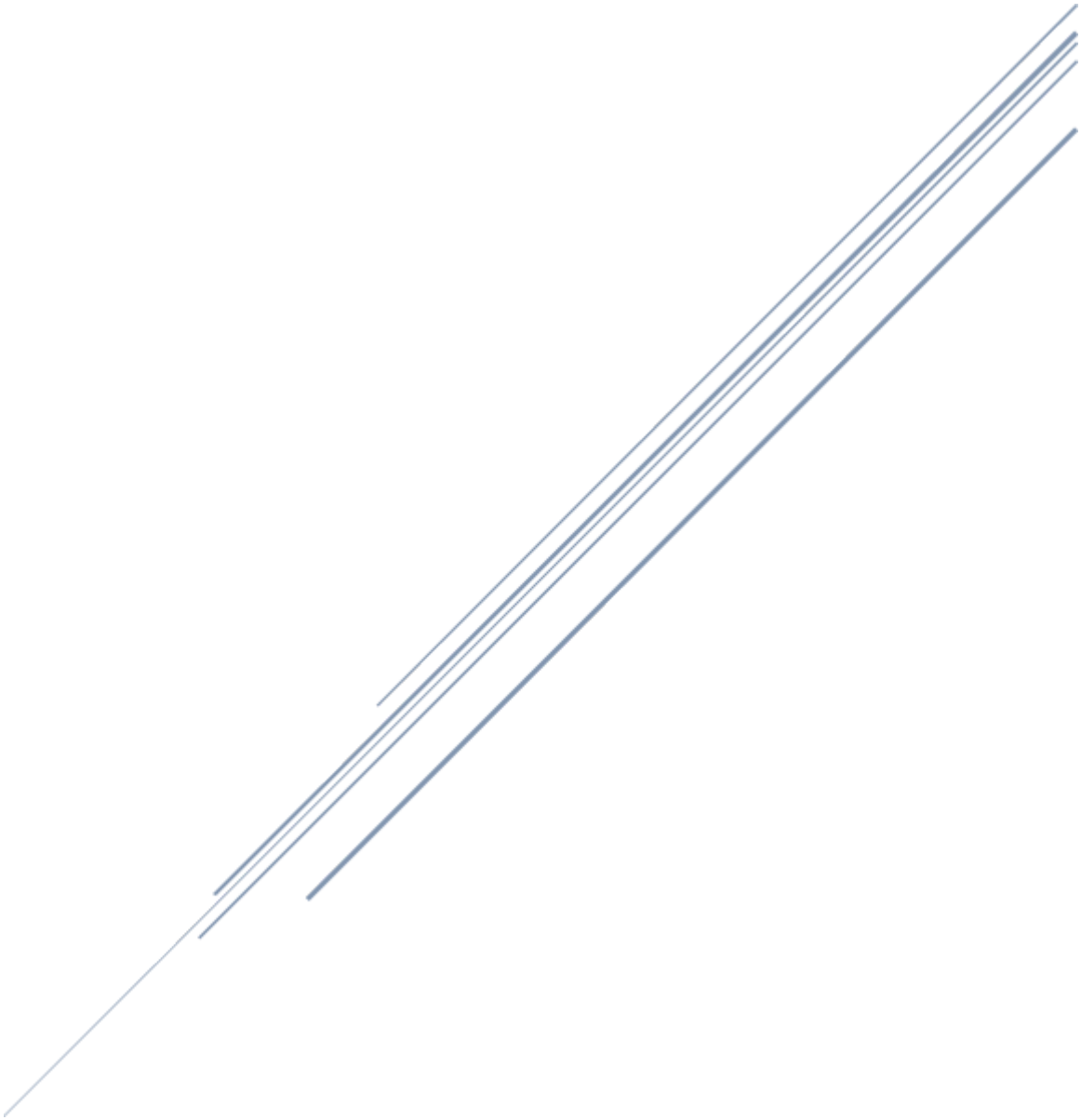
WHAT ‘S REALLY NEW WITH NEWSQL

Par Andrew PAVLO et Matthew ASLETT



Rédigé par :
Lydia OUTAYEB Amine LAGAB

Le terme “NewSQL” est apparu en 2011, il a été utilisé pour la première fois par un des auteurs de cet article dans un rapport qui discute de l'essor des nouveaux systèmes de base de données , depuis, plusieurs autres auteurs ont utilisé ce terme pour parler de leurs systèmes.

l'objectif de cet article est d’étudier et de déterminer quels avantages apporte “Newsql” par rapport aux produits existants depuis plusieurs années.

Avant tout, un peu d’histoire, au début des années 1960 jusqu'en 1979 la classe “relationnelle” à vu le jour , à la fin des années 80s et début des 90s , des SGBD “objet-relationnel” ont été conçus pour transcender les inadéquations entre le modèle relationnel et les données orientés objet, les années 1990 ont marquées l’essor des sgbd relationnels , puis l’an 2000 a vu le jaillissement d’internet et l’apparition des applications web apportant de nouveaux challenges aux DBMs tel que le « goulot d’étranglement » . Pour pallier ce mal, trois solutions majeures ont été mises en place :

- Favoriser l’installation des DBMs dans des machines plus performantes qui s'avèrent être une procédure complexe et coûteuse en temps et en ressources.

- L’approche des middleware qui simule un SGBD distribué en utilisant des clusters et un middleware permettant de communiquer avec les applications en leurs présentant un schéma conceptuel global de la bdd distribué, Cette approche est devenue favorable aux entreprises comme Facebook qui l’a intégré dans son système MySQL . L’approche étant très efficace sur des requêtes faciles , elle reste cependant moins performante sur des requêtes complexes telles que la MAJ sur plusieurs tuples.

- Les nombreuses complications rencontrées constituent une motivation suffisante pour permettre aux entreprises de se lancer vers la création de leurs propres sgbd distribués.

Le retour du Nosql

Les multiples problèmes rencontrés lors de l’utilisation des SGBD relationnel est l’évolution de l’information qui devient de plus en plus complexe à décrire dans un schéma relationnel , ont poussé les entreprises à adopter un autre type de sgbd objet-relationnel , un modèle facilement scalable qui peut manipuler différentes sortes de données tels que les documents , graphes , key/value mais qui gère moins les propriétées ACID contrairement au modèle relationnel, c’est à ce moment là précis que la classe NewSQL fait surface .

Le NewSQL

Le NewSQL établit son fondement sur les avantages combinés du Nosql et les BDDs relationnels classiques .

Le système NewSQL est séparé en trois catégories : les systèmes basés sur une nouvelle architecture, les systèmes utilisant un middleware avec sharding et les « database as a service ».

New architecture

Les SGBD NewSQL de cette catégorie sont basé sur le « shared nothing architecture » dont l’implémentation a été totalement renouvelée, ils sont parfaitement conçus pour l’environnement multi-nœud en utilisant le Query optimisé et la communication entre les nœuds dont l'habileté réside sur le fait de pouvoir envoyer des requêtes directement entre les nœuds sans passer par l’implémentation d’une logique de middleware .

L'un des aspect les plus important de ces SGBD est la gestion de leurs propres stockage en mémoire ou en disque autrement dit la distribution de la bdd sur les ressources des nœuds sans l'utilisation d'un système de fichier distribué (hdfs) , cela permet au DBMs d'utiliser des technologies sophistiqués de réplication et de transmettre les requêtes vers la bdd au lieu de l'inverse , ce qui réduit efficacement le trafic sur le réseau . Cependant, l'utilisation des nouvelles architectures comprend des risques puisque la technologie n'est pas encore à point (pas stable car encore nouvelle).

Sharding middleware

Cette classe fonctionne sur un principe similaire aux middlewares , Un middleware centralisé permet le routage des requêtes vers les partitions concernées, la coordination des transactions, la gestion des réplications et le partitionnement de la bdd. L'utilisation du sharding permet de découper la bdd en petites partitions facilement stockable dans des nœuds, ceci est facilement gérable par le SGBD surtout si l'accès vers la bdd est désigné pour une seule application. Le sharding middleware permet de rendre facilement une le sgbd scalable, cependant elle utilise encore avec les anciens architectures DBMS sur ses nœuds.

Data base as a service

Représente l'utilisation des SGBD NewSQL comme DBaaS, une pratique utilisée par les plus grandes firmes du cloud computing tel qu'Amazon qui fournit un SGBD complet, scalable permettant aux entreprises de ne plus considérer un coût pour l'implémentation d'un SGBD et de son maintien. Cependant, certains types de DBaaS ne sont pas considérés comme NewSQL puisqu' ils sont basés sur des architectures traditionnelles orientées disque.

Les deux auteurs de l'article s'intéressent ensuite aux fonctionnalités principales des systèmes NewSQL en les comparant avec celles des DBMS classiques :

- **Le stockage en mémoire principale**

La majorité des SGBDs commerciaux utilisent l'architecture de disk-oriented storage, qui se base sur le stockage block-adressable comme des disques durs ou l'SDD, l'utilisation de cette architecture bien qu'elle permet de stocker la base de données sur un large support s'avère moins performante en termes d'accès au disque L/E contrairement à l'architecture memory Storage. L'utilisation de cette architecture traditionnelle n'était pas un choix mais une obligation compte tenu du coût élevé des supports mémoire, désormais , le coût d'un support mémoire est abordable pour la plupart des entreprises , ce qui permet en une éventuelle ouverture pour favoriser l'architecture main memory Storage et d'en bénéficier des optimisations qu'elle apporte .

Les SGBD NewSQL ont tendance à utiliser une architecture « orientée mémoire », l'une des caractéristiques de cette architecture est la capacité de détecter les tuples inutilisables afin de les «éjecter » temporairement de la bdd en mémoire et de les stocker dans un disque , ceci permet de libérer de la mémoire nécessaires au futures tuples.

- **Le sharding (partitionnement) :**

Le partitionnement c'est basiquement l'éclatement de la bdd horizontalement en plusieurs parties disjointes, chaque partition correspond à un nœud qui assure l'exécution des requêtes locales au sein de celle-ci . NewSQL possède un système pour distribuer des requêtes aux nœuds nécessaires et regroupe ensuite les partitions pour renvoyer le

résultat, le schéma de la bdd est transposé en une structure arborescente dont chaque nœud descendant (enregistrements de la table) a une liaison avec son nœud père (table).

- **Le crash recovery (Récupération des crashs) :**

Les SGBDs NewSQL doivent garantir la tolérance aux pannes et la disponibilité permanente de la bdd, on citera deux approches permettant de récupérer la bdd en cas de crash :

L'approche traditionnelle concerne les bdd basés sur une architecture single node sans replicas, en cas de panne, le sgbd charge le dernier point de control et exécute son WAL afin de retourner l'état de la bdd au moment de la panne. la récupération d'un noeud dans une architecture distribué multi noeud ne pourra pas se faire avec cette approche , quand un nœud tombe en panne, il ne doit pas affecter les autres nœuds, le récupérer ne rendra pas l'état précédent de la bdd puisque cela dépend de tous les états des nœuds dont l'etat a progressé depuis le crash .

L'autre approche concerne les bdd basées sur l'architecture distribuée avec replicas , il existe deux manières pour récupérer l'état du noeud en panne , la première est de charger le dernier point de contrôle et le WAL du noeud , ensuite, en utilisant les journaux"logs" des autres noeuds , le noeud exécutera les entrées manquées jusqu'à ce qu'il atteint éventuellement un état cohérent par rapport aux autre noeuds . La seconde manière ne prend pas en considération le dernier point de control , en revanche , le système lui affecte un nouvel état de récupération , cette manière est aussi utilisée pour créer une nouvelle réplique de nœud . Cette approche est favorisée par les sgbd NewSQL qui combinent les technologies existantes ainsi que leurs propres implémentations .

- **secondary indexes:**

Un index secondaire est une technique d'indexation qui repose sur des attributs différents de la clé primaire , cependant, la difficulté de l'implémenter sur des architectures distribuées réside sur la manière de choisir le partitionnement basé sur deux critères : Où stocker la table des indexes et Comment maintenir la mise à jour de celles-ci.

Les systèmes NewSQL qui se basent sur les nouvelles architectures sont décentralisés et utilisent tous l'indexage secondaire partitionné , autrement dit, chaque nœud contient une portion d'index correspondante à sa partition . Le choix d'utiliser des réplications totales des indexes ou de les repartitionner sur les nœuds réside sur la différence entre ces derniers, l'utilisation des réplicas permet de trouver en un seul nœud le tuple correspondant , si une MAJ à été répercutée sur les attributs correspondant à ceux des indexes , le système doit absolument MAJ chaque réplication d'indexes. Inversement, si il n'y a pas de réplication d'index , la recherche doit solliciter plusieurs nœuds pour trouver le tuple correspondant , en cas de MAJ , seulement la portion d'index correspondante sera modifié.

L'un des SGBD qui fusionne la réplication et le partitionnement des index secondaires est clusterix , ce sgbd utilise deux tables d'indexage secondaire ,une table correspondant aux nœuds répliqués dont les attributs sont ceux utilisés pour le partitionnement et une autre table d'indexage partitionné sur l'ensemble des nœuds . Cette approche hybride permet de trouver sur un seul nœud la partition , puis de trouver sur le nœud correspondant à la partition le tuple recherché .

- **la réplication:**

L'une des solutions envisageables implémentée par les entreprises dont l'efficacité réside sur le fait de pouvoir assurer la disponibilité et la durabilité de la bdd. Deux critères de décisions s'imposent à l'implémentation d'une réplication au

sein d'une architecture distribuée : le niveau de consistance choisi ainsi que la façon dont les SGBDs assurent la consistance des données. Dans le cas des SGBD NewSQL, il est plus important d'assurer un seul état cohérent de la bdd à chaque moment, leurs choix est évidemment porté sur le strong consistency réplication. Pour propager la réplication, la plupart des NewSQL utilisent un schéma non déterministe de contrôle de concurrence, favorisant ainsi la réplication active passive ou bien master-slave" replication..

- **Concurrency control (Contrôle d'accès concurrent) :**

Cet aspect des NewSQL n'est pas nouveau, il existe deux approches pour le réaliser , la première étant l'architecture centralisée dont les transactions sont contrôlées par un coordinateur, la seconde approche qui est favorisée par les NewSQL est l'architecture décentralisée qui se base sur la synchronisation & coordination entre les nœuds pour éviter les conflits.

NewSql favorise le protocole hybride MVCC & 2pl pour mettre en place cette coordination décentralisée. Basée sur 2pl & MVCC, SpannerSQL est l'un des sgbd récents des NewSQL, pour plus de précision en termes de synchronisation, des Atomic lock sont utilisées. Contrairement à spannerSQL , VoltDB utilise les TO ainsi qu'une architecture hybride dont le concept est d'utiliser une planification centralisée avec des transactions multi partitions et l'inverse qui est l'utilisation d'une planification décentralisée avec des transactions single partitions .

Future Trends :

dans le futur proche , les sgbd auront la possibilité d'exécuter des requêtes analytiques en temps réel (HTAP)sur des données fraîchement obtenues .

il existe deux approches qui s'appuient sur le HTAP :

1) séparer les transactions des requêtes analytiques : en utilisant un sgbd front end oltp pour le stockage des résultats obtenues des transactions , puis se charge de les migrer vers un dbms back end en s'appuyant sur une ETL . Le sgbd back end OLAP se chargera d'exécuter les requêtes analytiques (olap complexe queries) . toute modification est répercutée sur le sgbd front end .
2) utilisation d'un seul sgbd htap dbms pour les transactions et les requêtes analytiques qui implémente les performances combinées d'OLTP (lock free , in memory storage) et olap (columnar storage , vectorised execution)

La dernière approche est plus intéressante puisqu'elle utilise un seul sgbd HTAP pour les requêtes olap et oltp , on peut trouver des versions commercialisées en tant que "systèmes HTAP" telles que sap hana & memsql .

Pour conclure, les auteurs indiquent que les systèmes de base de données NewSQL ne sont pas un écart radical par rapport au système existant mais représentent plutôt la prochaine étape dans le développement continu des technologies de base de données. L'objectif de NewSQL est qu'ils regroupent beaucoup de technologies déjà existantes, mais dans des plates-formes simples. Malgré que pour l'instant a du mal à se populariser par rapport à la solution NoSQL, cependant NewSql reste une étape vers la solution de stockage qui regroupe les DBMS OLAP, OLTP, NoSQL et NewSQL.