Back to Company

COMPANY

•SEPTEMBER 29, 2021

SQL ou NoSQL : avantages et inconvénients

Rich Edwards

Rich Edwards

SQL ou NoSQL : avantages et inconvénients

Quel est le meilleur moyen de stocker, protéger et accéder à vos données ? Une décision fondamentale et déterminante. Parce que les données sont la pierre angulaire de la réussite de toute organisation moderne. Pour la plupart des entreprises, le choix devra se faire entre bases de données SQL et NoSQL. Chaque option a ses avantages et ses inconvénients.

Les bases de données SQL ont fait leurs preuves, depuis les années 1970. Elles se composent de tables hautement structurées, constituées de lignes et de colonnes, reliées les unes aux autres par des attributs communs. Chaque colonne doit obligatoirement avoir une valeur pour sa ligne correspondante. Les bases de données NoSQL (« not only SQL » ou « non-SQL ») sont arrivées plus tard, brisant le carcan des tables relationnelles pour permettre de stocker ensemble et d’accéder à tous les types de données, structurées et non structurées. Elles sont extrêmement flexibles et faciles à gérer et à modifier pour les développeurs. Apprenez-en plus sur les bases de données SQL et NoSQL et leurs différences fondamentales.

Le meilleur choix ? Eh bien, cela dépend de toute une série de facteurs, notamment de vos besoins en matière de requêtes, de disponibilité et de conformité, de la variété des types de données que vous utilisez, ou encore de la croissance attendue.

Comparatif SQL / NoSQL

SQL

NoSQL

Avantages

Inconvénients

Avantages

Inconvénients

Schéma standardisé

Matériel

Disponibilité continue

Pas de langage standardisé

Vaste communauté d’utilisateurs

Normalisation des données

Vitesse de requête

Communauté d’utilisateurs plus petite

Aucun code requis

Rigidité

Agilité

Inefficacité en cas de requêtes complexes

Conformité ACID

Extension exigeante en ressources

Coût

Incohérence dans l’extraction des données

Regardons de plus près les avantages et les inconvénients des modèles SQL et NoSQL afin de vous aider à faire le bon choix.

Avantages de SQL

Schéma standardisé

Le schéma standardisé des bases de données SQL les rend certes rigides et difficiles à modifier, mais il présente aussi quelques avantages. Toutes les données ajoutées à la base de données doivent correspondre au fameux schéma de tables reliées, constituées de lignes et de colonnes. On peut trouver cela limitatif ou restrictif, mais c’est utile lorsque la cohérence, l’intégrité, la sécurité et la conformité des données priment.&nbsp;

Vaste communauté d’utilisateurs

À près de 50 ans, le langage de programmation SQL est extrêmement mature et toujours très utilisé. Il possède une forte communauté, avec une multitude d’experts tout disposés à partager conseils et bonnes pratiques bien établies. Les opportunités d’affûter ses compétences et de collaborer sont nombreuses. Si nécessaire, des consultants et fournisseurs SQL peuvent proposer un soutien supplémentaire. Avec SQL, vos développeurs seront en mesure de trouver les réponses dont ils ont besoin.

Aucun code requis

SQL est un langage convivial. La gestion et l’interrogation de la base de données se fait par simples mots-clés, avec très peu de code voire sans. La plupart des développeurs ont appris le langage SQL à l’université.

Conformité ACID

La nature extrêmement structurée des tables de bases de données relationnelles permet de garantir la conformité des bases de données SQL avec les propriétés ACID. Ce niveau de conformité assure la synchronisation des tables et garantit la validité des transactions. C’est sûrement le bon choix si vous exécutez des applications qui ne laissent pas de place à l’erreur et requièrent le plus haut niveau d’intégrité des données.

Voici les propriétés ACID :

Atomicité : toutes les modifications apportées aux données et aux transactions sont exécutées complètement, en une seule opération. Si ce n’est pas possible, aucune modification ne sera effectuée. C’est tout ou rien.

Cohérence : les données doivent être valides et cohérentes du début à la fin d’une transaction.

Isolation : les transactions sont exécutées simultanément, sans être en concurrence les unes avec les autres. Au contraire, elles se comportent comme si elles se déroulaient successivement.

Durabilité : lorsqu’une transaction est achevée, les données associées sont permanentes et ne peuvent pas être modifiées.

Inconvénients de SQL

Matériel

La norme pour les bases de données SQL est d’évoluer verticalement. La contenance ne peut être étendue que par l’augmentation des capacités – RAM, CPU et SSD – sur le serveur existant ou par la migration vers une version plus grande et plus chère. Plus vos données augmentent, plus vous devrez créer d’espace sur disque dur et plus vous aurez besoin de machines rapides pour faire fonctionner des technologies toujours plus sophistiquées, en évolution permanente. Le fournisseur de base de données que vous utilisez vous demandera certainement de mettre régulièrement à niveau votre matériel, simplement pour assurer la compatibilité avec ses nouvelles versions. Dans cet environnement, le matériel devient vite obsolète. Chaque mise à niveau est coûteuse, financièrement et en ressources. Il faut également penser aux coûts réguliers liés à la maintenance quotidienne et à l’exploitation du matériel SQL. C’est une course sans fin.

Normalisation des données

Développées à une époque où les coûts du stockage des données étaient élevés, les bases de données relationnelles tentent de neutraliser la duplication de données. Chaque table comporte des informations différentes et les tables peuvent être reliées et interrogées par le biais de valeurs communes. Mais plus les bases de données SQL s’agrandissent, plus les recherches et les jointures nécessaires entre plusieurs tables peuvent ralentir les choses.&nbsp;

Rigidité

Le schéma d’une base de données SQL doit être défini avant son utilisation. Une fois mis en place, il est rigide. Toute modification sera difficile et exigera des ressources considérables. Il faut donc investir beaucoup de temps dans la planification en amont, avant que la base de données soit mise en production. Par conséquent, ce modèle n’entre en ligne de compte que si toutes vos données sont également structurées et que vous n’attendez pas beaucoup de changement, que ce soit en termes de volume ou de types de données.

Extension exigeante en ressources

Comme nous l’avons évoqué plus haut, les bases de données SQL suivent normalement une évolution verticale via l’extension des investissements en matériel. Cela coûte cher et prend du temps. Dans certains cas, une organisation peut tenter une extension horizontale de sa base de données SQL par le biais d’un partitionnement. Ce surcroît de complexité amplifie le temps et les ressources engagés. Cet effort nécessitera probablement du code, et donc des développeurs très compétents et très bien payés. Lorsque le volume de données augmente, faire évoluer votre base de données revient en quelque sorte à jouer sans fin au chat et à la souris, la configuration parfaite restant toujours hors d’atteinte. Les bases de données NoSQL, en revanche, permettent cette évolutivité horizontale, ce qui facilite l’extension des capacités à moindres frais. Elles se prêtent bien au cloud computing et à la gestion de très grands ensembles de données à croissance rapide.

Avantages de NoSQL

Disponibilité continue

Avec NoSQL, les données sont réparties entre plusieurs serveurs et régions, de sorte qu’il n’y a pas de point de défaillance unique. Ainsi, les bases de données NoSQL sont plus stables et résistantes, elles offrent une disponibilité continue sans temps d’arrêt.

Vitesse de requête

Les bases de données NoSQL étant dénormalisées, et la duplication de données n’étant pas un souci, toutes les informations nécessaires à une requête particulière sont souvent déjà stockées ensemble – aucune jointure nécessaire. Cela peut faciliter les recherches, notamment quand on travaille avec d’importants volumes de données. Cela signifie également que NoSQL peut être très rapide pour les requêtes simples. Ne nous y trompons pas, les bases de données SQL peuvent aussi répondre très vite à des requêtes. Elles prennent également en charge des requêtes de haute complexité pour des données structurées. Cependant, la vitesse de requête peut rapidement diminuer lorsque les bases de données s’agrandissent et nécessitent davantage de jointures complexes.

Agilité

Les bases de données NoSQL ont été développées lorsque les coûts de stockage de données commençaient à baisser fortement et que le coût des développeurs grimpait. La duplication de données n’était plus problématique. Au contraire, elles ont été conçues pour donner aux développeurs autant de flexibilité que possible pour booster leur créativité et leur productivité. Libérés des contraintes des lignes et colonnes, les schémas de bases de données NoSQL n’ont pas à être prédéfinis. Ils sont au contraire dynamiques et capables de gérer tous les types de données – structurées, semi-structurées, non structurées et polymorphes. Vous pouvez lancer des bases de données NoSQL sans perdre de temps à définir leur structure et ajouter simplement des types de données et des champs sans temps d’arrêt, à la volée. Tout cela fait de NoSQL la solution idéale pour les équipes de développement modernes et agiles. Les développeurs peuvent se lancer dans la création d’une base de données sans avoir à consacrer du temps et des efforts à une planification préalable. Elle leur permettra d’effectuer rapidement des modifications au fur et à mesure que les exigences changent et que de nouveaux types de données viennent s’ajouter. La flexibilité et la nature adaptative des bases de données NoSQL en fait une excellente option pour les organisations qui ont affaire à différents types de données et qui comptent ajouter continuellement de nouvelles caractéristiques et fonctionnalités.

Les bases de données NoSQL ne sont pas toutes identiques. À la différence des bases de données SQL, elles ne sont pas limitées par un modèle de données centralisé, rigide, probablement hébergé sur un seul serveur. Au contraire, NoSQL offre la flexibilité de connecter des modèles de bases de données divers qui peuvent être répartis sur plusieurs serveurs. NoSQL inclut plusieurs types de bases de données, permettant aux développeurs de trouver le cocktail idéal pour leurs données et cas d’utilisation. Les principaux types de bases de données NoSQL sont les formats clé-valeur, document, tabulaire (ou à larges colonnes), graphe ou multi-modèle.

Faible coût

Les bases de données NoSQL évoluent horizontalement, ce qui permet une extension des capacités à peu de frais. Au lieu de mettre à niveau du matériel onéreux, elles peuvent être élargies de manière économique en ajoutant simplement des serveurs standard ou des instances cloud. Les bases de données NoSQL open source fournissent en outre des options abordables pour de nombreuses organisations. Elles se prêtent bien au cloud computing et à la gestion de très grands ensembles de données à croissance rapide.

Inconvénients de NoSQL

Pas de langage standardisé

Il n’y a pas de langage standard pour effectuer des requêtes NoSQL. La syntaxe utilisée pour interroger des données varie pour les différents types de bases de données NoSQL. À la différence de SQL, où il n’y a qu’un langage à maîtriser, facile à apprendre, NoSQL présente une courbe d’apprentissage plus abrupte. Par exemple, un développeur pourra avoir du mal à maîtriser rapidement le travail sur une base de données à larges colonnes s’il était habitué auparavant uniquement à créer et à gérer des base de données orientées graphe.

Communauté d’utilisateurs plus petite

Les développeurs utilisent les bases de données NoSQL depuis plus de dix ans et la communauté connaît une croissance rapide. Elle reste cependant moins mature que la communauté SQL. Il peut donc être plus difficile de résoudre des problèmes non documentés. Il y a également moins de consultants et d’experts côté NoSQL.

Inefficacité en cas de requêtes complexes

La flexibilité a un prix. L’efficacité des requêtes pâtit de la diversité des structures de données trouvées dans les bases de données NoSQL. Contrairement aux bases de données SQL, il n’existe pas d’interface standard pour effectuer des requêtes complexes. Même les requêtes NoSQL simples demandent généralement de l’expérience en programmation. Autrement dit, une équipe plus technique et coûteuse, comme des développeurs ou des data scientists, pour effectuer les requêtes.

Incohérence dans l’extraction des données

La nature distribuée des bases de données NoSQL permet de disposer des données plus vite. Mais il peut aussi être plus difficile de garantir que les données soient toujours cohérentes. Les requêtes peuvent ne pas toujours renvoyer des données à jour et il est possible de recevoir des informations imprécises. De par son approche distribuée, la base de données peut rendre des valeurs différentes au même moment, en fonction du serveur interrogé. C’est l’une des raisons pour lesquelles NoSQL n’atteint pas une conformité de niveau ACID. La cohérence, le « C » dans ACID, indique que les données doivent être valides et cohérentes du début à la fin d’une transaction. En revanche, la plupart des bases de données NoSQL adhèrent au modèle de cohérence BASE, où le « E » correspond à « eventual consistency ». Autrement dit, les données seront cohérentes ultérieurement. Dans le monde réel, il s’agit souvent d’un petit retard de quelques millisecondes. Pour bon nombre d’applications, cela ne jouera pas un grand rôle, comme pour les posts de réseaux sociaux publiés, ou la mise à jour d’un panier d’achats en ligne. Dans ces situations, une disponibilité plus rapide pour la majorité du réseau l’emporte sur l’importance de fournir des données exactement identiques à tous les utilisateurs au même moment. Cependant, cela peut avoir son importance dans certains cas, comme lorsque vous faites un achat sur un stock en ligne. NoSQL privilégie la vitesse et la disponibilité au détriment de la cohérence. Chaque organisation doit savoir si cela est compatible avec ses objectifs.

Évaluer vos options

Les bases de données SQL et NoSQL répondent toutes deux parfaitement à des besoins et cas d’utilisation spécifiques. Selon l’environnement de données de votre organisation et ses objectifs, certains avantages et inconvénients de chacune peuvent être amplifiés. Vous trouverez peut-être que la meilleure solution est d’utiliser les deux et de laisser chaque type de base de données livrer le meilleur d’elle-même. Beaucoup d’organisations utilisent à la fois des bases de données SQL et NoSQL dans leur architecture cloud, parfois même au sein de la même application. Cela dit, la meilleure option serait sans doute de trouver une solution, comme DataStax Astra DB, qui profite des avantages inhérents à NoQSL, comme la flexibilité, la disponibilité continue et l’évolutivité, tout en réduisant à un minimum ses inconvénients.

Astra est une base de données multi-cloud en tant que service (DBaaS), fondée sur Apache Cassandra et Kubernetes et dotée d’une architecture de microservices. Elle vous permet d’être opérationnel en quelques clics dans le cloud de votre choix – Azure, Google Cloud Platform ou AWS. Une fois là, elle simplifie radicalement le développement d’applications. Intégrée à Astra, Stargate est une couche d’API de données open source qui permet de se passer des pilotes et d’interroger vos données ou de créer des tables et schémas sans apprendre le langage CQL (Cassandra Query Language). Stargate vous permet d’interagir avec les données par le biais d’API modernes, notamment Schemaless JSON, REST et GraphQL. Grâce au SAI (Storage Attached Indexing) d’Astra, vous pouvez interroger n’importe quelle colonne de la table, sans être limité à la clé primaire.

Regardez Astra de plus près.

SHARE

More Company

View All

DataStax Astra DB Nabs Three Prestigious 2023 TrustRadius “Best of” Awards, Dominates the Vector Databases Category

COMPANY

NOVEMBER 8, 2023

DataStax Astra DB Nabs Three Prestigious 2023 TrustRadius “Best of” Awards, Dominates the Vector Databases Category

DataStax Has Achieved ISO 27001 Certification

COMPANY

OCTOBER 17, 2023

DataStax Has Achieved ISO 27001 Certification

Keeping the Momentum Rolling with Google Cloud

COMPANY

AUGUST 24, 2023

Keeping the Momentum Rolling with Google Cloud

My Internship Journey: A Dive into AI

COMPANY

AUGUST 17, 2023

My Internship Journey: A Dive into AI

Build generative AI apps at scale on Astra DB vector

Astra DB gives developers the APIs, real-time data and complete ecosystem integrations to put accurate Gen AI apps in production - FAST.

LEARN MORE

GET STARTED FOR FREE

COMPANY

About Us

Leadership

Board of Directors

Contact Us

Partners

Careers

Newsroom

RESOURCES

Blog

Events

eBooks

Whitepapers

Webinars

Legal

Security

Awesome Astra

CLOUD PARTNERS

Amazon Web Services

Google Cloud Platform

Microsoft Azure

TRENDING GUIDES

What is Vector Search?

What is a Vector Database?

What is Retrieval Augmented Generation?

© 2023 DataStax

Privacy Policy

Terms of Use

Trademark Notice

Brand Resources

Cookies Settings

France

Texte d'origine

Évaluez cette traduction

Votre avis nous aidera à améliorer Google Traduction