databricks

e-book

Le Big Book de la BI et du Data Warehousing

Table des matières

Les données sont essentielles au succès de toute entreprise. Du fait de leur rôle central, il est aujourd'hui crucial de les encadrer par une gestion et une gouvernance efficaces. Pour relever ce défi, la Data Intelligence Platform de Databricks simplifie la gestion, la consultation et l'exploitation des données et de l'IA. Bâtie sur l'architecture lakehouse, elle réunit les atouts des data lakes et des data warehouses pour réduire les coûts et accélérer les initiatives de données et d'IA. La plateforme unifie la gouvernance des données et de l'IA, et fournit un moteur de requête polyvalent pour l'ETL, SQL, le machine learning et la BI.

Présentation de la Data Intelligence Platform

La Data Intelligence Platform de Databricks donne aux entreprises un moyen efficace de gérer, consulter et exploiter les données et l'IA. Bâtie sur l'architecture lakehouse, la plateforme ajoute une couche de gouvernance unifiée couvrant les données et l'IA, ainsi qu'un moteur de requête commun pour l'ETL, SQL, l'IA et la BI. Elle réunit ainsi les atouts des data lakes et des data warehouses pour réduire les coûts et accélérer les initiatives de données et d'IA.

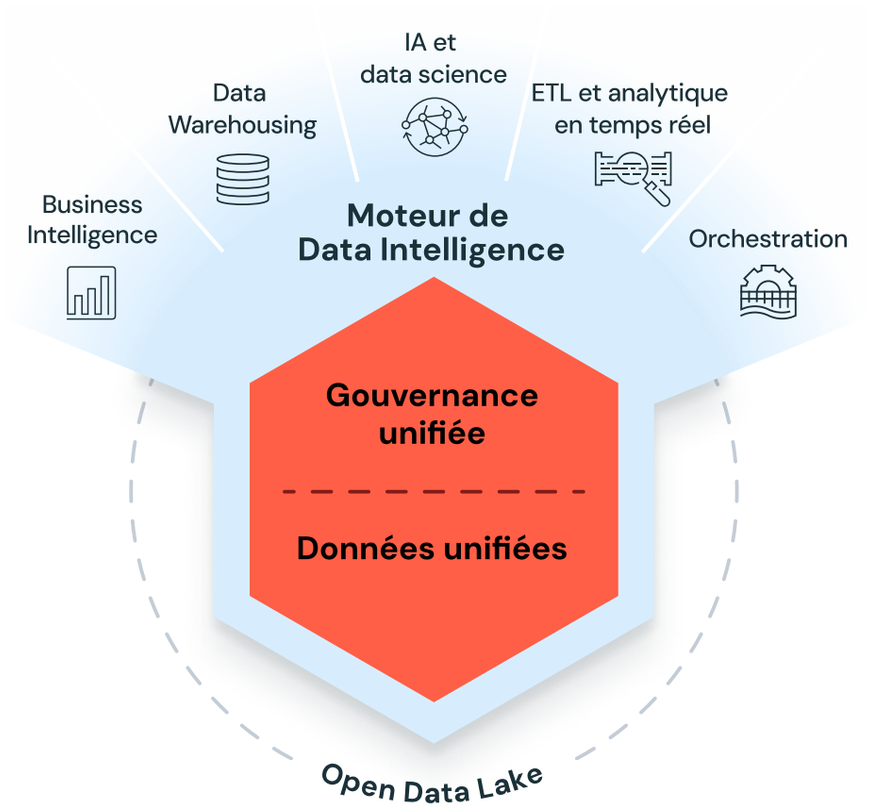
En associant l'IA générative au pouvoir d'unification d'un lakehouse, la Data Intelligence Platform fournit un moteur de Data Intelligence, DatabricksIQ, qui comprend la sémantique propre aux données de votre entreprise. DatabricksIQ analyse automatiquement tous les aspects des données : contenu, métadonnées et modèles d'utilisation (requêtes, rapports et lineage). Grâce à cette analyse complète, la plateforme apprend en continu, s'améliore et s'enrichit de nouvelles fonctionnalités afin d'optimiser la gestion des données et les applications d'IA. Cette compréhension détaillée des données est la base des capacités de la Data Intelligence Platform de Databricks :

- **Accès en langage naturel :** Grâce aux modèles d'IA, la Data Intelligence Platform permet de travailler avec des données en langage naturel et comprend le jargon et les acronymes de chaque organisation. La plateforme étudie l'utilisation des données dans les charges de travail existantes pour apprendre la terminologie de l'organisation et fournir une interface en langage naturel à tous les utilisateurs, spécialistes ou non.
- **Catalogage et découverte sémantique :** Parce qu'elle comprend le modèle de données, les métriques et les KPI de l'entreprise, l'IA générative offre des fonctionnalités de découverte inégalées et sait identifier les écarts dans l'utilisation des données.
- **Gestion automatisée et optimisation :** Les modèles d'IA peuvent optimiser la disposition, le partitionnement et l'indexation des données en fonction de leur utilisation afin de minimiser les réglages manuels.

databricks

- **Améliorer la gouvernance et la protection de la vie privée :** Les plateformes de Data Intelligence peuvent détecter, classer et prévenir automatiquement l'utilisation abusive des données sensibles, tout en simplifiant la gestion grâce au langage naturel.

- **Une prise en charge optimale des charges d'IA :** Les plateformes de Data Intelligence peuvent améliorer une application d'IA d'entreprise en la connectant aux données métier pertinentes et la faire profiter de la sémantique (métriques, KPI…) apprise par la plateforme pour fournir des résultats précis. Les développeurs d'applications d'IA n'ont plus besoin de faire du « bricolage » avec une ingénierie de prompt précaire.



La plateforme Databricks simplifie également le développement d'applications d'IA d'entreprise. En effet, les applications d'IA créées sur la plateforme comprennent leurs données. La plateforme Databricks offre plusieurs moyens d'intégrer directement les données d'entreprise dans les systèmes d'IA :

- RAG (génération augmentée par récupération) de bout en bout pour créer des agents conversationnels de haute qualité avec vos données
- Entraînement de modèles personnalisés sur la seule base des données de l'organisation, ou par pré-entraînement continu de modèles existants tels que MPT et Llama 2, pour spécialiser les applications d'IA dans un domaine cible.
- Inférence serverless efficace et sécurisée sur les données de votre entreprise, adossée à une gouvernance unifiée et une fonction de suivi de la qualité.
- MLOps de bout en bout basé sur le projet open source MLflow ; toutes les données produites sont automatiquement exploitables, suivies et contrôlables dans le lakehouse.

Ce guide de référence présente les bonnes pratiques du data warehousing sur la Data Intelligence Platform de Databricks et tire ses cas d'usage d'exemples réels. Découvrez comment la plateforme aide tous les profils d'entreprise à convertir des données brutes en informations exploitables avec SQL, de l'importation au traitement dans des projets d'IA, de LLM, d'analyse et de BI. Vous y trouverez des architectures de référence et des échantillons de code pour explorer tous les aspects du cycle de vie des données sur la Data Intelligence Platform.

Pour en savoir plus sur Databricks SQL, lisez notre e-book Pourquoi le lakehouse est votre prochain data warehouse.

SECTION

02

Intégrer l'intelligence et l'automatisation au data warehousing

databricks

La réussite des initiatives de données et d'IA basées sur le lakehouse repose sur une architecture de données simplifiée, le contrôle qualité et la gouvernance, ainsi que l'évolutivité et la performance. Ensemble, ces piliers forment les fondements des stratégies de données des entreprises et les guident dans le grand labyrinthe du paysage moderne de la gestion des données et de l'analytique.

Architecture de données simplifiée

L'architecture de données du lakehouse résout les problèmes de silos tout en offrant les capacités d'un data warehouse. Elle consiste d'abord à centraliser les données dans un data lake basé sur le cloud. Cette base, fournie par **Delta Lake**, permet d'adosser des cas d'usage d'analytique et d'IA sur une seule source de données pour réduire les coûts de stockage et rationaliser la data engineering.

L'architecture lakehouse intègre une structure de gouvernance et de sécurité unifiée, **Unity Catalog**, qui exerce un contrôle granulaire sur les données et offre un accès rapide aux différentes équipes.

L'approche holistique de l'architecture lakehouse englobe l'ensemble du cycle de vie, de la transformation et de l'impact des données sur diverses charges de travail d'analytique et d'IA. Comme tous les workloads partagent les mêmes données et respectent des politiques de sécurité et de gouvernance uniformes, vous pouvez compter sur l'exactitude des données. L'atténuation des silos fonctionnels fluidifie la collaboration et favorise la productivité dans la livraison des produits de données.

Une architecture de données simplifiée présente plusieurs avantages :

- Reposant sur du code open source et des standards ouverts, le lakehouse élimine les silos pour faciliter les opérations de données et d'IA.
- Intégration, stockage, traitement, gouvernance, partage, analytique et IA : tout est centralisé sur une même plateforme. Elle unifie la gestion des données, structurées ou non, offre une vue complète sur leur lineage et leur provenance. Elle s'accompagne d'outils pour prendre en charge Python et SQL, les notebooks, les IDE, les charges en batch et en streaming et tous les principaux fournisseurs de cloud.
- Grâce à l'optimisation automatisée des performances et du stockage, la plateforme offre un TCO compétitif et définit un véritable standard pour le data warehousing et l'IA, grands modèles de langage (LLM) inclus.

Gouvernance et qualité des données

Du fait de leur rôle stratégique, de leur volume et de leur diversité, les données exigent une gouvernance et des contrôles de qualité rigoureux. Les organisations doivent faire de l'exactitude, de la fiabilité et de la conformité une priorité pour maximiser le potentiel de leur lakehouse. Des données de piètre qualité fausseront les insights, et une gouvernance laxiste peut entraîner des failles de conformité et de sécurité. La réponse de la Data Intelligence Platform de Databricks : Unity Catalog, un framework intégré pour gérer et améliorer la qualité des données tout au long de leur cycle de vie. Avec la gouvernance sur l'architecture lakehouse, vous pouvez :

- Découvrir, classer et rassembler vos assets de données et d'IA résidant dans différents clouds et plateformes pour centraliser leur exploration et l'extraction d'insights en langage naturel.
- Simplifier la gestion des accès grâce à une interface unifiée, afin d'uniformiser et sécuriser l'accès à l'ensemble des clouds et plateformes en appliquant des contrôles granulaires à l'aide de politiques low-code évolutives.
- Exploiter l'IA pour automatiser la supervision des données et des modèles ML, recevoir des alertes en cas de problème, rationaliser le débogage et mettre en place une observabilité complète à l'aide de tables système intégrées.
- Partager efficacement les assets de données et d'IA entre différents clouds, régions et plateformes à l'aide de Delta Sharing (open source) dans Unity Catalog, pour sécuriser la collaboration et créer de la valeur sans processus complexe ni réplication.

Évolutivité et performance

Face à l'augmentation des volumes de données, l'architecture lakehouse distribue les fonctions de calcul indépendamment du stockage, afin de garantir des performances constantes à des coûts optimaux. La Data Intelligence Platform de Databricks est conçue dans une optique d'élasticité et permet aux organisations de dimensionner leurs opérations de données en fonction de leurs besoins. Cette évolutivité de manifeste à plusieurs titres :

SERVERLESS

La plateforme Databricks utilise des ressources informatiques serverless basées sur le cloud, pour ajuster les charges de travail de manière élastique en fonction des besoins en calcul. Cette affectation dynamique des ressources garantit la rapidité du traitement et de l'analyse des données, même en cas de pic de demande.

CONCURRENCE

Grâce au calcul serverless et aux optimisations par IA, la plateforme Databricks permet de paralléliser le traitement des données et l'exécution des requêtes. Cela permet à plusieurs utilisateurs et équipes de lancer simultanément des tâches d'analytique, sans dégradation de performance.

STOCKAGE

La plateforme s'intègre parfaitement aux lacs de données, pour stocker les gros volumes de données à moindre coût, tout en garantissant leur disponibilité et leur fiabilité. Elle optimise également le stockage dans un souci de performance pour réduire encore les coûts.

L'évolutivité est essentielle, mais elle n'est rien sans la performance. À cet égard, la plateforme Data Intelligence de Databricks se démarque par un large éventail d'optimisations basées sur l'IA :

OPTIMISATION DU TRAITEMENT DES REQUÊTES

La plateforme emploie des techniques d'optimisation basées sur le machine learning pour accélérer l'exécution des requêtes. Elle mise sur l'indexation automatique, la mise en cache et le refoulement des prédicats pour traiter efficacement les requêtes et produire rapidement des insights.

AUTOSCALING

La plateforme Databricks adapte intelligemment les ressources serverless à vos charges de travail : vous bénéficiez de performances optimales, mais vous ne payez que pour le calcul que vous utilisez.

PHOTON

Ce nouveau moteur de traitement massivement parallèle (MPP), natif de la plateforme Databricks, offre des performances de requête extrêmes à faible coût. Il s'utilise directement sur votre data lake, pour l'importation des données, l'ETL, le streaming, la data science et les requêtes interactives. Photon est compatible avec les API Apache Spark™, sans modifier le code ni être tributaire d'un fournisseur.

DELTA LAKE

Avec Unity Catalog et Photon, Delta Lake offre le meilleur rapport performance-prix dès le départ, sans ajustement manuel. La plateforme Databricks utilise des modèles d'IA pour relever les défis courants liés au stockage de données. Le résultat : des performances supérieures sans avoir à gérer les tables, même si elles évoluent au fil du temps.

- Les E/S prédictives pour les mises à jour optimisent vos plans de requêtes et la disposition des données, en équilibrant intelligemment les performances de lecture et d'écriture. Vous profitez pleinement de vos données sans avoir à choisir entre des stratégies comme la copie à l'écriture et la fusion à la lecture.
- Le clustering fluide offre les performances d'une table bien réglée et bien partitionnée sans les problèmes traditionnellement liés au partitionnement. Ne vous demandez plus si vous pouvez partitionner des colonnes à cardinalité élevée ou quel est le coût de la réécriture lors du changement des colonnes de partition. Le résultat est : des tables ultra-rapides et un clustering idéal avec le minimum de configuration.
- L'optimisation prédictive adapte automatiquement vos données pour équilibrer performance et prix. Elle apprend vos pratiques d'utilisation des données, élabore un plan d'optimisation puis le met en œuvre sur une infrastructure serverless hyper-optimisée.

Maintenant que nous avons vu les bases de l'architecture Lakehouse, nous allons voir comment les capacités de data warehouse d'analytique sont fournies sur Databricks avec les structures de données appropriées et les fonctionnalités de gestion offertes par Databricks SQL (DB SQL).

Databricks SQL Serverless : Le meilleur data warehouse pour une architecture lakehouse

Databricks SQL a été créé pour améliorer les capacités du data warehouse et offrir une excellente prise en charge de SQL sur la Data Intelligence Platform de Databricks.

Il rationalise la transformation, l'exploration et l'analyse des données basées sur SQL et répond aux besoins de différents profils d'utilisateurs. Pour les analystes BI et les architectes de données comme les ingénieurs analytiques, cette interface SQL intuitive simplifie les requêtes et les opérations de données complexes sans programmation spécialisée. Cet accès élargi aux données favorise une culture centrée sur l'organisation et donne à de nouvelles équipes les moyens de fonder leurs décisions sur des données.

Databricks SQL se distingue par sa capacité à gérer des datasets massifs avec rapidité et efficacité. Photon, le moteur nouvelle génération de Databricks optimisé par l'IA, accélère le traitement et l'analyse des données, réduisant considérablement la durée d'exécution des requêtes. Les organisations qui doivent relever des défis liés aux données ont besoin de performances d'exception pour extraire des insights de datasets variés. Databricks SQL a également l'avantage de favoriser la collaboration en mettant à la disposition des professionnels un espace de travail où ils peuvent partager des requêtes, des résultats et des interprétations. Cet environnement de partage encourage la diffusion des connaissances et permet de parvenir plus rapidement à une solution en misant sur l'intelligence collective des équipes.

Databricks SQL intègre en outre des capacités sophistiquées pour la gouvernance des données, la sécurité et la conformité. Grâce à elles, les organisations peuvent maintenir la qualité des données, encadrer les accès, superviser les activités, protéger les données sensibles et respecter leurs obligations réglementaires. Faisons le point sur les avantages de Databricks SQL :

- **Des insights plus rapidement**
  Utilisez de l'anglais courant pour accéder aux données : les requêtes SQL sont automatiquement créées pour vous. Cela permet d'affiner plus rapidement les requêtes et de les mettre à disposition de toute l'entreprise.
- **Meilleur rapport performance-prix**
  Le calcul serverless associé au traitement optimisé par IA permet d'atteindre des performances exceptionnelles à bas coût, sans avoir à gérer une infrastructure cloud.
- **Gouvernance unifiée**
  Mettez en place une couche de gouvernance unifiée pour encadrer toutes vos données, où qu'elles résident
- **Complexité allégée**
  Unifiez les données, l'analytique et l'IA sur une même plateforme prenant en charge SQL et Python, des notebooks et des IDE, des charges en batch et en streaming, ainsi que tous les grands fournisseurs de cloud.
- **Un écosystème riche**
  Utilisez SQL et vos outils habituels comme Power BI, Tableau, dbt et Fivetran avec Databricks pour la BI, l'importation et la transformation des données.

Conclusion

La Data Intelligence Platform de Databricks représente une percée significative dans le monde du data warehouse et de l'analytique. Elle répond à un besoin crucial : améliorer l'efficacité du data warehouse dans le paysage commercial data-driven d'aujourd'hui. L'architecture de données simplifiée de la plateforme centralise les données, unifie le batch et le streaming et fournit des capacités de data warehousing complètes pour réduire les coûts et le temps nécessaire pour transformer les données brutes en insights exploitables à grande échelle. Unity Catalog assure la qualité des données et exerce une gouvernance granulaire avec suivi du lineage pour faciliter la découverte, la sécurisation et la gestion de toutes vos données sur l'ensemble des clouds avec le Unity Catalog de Databricks.

C'est avant tout par son évolutivité et ses performances que la plateforme Databricks se démarque, mais aussi grâce au calcul serverless, à la prise en charge de la concurrence et à l'optimisation du stockage. Les optimisations par IA améliorent le traitement des requêtes, les performances et les données pour un rapport performance-prix optimal : l'analyse est à la fois plus rapide et plus rentable.

Enfin, Databricks SQL enrichit les capacités de la plateforme en offrant une excellente prise en charge de SQL et en facilitant la transformation et l'analyse des données pour de nombreux profils d'utilisateur. Ce langage favorise la collaboration, la gouvernance des données et le développement d'un riche écosystème d'outils. En éliminant les silos, les organisations peuvent exploiter tout le potentiel de leurs données. Examinons maintenant quelques cas d'usage pour mieux comprendre comment exécuter vos charges de data warehousing et de BI sur la plateforme Databricks.

EN SAVOIR PLUS

- → Pourquoi le lakehouse est votre prochain data warehouse : 2e édition
- → Les nouveautés de Databricks SQL
- → De nouvelles capacités de fédération des lakehouses dans Unity Catalog
- → Présentation des fonctions d'IA : intégrer les grands modèles de langage avec Databricks SQL

SECTION

03

Bonnes pratiques de data warehousing sur le lakehouse

SECTION 3.1

Techniques de modélisation du data warehouse et mise en œuvre sur le lakehouse

Utilisation de data vaults et de schémas en étoile sur le lakehouse

par Soham Bhatt et Deepak Sekar

Le lakehouse est un nouveau paradigme de plateforme de données qui combine les atouts des data warehouses et des data lakes. Il est conçu comme une plateforme de données d'entreprise de grande échelle, capable d'abriter de nombreux cas d'usage et produits de données. Il peut servir de dépôt de données central et unifié afin de regrouper :

- les domaines de données,
- les cas d'usage de streaming en temps réel,
- les data marts,
- les data warehouses disparates,
- les magasins de fonctionnalités et les sandboxes utilisés en data science, et
- les sandboxes d'analytique en libre-service des différents départements.

Face à la diversité des cas d'usage, les différents projets hébergés sur un même lakehouse peuvent être régis par différents principes d'organisation et utiliser des techniques de modélisation variables. Techniquement, une architecture de lakehouse peut prendre en charge de nombreux styles de modélisation différents. Dans cet article, nous expliquerons comment mettre en œuvre les principes d'organisation Bronze/Silver/Gold du lakehouse et quelles techniques de modélisation correspondent à chaque couche.

Qu'est-ce qu'un data vault ?

Le data vault est une méthode de modélisation plus récente, employée pour mettre sur pied des data warehouses destinés à de l'analytique à l'échelle de l'entreprise, que l'on compare généralement aux méthodes Kimball et Inmon.

Un data vault organise les données en trois types différents : centres, liens et satellites. Les centres, ou hubs, correspondent aux entités commerciales, les liens représentent les relations entre les centres, et les satellites stockent les attributs des centres et des liens.

Le data vault permet le développement d'un data warehouse agile, qui donne la priorité à l'évolutivité, à l'intégration des données/ETL et à la vitesse de développement. La plupart des clients mettent en place une zone d'accueil, une zone vault et une zone data mart, qui correspondent aux couches Bronze, Silver et Gold du paradigme de Databricks. La modélisation en centres, liens et tables satellites du data vault convient parfaitement à la couche Silver du lakehouse.

Pour en savoir plus sur la modélisation en data vault, consultez le site de la Data Vault Alliance.

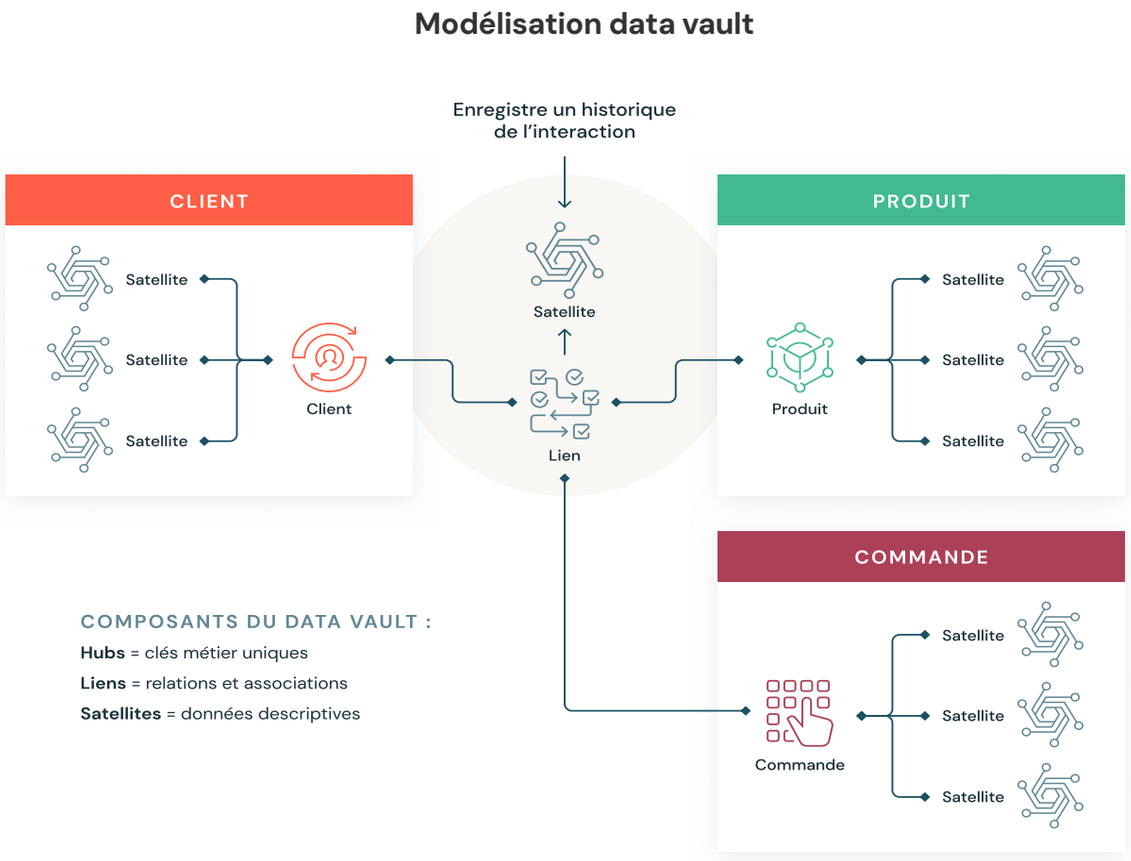
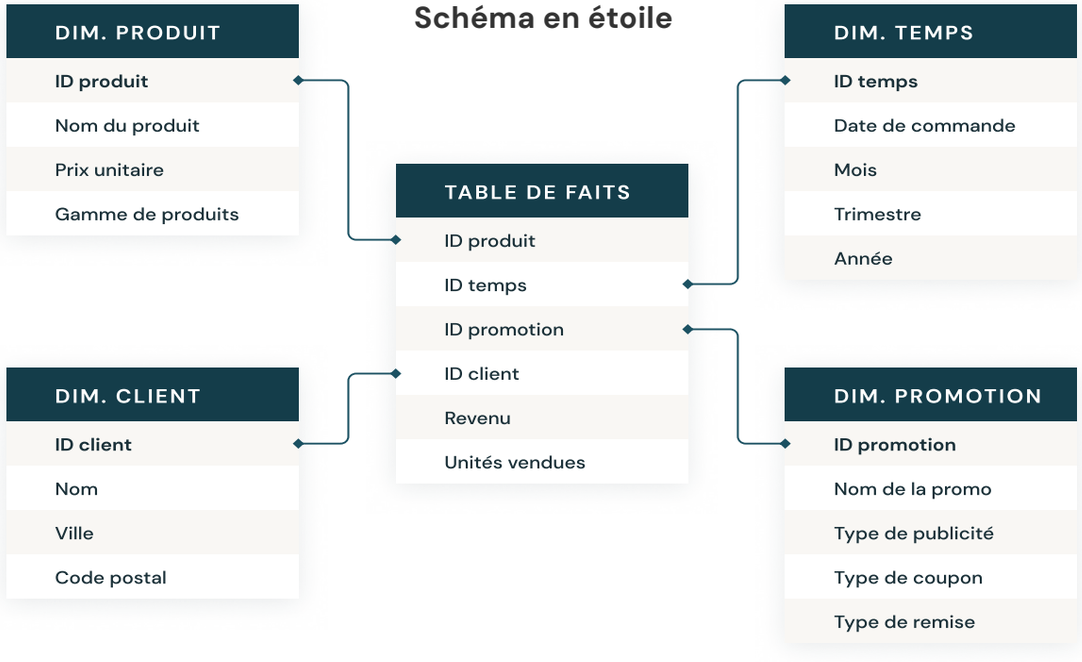


Schéma du fonctionnement de la modélisation data vault, illustrant la façon dont les centres, les liens et les satellites se connectent entre eux.

Qu'est-ce que la modélisation dimensionnelle ?

La modélisation dimensionnelle est une approche ascendante de la conception des data warehouses qui vise à les optimiser pour l'analytique. Les modèles dimensionnels sont utilisés pour décomposer les données métiers en dimensions (temps, produit, etc.) et en faits (montants et quantités des transactions, par ex.). Les domaines d'intérêt sont reliés entre eux via des dimensions conformées pour naviguer entre différentes tables.

La forme la plus courante de modélisation dimensionnelle est le schéma en étoile. Un schéma en étoile est un modèle de données multidimensionnel qui organise les données pour faciliter leur interprétation et leur analyse, et rend la création de rapports très intuitive. Les schémas en étoile et les modèles dimensionnels Kimball sont la norme pour la couche de présentation des data warehouses et des data marts, et même pour les couches sémantiques et de rapport. Le schéma en étoile est optimisé pour l'interrogation des grands datasets.



Exemple de schéma en étoile

La Data Intelligence Platform de Databricks accueille aussi bien les modèles dimensionnels normalisés de type data vault, optimisés pour l'écriture, que les modèles dénormalisés, optimisés pour l'écriture. Les centres et les satellites du data vault de la couche Silver abritent les dimensions du schéma en étoile, et les tables de liens servent de tables pilotes pour charger les tables de faits dans le modèle de dimensions. Pour en savoir plus sur la modélisation dimensionnelle, consultez le site du Kimball Group.

Principes d'organisation des données dans chaque couche du lakehouse

Le lakehouse moderne est une plateforme globale pour toutes les données d'entreprise. Son haut niveau d'évolutivité et de performance permet de traiter différents cas d'usage – ETL, BI, data science, streaming, etc. – selon différentes approches de modélisation. Examinons l'organisation typique d'un lakehouse :

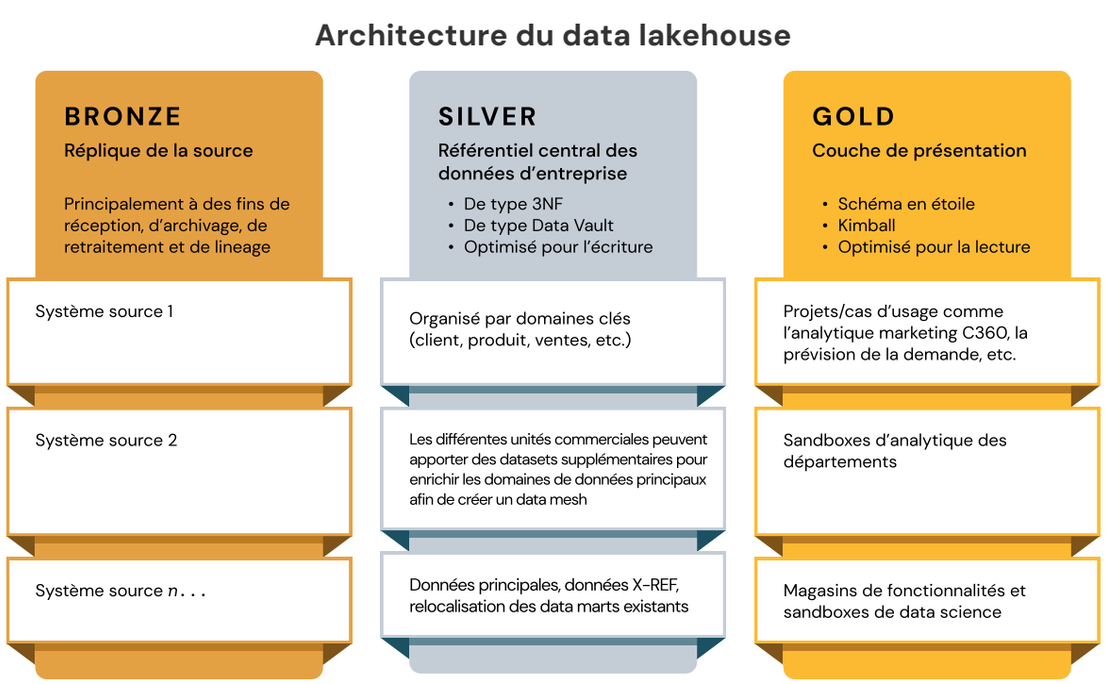


Schéma représentant les caractéristiques des couches Bronze, Silver et Gold de l'architecture du data lakehouse.

Couche Bronze : la zone de réception

C'est dans la couche Bronze que parviennent les données de tous les systèmes source. Ses structures de table correspondent « telles quelles » à celles du système source, hormis les colonnes de métadonnées facultatives pouvant capturer la date/l'heure de chargement, l'ID de processus, etc. Cette couche cible la capture des modifications de données (CDC) et peut fournir une archive historique des données source (stockage à froid), le data lineage et un support d'audit, tout en facilitant le retraitement au besoin, sans qu'il faille relire les données depuis la source.

On conservera généralement les données de la couche Bronze au format Delta pour améliorer les performances des lectures suivantes à des fins d'ETL et faciliter les mises à jour et l'inscription des modifications CDC. Certains clients réceptionnent les données au format JSON ou XML dans leur format d'origine et les préparent et les convertissant au format Delta. Ils utilisent donc la couche Bronze logique comme zone de réception physique et de préparation.

Stocker les données brutes dans leur format d'origine contribue également à leur cohérence si vous importez les données à l'aide d'outils qui ne prennent pas en charge Delta, ou si les systèmes sources déversent directement leurs données dans des magasins d'objets. Cette approche fonctionne bien avec le cadre d'ingestion autoloader : les sources envoient les données des fichiers bruts dans la zone de réception, puis Databricks Autoloader les convertit au format Delta dans la couche de préparation.

Couche Silver : le référentiel central de l'entreprise

Dans la couche Silver du lakehouse, les données de la couche Bronze sont rapprochées, fusionnées, mises en forme et nettoyées (« le strict minimum ») de façon à produire une « vue entreprise » de l'ensemble des entités, transactions et concepts commerciaux. On peut la comparer à un magasin de données opérationnelles (ODS), à un référentiel central ou aux domaines de données d'un data mesh (clients principaux, produits, transactions sans doublon et tables de références croisées). En rassemblant les données de différentes sources, cette vue d'entreprise rend possibles l'analytique en libre service et les rapports ad hoc, l'analytique avancée et le ML. Elle sert aussi de source pour les analystes départementaux, les ingénieurs data et les data scientists qui pourront ensuite créer dans la couche Gold des projets de données et des analyses afin de résoudre des problèmes métier.

Selon le paradigme de data engineering du lakehouse, la méthodologie ELT (extraire, charger, transformer) est préférée à l'approche ETL (extraire, transformer, charger) conventionnelle. Avec l'ELT, seul le « strict minimum » de transformation et de nettoyage est appliqué au moment du chargement de la couche Silver. Toutes les règles de « niveau entreprise » sont appliquées dans la couche Silver, tandis que les règles de transformation spécifiques le sont dans la couche Gold. À ce stade, la priorité est à la vitesse et à l'agilité de l'importation et de la livraison des données.

Du point de vue de la modélisation des données, la couche Silver contient davantage de modèles de type 3NF. Il est possible d'y utiliser des architectures et des modèles de données axés sur les performances d'écriture, comme data vault. Si la méthodologie data vault est choisie, le data vault brut comme le business vault trouvent leur place dans la couche Silver. Les vues à un point dans le temps (PIT) et les vues matérialisées seront présentées dans la couche Gold.

Couche Gold : la couche de présentation

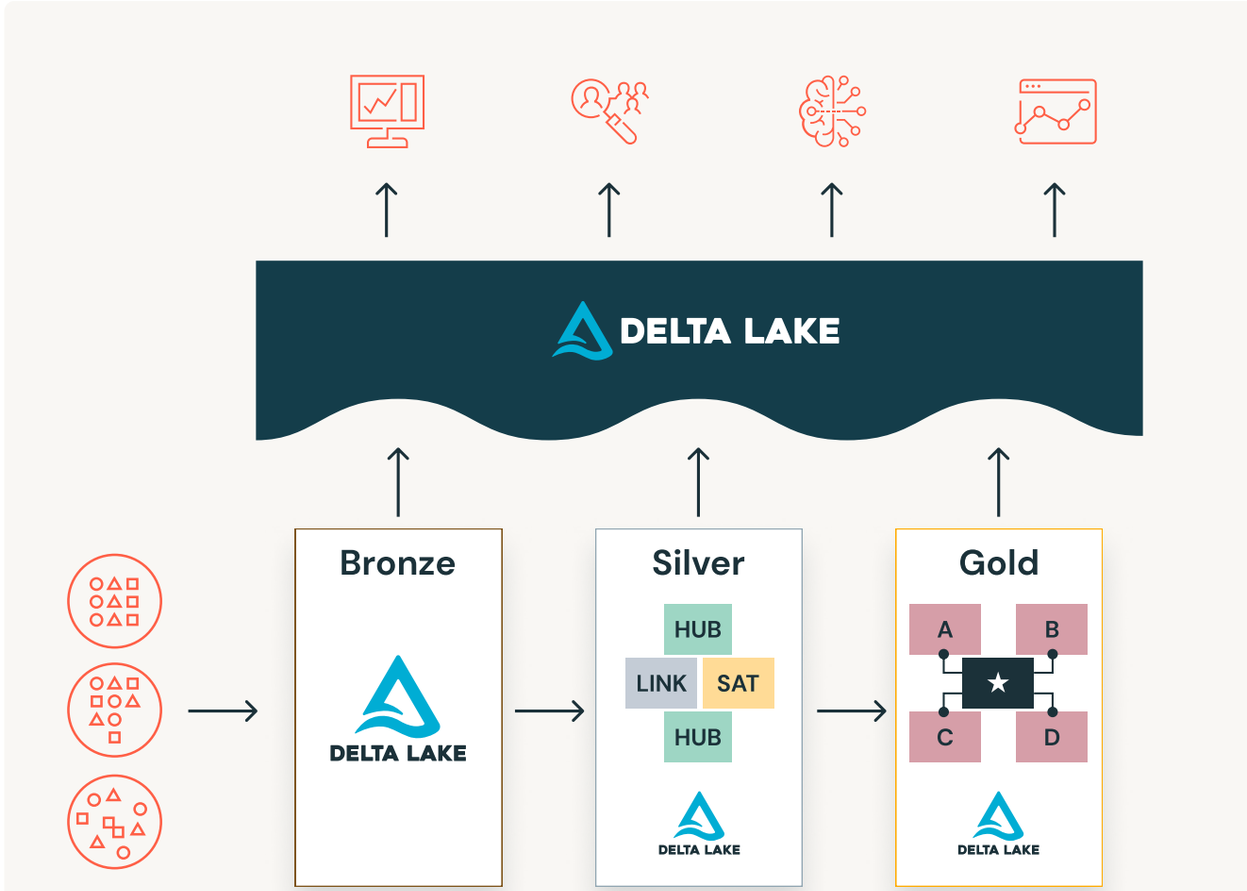
C'est dans la couche Gold que des data marts ou des data warehouses sont créés selon la méthodologie de modélisation dimensionnelle ou Kimball. Comme nous l'avons vu plus tôt, la couche Gold est destinée aux rapports et utilise davantage de modèles de données dénormalisés, optimisés pour la lecture, avec moins de jointures que dans la couche Silver. Il arrive que des tables de la couche Gold soient entièrement dénormalisées, en particulier si les data scientists en ont besoin pour alimenter leurs algorithmes d'ingénierie de fonctionnalités.

Les règles ETL et de qualité des données « spécifiques à un projet » sont appliquées au moment de la transformation des données, entre les couches Silver et Gold. Cette couche accueille les présentations finales, comme les data warehouses, les data marts ou les produits de données (analytique client, analytique produit/qualité, analytique d'inventaire, segmentation des clients, recommandations de produits, analytique marketing/commerciale, etc.). Des modèles de données basés sur le schéma en étoile Kimball et des data marts Inmon y sont notamment hébergés. Les laboratoires de data science et les sandboxes départementales utilisées pour l'analytique en libre-service ont également leur place dans la couche Gold.

Paradigme d'organisation des données du lakehouse

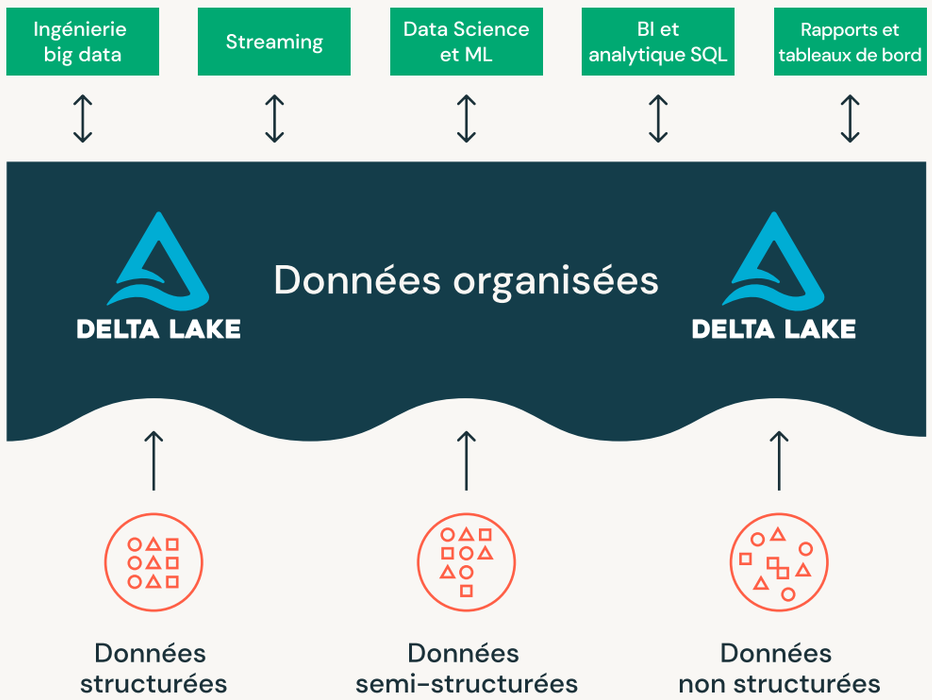
Pour résumer, les données sont organisées à chaque passage d'une couche à l'autre de l'architecture lakehouse.

- La couche Bronze utilise les modèles de données des systèmes source. Si des données arrivent dans des formats bruts, elles sont converties au format DeltaLake dans cette couche.
- La couche Silver rassemble pour la première fois les données de différentes sources et les met en forme pour produire une vue d'entreprise. Le plus souvent, on emploie des modèles normalisés, optimisés pour l'écriture, de type 3NF ou data vault.
- La couche Gold, qui est la couche de présentation, contient davantage de modèles de données dénormalisés ou aplatis que la couche Silver. Elle repose généralement sur des modèles dimensionnels de type Kimball ou des schémas en étoile. La couche Gold héberge également les sandboxes départementales et de data science qui permettent d'utiliser l'analytique et la data science en libre-service dans toute l'entreprise. Ces sandboxes, associées à leurs propres clusters de calcul distincts, évitent aux équipes métier de devoir créer leurs propres copies des données en dehors du lakehouse.



Cette méthode d'organisation des données du lakehouse est destinée à briser les silos, à fédérer les équipes et à leur permettre de réaliser des tâches d'ETL, de streaming, de BI et d'IA sur une même plateforme, avec une gouvernance robuste. Les équipes data centrales doivent jouer le rôle de catalyseurs d'innovation en aidant les nouveaux utilisateurs à se familiariser avec le libre-service et en facilitant le développement parallèle de nombreux projets de données, sans le goulet d'étranglement du processus de modélisation. Le Unity Catalog Databricks fournit des fonctions de recherche, de découverte, de gouvernance et de lineage pour maintenir l'encadrement des données.

Créez vos data vaults et vos data warehouses à schéma en étoile avec Databricks SQL. →



Les données sont organisées à chaque passage d'une couche à l'autre de l'architecture lakehouse.

EN SAVOIR PLUS

- → Implémenter un schéma en étoile en cinq étapes simples dans Databricks avec Delta Lake
- → Bonnes pratiques de mise en œuvre d'un modèle data vault dans le lakehouse Databricks
- → Bonnes pratiques de modélisation dimensionnelle et mise en œuvre sur un lakehouse moderne
- → Des colonnes d'identité pour générer des clés artificielles sur votre lakehouse
- → Charger un modèle dimensionnel EDW en temps réel avec le lakehouse Databricks

SECTION 3.2

3.2 Bonnes pratiques de modélisation dimensionnelle et mise en œuvre sur une architecture de lakehouse moderne

par Leo Mao, Abhishek Dey, Justin Breese et Soham Bhatt

Beaucoup de nos clients abandonnent leurs data warehouses pour le lakehouse Databricks afin de moderniser leur infrastructure et obtenir un accès instantané à une plateforme mature de streaming et d'analytique avancée. Très polyvalent, le lakehouse répond à tous les besoins de streaming, d'ETL, de BI et d'IA et permet à vos équipes métier et data de collaborer sur une même plateforme.

Sur le terrain, nos clients sont en demande de bonnes pratiques pour la modélisation des données et l'implémentation des modèles de données physiques dans Databricks.

Dans cet article, nous explorons les bonnes pratiques de la modélisation dimensionnelle sur la Data Intelligence Platform de Databricks. Nous étudierons un exemple concret d'implémentation de modèle physique conforme aux bonnes pratiques de création de tables et de DDL.

Nous aborderons plusieurs thèmes clés :

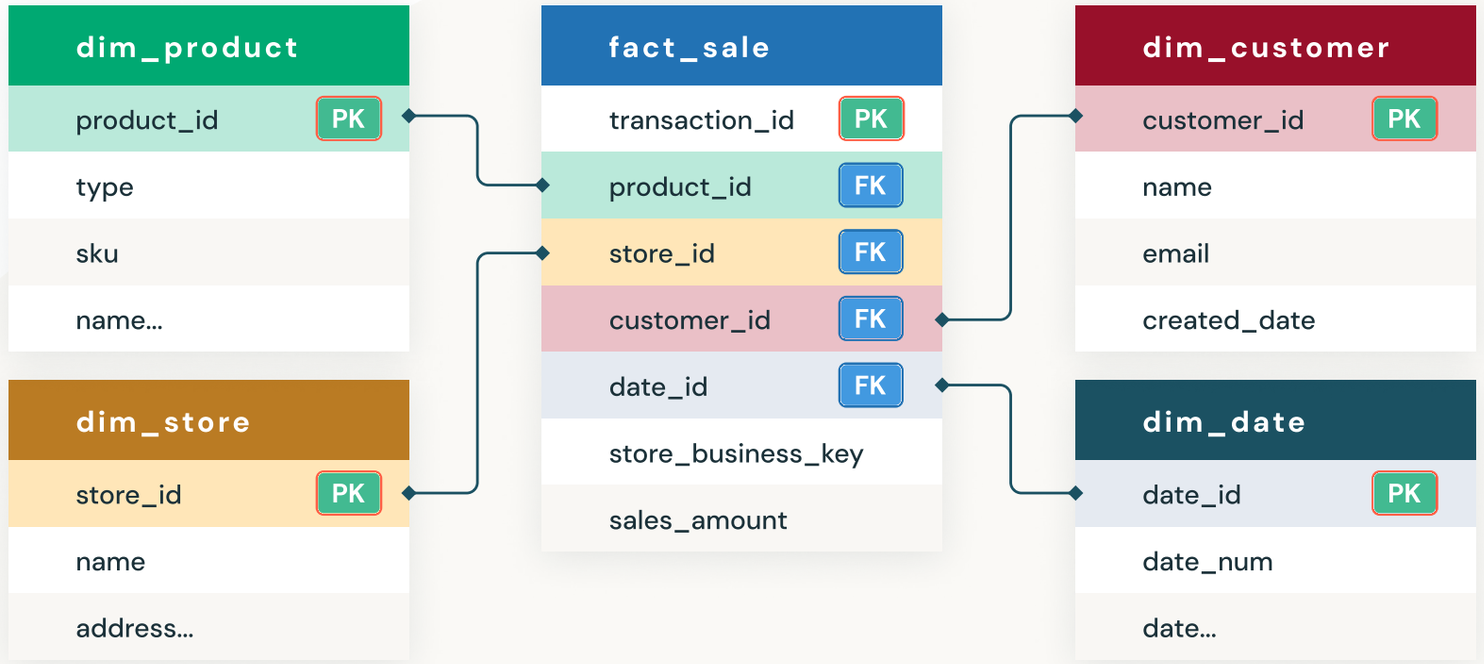
- L'importance de la modélisation des données
- Les techniques courantes de modélisation des données
- L'implémentation DDL de la modélisation de data warehouse
- Bonnes pratiques et recommandations pour la modélisation des données sur le lakehouse

L'importance de la modélisation des données pour le data warehouse

Les modèles de données constituent le fondement du data warehouse. Le processus commence généralement par le modèle d'informations commerciales sémantiques, se poursuit avec un modèle de données logique et se termine par un modèle de données physiques (PDM). La première phase est celle de la conception et de l'analyse des systèmes : un modèle d'informations commerciales et des flux de processus sont créés, puis les entités, attributs et interactions métier clés sont capturés conformément aux processus métier de l'organisation. Un modèle de données logique est ensuite créé. Technologiquement agnostique, il décrit les relations entre les différentes entités. Enfin, un PDM est créé sur la base de la plateforme technologique sous-jacente pour garantir l'efficacité des opérations de lecture et d'écriture. Ce n'est pas un secret, les styles de modélisation axés sur l'analytique, comme le schéma en étoile et le data vault, sont très utilisés pour le data warehousing.

Bonnes pratiques de création d'un modèle de données physique dans Databricks

Selon le problème business défini, la conception du modèle de données a pour but de représenter les données dans une optique de réutilisabilité, de flexibilité et d'évolutivité. Dans ce modèle de données de type « schéma en étoile », une table de faits Ventes contient chaque transaction, et diverses tables de dimension (clients, produits, boutiques, date, etc.) permettent de découper et trier les données. Les dimensions peuvent être jointes à la table de faits pour répondre à des questions commerciales précises. Il est ainsi possible de savoir quels sont les produits les plus populaires dans un mois donné ou quelles boutiques ont affiché les meilleures performances du trimestre. Voyons comment le mettre en œuvre dans Databricks.



Modèle dimensionnel sur l'architecture lakehouse

À savoir : chaque table de dimension possède des colonnes **__START_AT** et **__END_AT** à des fins de compatibilité avec SCD Type 2, mais elles ne sont pas représentées ici faute de place.

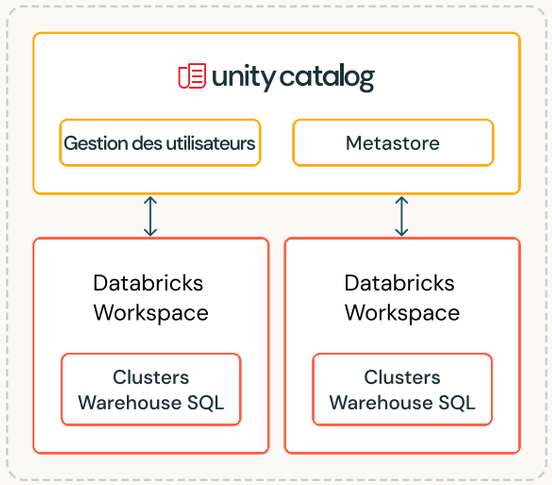
Implémentation DDL de la modélisation de data warehouse sur Databricks

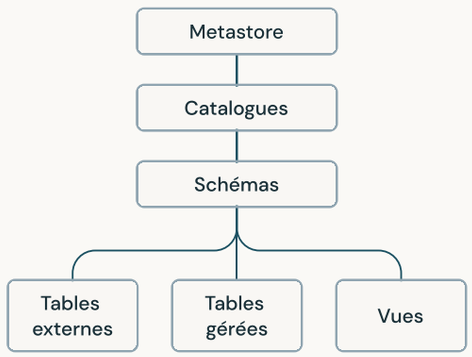
Dans les prochaines sections, nous illustrerons les aspects suivants à travers des exemples.

- Créer un catalogue à 3 niveaux avec base de données et tables
- Définition des clés primaires et des clés étrangères
- Colonnes d'identité pour les clés artificielles
- Contraintes des colonnes pour la qualité des données
- Indexation, optimisation et analyse
- Techniques avancées

1. Unity Catalog – Espace de nommage à 3 niveaux

Unity Catalog représente la couche de gouvernance de Databricks. Utilisé par les administrateurs et les data stewards, il centralise dans un metastore la gestion des utilisateurs et de l'accès aux données sur tous les espaces de travail d'un compte Databricks. Les utilisateurs de différents espaces de travail partagent l'accès aux mêmes données en fonction des privilèges accordés par Unity Catalog. Unity Catalog emploie un espace de nommage à 3 niveaux (catalog. schema(database).table) pour organiser vos données.
Consultez cette page pour en savoir plus sur Unity Catalog.





Voici comment configurer le catalogue et le schéma avant de créer des tables dans la base de données. Pour notre exemple, nous créons un catalogue **US_Stores** et un schéma (base de données) **Sales_DW** comme indiqué ci-dessous. Nous les utiliserons dans la suite de cette section.

```
CREATE CATALOG IF NOT EXISTS US_Stores;
USE CATALOG US_Stores;
CREATE SCHEMA IF NOT EXISTS Sales_DW;
USE SCHEMA Sales_DW;
```

Création du catalogue et de la base de données

Exemple d'interrogation portant sur la table **fact_sales** avec un espace de nommage à 3 niveaux.

```
SELECT *
FROM US_Stores.Sales_DW.fact_sales
```

▸ (2) Spark Jobs

Table | Data Profile

	transaction_id	date_id	customer_id	product_id	store_id	store_business_key	sales_amount
1	10001	20211001	1	1	1	PER01	50
2	10004	20211003	2	1	2	BNE02	60
3	10005	20211003	3	2	1	PER01	79
4	10002	20211002	2	1	2	BNE02	79
5	10003	20211002	1	2	2	BNE02	79

Showing all 5 rows.

Exemple d'interrogation d'une table avec catalog.database.tablename

2. Définition des clés primaires et des clés étrangères

La définition des clés primaires et des clés étrangères est très importante lors de la création d'un modèle de données. La prise en charge de la définition des CP/CE simplifie considérablement la définition d'un modèle de données dans Databricks. Elle aide également les analystes à identifier rapidement les relations de jointure dans le warehouse Databricks SQL pour rédiger efficacement des requêtes. Comme dans d'autres cas de traitement massivement parallèle (MPP), d'EDW et de data warehouses cloud, les contraintes de CP/CE n'ont qu'une valeur informative. Databricks ne prend pas en charge l'application de la relation CP/CE, mais la plateforme permet de les définir pour simplifier la conception de modèles de données sémantiques.

Prenons l'exemple de la création d'une table **dim_store** avec **store_id** comme colonne d'identité, qui sera aussi définie comme clé primaire.

```
-- Dimension Store
CREATE OR REPLACE TABLE dim_store(
  store_id BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
  business_key STRING,
  name STRING,
  email STRING,
  city STRING,
  address STRING,
  phone_number STRING,
  created_date TIMESTAMP,
  updated_date TIMESTAMP,
  start_at TIMESTAMP,
  end_at TIMESTAMP
);
```

Implémentation DDL pour la création de la dimension « store » avec définition des clés primaires.

Une fois la table créée, nous voyons que la clé primaire (store_id) est créée en tant que contrainte dans la définition de table ci-dessous.

Describe Dim Store table information

```
DESC TABLE EXTENDED US_Stores.Sales_DW.dim_store
```

Table | Data Profile

	col_name	data_type
20	Owner	leo.mao@databricks.com
21	Is_managed_location	true
22	Table Properties	[delta.minReaderVersion=1,delta.minWriterVersion=6]
23		
24	# Constraints	
25	dim_store_pk	PRIMARY KEY (`store_id`)

Showing all 25 rows.

La clé primaire store_id se présente comme contrainte de table

À titre d'exemple, voici la création d'une table **fact_sales** avec **transaction_id** comme clé primaire, ainsi que des clés étrangères qui référencent les tables de dimension.

```
-- Fact Sales
CREATE OR REPLACE TABLE fact_sales(
  transaction_id BIGINT PRIMARY KEY,
  date_id BIGINT NOT NULL CONSTRAINT dim_date_fk FOREIGN KEY REFERENCES dim_date,
  customer_id BIGINT NOT NULL CONSTRAINT dim_customer_fk FOREIGN KEY REFERENCES
dim_customer,
  product_id BIGINT NOT NULL CONSTRAINT dim_product_fk FOREIGN KEY REFERENCES
dim_product,
  store_id BIGINT NOT NULL CONSTRAINT dim_store_fk FOREIGN KEY REFERENCES dim_
store,
  store_business_key STRING,
  sales_amount DOUBLE
);
```

Implémentation DDL pour la création d'une table de faits Sales avec des définitions de clés étrangères

Une fois la table de faits créée, nous voyons que la clé primaire **(transaction_id)** et les clés étrangères sont créées en tant que contraintes dans la définition de table ci-dessous.

Describe Fact Sales Table Info

```
DESC TABLE EXTENDED US_Stores.Sales_DW.fact_sales
```

Table | Data Profile

	col_name	data_type
20	# Constraints	
21	dim_product_fk	FOREIGN KEY (`product_id`) REFERENCES `us_stores`.`sales_dw`.`dim_product` (`product_id`)
22	dim_date_fk	FOREIGN KEY (`date_id`) REFERENCES `us_stores`.`sales_dw`.`dim_date` (`date_id`)
23	dim_customer_fk	FOREIGN KEY (`customer_id`) REFERENCES `us_stores`.`sales_dw`.`dim_customer` (`customer_id`)
24	dim_store_fk	FOREIGN KEY (`store_id`) REFERENCES `us_stores`.`sales_dw`.`dim_store` (`store_id`)
25	fact_sales_pk	PRIMARY KEY (`transaction_id`)

Showing all 25 rows.

Définition de table de faits avec une clé primaire et des clés étrangères qui référencent des dimensions

3. Colonnes d'identité pour les clés artificielles

Dans une base de données, une colonne d'identité est une colonne qui génère automatiquement un ID numérique unique pour chaque nouvelle ligne de données. Elle est couramment utilisée pour créer des clés artificielles dans les data warehouses. Les clés artificielles sont générées par le système et n'ont pas de sens. Elles évitent d'avoir à utiliser différentes clés primaires naturelles et des concaténations de plusieurs champs pour identifier l'unicité d'une ligne. Typiquement, ces clés artificielles sont utilisées comme clés primaires et étrangères dans les data warehouses. Cet article aborde les colonnes d'identité de façon plus détaillée. Voici un exemple de création de colonne d'identité nommée customer_id, avec une valeur assignée automatiquement en partant de 1, avec un incrément de 1.

```
1  -- Dimension Customer
2  CREATE OR REPLACE TABLE dim_customer(
3    customer_id BIGINT GENERATED ALWAYS AS IDENTITY (START WITH 1 INCREMENT BY 1)
4  PRIMARY KEY,
5    name STRING,
6    email STRING,
7    address STRING,
8    created_date TIMESTAMP,
9    updated_date TIMESTAMP,
10   start_at TIMESTAMP,
11   end_at TIMESTAMP
12 );
```

Implémentation DDL pour la création de la dimension « customer » avec une colonne d'identité

4. Contraintes des colonnes pour la qualité des données

En plus des contraintes informationnelles entourant les clés primaires et étrangères, Databricks prend également en charge les contraintes de contrôle qualité au niveau des colonnes, qui doivent assurer la qualité et l'intégrité des données ajoutées à la table. Les contraintes sont automatiquement vérifiées. Les contraintes NOT NULL et les contraintes de valeur de colonne en sont de bons exemples. Contrairement au data warehouse cloud, Databricks va plus loin avec des contraintes de valeur de colonne, très utiles pour garantir la qualité des données d'une colonne spécifique. Comme on le voit ci-dessous, la contrainte **valid_sales_amount** vérifie que les lignes existantes satisfont la contrainte (**sales amount** > 0, par exemple) avant de les ajouter à la table. Vous trouverez ici davantage d'informations à ce sujet.

Voici deux exemples de contrainte pour dim_store et fact_sales, vérifiant respectivement que les valeurs de store_id et sales_amount sont valides.

```
-- Une contrainte est ajoutée à dim_store pour vérifier que la valeur de la
colonne store_id est comprise entre 1 et 9998
ALTER TABLE US_Stores.Sales_DW.dim_store ADD CONSTRAINT valid_store_id CHECK
(store_id > 0 and store_id < 9999);

-- Une contrainte est ajoutée à fact_sales pour vérifier que la colonne sales_
amount contient une valeur valide
ALTER TABLE US_Stores.Sales_DW.fact_sales ADD CONSTRAINT valid_sales_amount CHECK
(sales_amount > 0);
```

Ajouter une contrainte de colonne aux tables existantes pour assurer la qualité des données

5. Indexation, optimisation et analyse

Les bases de données traditionnelles ont des index de type b-arbre et bitmap. Databricks, en revanche, emploie l'indexation cluster d'ordre Z multi-dimension, et prend en charge l'indexation de filtre Bloom. Tout d'abord, le format Delta s'appuie sur le format Parquet, un format de fichier compressé en colonne, déjà très efficace pour l'élagage de colonne. De plus, l'indexation d'ordre Z vous permet de parcourir des pétaoctets de données en quelques secondes. L'indexation d'ordre Z et de filtre Bloom réduit considérablement la quantité de données à scanner pour répondre à des requêtes très sélectives portant sur de vastes tables Delta. Le résultat : des durées d'exécution et des coûts réduits généralement de plusieurs ordres de grandeur. Utilisez l'ordre Z sur les clés primaires et les clés étrangères employées pour vos jointures les plus courantes. Et complétez avec l'indexation de filtre Bloom selon les besoins.

```
-- Nous optimisons la table fact_sales par customer_id et product_id pour
améliorer les performances de requête et de jointure
OPTIMIZE US_Stores.Sales_DW.fact_sales
ZORDER BY (customer_id, product_id);
```

Optimiser fact_sales selon customer_id et product_id pour améliorer les performances

```
-- Nous créons un index bloomfilter pour activer le saut de données sur store_
business_key
CREATE BLOOMFILTER INDEX
ON TABLE US_Stores.Sales_DW.fact_sales
FOR COLUMNS(store_business_key)
```

Créer un index bloomfilter pour activer le saut de données sur une colonne donnée

Et comme avec tout autre data warehouse, vous pouvez utiliser ANALYZE TABLE pour mettre à jour les statistiques. L'optimiseur de requête disposera ainsi des meilleures statistiques pour créer le meilleur plan de requête.

```
-- collecte des statistiques pour toutes les colonnes afin d'améliorer les performances
ANALYZE TABLE US_Stores.Sales_DW.fact_sales COMPUTE STATISTICS FOR ALL COLUMNS;
```

Collecter des statistiques pour toutes les colonnes afin d'améliorer le plan d'exécution des requêtes

6. Techniques avancées

Databricks prend en charge des techniques avancées comme le partitionnement de tables, mais nous recommandons de les utiliser uniquement si vous avez des téraoctets de données compressées. La plupart du temps, nos index OPTIMIZE et Z-ORDER produiront le meilleur élagage pour les fichiers et les données, faisant presque du partitionnement par date ou mois une mauvaise pratique. Nous recommandons toutefois d'activer l'auto-optimisation et l'auto-compactage des DDL de vos tables. De cette façon, les données écrites fréquemment en petits fichiers seront compactées dans des formats compressés en colonne Delta.

Vous voulez utiliser un outil de modèle de données visuel ? Notre partenaire erwin Data Modeler de Quest permet de faire de la rétro-ingénierie, de créer et d'implémenter des schémas en étoile, des data vaults et tout autre modèle de données dans Databricks en quelques clics.

Exemple de notebook Databricks

La plateforme Databricks facilite la conception et la mise en œuvre de différents modèles de données. Consultez cet exemple qui rassemble tous les éléments ci-dessus en un workflow complet.

Lisez également notre article de blog :
Implémenter un schéma en étoile en cinq étapes simples dans Databricks avec Delta Lake →

databricks

SECTION 3.3

Charger un modèle de données data warehouse en temps réel avec la Data Intelligence Platform de Databricks

Mettre en œuvre la modélisation dimensionnelle sur le lakehouse moderne à l'aide de Delta Live Tables

par Leo Mao, Soham Bhatt et Abhishek Dey

La modélisation dimensionnelle est l'une techniques les plus répandues pour créer un data warehouse moderne. Elle permet de développer rapidement des faits et des dimensions répondant aux besoins commerciaux d'une entreprise. En aidant nos clients sur le terrain, nous avons compris qu'ils étaient en demande de bonnes pratiques et d'architectures de référence recommandées par Databricks.

Dans cet article, nous explorons les bonnes pratiques de la modélisation dimensionnelle sur l'architecture lakehouse. Nous étudions aussi un exemple concret de chargement de modèle dimensionnel EDW en temps réel à l'aide de Delta Live Tables.

Dans ce chapitre, nous allons aborder les grandes étapes de l'implémentation :

- Définition du problème métier
- Conception du modèle dimensionnel
- Bonnes pratiques et recommandations pour la modélisation dimensionnelle
- Mise en œuvre d'un modèle dimensionnel sur une architecture lakehouse
- Conclusion

Définition du problème métier

La modélisation dimensionnelle est orientée métier : elle part toujours d'une problématique commerciale. Avant de créer un modèle dimensionnel, il faut comprendre le problème métier à résoudre, car cela détermine la façon dont l'asset sera présenté et consommé. Le modèle de données doit être conçu de façon à faciliter et accélérer les requêtes.

La matrice métier, illustrée ci-dessous, est un concept fondamental de la modélisation dimensionnelle. Les colonnes sont des dimensions partagées et les lignes, des processus métier. Le problème métier défini détermine la granularité des données de fait et les dimensions requises. Tout repose sur une idée clé : il est possible de créer des assets supplémentaires de façon incrémentale sur la base de cette matrice et de ses dimensions, partagées ou conformées.

Dimensions partagées →

Processus métier ↓	Date	Client	Produit	Fournisseur	Promotion	Revendeur	Territoire commercial	Collaborateur	Compte	Organisation
Ventes en ligne	✓	✓	✓	✓	✓		✓			
Ventes du revendeur	✓		✓		✓	✓	✓	✓		
Grand livre	✓								✓	✓
Plan commercial	✓		✓				✓			
Inventaire	✓		✓	✓					✓	
Enquêtes client	✓	✓								
Appels au service client	✓	✓	✓					✓		

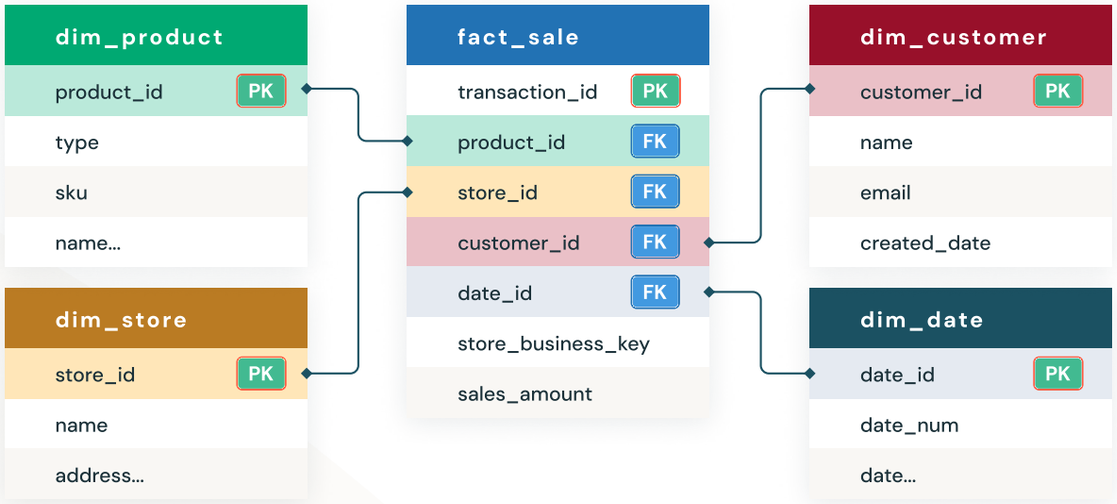
Matrice métier, avec des dimensions partagées et des processus métier

Dans notre exemple, le commanditaire métier a besoin d'un rapport donnant des informations sur :

- les produits qui se vendent le mieux, afin de comprendre la popularité des produits
- les boutiques les plus performantes, afin d'en étudier les bonnes pratiques

Conception du modèle dimensionnel

Selon le problème commercial défini, la conception du modèle de données a pour but de représenter les données dans une optique de réutilisabilité, de flexibilité et d'évolutivité. Voici un modèle de données de haut niveau capable de répondre aux questions métier ci-dessus.



Modèle dimensionnel sur l'architecture lakehouse

À savoir : chaque table de dimension possède des colonnes **__START_AT** et **__END_AT** à des fins de compatibilité avec SCD Type 2, mais elles ne sont pas représentées ici faute de place.

La conception doit être aisément compréhensible et efficace pour plusieurs types de requêtes. À partir du modèle, nous concevons la table de faits Sales pour répondre aux questions métier. Comme vous le voyez, hormis les clés étrangères (FK) qui référencent les dimensions, elle ne contient que les métriques numériques métier, comme le montant des ventes (sales_amount).

Nous avons également créé des tables de dimensions – Product, Store, Customer, Date – qui apportent du contexte aux données de fait. Les tables de dimension sont généralement jointes aux tables de faits pour répondre à des questions commerciales : quels sont les produits les plus populaires dans un mois donné, ou quelles boutiques ont affiché les meilleures performances du trimestre, par exemple.

Bonnes pratiques et recommandations pour la modélisation dimensionnelle

La Data Intelligence Platform de Databricks simplifie la conception et l'implémentation de modèles dimensionnels ainsi que la création des tables de dimensions et de faits d'un thème donné.

Voici quelques bonnes pratiques à suivre lors de la mise en œuvre d'un modèle dimensionnel :

- Il faut dénormaliser les tables de dimension. Contrairement au modèle de type 3NF ou flocon de neige, les tables de dimension sont hautement dénormalisées et contiennent des relations plusieurs-vers-un aplaties.
- Utilisez des tables de dimension conformées lorsque les attributs de différentes tables de dimension ont les mêmes noms de colonne et contenus de domaine. De cette manière, les données de différentes tables de faits pourront être combinées en un même rapport à l'aide des attributs conformés associés à chaque table de faits.
- Une pratique courante consiste à suivre les modifications apportées aux dimensions au fil du temps pour produire des rapports sur l'état présent ou un état antérieur. Selon vos besoins, vous allez utiliser l'une des techniques simples ci-dessous pour manipuler les dimensions.
  - La technique 1 consiste à écraser la valeur initiale de l'attribut de dimension.
  - La technique 2, technique SCD la plus courante, permet de suivre avec précision les changements au fil du temps.

  Une implémentation de Delta Live Tables le fait simplement sans autre intervention.
  - Nous pouvons appliquer la technique SCD type 1 ou SCD type 2 à l'aide de Delta Live Tables, en utilisant APPLY CHANGES INTO
- Clé primaire et clé étrangère **Les contraintes** permettent aux utilisateurs finaux de comprendre les relations qui existent entre les tables.
- **L'utilisation de la colonne IDENTITY** génère automatiquement des valeurs entières uniques lors de l'ajout d'une nouvelle ligne. Les colonnes d'identité sont une forme de clé artificielle. Consultez cet article de blog pour plus d'informations.
- **Appliquez des contraintes CHECK** pour ne plus jamais vous inquiéter de la qualité ou de l'exactitude des données.

Mise en œuvre d'un modèle dimensionnel sur une architecture lakehouse

Voyons maintenant un exemple d'implémentation d'une modélisation dimensionnelle basée sur Delta Live Tables :

L'exemple de code ci-dessous permet de créer une table de dimension (dim_store) avec la technique SCD type 2, qui capture les changements auprès du système source.

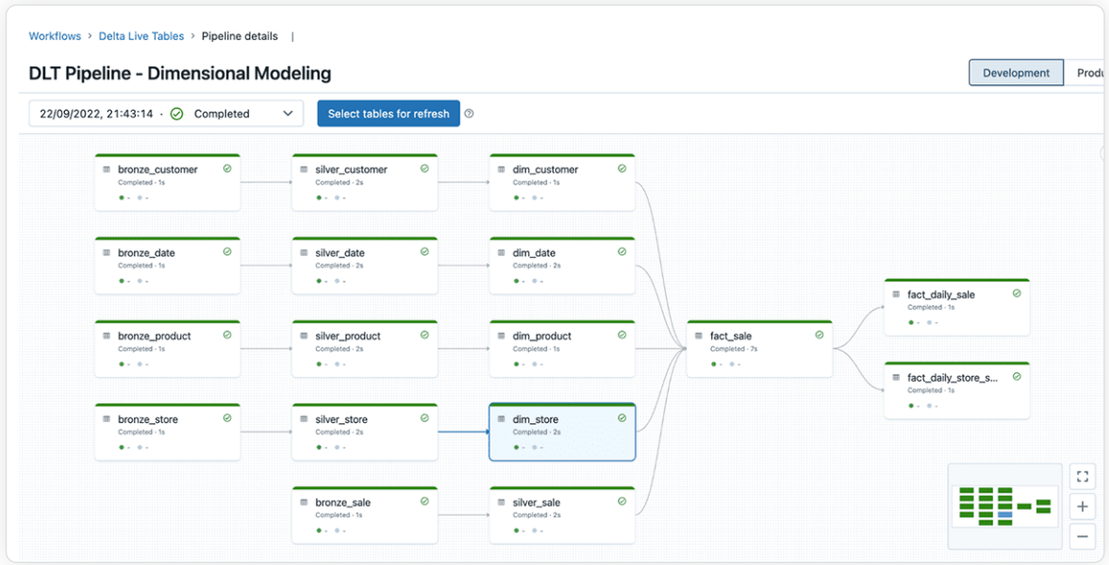
```
-- création de la table gold
CREATE INCREMENTAL LIVE TABLE dim_store
TBLPROPERTIES ("quality" = "gold")
COMMENT "SCD Type 2 pour la dimension store dans la couche gold";

-- enregistrer toutes les modifications sous forme SCD2
APPLY CHANGES INTO live.dim_store
FROM STREAM(live.silver_store)
  KEYS (store_id)
  SEQUENCE BY updated_date
  COLUMNS * EXCEPT (_rescued_data, input_file_name)
  STORED AS SCD TYPE 2;
```

L'exemple de code ci-dessous indique comment créer une table de faits (fact_sale). La contrainte valid_product_id nous assure que tous les enregistrements de fait chargés sont associés à un produit valide.

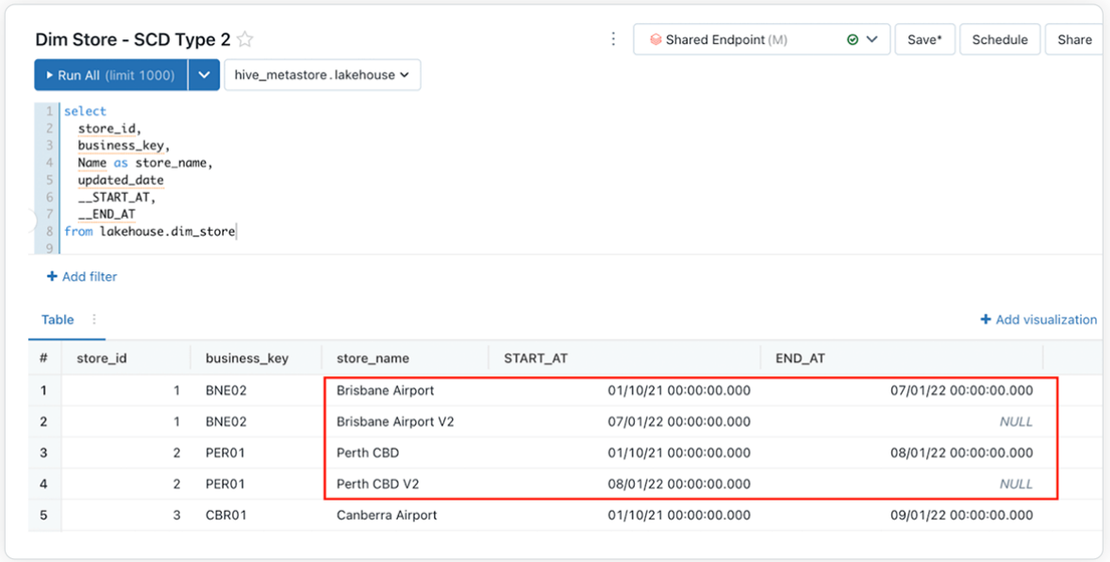
```
-- création de la table de faits des ventes dans la couche gold
CREATE STREAMING LIVE TABLE fact_sale (
  CONSTRAINT valid_store_business_key EXPECT (store_business_key IS NOT NULL) ON
VIOLATION DROP ROW,
  CONSTRAINT valid_product_id EXPECT (product_id IS NOT NULL) ON VIOLATION DROP
ROW
)
TBLPROPERTIES ("quality" = "gold", "ignoreChanges" = "true")
COMMENT "table de faits des ventes dans la couche gold" AS
  SELECT
    sale.transaction_id,
    date.date_id,
    customer.customer_id,
    product.product_id AS product_id,
    store.store_id,
    store.business_key AS store_business_key,
    sales_amount
  FROM STREAM(live.silver_sale) sale
  INNER JOIN live.dim_date date
  ON to_date(sale.transaction_date, 'M/d/yy') = to_date(date.date, 'M/d/yyyy')
  -- joindre uniquement avec les clients actifs
  INNER JOIN (SELECT * FROM live.dim_customer WHERE __END_AT IS NULL) customer
  ON sale.customer_id = customer.customer_id
  -- joindre uniquement avec les produits actifs
  INNER JOIN (SELECT * FROM live.dim_product WHERE __END_AT IS NULL) product
  ON sale.product = product.SKU
  -- joindre uniquement avec les boutiques actives
  INNER JOIN (SELECT * FROM live.dim_store WHERE __END_AT IS NULL) store
  ON sale.store = store.business_key
```

Vous trouverez sur cette page l'exemple de pipeline Delta Live Tables. Consultez le Guide de démarrage rapide de Delta Live Tables pour savoir comment créer un pipeline DLT. Comme nous le voyons, DLT offre une visibilité complète sur le pipeline ETL et les dépendances entre les différents objets des couches Bronze, Silver et Gold, conformément à l'architecture en médaillon du lakehouse.



Pipeline DLT de bout en bout

Voyons à travers un exemple comment la table de dimension dim_store est mise à jour en cas de modification entrante. La boutique Brisbane Airport a été mise à jour vers Brisbane Airport V2. Grâce à la prise en charge native de SCD type 2, l'enregistrement d'origine a été interrompu le 7 janvier 2022 et un nouvel enregistrement a été créé le même jour avec une date de fin non définie (NULL), indiquant que c'est l'enregistrement le plus récent pour l'aéroport de Brisbane.



#	store_id	business_key	store_name	START_AT	END_AT
1	1	BNE02	Brisbane Airport	01/10/21 00:00:00.000	07/01/22 00:00:00.000
2	1	BNE02	Brisbane Airport V2	07/01/22 00:00:00.000	NULL
3	2	PER01	Perth CBD	01/10/21 00:00:00.000	08/01/22 00:00:00.000
4	2	PER01	Perth CBD V2	08/01/22 00:00:00.000	NULL
5	3	CBR01	Canberra Airport	01/10/21 00:00:00.000	09/01/22 00:00:00.000

SCD Type 2 pour la dimension Store

Pour plus de détails sur la mise en œuvre, consultez ici l'exemple de notebook complet.

Conclusion

Dans ce chapitre, nous avons étudié les concepts de la modélisation dimensionnelle, ses bonnes pratiques et comment la mettre en œuvre avec Delta Live Tables.

Pour en savoir plus sur la modélisation dimensionnelle, consultez le site Kimball Technology.

SECTION 3.4

Les nouveautés de Databricks SQL

Optimisations basées sur l'IA, fédération des lakehouses et plus encore pour une BI professionnelle

par Alex Lichen, Miranda Luna, Can Efeoglu and Cyrielle Simeone

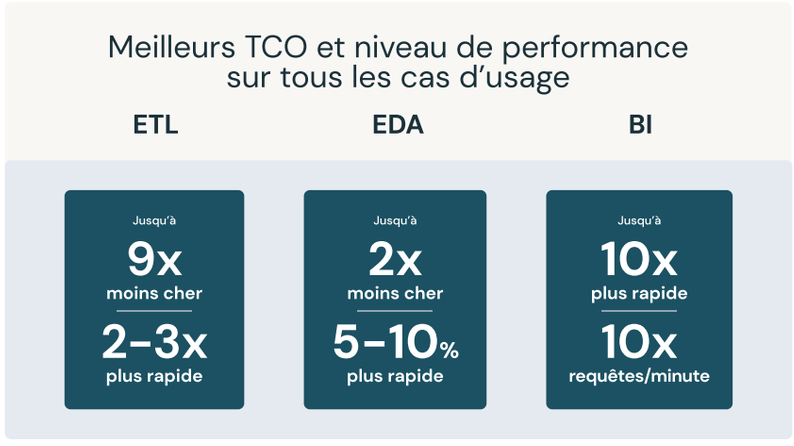
Lors du Data + AI Summit de cette année, Databricks SQL a encore repoussé les limites du data warehouse en appliquant l'IA sur toute la surface du produit. L'objectif : étendre notre leadership en matière de performance et d'efficacité, simplifier l'expérience des utilisateurs et ouvrir de nouvelles opportunités à nos clients. Parallèlement à cela, nous avons encore amélioré nos capacités fondamentales pour vous aider à unifier votre pile de données dans l'architecture lakehouse de la plateforme Data Intelligence de Databricks.

Dans ce chapitre, nous avons le plaisir de vous faire découvrir les nouveautés de Databricks SQL :

- Des optimisations de performance basées sur l'IA, comme les E/S prédictives qui réduisent les coûts et améliorent les performances sans intervention manuelle.
- De nouvelles expériences : fonctions IA, warehouses SQL dans les notebooks, nouveaux tableaux de bord et fonctions Python définies par l'utilisateur.
- Prise en charge enrichie des sources externes avec Lakehouse Federation.
- De nouveaux moyens d'accéder à vos données avec l'API SQL Statement Execution.
- Un traitement simple et efficace des données avec les tables en streaming, les vues matérialisées et l'intégration des workflows.
- Assistance intelligente de notre moteur de connaissances DatabricksIQ.
- De meilleurs outils d'administration avec les tables système Databricks SQL et les profils de requête en direct.
- De nouvelles fonctionnalités pour les intégrations de nos partenaires Fivetran, dbt labs et PowerBI.

Le warehouse optimisé par IA : accueille toutes vos tâches sans aucun ajustement

Nous pensons que le meilleur data warehouse est un lakehouse, et c'est pourquoi nous élargissons notre leadership en matière d'ETL et misons sur la puissance de l'IA. Databricks SQL offre maintenant des performances de référence pour vos charges EDA et BI tout en réduisant les coûts, sans ajustement manuel.



Vous ne créerez plus jamais d'index à la main. Avec les E/S prédictives pour la lecture (disponible) et la mise à jour (accès anticipé public), Databricks SQL analyse désormais l'historique de lecture et d'écriture pour créer des index et optimiser les charges intelligemment. Certains clients ont vu l'efficacité des recherches ponctuelles multipliée par 35. Les opérations MERGE sont de 2 à 6 fois plus performantes, et les opérations DELETE, de 2 à 10 fois.

Avec les optimisations prédictives (accès anticipé public), Databricks optimise la taille des fichiers et le clustering en exécutant à votre place les commandes OPTIMIZE, VACUUM, ANALYZE et CLUSTERING. Anker Innovations, par exemple, bénéficie de requêtes 2,2 fois plus performantes et réalise 50 % d'économies sur ses coûts de stockage.

> **Grâce à l'optimisation prédictive de Databricks sur notre Unity Catalog, nous avons réalisé 50 % d'économies en frais de stockage annuels et la vitesse de nos requêtes a plus que doublé. Elle donne la priorité à nos tables les plus vastes et les plus utilisées. Et tout cela automatiquement, ce qui fait gagner un temps précieux à notre équipe.**
>
> — ANKER INNOVATIONS

Vous devez gérer des warehouses différents en fonction de la taille des charges, ou régler des paramètres d'évolutivité à la main ? Les fonctionnalités d'Intelligent Workload Management préservent la performance des requêtes et optimisent les coûts. Cet outil analyse les modèles d'utilisation en temps réel afin que vos charges reçoivent la capacité de calcul optimale. L'objectif : exécuter les requêtes SQL entrantes sans perturber celles qui sont déjà en cours.

Grâce aux optimisations axées sur l'IA, Databricks SQL délivre un TCO et des performances de pointe pour tous les types de charge de travail, sans aucun ajustement manuel. Pour en savoir plus sur les aperçus d'optimisation disponibles, regardez la présentation de Reynold Xin et Databricks SQL Serverless sous le capot : utiliser le ML pour obtenir le meilleur rapport prix-performance lors du Data+AI Summit.

Unifiez les données fragmentées avec Lakehouse Federation

Les organisations d'aujourd'hui ont des difficultés à découvrir, régir et interroger les données enfermées dans des silos et des systèmes fragmentés. Avec Lakehouse Federation, les équipes data peuvent utiliser Databricks SQL pour découvrir, interroger et gérer les données de plateformes externes telles que MySQL, PostgreSQL, Amazon Redshift, Snowflake, Azure SQL Database, Azure Synapse, Google's BigQuery (prochainement) et bien d'autres.

Autre avantage, Lakehouse Federation s'intègre parfaitement aux fonctionnalités avancées de Unity Catalog lors de l'accès à des sources de données externes depuis Databricks. Appliquez des règles de sécurité à l'échelle des lignes et des colonnes pour limiter l'accès aux informations sensibles. Remontez à l'origine de vos données grâce au data lineage afin de garantir leur qualité et leur conformité. Ajoutez des tags aux assets de votre catalogue fédéré pour simplifier la découverte des données.

Enfin, pour accélérer les transformations ou les jointures complexes sur des sources fédérées, Lakehouse Federation prend en charge les vues matérialisées qui réduisent la latence des requêtes.

Pour plus d'informations, regardez la présentation du Data+AI Summit à ce sujet, Lakehouse Federation : utilisation et gouvernance des sources de données externes à partir de Unity Catalog.

Développer sur l'architecture lakehouse avec l'API SQL Statement Execution

L'API SQL Statement Execution vous permet d'accéder à votre warehouse Databricks SQL via une API REST pour interroger ses données et récupérer les résultats. Et comme il existe des frameworks HTTP pour presque tous les langages de programmation, vous pourrez connecter directement un large éventail d'applications et de plateformes à un warehouse Databricks SQL.

L'API SQL Statement Execution de Databricks est disponible aux niveaux Databricks Premium et Entreprise. Pour en savoir plus, regardez notre présentation, suivez notre tutoriel (AWS | Azure), lisez la documentation (AWS | Azure) ou explorez notre dépôt d'échantillons de code.

Simplifiez le traitement de vos données avec les tables en streaming, les vues matérialisées et DB SQL dans les workflows

Avec les tables en streaming, les vues matérialisées et DB SQL dans les workflows, tout utilisateur SQL peut désormais traiter des données selon les bonnes pratiques de data engineering Importez, transformez, orchestrez et analysez les données en quelques lignes de SQL.

Les tables en streaming sont idéales pour alimenter la couche « Bronze ». En une seule déclaration SQL, importez les données depuis des stockages cloud (S3, ADLS, GCS), des bus de messages (EventHub, Kafka, Kinesis) et d'autres sources, quelle que soit l'échelle. L'importation se fait de façon incrémentale pour réduire la latence et optimiser les pipelines. Et il n'y a pas d'infrastructure complexe à gérer.

```
CREATE STREAMING TABLE web_clicks
AS
SELECT *
FROM STREAM
  read_files('s3://mybucket')
```

Les vues matérialisées réduisent les coûts et la latence des requêtes en précalculant les requêtes lentes et les opérations fréquentes. Leur rafraîchissement incrémentiel réduit la latence globale. En data engineering, elles servent à transformer les données. Les analystes les utilisent aussi dans un contexte de data warehousing parce qu'elles (1) accélèrent les requêtes des utilisateurs et les tableaux de bord de BI et (2) sécurisent le partage des données. Il suffit de quatre lignes de code pour créer une vue matérialisée qui va optimiser le traitement des données.

```
CREATE MATERIALIZED VIEW customer_orders
AS
SELECT
  customers.name,
  sum(orders.amount),
  orders.orderdate
FROM orders
  LEFT JOIN customers ON
    orders.custkey = customers.c_custkey
GROUP BY
  name,
  orderdate;
```

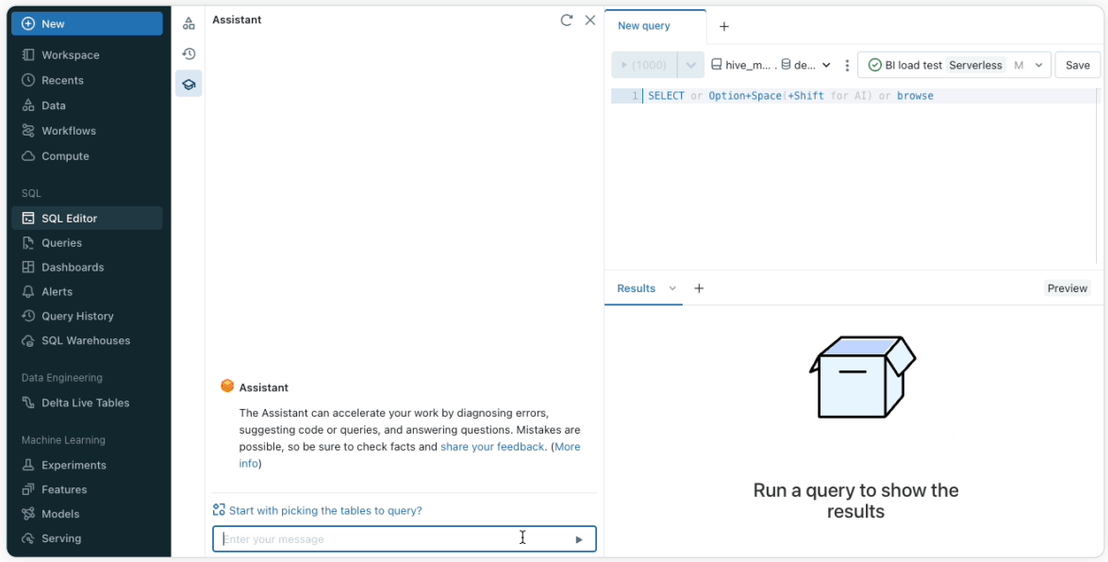
Besoin d'orchestrer avec DB SQL ? Les workflows vous permettent maintenant de planifier des requêtes SQL, des tableaux de bord et des alertes. Gérez des dépendances complexes entre les tâches et contrôlez les opérations antérieures grâce à l'UI Workflows ou via l'API.

Les tables en streaming et les vues matérialisées sont désormais en accès anticipé public. Pour en savoir plus, lisez notre article. Pour profiter de l'accès anticipé, remplissez ce formulaire. Les Workflows de DB SQL sont accessibles à tous : explorez la documentation (AWS | Azure) pour en savoir plus.

databricks

Databricks Assistant : rédigez du SQL de qualité plus vite avec le langage naturel

Databricks Assistant est un assistant IA sensible au contexte, intégré à Databricks Notebooks et à l'éditeur SQL. Sur la base d'une question en langage naturel, Databricks Assistant suggère la requête SQL correspondante. Pour comprendre une requête complexe, les utilisateurs peuvent demander à l'Assistant de l'expliquer en langage naturel, mettant ainsi la logique sous-jacente des résultats à la portée de tous.



Databricks Assistant exploite plusieurs signaux pour fournir des résultats plus précis et pertinents. Il utilise le contexte des cellules de code, des bibliothèques, des tables courantes, des schémas Unity Catalog et des tags pour traduire le langage naturel en requêtes et en code.

Nous prévoyons d'ajouter des intégrations avec DatabricksIQ pour apporter toujours plus de contexte à vos requêtes.

Gérez votre data warehouse en toute confiance

Les administrateurs et les équipes IT ont besoin d'outils pour comprendre l'utilisation de leur data warehouse. Avec les tables système, les profils de requête en direct et l'expiration des déclarations, les administrateurs peuvent corriger les problèmes dès qu'ils surviennent pour préserver les performances du data warehouse.

Obtenez davantage de visibilité et d'insights sur votre environnement SQL avec les tables système. Fournies par Databricks, elles contiennent des informations sur l'exécution des précédentes requêtes, les coûts, le lineage et d'autres aspects. En interrogeant les métadonnées et les métriques d'utilisation, vous pourrez répondre à des questions telles que « Quelles requêtes ont été exécutées, et par qui ? », « Quand et comment mes warehouses ont-ils été redimensionnés ? » ou « Qu'est-ce qu'on m'a facturé ? ». Comme les tables système sont intégrées à Databricks, vous avez accès à des fonctions natives comme les alertes SQL et les tableaux de bord SQL pour automatiser le processus de supervision et de signalement.

Trois tables système sont en accès anticipé public actuellement : logs d'audit, utilisation facturable et lineage (AWS | Azure). D'autres tables système dédiées aux événements du warehouse et à l'historique des requêtes seront ajoutées prochainement.

Par exemple, pour calculer le nombre de DBU utilisées chaque mois par SKU, vous pouvez interroger les tables système d'utilisation facturable.

```
SELECT sku_name, usage_date, sum(usage_quantity) as `DBUs`
    FROM system.billing.usage
WHERE
    month(usage_date) = month(NOW())
    AND year(usage_date) = year(NOW())
GROUP BY sku_name, usage_date
```

Grâce aux profils de requête en direct, les utilisateurs peuvent optimiser les charges à la volée sur la base d'informations de performance en temps réel. Visualisez les plans d'exécution et évaluez l'exécution des tâches en direct pour corriger l'éclatement des jointures ou les lectures de tables complètes. Le profilage de requête vous permet également d'optimiser les requêtes en cours sur votre data warehouse. Lisez la documentation (AWS | Azure) pour en savoir plus.

Vous voulez automatiser des contrôles ? Appliquez un délai d'expiration automatisé à un espace de travail ou une requête. Si la durée d'exécution dépasse le seuil d'expiration, la requête est automatiquement interrompue. Lisez la documentation (AWS | Azure) pour en savoir plus.

De nouvelles expériences puissantes dans DBSQL

Au cours de l'année dernière, nous avons travaillé à ajouter de nouvelles expériences de pointe à Databricks SQL. Nous sommes ravis de présenter de nouveaux outils qui mettent la puissance de l'IA au service des utilisateurs SQL : warehouses SQL accessibles sur toute la plateforme Databricks, nouvelle génération de tableaux de bord SQL et intégration de Python dans Databricks SQL.

Démocratiser l'analyse des données non structurées avec les fonctions IA

Avec les Fonctions IA, DB SQL met la puissance de l'IA à la portée du warehouse SQL. Profitez de tout le potentiel des données non structurées en réalisant des analyses de sentiment, des classifications de texte, des résumés, des traductions, etc. Les analystes pourront appliquer des modèles d'IA en libre-service, tandis que les ingénieurs data pourront développer des pipelines avec IA en toute indépendance.

Les fonctions IA sont très simples à utiliser. Imaginons par exemple qu'un utilisateur souhaite classer le sentiment de certains articles en différentes catégories : frustré (Frustrated), heureux (Happy), neutre (Neutral) ou satisfait (Satisfied).

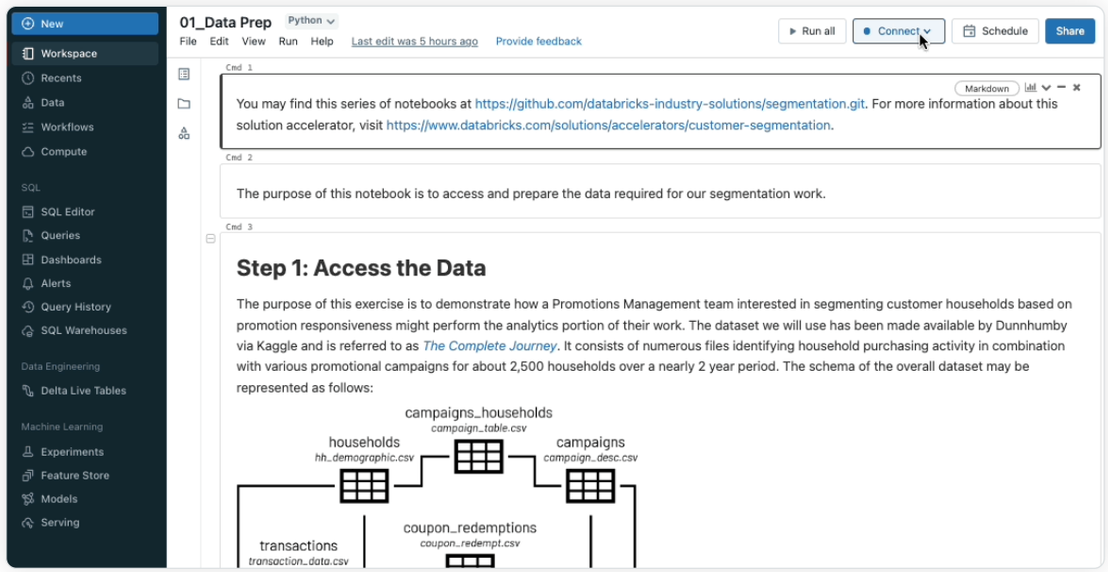
```
-- création d'un udf pour la classification du sentiment
CREATE FUNCTION classify_sentiment(text STRING)
  RETURNS STRING
  RETURN ai_query(
    'Dolly', -- nom du point de terminaison de mise à disposition du modèle
    named_struct(
      'prompt',
      CONCAT('Classifier le texte qui suit dans l'une des quatre catégories
[Frustrated, Happy, Neutral, Satisfied] :\n',
        text),
      'temperature', 0.5),
    'returnType', 'STRING');
```

```
-- utiliser l'udf
SELECT classify_sentiment(text) AS sentiment
FROM reviews;
```

Les fonctions IA sont désormais en accès anticipé public. Pour accéder à l'accès anticipé, remplissez ce formulaire. Pour en savoir plus, lisez également notre article de blog détaillé ou consultez la documentation (AWS | Azure).

La puissance des warehouses SQL au service des notebooks

Les warehouses Databricks SQL sont désormais accessibles dans les notebooks en version anticipée afin d'associer la souplesse des notebooks aux performances et au TCO des warehouses Databricks SQL Serverless et Pro. Pour activer les warehouses SQL dans les notebooks, sélectionnez simplement un warehouse disponible dans le menu déroulant du notebook.



Connecter des warehouses SQL serverless à partir des notebooks Databricks

Découvrir et partager des insights avec des tableaux de bord de nouvelle génération

Découvrez une nouvelle expérience de création de tableau de bord dans l'architecture lakehouse. Il suffit de sélectionner un dataset pour créer des visualisations impressionnantes avec, en option, une expérience SQL. Vous n'aurez plus besoin de gérer des requêtes et des objets de tableau de bord séparément : le modèle de contenu unifié simplifie les autorisations et la gestion. Enfin, diffusez votre tableau de bord dans toute l'organisation : tout utilisateur authentifié auprès de votre fournisseur d'identité pourra y accéder via un lien web sécurisé, même sans accès à Databricks.

Les nouveaux tableaux de bord Databricks SQL sont actuellement en accès anticipé privé. Contactez votre équipe de compte pour en savoir plus.

Profitez de la souplesse de Python en SQL

Profitez de la souplesse de Python dans Databricks SQL avec les fonctions Python définies par l'utilisateur (UDF). Intégrez des modèles de machine learning ou appliquez une logique de rédaction personnalisée pour le traitement et l'analyse des données en appelant des fonctions Python personnalisées directement dans votre requête SQL. Les UDF sont réutilisables et vous permettent d'appliquer un traitement homogène à vos pipelines de données et vos analyses.

Par exemple, pour supprimer les adresses e-mail et les numéros de téléphone d'un fichier, utilisez cette déclaration CREATE FUNCTION :

```
CREATE FUNCTION redact(a STRING)
RETURNS STRING
LANGUAGE PYTHON
AS $$
import json
keys = [“email”, “phone”]
obj = json.loads(a)
for k in obj:
  if k in keys:
    obj[k] = “REDACTED”
return json.dumps(obj)
$$;
```

Pour savoir comment participer à l'accès anticipé, cliquez ici.

Intégrations avec votre écosystème de données

Lors du Data+AI Summit, Databricks SQL a annoncé de nouvelles intégrations pour vous permettre d'utiliser vos outils en toute simplicité.

Databricks + Fivetran

Nous avons le plaisir d'annoncer la disponibilité générale de l'accès Fivetran dans Partner Connect pour tous les utilisateurs, administrateurs ou non, ayant des privilèges d'accès suffisants. Grâce à cette innovation, il est dix fois plus simple d'importer des données dans Databricks avec Fivetran. C'est une avancée majeure pour tous les clients Databricks qui peuvent désormais importer des données dans l'architecture lakehouse à l'aide de centaines de connecteurs Fivetran, dont Salesforce et PostgreSQL. Et maintenant, Fivetran prend pleinement en charge les warehouses serverless !

Lisez cet article de notre blog pour en savoir plus.

Databricks + dbt Labs

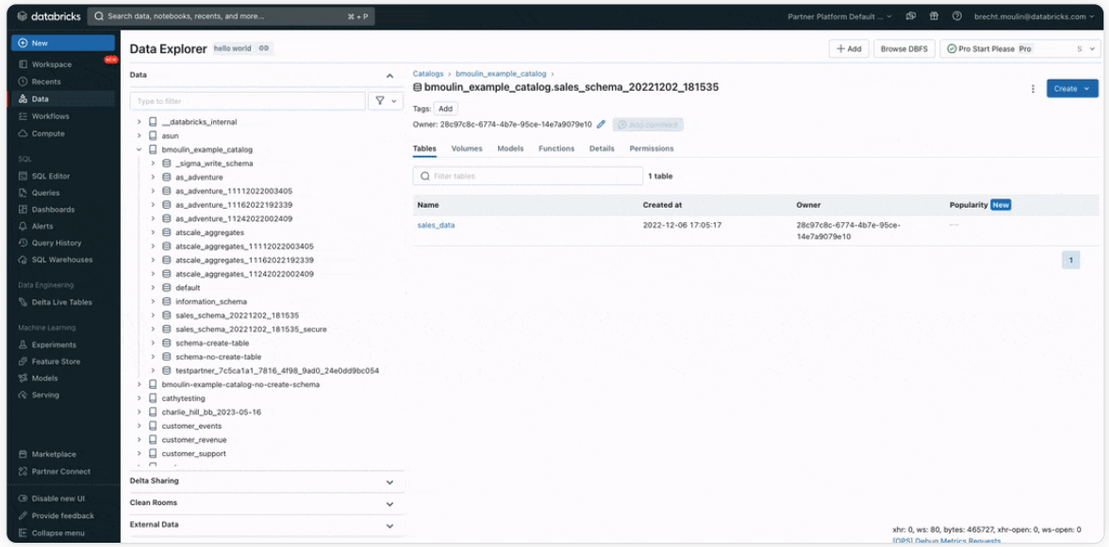
Avec Databricks et dbt Labs, simplifiez l'ingénierie analytique en temps réel sur l'architecture lakehouse. Le framework d'ingénierie analytique de dbt, extrêmement populaire, s'associe à la plateforme Databricks pour délivrer de puissantes fonctionnalités :

- dbt + Tables en streaming : l'importation en streaming depuis n'importe quelle source est désormais intégrée aux projets dbt. Les ingénieurs en analytique peuvent utiliser SQL pour définir et importer des données cloud ou en streaming dans leurs pipelines dbt.
- dbt + Vues matérialisées : dbt facilite considérablement la création de pipelines efficace en exploitant les puissantes capacités d'actualisation incrémentale de Databricks. Les utilisateurs peuvent utiliser dbt pour créer et exécuter des pipelines adossés à des VM afin de bénéficier de calculs incrémentiels plus efficaces pour réduire les coûts d'infrastructure.

Pour en savoir plus, lisez notre article détaillé.

Databricks + PowerBI : publiez dans des espaces de travail PowerBI

Publiez des datasets à partir de votre espace de travail Databricks vers votre espace de travail PowerBI Online en quelques clics ! Vous n'avez plus besoin de gérer des connexions odbc/jdbc : sélectionnez simplement le dataset que vous voulez publier. Il suffit de choisir les datasets ou le schéma que vous voulez publier et de sélectionner votre espace de travail PBI ! Les administrateurs et les créateurs de rapport BI pourront ainsi facilement prendre en charge des espaces de travail PowerBI sans avoir à utiliser l'application de bureau.



Démarrer avec Databricks SQL

Suivez le guide (AWS | Azure | GCP) pour mettre en place un warehouse SQL et prendre un bon départ avec Databricks SQL. Databricks SQL Serverless est actuellement proposé avec une remise de plus de 20 % : visitez notre page Tarifs pour en savoir plus.

Vous pouvez également regarder **Databricks SQL : le meilleur data warehouse serverless est un lakehouse** et **Nouveautés de Databricks SQL – Avec démos en direct** pour une vue d'ensemble complète.

SECTION 3.5

Gouvernance des données distribuées et environnements isolés avec Unity Catalog

par Max Nienu, Zeashan Pappa, Paul Roome et Sachin Thakur

Toute organisation qui s'appuie sur les données, l'analytique et l'IA a besoin d'une gouvernance efficace pour ses données. Et beaucoup prennent conscience de la valeur que peut apporter une gouvernance centralisée des données. Mais en dépit des meilleures intentions, la mise en œuvre de cette approche peut être difficile sans les processus et les ressources adéquats. Le rôle du directeur des données (CDO) ne fait qu'émerger, et la responsabilité de la définition et de l'exécution des politiques de gouvernance à l'échelle de l'organisation n'est pas encore claire.

Ce manque fréquent de centralisation entraîne des disparités dans les politiques et une multiplication des entités responsables, qui ne correspondent pas toujours aux délimitations des domaines d'activité, des sous-unités et autres divisions. Pour plus de simplicité, nous parlerons dans ce cas de gouvernance distribuée : les distinctions entre ces entités responsables font l'objet d'un consensus, mais il n'y a pas de fonction centrale de gouvernance des données.

Dans ce chapitre, nous verrons comment mettre en place un modèle de gouvernance distribué avec Unity Catalog de Databricks, qui fournit une solution de gouvernance unifiée pour les données, l'analytique et l'IA sur le lakehouse.

Évolution de la gouvernance des données dans Databricks

Avant l'introduction de Unity Catalog, le concept d'espace de travail était monolithique : chaque espace avait son propre metastore, sa gestion des utilisateurs et ses tables ACL. En raison des frontières de gouvernance créées par cette approche, toute tentative de mise en cohérence reposait sur d'importants efforts de duplication. Pour y faire face, certains clients exécutaient des pipelines ou du code visant à synchroniser leurs metastores et leurs ACL, tandis que d'autres créaient des metastores indépendants utilisés par l'ensemble des espaces de travail. Ces solutions, chères à maintenir, exigeaient de prendre dès le départ des décisions d'architecture sur la segmentation des données de l'organisation.

Gouvernance des données avec Unity Catalog

Pour surmonter ces obstacles, Databricks a mis au point Unity Catalog, qui simplifie la gouvernance des données tout en cherchant à faciliter la collaboration et le partage. Premièrement, il fallait établir un espace de nommage commun donnant l'accès à toutes les données d'une organisation.

C'était un véritable défi dans le contexte d'une gouvernance distribuée, mais Unity Catalog fournit de nouveaux mécanismes d'isolement au sein de l'espace de nommage, pour remplacer la multiplication des metastores Hive. Ces mécanismes permettent à des groupes de fonctionner de façon indépendante, avec peu ou pas d'interaction, et d'isoler, par exemple, les environnements de développement et de production.

Hive Metastore et Unity Catalog dans Databricks

Avec Hive, le metastore délimite un service. Multiplier les metastores implique donc d'héberger plusieurs services Hive et bases de données. Unity Catalog est un service à l'échelle de la Data Intelligence Platform de Databricks : il n'impose aucune délimitation.

Unity Catalog fournit un espace de nommage commun pour centraliser l'encadrement et l'audit de vos données.

Avec Hive, il fallait généralement utiliser plusieurs metastores ayant chacun leur espace de nommage, pour isoler les environnements de développement et de production ou séparer les données des différentes unités commerciales.

Unity Catalog remplit cette fonction grâce à des mécanismes d'isolement dynamiques qui ne compromettent pas les possibilités de partage de données et de collaboration, et n'exigent pas de prendre des décisions d'architecture initiales.

Collaborer entre équipes et sur différents environnements

Avec une plateforme de données, il est souvent indispensable d'isoler différents environnements : développement et production, groupes commerciaux, équipes, unités commerciales, etc.

Commençons par définir ce que sont ces frontières dans une plateforme de données comme Databricks :

- Les utilisateurs ne doivent avoir accès aux données qu'en fonction de règles établies
- Les données peuvent être gérées par des personnes ou des équipes désignées
- Les données doivent être physiquement séparées dans le stockage
- Les données ne doivent être accessibles que dans des environnements désignés

Les utilisateurs ne doivent avoir accès aux données qu'en fonction de règles établies

Les organisations ont généralement des exigences strictes et des obligations réglementaires concernant l'accès aux données pour assurer leur sécurité. Les informations liées aux salaires ou les numéros de carte bancaire, par exemple, sont très encadrés.

L'accès à ce type d'informations est étroitement surveillé et fait l'objet de contrôles réguliers. Unity Catalog fournit un contrôle granulaire sur les assets qu'il contient afin de respecter ces normes. Grâce à ces mécanismes, les utilisateurs ne peuvent voir et interroger que les données qu'ils sont autorisés à consulter.

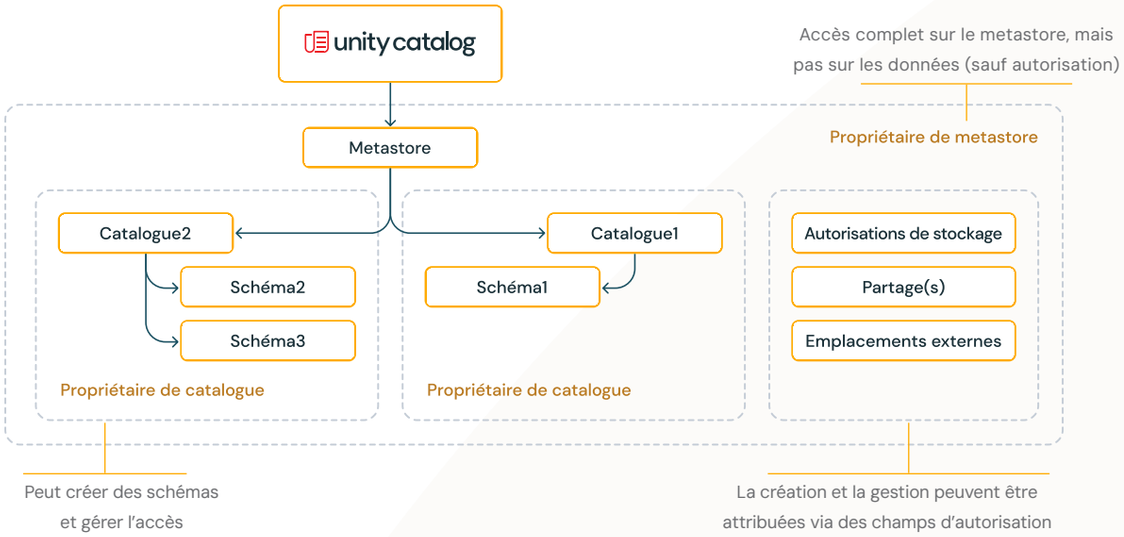
Les données peuvent être gérées par des personnes ou des équipes désignées

Unity Catalog donne le choix entre la gouvernance centralisée et la gouvernance distribuée.

Selon le modèle de gouvernance centralisée, les administrateurs sont responsables du metastore. Ils peuvent devenir propriétaires de tout objet et définir des ACL et des politiques.

Dans le modèle de gouvernance distribuée, un catalogue - ou un ensemble de catalogues - est considéré comme un domaine de données. Le propriétaire du catalogue peut créer et posséder tous les assets. Il peut aussi gérer la gouvernance de ce domaine. Les propriétaires de domaines peuvent agir indépendamment de ceux des autres domaines.

Nous recommandons vivement de définir un groupe comme propriétaire ou donneur d'ordre pour ces deux options si la gestion se fait via des outils.



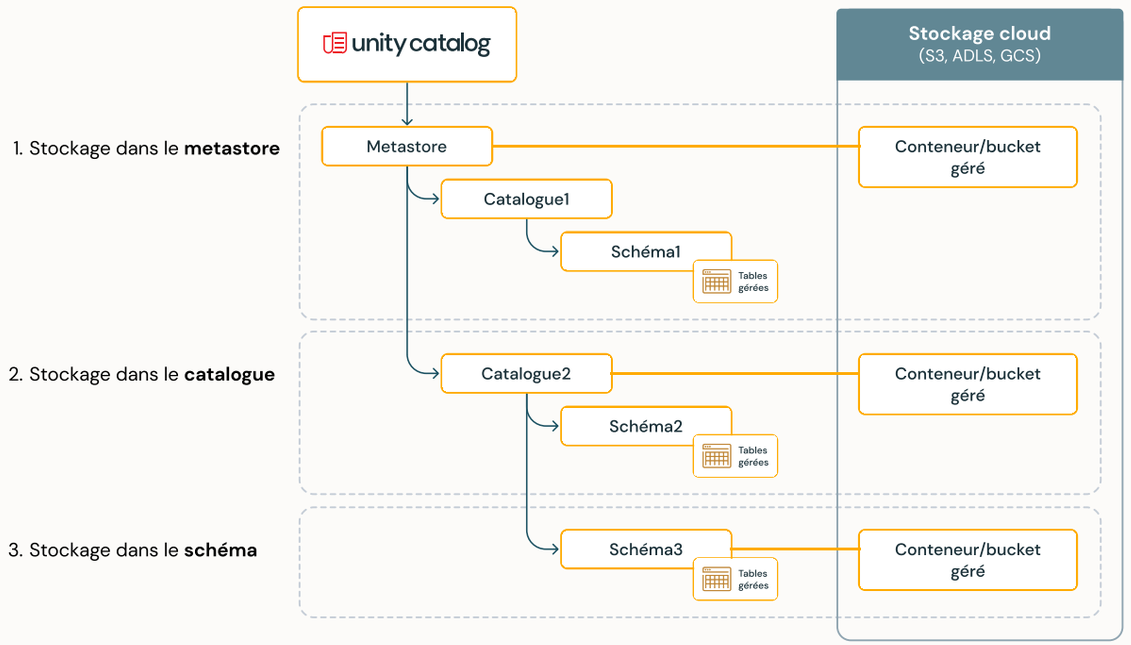
Les données doivent être physiquement séparées dans le stockage

Par défaut, lors de la création d'un metastore UC, l'administrateur de compte Databricks fournit un seul emplacement cloud par défaut et des identifiants pour les tables gérées.

Les organisations qui doivent isoler physiquement les données pour des raisons réglementaires, pour reproduire des portées SDLC, séparer des unités commerciales ou même affecter les coûts, doivent envisager les fonctions de source de données gérée au niveau du catalogue et du schéma.

Unity Catalog vous permet de choisir des options par défaut pour la séparation des données. Par défaut, toutes les données sont stockées dans le metastore. Mais avec la prise en charge des sources de données gérées au niveau des catalogues et des schémas, vous pouvez isoler physiquement les stockages et les accès afin de respecter les obligations de gouvernance et de gestion de l'entreprise.

Lors de la création des tables gérées, le stockage des données se fait selon l'emplacement du schéma (le cas échéant) suivi de l'emplacement du catalogue (le cas échéant), et n'utilise celui du metastore que si les deux précédents n'ont pas été définis.

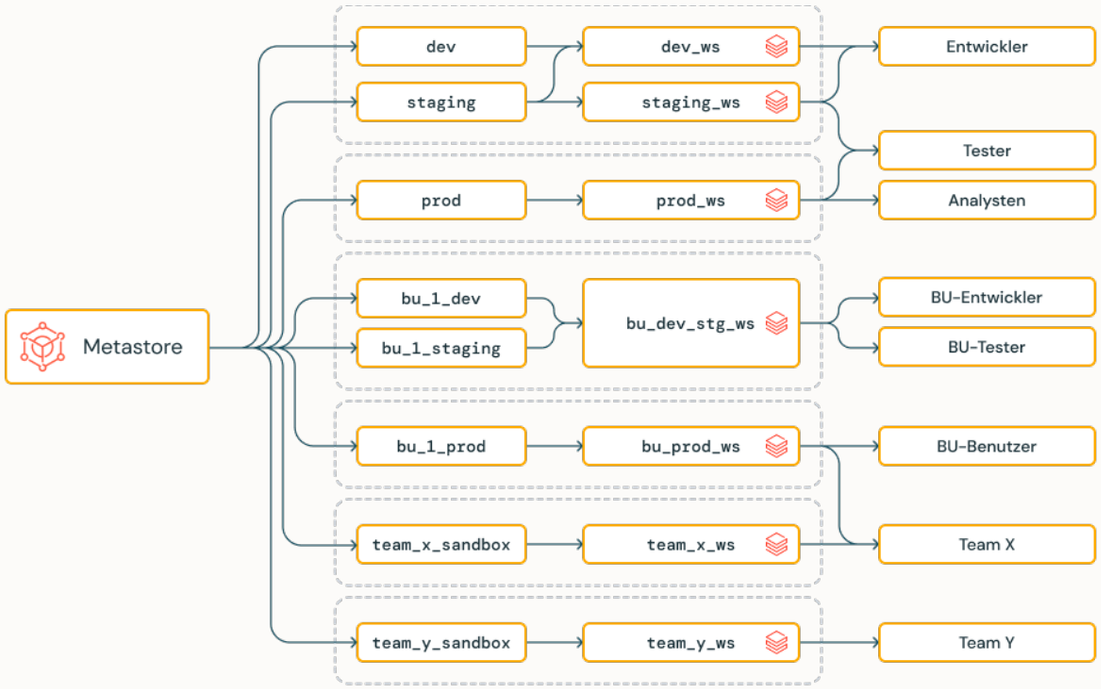


databricks

Les données ne doivent être accessibles que dans des environnements désignés, selon leur fonction.

Les politiques internes et les obligations de conformité exigent souvent que certaines données ne soient accessibles que dans certains environnements. Pensez, par exemple, aux environnements de développement et de production, ou aux environnements HIPAA et à ceux qui contiennent des données personnelles à des fins d'analyse : l'accès aux données et le contexte d'utilisation sont strictement encadrés. Dans certains cas, il faut interdire que certains groupes ou domaines de données ne soient croisés ou combinés.

Dans Databricks, un environnement est un espace de travail. Unity Catalog possède une fonctionnalité qui permet de « lier » des catalogues à un espace de travail. Grâce à ces ACL sensibles à l'environnement, vous pouvez faire en sorte que seuls certains catalogues soient accessibles dans un espace de travail donné, quels que soient les ACL de l'utilisateur. Autrement dit, l'administrateur du metastore ou le propriétaire du catalogue peuvent préciser dans quels espaces de travail un catalogue de données est accessible. Ces règles sont définies à l'aide de notre UI ou par API/terraform pour simplifier l'intégration. Nous avons récemment publié un article expliquant comment contrôler Unity Catalog via terraform en fonction de votre modèle de gouvernance.



L'accès aux données et leur disponibilité peuvent être isolés selon les contours des espaces de travail et des groupes.
Les utilisateurs ne peuvent accéder à certains catalogues que dans des environnements donnés.

Conclusion

En plaçant Unity Catalog au centre de votre architecture lakehouse, vous pouvez mettre en place une gouvernance souple et évolutive sans diminuer votre capacité à gérer et partager efficacement les données. Avec Unity Catalog, vous pouvez vous libérer des contraintes de votre metastore Hive, isoler des données et faciliter la collaboration en fonction des besoins spécifiques de votre entreprise. Suivez les guides Unity Catalog (AWS, Azure) pour commencer. Téléchargez cet e-book gratuit sur la gouvernance des données, de l'analytique et de l'IA pour connaître les bonnes pratiques et adopter une stratégie efficace pour votre data lakehouse.

SECTION 3.6

Le Guide du routard des modèles de privilèges et du contrôle d'accès dans Unity Catalog

Une approche simple et claire des concepts du modèle de privilèges de Unity Catalog pour répondre à différents usages et besoins

par Som Natarajan et Vuong Nguyen

Avec l'augmentation du volume, de la vélocité et de la diversité des données, les organisations doivent miser sur de solides pratiques de gouvernance pour atteindre leurs objectifs. Unity Catalog est une solution de gouvernance granulaire pour les données et l'IA sur la Data Intelligence Platform de Databricks. Il simplifie la sécurité et la gouvernance des données d'entreprise en offrant un mécanisme centralisé pour l'administration et le contrôle des accès.

Revenons un temps en arrière. Avant que Unity Catalog n'unifie le modèle d'autorisation des fichiers et des tables, en prenant en charge tous les langages, les clients utilisaient les tables ACL (TACL) appliquées à l'espace de travail, qui étaient limitées à certaines configurations de cluster et n'étaient compatibles qu'avec Python et SQL. Unity Catalog et les TACL vous permettent de contrôler l'accès aux objets à sécuriser comme les catalogues, les schémas (bases de données), les tables et les vues, mais des nuances les distinguent.

Il est essentiel de comprendre le modèle d'accès aux objets pour mettre en place une gouvernance des données à grande échelle avec Unity Catalog. C'est particulièrement vrai si vous avez déjà implémenté le modèle TACL et cherchez à adopter Unity Catalog pour profiter de ses nouvelles fonctionnalités – prise en charge de plusieurs langues, contrôle d'accès centralisé et data lineage.

Les axiomes du modèle d'accès de Unity Catalog

- Les privilèges de Unity Catalog sont définis au niveau du metastore : les autorisations de UC portent toujours sur des identités au niveau du compte, alors que les autorisations TACL, définies dans le catalogue hive_metastore, portent sur des identités de l'espace de travail.
- Héritage des privilèges : les objets de Unity Catalog sont hiérarchisés et les privilèges s'héritent du haut vers le bas. L'objet de plus haut niveau dont on puisse hériter des privilèges est le catalogue.
- La propriété des objets joue un rôle clé : les privilèges ne peuvent être accordés que par un administrateur de metastore, le propriétaire d'un objet ou celui du catalogue ou du schéma qui le contient. Un objet ne peut être supprimé que par son propriétaire ou celui du catalogue ou du schéma qui le contient.
- Privilèges USE au niveau des frontières : USE CATALOG/SCHEMA est requis pour interagir avec les objets d'un catalogue ou d'un schéma. Le privilège USE ne permet toutefois pas de parcourir les métadonnées de l'objet hébergées dans le catalogue ou le schéma.
- Les autorisations sur les objets dérivés sont simplifiées : selon Unity Catalog, le propriétaire d'une vue n'a besoin que du privilège SELECT, de USE SCHEMA sur le schéma parent de la vue et de USE CATALOG sur le catalogue parent. Avec les TACL, en revanche, le propriétaire d'une vue doit être aussi propriétaire de toutes les tables et vues référencées.

Quelques axiomes plus complexes

- Sécurisé par défaut : seul les clusters ayant des modes d'accès propres à Unity Catalog (partagé ou utilisateur unique) ont accès aux données de Unity Catalog. Avec les TACL, tous les utilisateurs ont accès à toutes les données sur les clusters non partagés.
- Limitation des clusters à utilisateur unique : les clusters à utilisateur unique ne prennent pas en charge les vues dynamiques. Les utilisateurs doivent avoir le privilège SELECT sur toutes les tables et vues référencées pour lire une vue.
- Pas de prise en charge de ANY FILE ni de ANONYMOUS FUNCTION : Unity Catalog ne prend pas en charge ces autorisations qui peuvent être employées pour contourner des restrictions en autorisant un utilisateur non privilégié à exécuter du code privilégié.

Modèles intéressants

Le modèle d'accès de Unity Catalog permet de mettre en place de nombreux profils de gouvernance.

Exemple 1 : Autorisations homogènes sur l'ensemble des espaces de travail

L'axiome 1 permet aux équipes produit de définir des autorisations sur leur produit au sein de leur propre espace de travail, puis de les répercuter sur tous les autres espaces, quelle que soit la provenance des consommateurs.

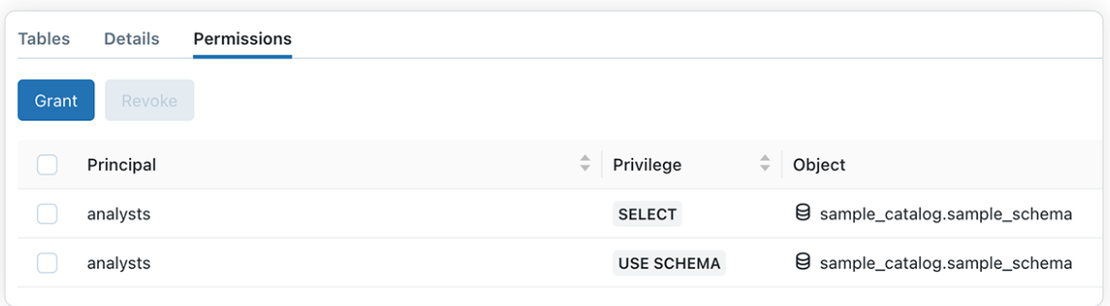
Exemple 2 : Définir des limites pour le partage des données

L'axiome 2 permet aux propriétaires de catalogue/schéma de créer des règles par défaut pour leurs données. Avec les commandes suivantes, par exemple, les membres de l'équipe machine learning vont créer des tables dans un schéma et pouvoir lire les tables des uns et des autres.

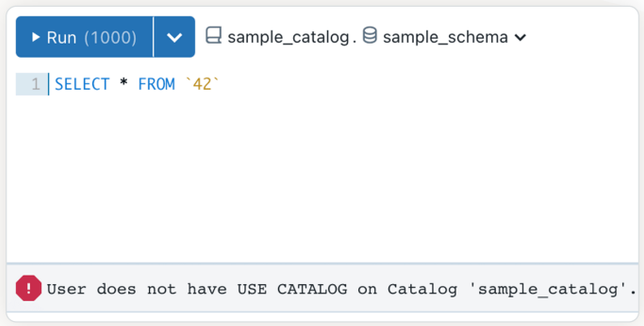
```
CREATE CATALOG ml;
CREATE SCHEMA ml.sandbox;
GRANT USE_CATALOG ON CATALOG ml TO ml_users;
GRANT USE_SCHEMA ON SCHEMA ml.sandbox TO ml_users;
GRANT CREATE TABLE ON SCHEMA ml.sandbox TO ml_users;
GRANT SELECT ON SCHEMA ml.sandbox TO ml_users;
```

L'axiome 4, quant à lui, permet désormais aux propriétaires de catalogue/schéma d'encadrer le partage des données qu'ils produisent. Un propriétaire de table qui accorde SELECT à un autre utilisateur ne lui donne pas un accès en lecture à la table tant qu'il ne lui a pas aussi donné des privilèges USE CATALOG sur le catalogue parent, et USE SCHEMA sur le schéma parent.

Dans l'exemple ci-dessous, sample_catalog est la propriété de l'utilisateur A, et l'utilisateur B a créé le schéma sample_schema et la table 42. Bien que les analystes aient reçu les autorisations USE SCHEMA et SELECT, ils ne peuvent pas interroger la table à cause des limites imposées par l'utilisateur A.



Page des autorisations montrant que le groupe des analystes a les autorisations SELECT et USE SCHEMA sur sample_catalog.sample_schema

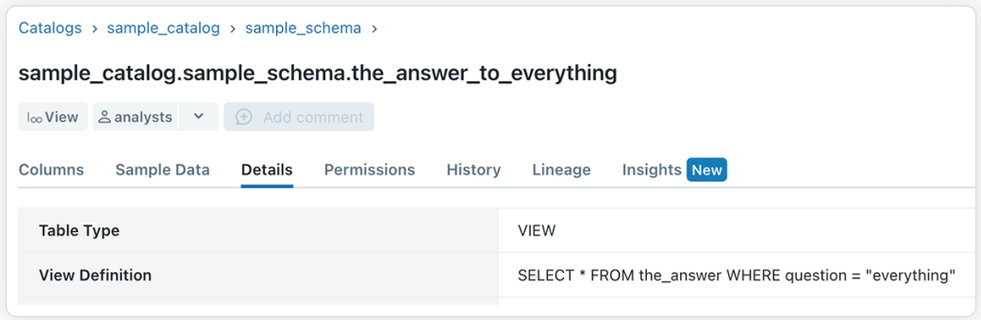


Page de requête avec message d'erreur

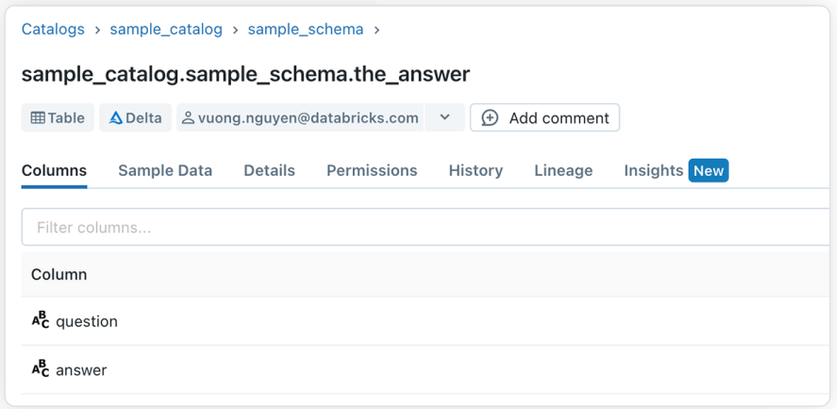
Exemple 3 : Simplifier le partage d'une logique métier

Les consommateurs de données ont besoin de partager leurs travaux et leur logique de transformation, et les vues permettent de le faire de façon réutilisable.

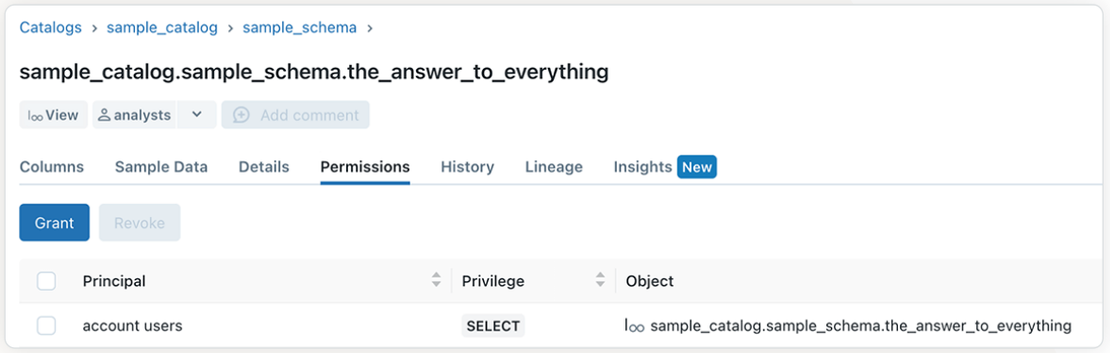
L'axiome 5 offre un moyen simple de créer des vues sans avoir à échanger avec les propriétaires des tables.



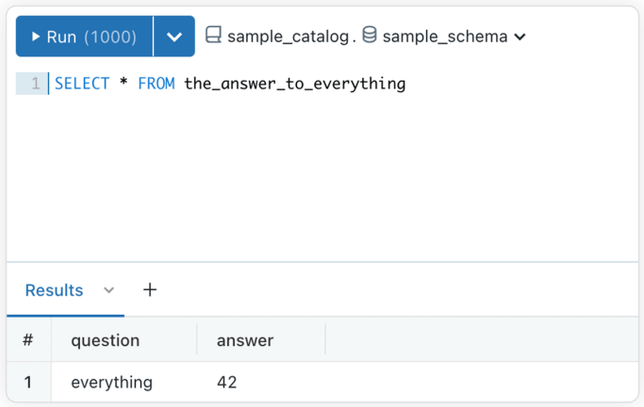
Définition de la vue



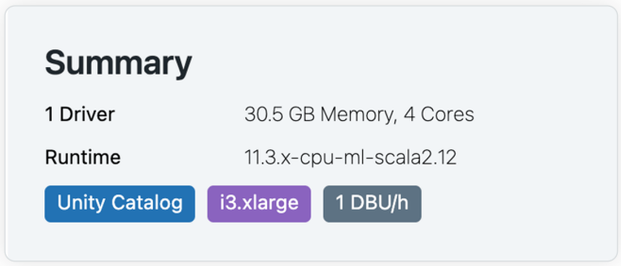
Propriété de la table



Page d'autorisation montrant une vue possédée par le groupe analystes ; le groupe utilisateurs de compte a l'autorisation SELECT



Page de requête montrant le résultat de la réponse à tout



Résumé du cluster indiquant la prise en charge de Unity Catalog

Exemple 4 : éliminer les fuites de données

Grâce à l'axiome 6, les propriétaires de donnée savent que leurs données sont protégées des accès non autorisés dus à un défaut de configuration du cluster. Un cluster qui n'est pas configuré avec le bon mode d'accès ne pourra pas accéder aux données de Unity Catalog.

Les utilisateurs peuvent vérifier si leurs clusters peuvent accéder aux données Unity Catalog grâce à une infobulle pratique sur la page Créer des clusters.

Quand les propriétaires de données comprennent le modèle de privilèges et le contrôle d'accès, ils peuvent utiliser Unity Catalog pour simplifier la gestion des accès à grande échelle.

Prochainement, de nouvelles fonctionnalités aideront les administrateurs et les propriétaires de données à créer des politiques d'accès plus complexes :

- **Filtrage de lignes et masquage de colonnes :** Utilisez des fonctions SQL standard pour définir des filtres de ligne et masquer des colonnes afin d'affiner le contrôle d'accès.
- **Contrôle d'accès basé sur les attributs :** Définissez des politiques d'accès en fonction des tags (attributs) de vos assets de données.

SECTION

04

Cas d’usage d’analytique avec Databricks

SECTION 4.1

Bâtir une solution d'analytique marketing avec Fivetran et dbt sur Databricks

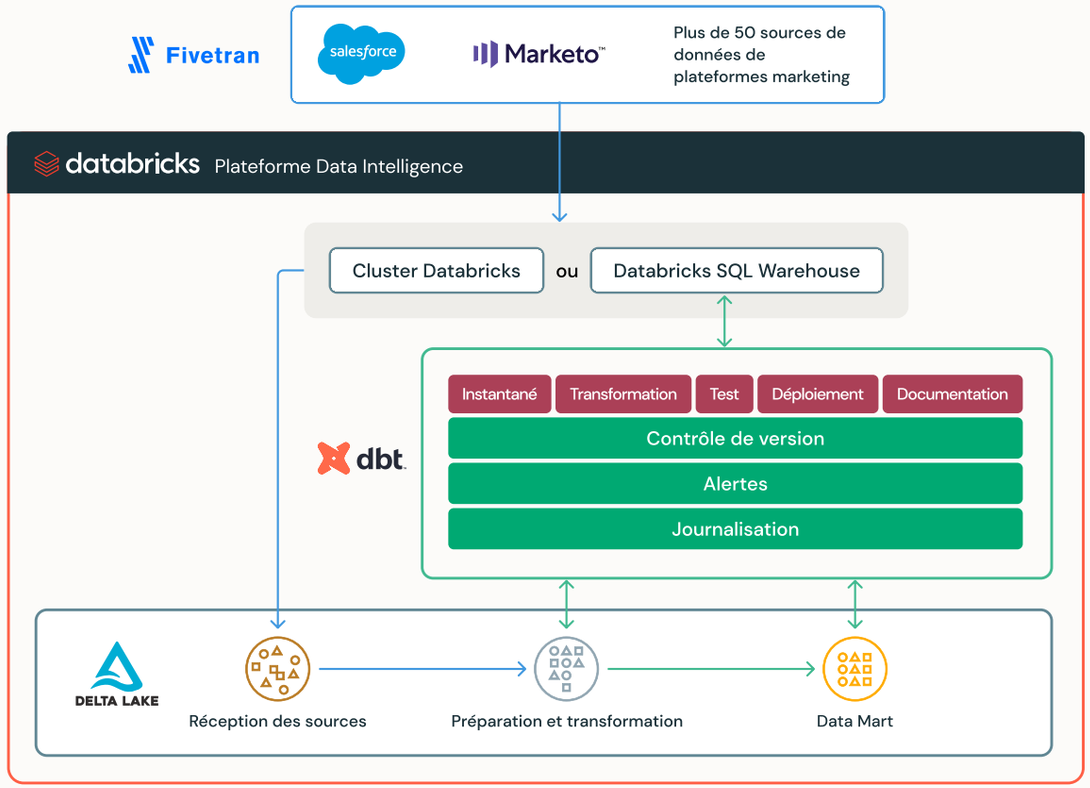
par Tahir Fayyaz, Bilal Aslam et Robert Saxby

Pour lancer des campagnes commerciales, les équipes marketing utilisent différentes plateformes qui génèrent des quantités de données précieuses, mais déconnectées. Rassembler toutes ces données peut être extrêmement rentable, comme l'a démontré le Groupe Publicis qui a augmenté de 50 % les revenus de ses campagnes.

La plateforme Databricks, qui unifie le data warehousing et les cas d'usage de l'IA, est idéale pour bâtir une solution d'analytique : elle maintient une source unique de vérité et prend en charge les applications d'IA/ML. Nous nous appuyons également sur deux solutions partenaires, Fivetran et dbt, pour soutenir de nombreuses utilisations en analytique marketing, comme l'analyse de l'attrition et de la valeur vie client, la segmentation des clients et l'efficacité de la publicité.

Fivetran permet d'importer les données de plus de 50 plateformes marketing dans Delta Lake sans avoir à créer et maintenir des pipelines complexes. En cas de modification ou de panne d'une API de la plateforme, Fivetran corrige les intégrations pour maintenir l'afflux de vos données.

dbt est un framework open source très utilisé qui permet de créer des pipelines de données très simplement en SQL. Tout est en texte brut et organisé en répertoires, pour faciliter le contrôle de version, le déploiement et les tests. Une fois les données importées dans Delta Lake, dbt sert à transformer, tester et documenter les données. Le data mart d'analytique marketing bâti sur les données importées est immédiatement utilisable dans le cadre de nouvelles campagnes et initiatives marketing.



Fivetran et dbt peuvent lire et écrire sur Delta Lake en utilisant un cluster Databricks ou un warehouse Databricks SQL.

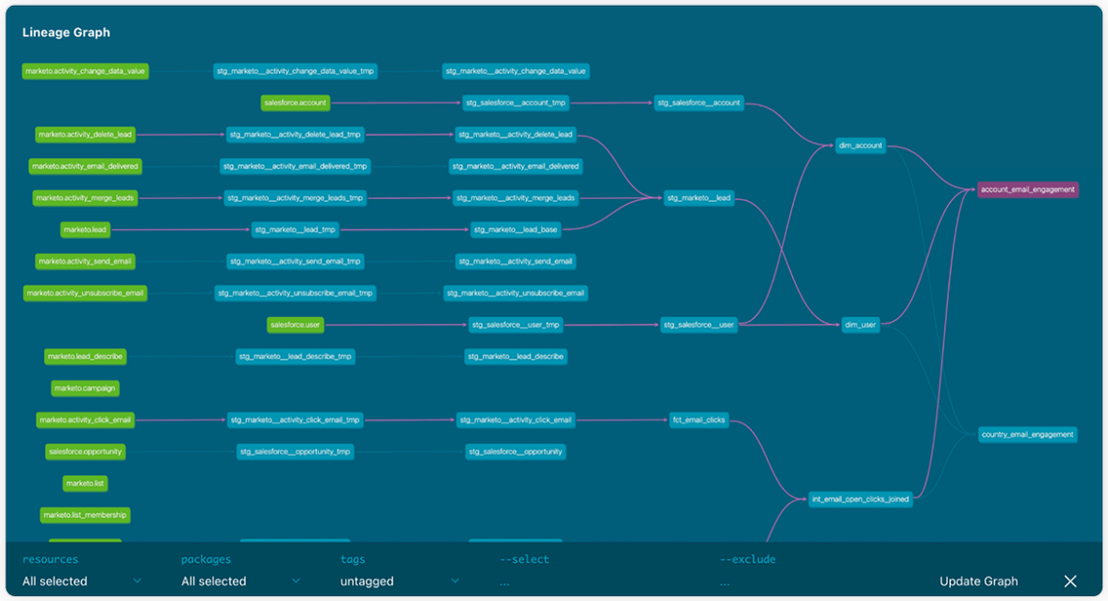
Fivetran et dbt font partie de Databricks Partner Connect, un portail central pour découvrir et connecter en toute sécurité des outils de données, d'analytique et d'IA dans la plateforme Databricks. Quelques clics suffisent à configurer et connecter ces outils (et bien d'autres) directement depuis votre espace de travail Databricks.

databricks

Bâtir une solution d'analytique marketing

Dans cette démo pratique, nous allons utiliser Fivetran pour importer des données Marketo et Salesforce dans Databricks, puis dbt pour transformer, tester et documenter votre modèle de données d'analytique marketing.

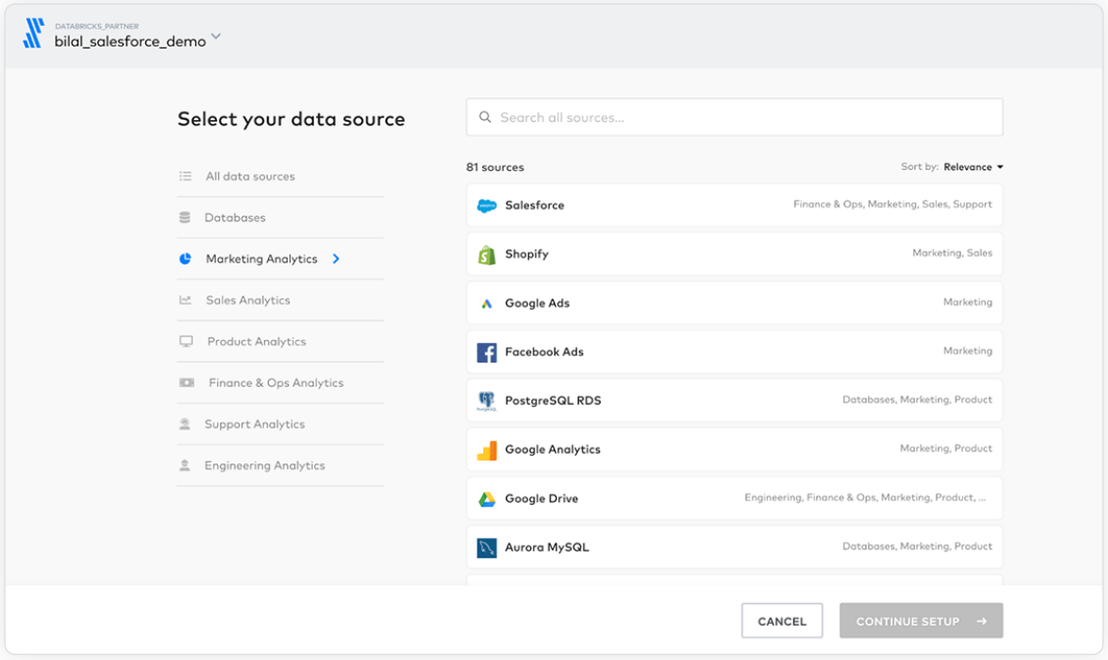
Tout le code de la démo est disponible sur Github, dans le référentiel workflows-examples.



Graphe de lineage dbt montrant les sources de données et les modèles

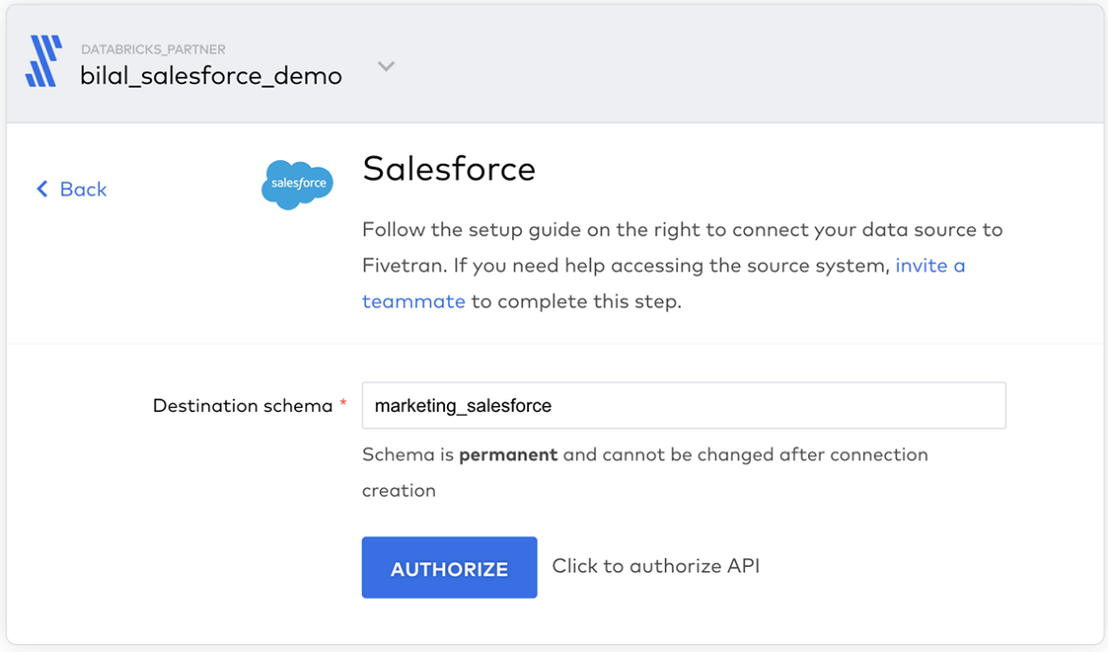
Le graphe de lineage du modèle dbt final aura cet aspect. Les tables sources Fivetran sont à gauche, en vert, et les modèles d'analytique marketing qui en résultent sont sur la droite. Sélectionnez un modèle pour mettre en évidence (en magenta) ses dépendances avec les autres modèles.

Ingestion des données avec Fivetran



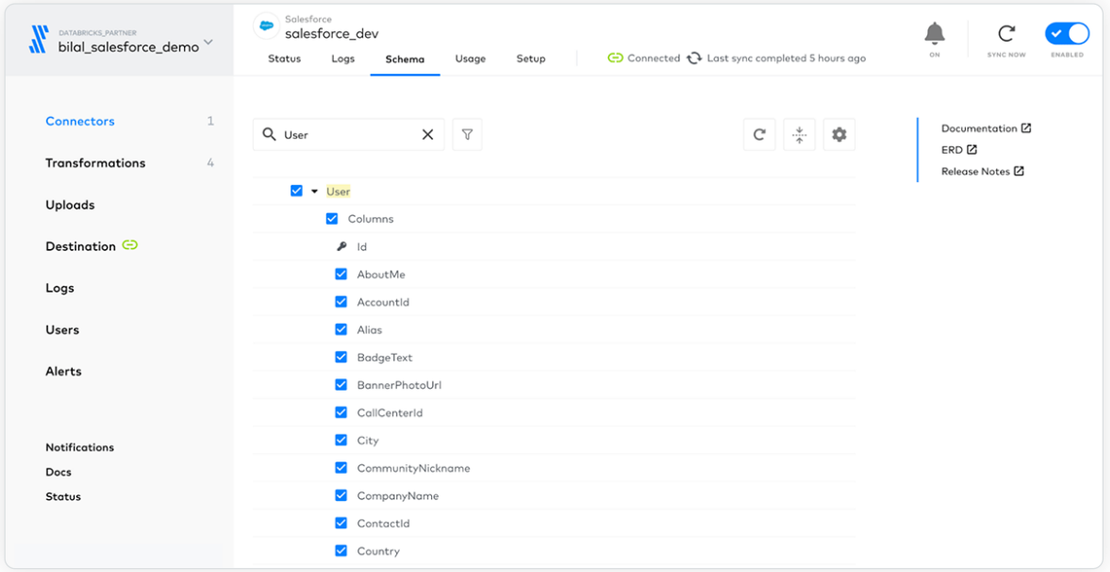
Fivetran propose de nombreux connecteurs de sources de données marketing.

Créez des connexions Salesforce et Marketo dans Fivetran pour commencer à importer les données marketing dans Delta Lake. Quand vous créez des connexions, Fivetran crée et gère automatiquement un schéma pour chaque source de données, dans Delta Lake. Par la suite, nous utiliserons dbt pour transformer, nettoyer et agréger ces données.



Définissez le schéma de destination dans Delta Lake pour la source de données Salesforce

Pour la démo, nommez les schémas qui seront créés dans Delta Lake marketing_salesforce et marketing_marketo. Si les schémas n'existent pas, Fivetran va les créer au cours de la phase initiale d'importation.



Sélectionnez les objets de source de données à synchroniser en tant que tables Delta Lake

Vous pouvez ensuite choisir les objets à synchroniser avec Delta Lake : chaque objet sera enregistré sous la forme d'une table. Avec Fivetran, il est également très simple de voir et gérer les colonnes synchronisées pour chaque table :

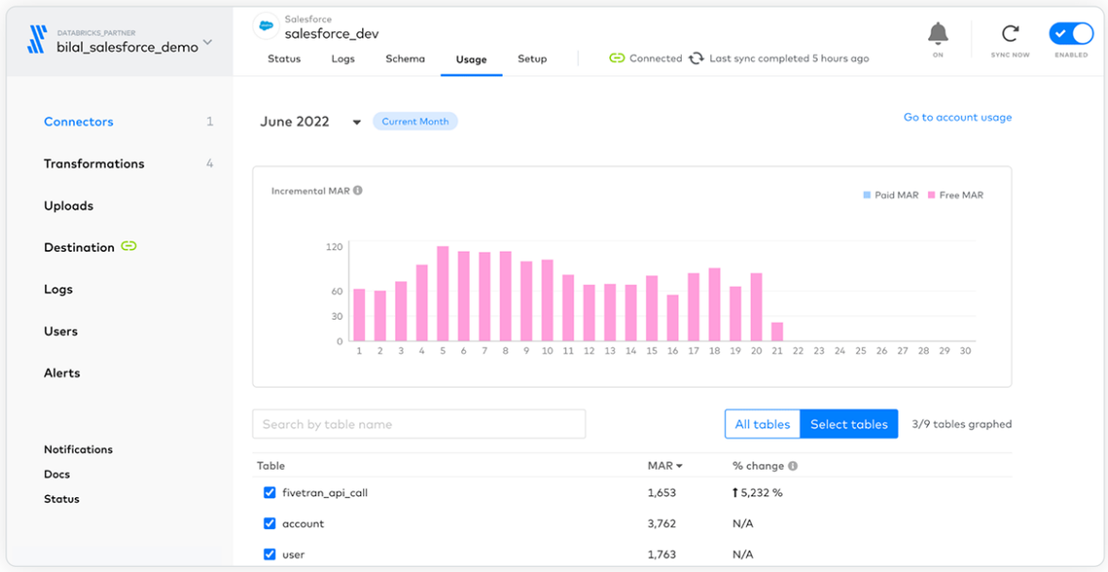


Tableau de bord de surveillance Fivetran pour monitorer les lignes actives synchronisées sur une base mensuelle

Fivetran propose également un tableau de bord de surveillance pour analyser le nombre de lignes actives mensuelles synchronisées chaque jour et chaque mois dans chaque table, entre autres statistiques et journaux utiles.

Modélisation des données avec dbt

Maintenant que toutes les données marketing sont dans Delta Lake, utilisez dbt pour créer votre modèle de données en suivant les étapes ci-dessous.

Création d'un projet dbt local et connexion à Databricks SQL

Créez votre environnement de développement dbt local dans l'IDE de votre choix en suivant les instructions pour dbt Core et dbt-databricks.

Montez un nouveau projet dbt et connectez-le à un warehouse Databricks SQL à l'aide de dbt init, qui vous demandera les informations suivantes :

```
$ dbt init
Enter a name for your project (letters, digits, underscore):
Which database would you like to use?
[1] databricks
[2] spark

Enter a number: 1
host (yourorg.databricks.com):
http_path (HTTP Path):
token (dapiXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX):
schema (default schema that dbt will build objects in):
threads (1 or more) [1]:
```

Une fois le profil configuré, vous pouvez tester la connexion :

```
$ dbt debug
```

Installation des paquets de modèle dbt Fivetran pour la phase de préparation

Pour utiliser les données Marketo et Salesforce, la première étape consiste à créer les tables qui seront la source de notre modèle. Heureusement, Fivetran propose des paquets de modèles dbt Fivetran qui simplifient et accélèrent ce processus. Pour cette démo, nous allons utiliser les paquets marketo_source et `salesforce_source`.

Pour installer les paquets, ajoutez simplement un fichier **`packages.yml`** à la racine de votre projet dbt, puis ajoutez les paquets `marketo-source`, `salesforce-source` et `fivetran-utils` :

```
packages:
  - package: dbt-labs/spark_utils
    version: 0.3.0
  - package: fivetran/marketo_source
    version: [“>=0.7.0”, “=0.4.0”, “

Pour télécharger et utiliser les paquets, lancez :
```

```
$ dbt deps
```

Vous devriez maintenant voir les paquets Fivetran installés dans le répertoire des paquets.

Mise à jour de dbt_project.yml avec les modèles dbt Fivetran

Vous devez modifier quelques configurations dans le fichier `dbt_project.yml` pour avoir la certitude que les paquets Fivetran fonctionneront correctement dans Databricks.

Le fichier `dbt_project.yml` se trouve dans le dossier racine de votre projet dbt.

Remplacer les macros dbt_utils avec spark_utils

Les modèles Fivetran dbt utilisent les macros du paquet dbt_utils, mais certaines d’entre elles doivent être modifiées pour fonctionner avec Databricks, ce dont se charge le paquet spark_utils.

Il fournit des modifications pour certaines macros `dbt_utils`, que vous pouvez appliquer à l’aide de la configuration du dispatch dans le fichier `dbt_project.yml`. Avec cette configuration, dbt cherchera d’abord les macros dans le paquet `spark_utils` lors de la résolution des macros dans l’espace de nommage `dbt_utils`.

```
dispatch:
 - macro_namespace: dbt_utils
   search_order: [‘spark_utils’, ‘dbt_utils’]
```

Variables des schémas marketo_source et salesforce_source

Les paquets Fivetran demandent de définir le catalogue (une base de données dans dbt) et le schéma de destination des données importées par Fivetran.

Ajoutez ces variables au fichier `dbt_project.yml` du projet. Le catalogue `hive_metastore` sera utilisé par défaut si `_database` est vide. Les noms des schémas sont ceux que vous aurez définis à la création des connexions dans Fivetran.

```
vars:
 marketo_source:
   marketo_database: # laisser vide pour utiliser le catalogue par défaut hive_
metastore
   marketo_schema: marketing_marketo
 salesforce_source:
   salesforce_database: # laisser vide pour utiliser le catalogue par défaut hive_
metastore
   salesforce_schema: marketing_salesforce
```

Schéma cible pour les modèles intermédiaires Fivetran

Pour éviter que toutes les tables intermédiaires créées par les modèles sources Fivetran soient également créées dans le schéma cible par défaut, il peut être utile de définir un schéma intermédiaire distinct.

Dans le fichier `dbt_project.yml`, ajoutez le nom du schéma intermédiaire, qui sera ajouté en suffixe au nom de schéma par défaut.

```
models:
 marketo_source:
   +schema: nom_schéma_intermédiaire # laisser vide pour utiliser target_schema
par défaut
 salesforce_source:
   +schema: nom_schéma_intermédiaire # laisser vide pour utiliser target_schema
par défaut
```

D'après ce qui est indiqué ci-dessus, si votre schéma cible défini dans `profiles.yml` est **`mkt_analytics`**, le schéma utilisé pour les tables `marketo_source` et `salesforce_source` sera **`mkt_analytics_nom_schéma_intermédiaire`**.

Désactivation des tables manquantes

À ce stade, vous pouvez exécuter les paquets de modèles Fivetran pour vérifier qu'ils fonctionnent.

```
dbt run -select marketo_source
```

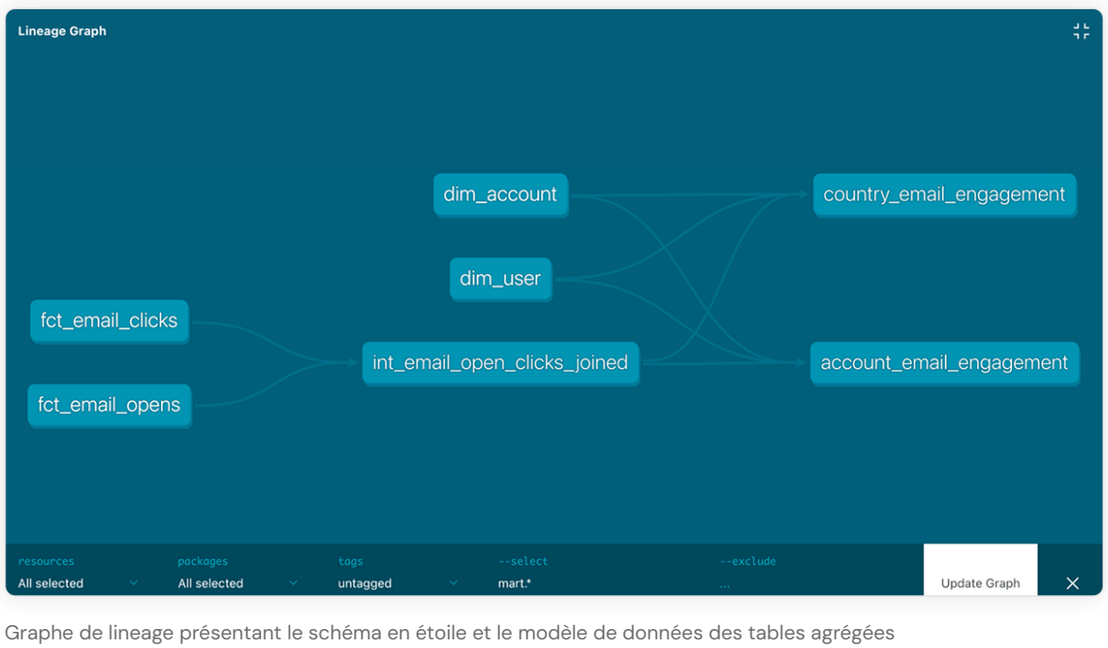

```
dbt run -select marketo_source
```


Si l'un des modèles échoue à cause de l'absence des tables que vous avez choisi de ne pas synchroniser dans Fivetran, désactivez ces modèles dans votre schéma source en modifiant le fichier **`dbt_project.yml`**.

Par exemple, si les tables email bounced et email template ne sont pas présentes dans le schéma source Marketo, vous pouvez désactiver les modèles de ces tables en ajoutant ce qui suit sous la configuration des modèles :

```
models:
 marketo_source:
   +schema: nom_schéma_intermédiaire
   tmp:
     stg_marketo__activity_email_bounced_tmp:
       +enabled: false
     stg_marketo__email_template_history_tmp:
       +enabled: false
   stg_marketo__activity_email_bounced:
     +enabled: false
   stg_marketo__email_template_history:
     +enabled: false
```

Développement des modèles d'analytique marketing



Graphe de lineage présentant le schéma en étoile et le modèle de données des tables agrégées

Une fois que les paquets Fivetran se sont chargés de créer et tester les modèles intermédiaires, vous pouvez développer les modèles de vos cas d'usage d'analytique marketing : un schéma en étoile et des tables agrégées matérialisées.

Par exemple, pour le premier tableau de bord d'analytique marketing, vous pouvez évaluer l'engagement de certaines sociétés et régions commerciales en fonction du nombre d'e-mails qu'elles ont ouverts et activés.

Pour ce faire, joignez les tables Salesforce et Marketo en fonction de l'e-mail de l'utilisateur Salesforce, de l'`account_id` Salesforce et du `lead_id` Marketo.

Les modèles seront structurés sous le répertoire mart comme suit :

```
marketing_analytics_demo
|-- dbt_project.yml
|-- packages.yml
|-- models
      |-- mart
            |-- core
            |-- intermediate
            |-- marketing_analytics
```

Le code des modèles est disponible sur Github dans le répertoire /models/mart. Vous trouverez ci-dessous une description et un exemple de ce que contient chaque répertoire.

Modèles fondamentaux

Les modèles fondamentaux sont les tables de faits et de dimension qui seront utilisés par tous les modèles en aval.

Code SQL dbt du modèle `dim_user` :

```sql
with salesforce_users as (
   select
       account_id,
       email
   from {{ ref('stg_salesforce__user') }}
   where email is not null and account_id is not null
),
marketo_users as (
   select
       lead_id,
       email
   from {{ ref('stg_marketo__lead') }}
),
joined as (
   select
     lead_id,
     account_id
   from salesforce_users
     left join marketo_users
     on salesforce_users.email = marketo_users.email
)

select * from joined
```

Vous pouvez également ajouter de la documentation et des tests pour les modèles à l'aide d'un fichier yaml placé dans le dossier.

Le fichier `core.yml` qui a été ajouté contient deux tests simples.

```yaml
version: 2

models:
 - name: dim_account
   description: "The Account Dimension Table"
   columns:
     - name: account_id
       description: "Primary key"
       tests:
         - not_null
 - name: dim_user
   description: "The User Dimension Table"
   columns:
     - name: lead_id
       description: "Primary key"
       tests:
         - not_null
```

Modèles intermédiaires

Certains modèles finaux en aval peuvent s'appuyer sur les mêmes métriques calculées. Pour éviter de répéter la partie SQL, vous pouvez créer des modèles intermédiaires réutilisables.

Le code SQL dbt du modèle `int_email_open_clicks_joined` :

```sql
with opens as (
        select *
        from {{ ref('fct_email_opens') }}
), clicks as (
        select *
        from {{ ref('fct_email_clicks') }}
), opens_clicks_joined as (

    select
      o.lead_id as lead_id,
      o.campaign_id as campaign_id,
      o.email_send_id as email_send_id,
      o.activity_timestamp as open_ts,
      c.activity_timestamp as click_ts
    from opens as o
      left join clicks as c
      on o.email_send_id = c.email_send_id
      and o.lead_id = c.lead_id

)

select * from opens_clicks_joined
```

Modèles d'analytique marketing

Ces modèles finaux d'analytique marketing alimenteront les tableaux de bord et les rapports utilisés par les équipes marketing et commerciales.

Le code SQL dbt du modèle `country_email_engagement` :

```
with accounts as (
        select
        account_id,
        billing_country
        from {{ ref('dim_account') }}
), users as (
        select
        lead_id,
        account_id
        from {{ ref('dim_user') }}
), opens_clicks_joined as (

    select * from {{ ref('int_email_open_clicks_joined') }}

), joined as (

        select *
        from users as u
        left join accounts as a
        on u.account_id = a.account_id
        left join opens_clicks_joined as oc
        on u.lead_id = oc.lead_id

)

select
        billing_country as country,
        count(open_ts) as opens,
        count(click_ts) as clicks,
        count(click_ts) / count(open_ts) as click_ratio
from joined
group by country
```

Exécuter et tester des modèles dbt

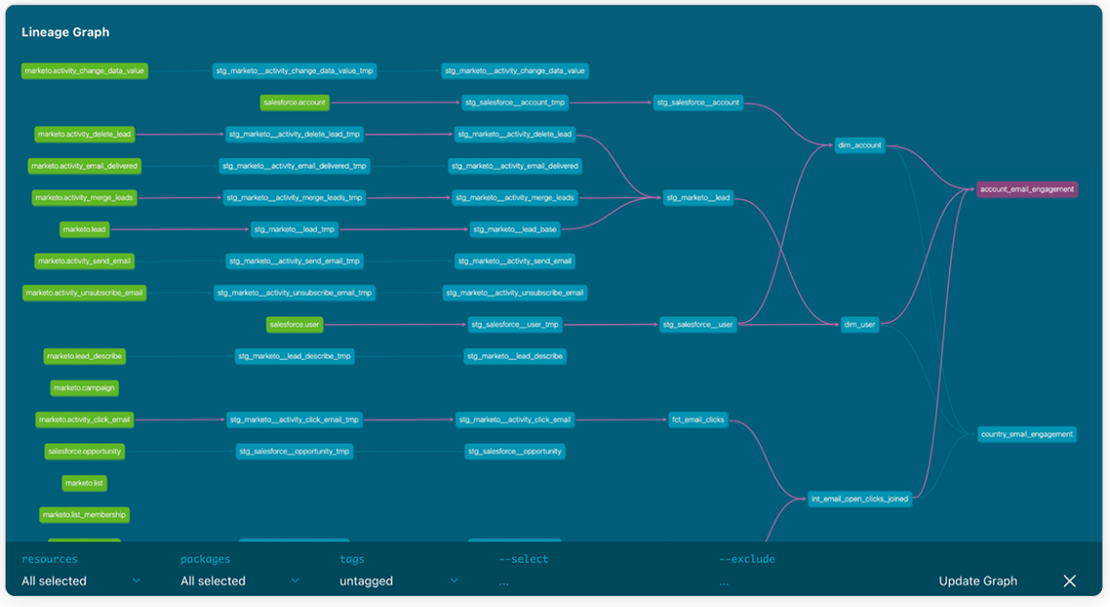
Une fois votre modèle prêt, vous pouvez exécuter tous les modèles en utilisant

```
dbt run
```

Puis exécuter des tests en utilisant

```
dbt test
```

Afficher les documents dbt et le graphe de lineage



Graphe de lineage dbt du modèle d'analytique marketing

Une fois que vous avez exécuté vos modèles, vous pouvez générer les documents et le graphe de lignage en utilisant

```
$ dbt docs generate
```

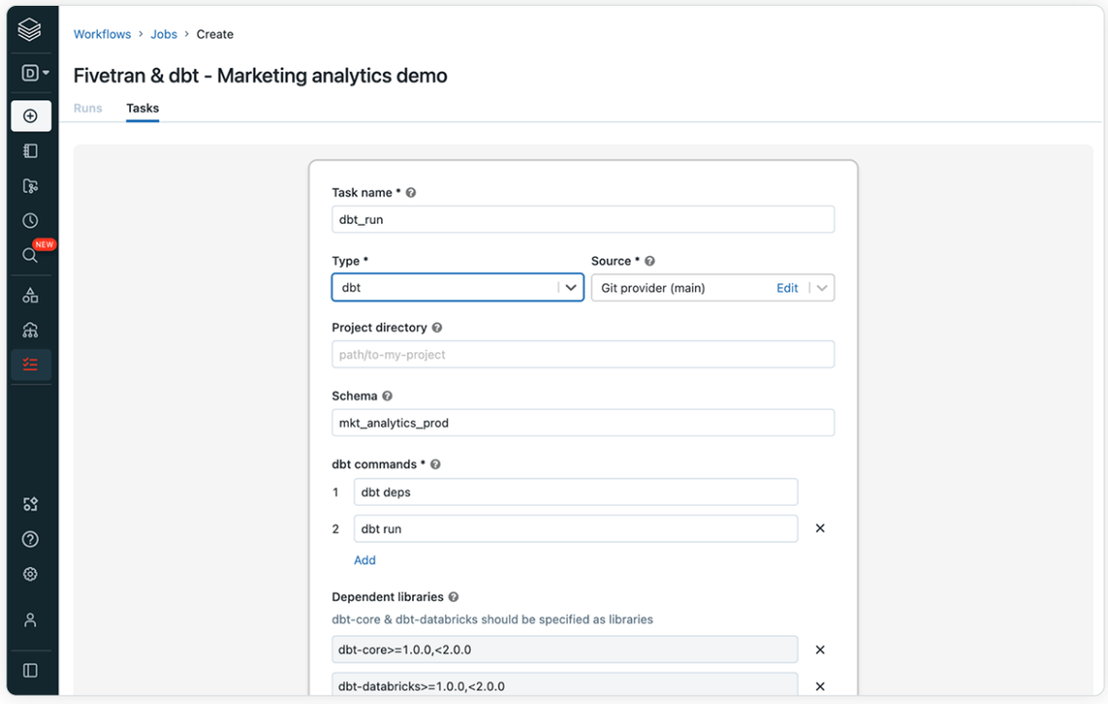
Puis, pour les visualiser localement, exécutez

```
$ dbt docs serve
```

Déploiement des modèles dbt en production

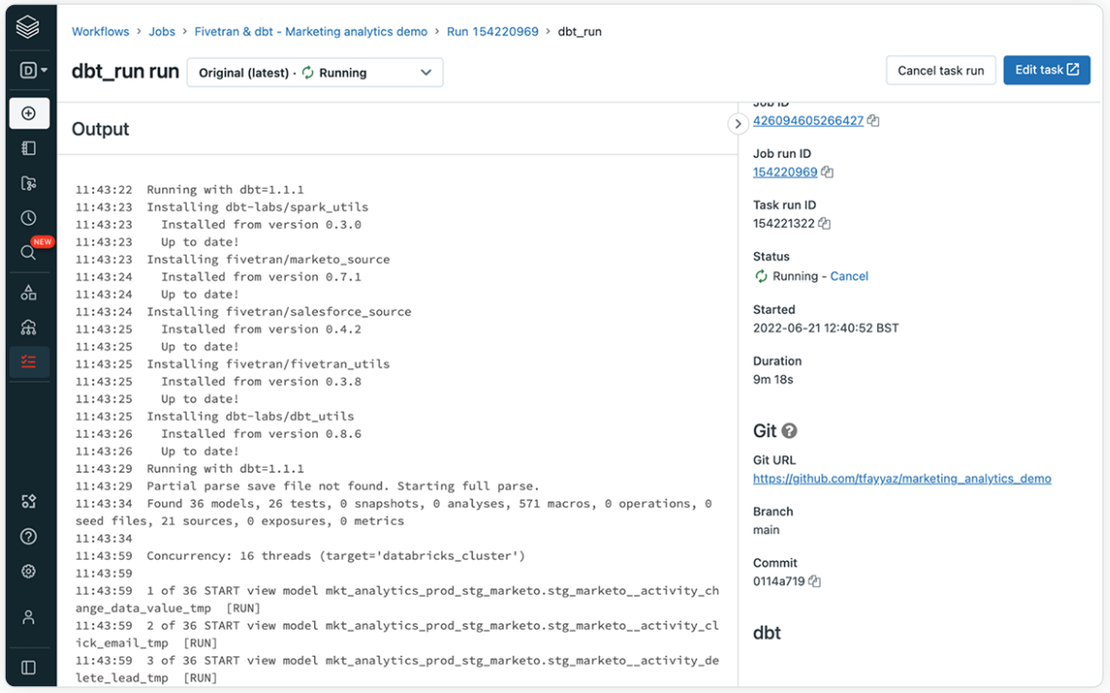
Une fois que vous avez développé et testé votre modèle dbt localement, le déploiement en production peut se faire de plusieurs manières, dont le nouveau type de tâche dbt dans Databricks Workflows (accès anticipé privé).

La gestion et le contrôle de version de votre projet dbt doivent se faire dans un dépôt Git. Vous pouvez créer une tâche dbt dans votre tâche Databricks Workflows pointant vers le dépôt Git.



Utiliser un type de tâche dbt dans Databricks Workflows pour orchestrer dbt

Comme vous utilisez des paquets dans votre projet dbt, la première commande doit être dbt deps, suivie de dbt run pour la première tâche et de dbt test pour la suivante.



Affichage des journaux dbt pour chaque exécution

Vous pouvez consulter les journaux de chaque commande dbt pour déboguer et corriger les problèmes éventuels.

Alimenter votre analytique marketing avec Fivetran et dbt

Comme nous le voyons, utiliser Fivetran et dbt sur la Data Intelligence Plateform de Databricks permet de créer et d'administrer simplement une puissante solution d'analytique marketing suffisamment flexible pour s'adapter à tous les besoins en modélisation de données.

Pour commencer à développer votre propre solution, consultez la documentation concernant l'intégration de Fivetran et dbt avec Databricks et réutilisez l'exemple de projet marketing_analytics_demo pour gagner du temps.

Le type de tâche dbt dans Databricks Workflows est en accès anticipé privé. Pour essayer le type de tâche dbt, consultez votre responsable de compte Databricks.

SECTION 4.2

Automatisation du traitement des sinistres sur Databricks

Des « smart claims » pour gagner en efficacité en automatisant tous les aspects du traitement des sinistres : importation, analyse et prise de décision.

par Anindita Mahapatra et Marzi Rasooli

Selon de récents rapports du cabinet de conseil international EY, l'avenir de l'assurance sera **data-driven** et **basé sur l'analytique**. L'avènement du cloud a facilité l'accès à une infrastructure technologique sophistiquée, mais la plupart des entreprises ont besoin d'aide pour mettre en œuvre et exploiter ces possibilités. Il est temps de chercher à opérationnaliser les services pour concrétiser le potentiel de valeur.

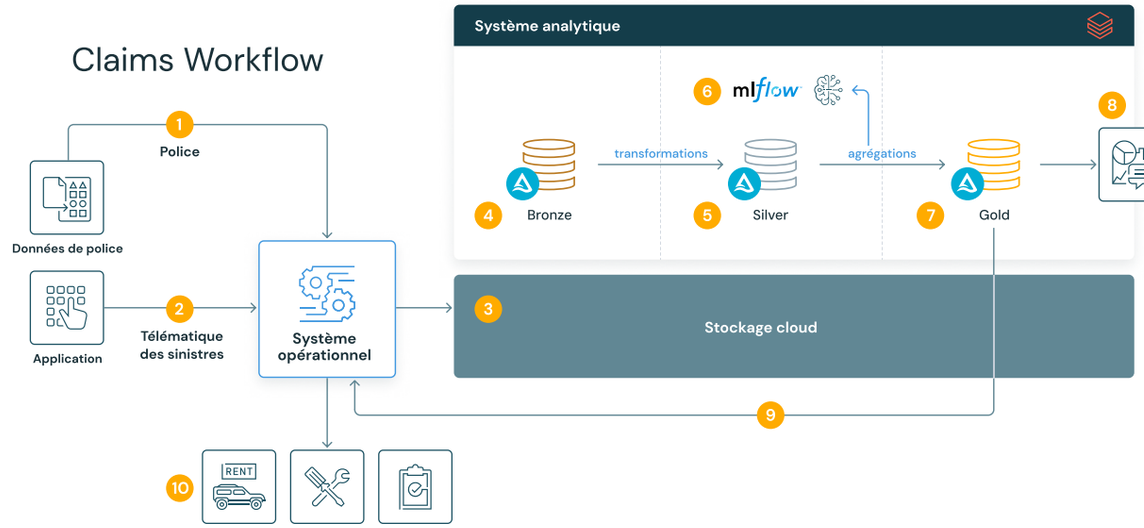
Dans la conjoncture économique actuelle, les compagnies d'assurance sont confrontées à des défis de plus en plus nombreux. Elles n'ont pas d'autre choix que d'exploiter les données à leur avantage et d'**innover** à un rythme soutenu. Pour les acteurs de l'assurance IARD, cela implique également de miser sur la **personnalisation** et la fidélisation de la clientèle. La fidélité aux marques n'a jamais été aussi faible : les clients recherchent sans cesse des tarifs plus compétitifs et des expériences de meilleure qualité, ce qui augmente le risque d'attrition. D'autre part, les marges sont érodées par l'augmentation des déclarations frauduleuses. Les assureurs doivent trouver de nouveaux moyens de réduire les coûts et mieux gérer les risques.

Parce qu'elles permettent de réduire les coûts en accélérant les processus et en réduisant le recours au capital humain, l'automatisation et l'optimisation du traitement des sinistres sont des domaines à privilégier. Pensez également que les données et l'analytique avancée peuvent produire des insights capables de réduire de façon importante l'exposition globale au risque.

Il est facile de comprendre ce qui a motivé le développement de l'accélérateur de solutions « Smart Claims » : améliorer le processus de traitement des sinistres pour accélérer leur résolution, réduire les coûts de traitement et obtenir rapidement des informations sur les fraudes potentielles, sur le lakehouse. En effet, le paradigme lakehouse simplifie le paysage architectural actuel et sert de fondement à de futures expansions à l'échelle de l'entreprise. Vous trouverez ici les assets connexes.

Architecture de référence et workflow

Un workflow de sinistre classique implique un certain degré d'orchestration entre les systèmes opérationnels tels que Guidewire, et les systèmes analytiques comme Databricks. Le schéma ci-dessous donne un exemple de ce workflow pour un assureur automobile.



Architecture de référence et workflow de Smart Claims

Pour automatiser et optimiser le traitement des sinistres, il faut une maîtrise parfaite de l'interaction du client avec les systèmes opérationnels et les différentes sources d'information disponibles.

Dans cet exemple, nous partons du principe que les clients utilisent essentiellement une application mobile pour déclarer les sinistres et suivre leurs dossiers en cours. Ce point de contact fournit des informations vitales sur le comportement des clients. Autre source essentielle : les dispositifs IoT installés dans les véhicules des clients. Les données télématiques peuvent être transmises aux systèmes opérationnels et analytiques pour mieux comprendre le comportement et les habitudes de conduite des assurés. Des données externes (météo, état des routes…) peuvent compléter les catégories traditionnelles, comme les caractéristiques du véhicule (modèle, année), celles du conducteur et la couverture (plafonds, franchises).

Les sources de données supplémentaires prennent de l'importance, surtout en l'absence de sources traditionnelles comme les agences de crédit. Les notes de crédit des agences forment généralement la base de la modélisation du risque ; elles évaluent l'exposition des conducteurs, ce qui influe sur leur prime.
À l'opposé, les données des applications mobiles et de l'IoT offrent une vue plus individualisée du comportement du client, et peuvent donc donner une indication plus précise du risque associé. Cette approche alternative de la modélisation du risque et de la tarification, **basée sur le comportement**, est indispensable pour une expérience client hyper-personnalisée.

L'architecture lakehouse, basée sur la Data Intelligence Plateform de Databricks, est la seule à combiner les fonctionnalités et les services nécessaires pour une approche moderne et durable du traitement des sinistres. Streaming, machine learning, création de rapports… Databricks est la meilleure plateforme pour créer une solution de bout en bout pour l'avenir du secteur de l'assurance.

Le flux se compose des étapes suivantes :

- Les données de police sont importées.
- Les données télématiques en provenance des capteurs IoT sont ingérées en continu. Un assuré soumet un sinistre via une application mobile.
- Toutes les données opérationnelles sont importées dans un stockage cloud.
- Elles sont chargées de façon incrémentale en tant que « données brutes » dans des tables Delta Bronze.
- Diverses transformations sont appliquées aux données pour les affiner.
- Les données sont évaluées à l'aide du modèle entraîné.
- Les prédictions sont chargées dans une table Gold.
- Le tableau de bord Claims est actualisé.
- Les insights produits sont renvoyés au système opérationnel. On obtient une boucle de rétroaction qui extrait des données de Guidewire et lui renvoie une « recommandation d'action » en temps réel afin de hiérarchiser les sinistres.
- Les workflows de prise de décision s'appuient sur ces insights pour acheminer correctement le dossier (approuver les dépenses de réparation, rembourser la location de véhicule, alerter les autorités, etc.).

Le rôle du paradigme lakehouse dans Smart Claims

L'architecture lakehouse permet à tous les profils d'utilisateurs (ingénieurs data, data scientists, ingénieurs analytiques et analystes BI) de collaborer sur une même plateforme. Parce qu'elle prend en charge toutes les tâches big data et tous les paradigmes (batch et streaming, DataOps, ML, MLOps et BI), elle simplifie considérablement l'architecture globale, améliore sa stabilité et réduit les coûts.

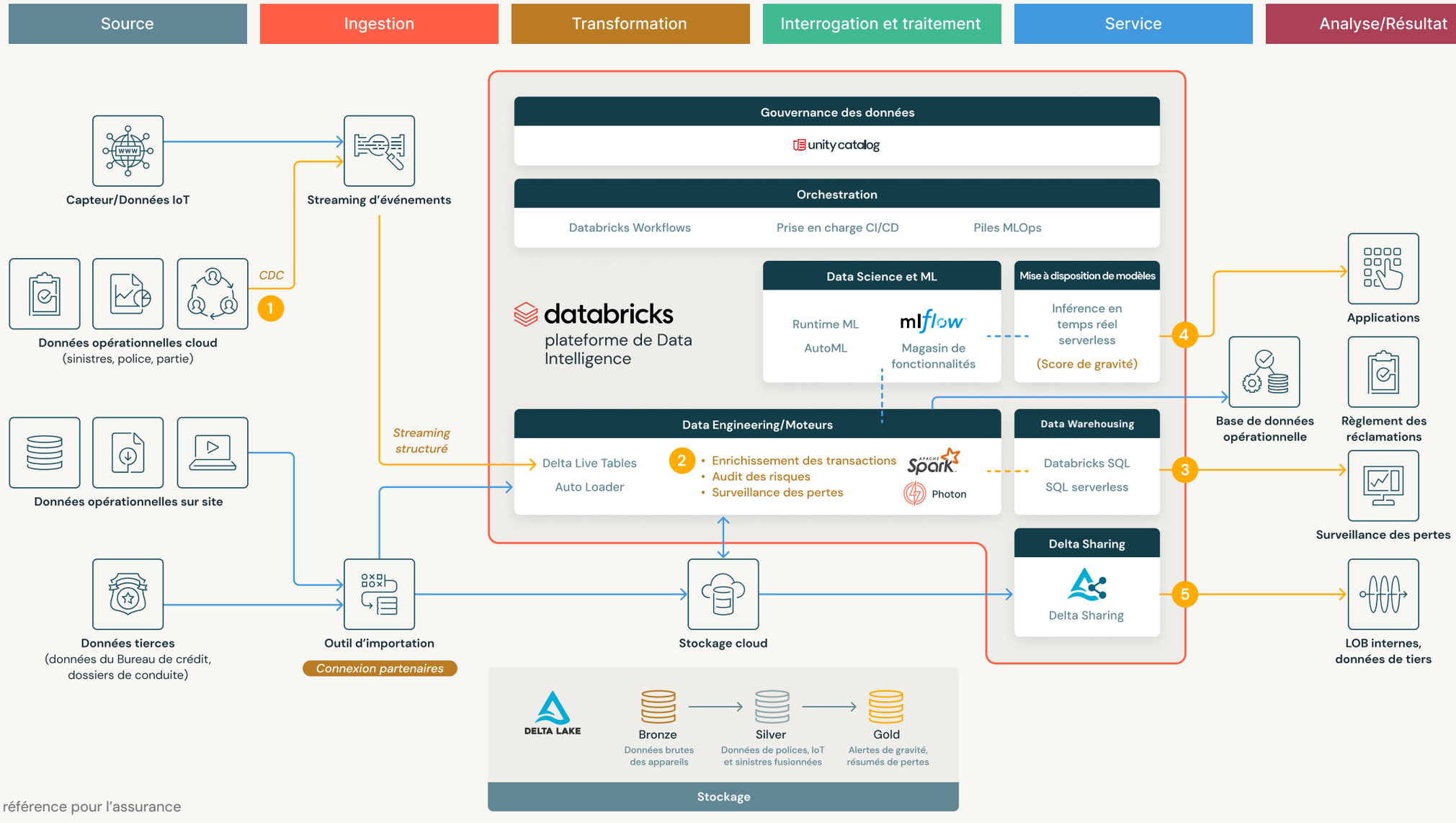
Les pipelines Delta Live Tables (DLT) de Databricks offrent un cadre déclaratif simple pour développer et implémenter des charges rapidement. Ils facilitent également la gestion de la qualité des données avec des contraintes granulaires qui garantissent l'intégrité des résultats.

MLflow permet de créer et gérer des charges ML et IA de façon reproductible et contrôlable. MLflow simplifie l'intégralité du cycle de vie du modèle, de l'expérimentation à l'archivage, en passant par le déploiement et la mise à disposition. Outre le texte, le ML peut s'appliquer à tous les types de données, structurées ou non (images, audio, vidéo, etc.). Ici, nous allons utiliser de la reconnaissance d'image pour évaluer les dommages d'un véhicule.

Enfin, le moteur Databricks SQL offre un moyen rapide et efficace d'interroger les données organisées et agrégées. Quelques minutes suffisent pour empaqueter ces insights et les diffuser au moyen d'un tableau de bord interactif.

Unity Catalog est une solution de gouvernance centralisée et multicloud pour tous les assets de données et d'IA – fichiers, tables, modèles de ML et tableaux de bord. Il intègre la recherche, la découverte et l'automatisation du lineage.

Le schéma ci-dessous présente une architecture lakehouse de référence dans le contexte d'un cas d'usage classique de l'assurance.



Architecture de référence pour l'assurance

Ingestion des données avec les workflows DLT et Multitask

Pour automatiser le traitement des sinistres, il faut d'abord optimiser le workflow d'importation et d'ingénierie des données. Le schéma ci-dessous résume les sources de données classiques – structurées, semi-structurées et non structurées. Certaines sources évoluent lentement, d'autres se mettent à jour plus rapidement. De plus, certaines sources sont additives : il faut ajouter régulièrement de nouvelles données. D'autres se mettent à jour de façon incrémentale et doivent être traitées comme des dimensions à évolution lente (SCD).

Volume	Vélocité	Variété
Élevé • Données télématiques	**Quasi-temps réel** • Données télématiques	**Structurées** • Police, données d'assuré, télématique
Intermédiaire • Données de sinistre	**Streaming** • Données de sinistre, données d'accident	**Semi-structurées** • Données de sinistre
Faible • Police, données d'assuré • Données d'accident	**Batch** • Police, données d'assuré	**Non structurées** • Données d'accident • Image, vidéo et texte

Datasets d'exemple utilisés dans le traitement des sinistres

DLT simplifie et opérationnalise le pipeline de traitement des données. Le cadre prend en charge Auto Loader pour faciliter l'importation des sources de streaming. Il applique un redimensionnement efficace pour gérer les changements brusques du volume de données et contribue à la résilience en relançant les tâches inabouties.

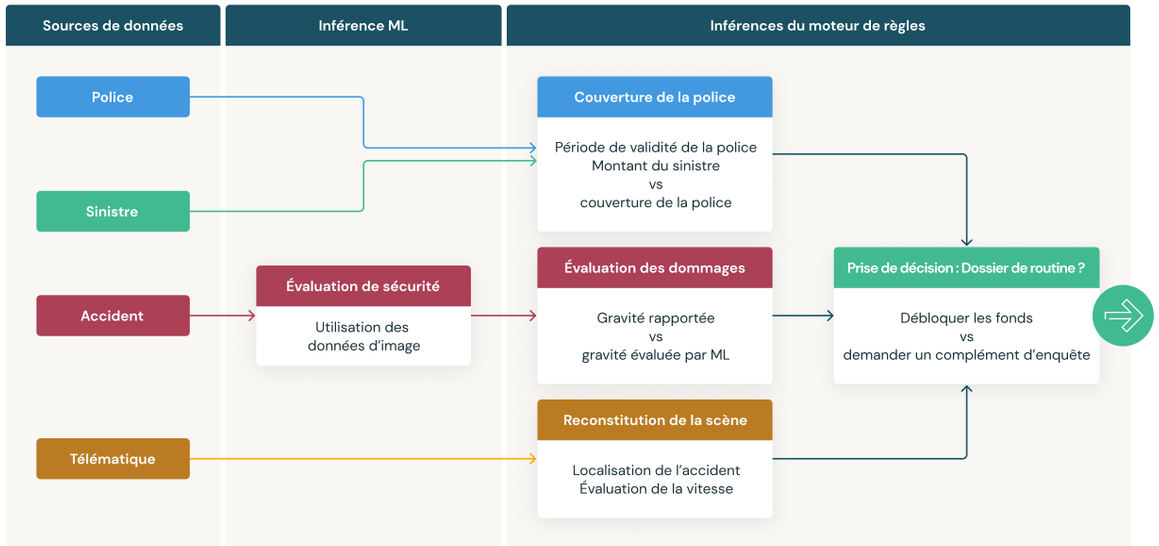
Les Databricks Workflows peuvent accueillir différentes tâches et charges (notebooks, DLT, ML, SQL). Ils sont compatibles avec le partage du calcul entre les tâches selon le principe « réparer et exécuter ». Ils garantissent ainsi la robustesse et l'évolutivité des charges tout en réduisant les coûts. Ces workflows peuvent enfin être automatisés par planification ou programmation à l'aide d'API REST.

Génération d'insights grâce au ML et au moteur de règles dynamiques

Le ML est indispensable pour mettre au jour des tendances invisibles, de nouveaux insights et des activités suspectes. Mais nous pouvons obtenir des résultats plus puissants encore en combinant le ML à des approches traditionnelles basées sur les règles.

Dans le processus de traitement des sinistres, le ML peut avoir plusieurs usages. Nous pouvons combiner la reconnaissance d'image et le ML pour évaluer les photos qui accompagnent les déclarations de sinistres automobiles. Les modèles peuvent être entraînés à interpréter la validité et la gravité des dommages. MLflow joue un rôle décisif ici, en simplifiant l'entraînement du modèle et sa mise à disposition grâce à ses capacités MLOps de bout en bout. MLflow met les modèles à disposition de façon serverless grâce à des API REST. Les modèles entraînés peuvent être opérationnalisés et mis en production en un clic.

D'autre part, les moteurs de règles permettent de définir de façon souple des vérifications de caractéristiques opérationnelles connues et de statistiques, qui peuvent être automatisées et appliquées sans intervention humaine. Les données qui ne sont pas conformes aux attentes génèrent des alertes et sont soumises à des vérifications humaines. L'intégration de ce type d'approche à des workflows ML apporte une supervision supplémentaire et réduit considérablement le temps nécessaire pour examiner les dossiers signalés.



Inférences ML et par moteur de règles

Visualisation des insights à l'aide de tableaux de bord

Dans cet exemple, nous avons créé deux tableaux de bord pour mettre en évidence des insights commerciaux stratégiques :

- Un tableau de bord **Résumé des pertes** qui offre une vue de haut niveau des opérations commerciales globales,
- Un tableau de bord **Enquête sur les sinistres**, qui présente une vue granulaire des sinistres pour comprendre les spécificités d'un dossier particulier.

L'analyse des tendances récentes facilite l'examen de dossiers similaires :

- Le ratio de perte est calculé en divisant la somme des sinistres dédommagés et des dépenses d'ajustement par le montant total des primes perçues. Par exemple, le ratio de perte moyen classique (toutes couvertures combinées, blessures et dommages) d'une assurance auto personnelle doit se situer autour de 65 %
- La visualisation résumée recense le nombre d'accidents par degré de gravité.
- Courbes de tendance concernant différentes caractéristiques et dimensions
- Répartition géographique des polices

Le tableau de bord Enquête sur les sinistres accélère les investigations en fournissant toutes les informations relatives à un sinistre. L'enquêteur peut explorer un sinistre en particulier et accéder, par exemple, aux images du véhicule endommagé, aux détails du sinistre, de la police et du conducteur. Les données télématiques retracent le parcours du véhicule et les données rapportées sont comparées aux insights évalués.

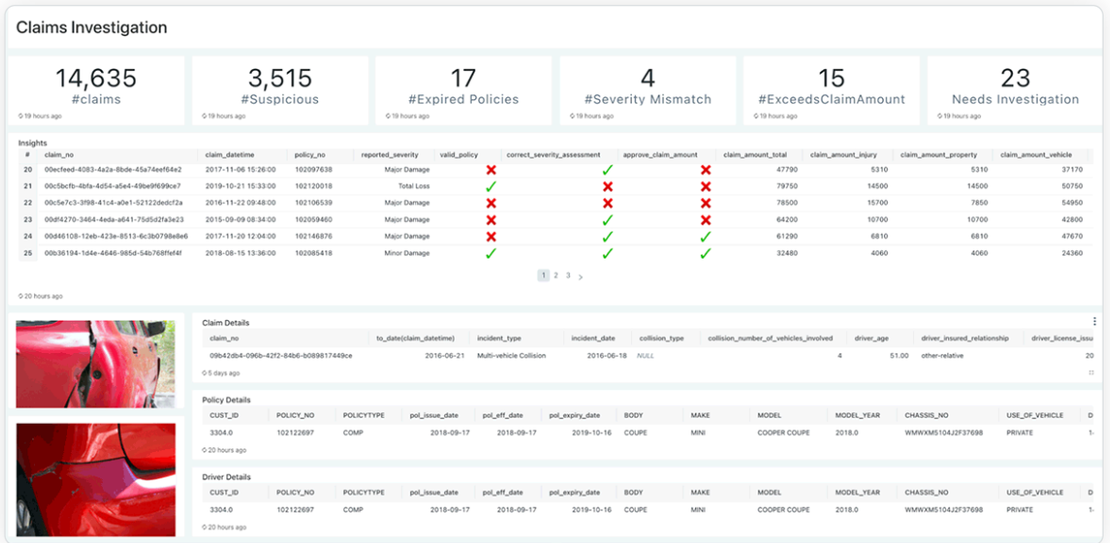


Tableau de bord d'enquête sur les sinistres

Il présente les sinistres récents évalués automatiquement dans le pipeline grâce à l'inférence ML et au moteur de règles.

- Une coche verte indique que l'évaluation automatique correspond à la description du sinistre
- Une croix rouge indique un écart qui justifie une investigation manuelle

Résumé

L'innovation et la personnalisation sont des axes majeurs pour les compagnies d'assurance qui veulent se démarquer de la concurrence. La Data Intelligence Plateform de Databricks permet aux assureurs d'innover à grande vitesse en s'appuyant sur une architecture ouverte, sécurisée et extensible qui s'intègre en souplesse aux outils et services tiers. L'accélérateur de solutions démontre comment ce paradigme peut être appliqué au traitement des sinistres. De plus, l'écosystème Databricks offre un large éventail d'outils qui permettent aux équipes de données comme aux acteurs commerciaux de collaborer, de produire des insights et de les partager, dans le but de soutenir les décisions stratégiques et d'exercer un impact tangible sur les bénéfices de l'entreprise.

Les assets techniques (configuration des pipelines, modèles et données d'exemple) sont disponibles ici ou directement sur Git.

SECTION 4.3

Modèles de conception pour le traitement batch dans les services financiers

Fondements de l'automatisation des workflows

par Eon Retief

Introduction

Volatilité des marchés, incertitudes politiques, évolution du paysage juridique et réglementaire… Dans le monde entier, les institutions de services financiers (ISF) doivent relever des défis sans précédent. Les entreprises n'ont d'autre choix que d'accélérer leurs initiatives de transformation numérique et d'automatiser les processus critiques pour réduire les coûts et les délais de réponse. Mais quand les données sont dispersées sur de nombreux systèmes, accéder aux informations nécessaires à l'exécution de ces projets n'a rien d'évident.

Et il peut sembler impossible de mettre sur pied un écosystème de services capables de soutenir la myriade de cas d'usage data-driven d'une entreprise entièrement digitalisée. Ce chapitre s'intéresse à un aspect crucial de la pile de données moderne : le traitement batch. S'il peut sembler obsolète, nous verrons pourquoi il reste un composant essentiel et viable de l'architecture de données. Et nous verrons comment Databricks peut aider les ISF à relever les défis majeurs qui se présentent lorsqu'une infrastructure de prise en charge de ces workflows intermittents est mise sur pied.

Importance de l'ingestion batch

Au cours des vingt dernières années, l'émergence de la société de l'instantané a contraint les entreprises à revoir leur modèle d'exploitation et d'engagement. La transformation numérique n'est plus une option : c'est une question de survie. Les besoins et les attentes des clients évoluent plus vite que jamais. Face au désir de gratification immédiate, les entreprises cherchent à s'équiper d'outils qui appuient le traitement et la prise de décision en temps réel. Dans ce contexte dynamique, nous pouvons nous interroger sur la pertinence du traitement batch.

Si les systèmes en temps réel et les services de streaming peuvent aider les ISF à conserver de l'agilité face à un marché volatil, ils ne répondent pas aux impératifs des fonctions back-office. La plupart des décisions commerciales ne sont pas prises dans l'instant, mais au terme d'un raisonnement posé et stratégique. Cela implique une revue systématique des données agrégées collectées au fil du temps. Dans ce contexte, le traitement batch reste la méthode la plus efficace et la plus rentable pour traiter de gros volumes de données agrégés. De plus, le traitement batch peut se faire hors ligne, ce qui réduit les coûts d'exploitation et offre davantage de contrôle sur le processus.

Le monde de la finance évolue, mais les acteurs historiques, tout comme les startups, continuent de recourir fortement au traitement batch pour alimenter leurs fonctions métier essentielles. Qu'il s'agisse de reporting, de gestion des risques ou de détection des anomalies, les ISF ont besoin du traitement batch pour limiter l'erreur humaine, accélérer la livraison et réduire les coûts d'exploitation.

Démarrer

Pour donner une vision d'ensemble, rappelons que la plupart des ISF ont une multitude de sources de données dispersées dans des systèmes sur site, dans le cloud et même dans des applications tierces. Pour créer un cadre d'importation batch prenant en charge toutes ces connexions, il faut une ingénierie complexe qui peut rapidement devenir un fardeau pour les équipes de maintenance. Et cela ne tient pas compte d'aspects comme la capture des modifications de données (CDC), la planification et l'évolution des schémas. Dans cette section, nous verrons comment le lakehouse pour les services financiers et son écosystème de partenaires peuvent relever ces défis clés et simplifier considérablement l'architecture globale.

L'architecture lakehouse fournit une plateforme unifiée qui prend en charge toutes les charges de données analytiques et scientifiques. La figure 1 montre l'architecture de référence d'une conception découplée qui facilite l'intégration d'autres plateformes prenant en charge l'écosystème de données moderne. Le lakehouse facilite la création de couches d'importation et de service fonctionnant quels que soient la source, le volume, la vitesse et la destination des données.

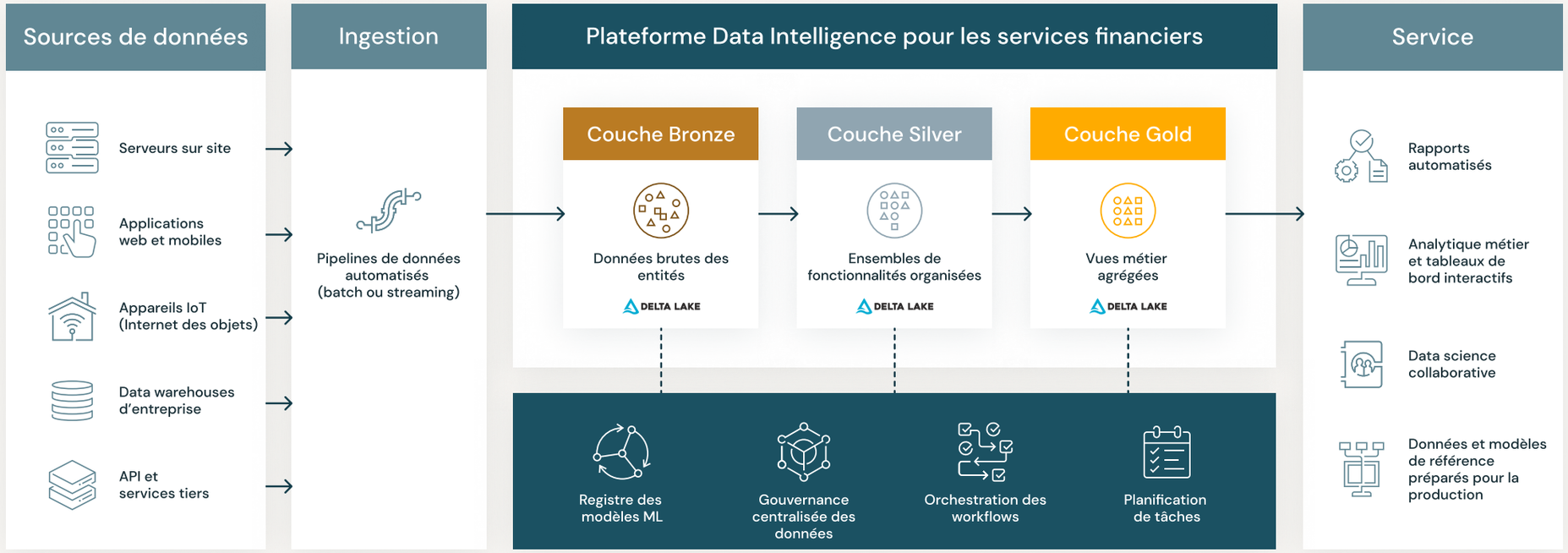


Figure 1 : Architecture de référence du lakehouse pour les services financiers.

Pour démontrer la puissance et l'efficacité du LFS, tournons-nous vers le monde de l'assurance. Nous examinons les exigences de signalement de base d'un workflow de sinistre classique. Dans ce scénario, l'organisation s'intéresse aux indicateurs clés générés par les processus de traitement des sinistres. Par exemple :

- Nombre de polices actives
- Nombre de sinistres
- Valeur des sinistres
- Exposition totale
- Ratio de perte

Il peut aussi être intéressant d'avoir un aperçu des sinistres suspects et une répartition des incidents par type et degré de gravité. Toutes ces métriques sont faciles à calculer à partir de deux sources : 1) le registre des polices et 2) les déclarations de sinistre. Les enregistrements de polices et de sinistres résident généralement dans des data warehouses d'entreprise (EDW) et des bases de données opérationnelles. Le principal défi consiste à se connecter à ces sources et à importer les données dans le lakehouse afin d'exploiter la puissance de Databricks pour réaliser les calculs souhaités.

Heureusement, la souplesse du LFS permet de s'appuyer sur un large éventail de technologies et d'outils SaaS de premier plan pour gérer des tâches spécifiques. Dans notre cas d'usage, il est possible d'utiliser Fivetran pour le plan d'ingestion par lots. Fivetran fournit une plateforme simple et sécurisée pour se connecter à de nombreuses sources de données et fournir des données directement à la Data Intelligence Platform de Databricks. Fivetran prend également en charge la CDC, l'évolution des schémas et la planification des charges de travail. À la figure 2, nous présentons l'architecture technique d'une solution pratique pour ce cas d'usage.

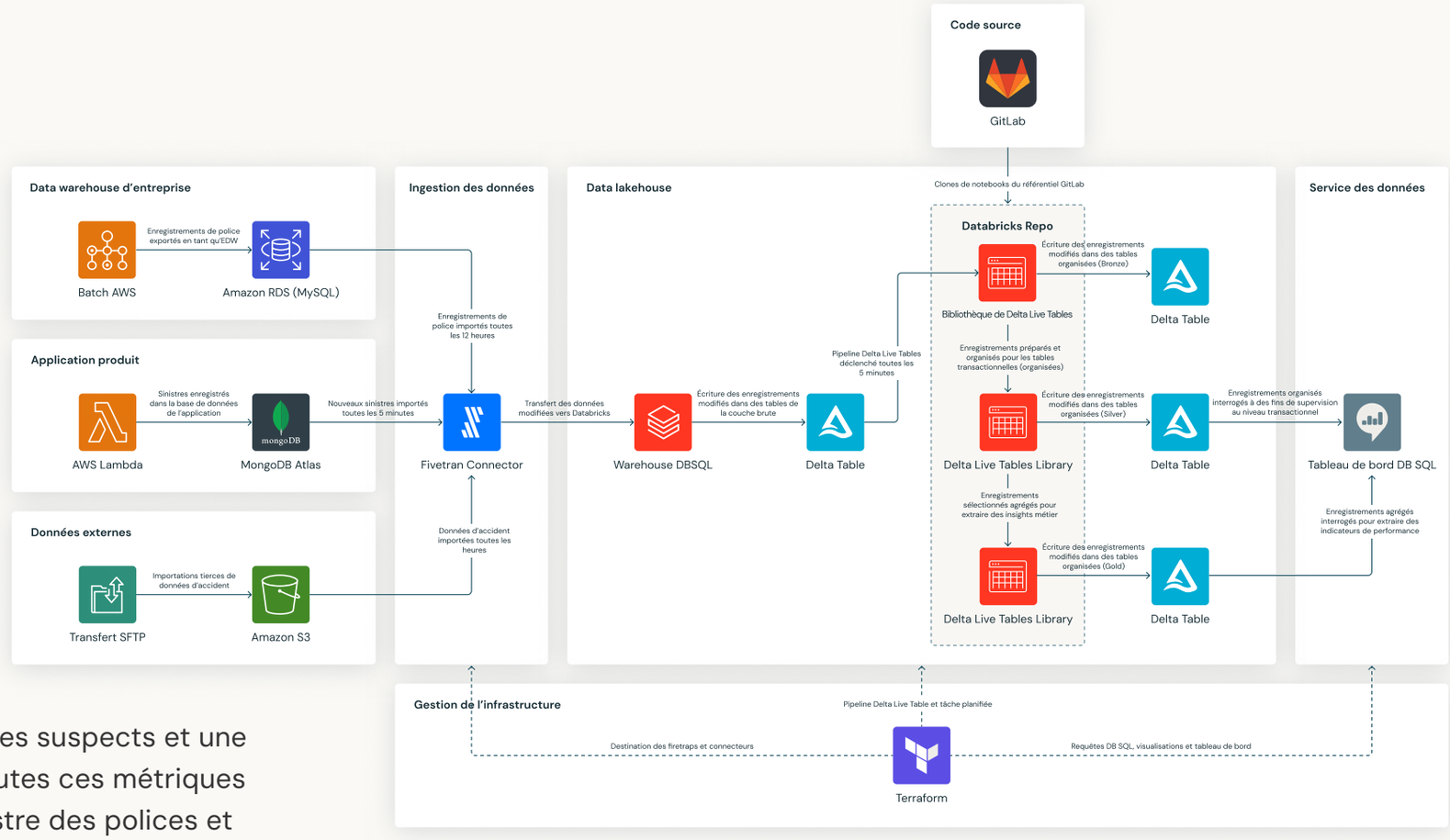


Figure 2 : Architecture technique d'un workflow simple de traitement de sinistre.

Une fois les données importées dans le LFS, nous pouvons utiliser Delta Live Tables (DLT) pour l'ensemble du workflow d'ingénierie. DLT offre un cadre déclaratif simple et évolutif pour automatiser des workflows complexes et contrôler la qualité des données. Les résultats de notre workflow DLT – nos assets organisés et agrégés – peuvent être interrogés à l'aide de Databricks SQL (DB SQL). DB SQL apporte la couche de data warehousing au LFS pour prendre en charge les tâches analytiques stratégiques. Les résultats des requêtes DB SQL peuvent être présentés dans des tableaux de bord intuitifs à destination des utilisateurs métier.

Étape 1 : Création de la couche d'importation

La configuration d'une couche d'importation avec Fivetran se fait en deux étapes. Nous configurons d'abord une « destination » où les données seront livrées, puis nous établissons une ou plusieurs connexions avec les systèmes sources. Pour la première étape, l'interface Partner Connect vous aide à connecter Fivetran à un warehouse Databricks. Fivetran l'utilisera pour convertir les données sources brutes en tables Delta et stockera les résultats dans la Data Intelligence Platform de Databricks. Les figures 3 et 4 illustrent les étapes de configuration dans les interfaces Partner Connect et Fivetran.

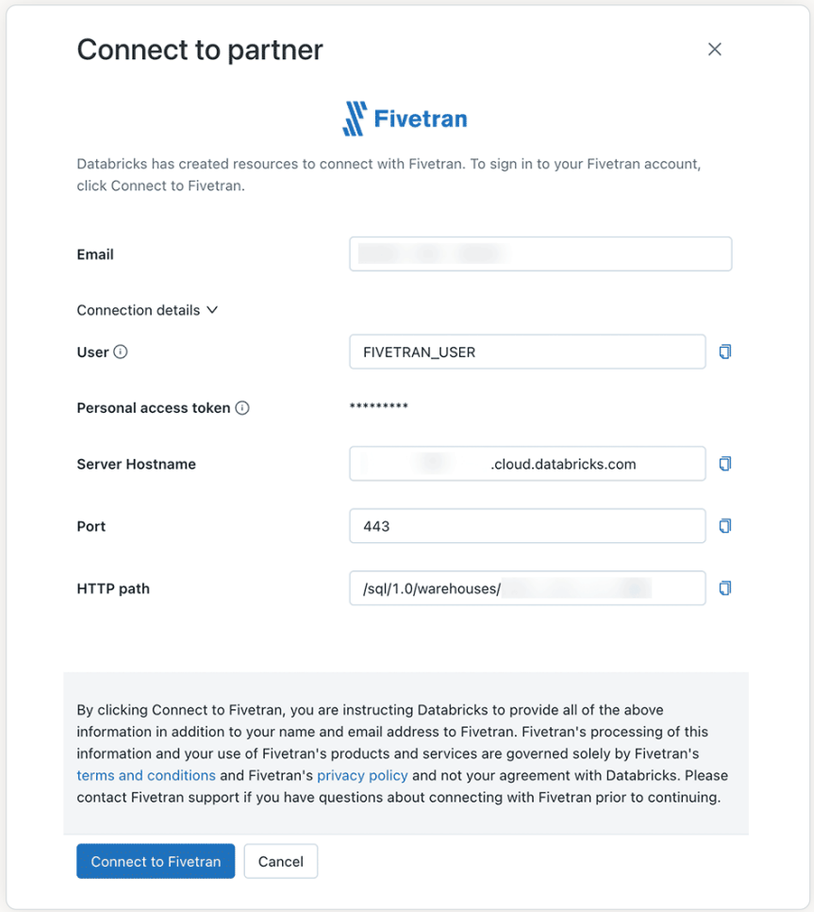


Figure 3 : Interface Databricks Partner Connect pour la création d'une nouvelle connexion.

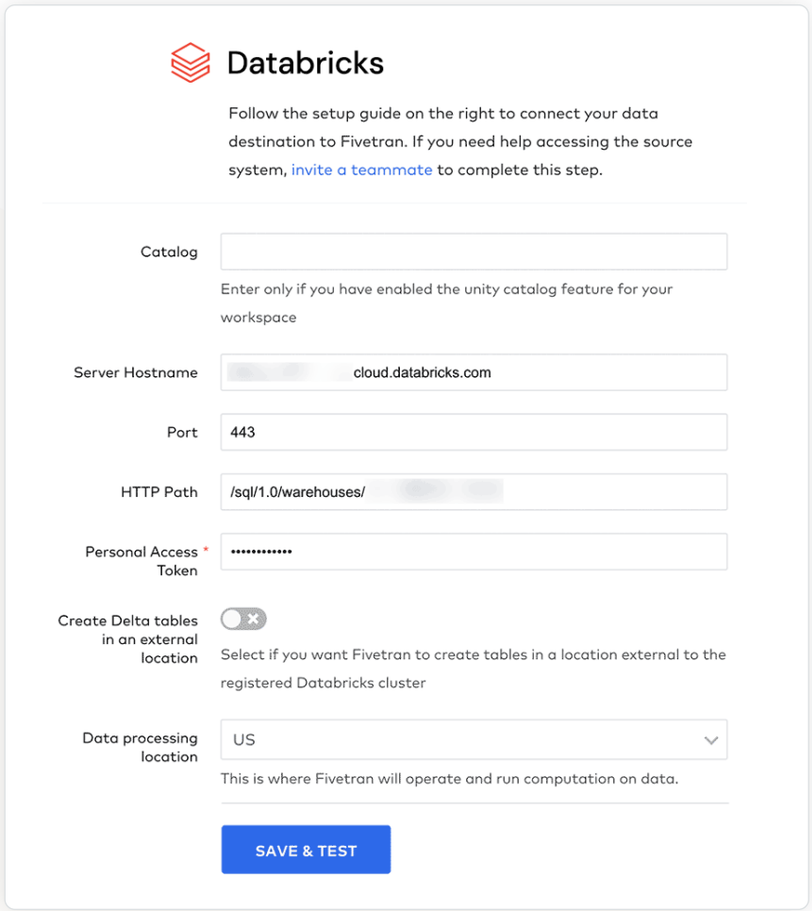


Figure 4 : Interface Fivetran pour la confirmation d'une nouvelle destination.

Passons dans l'interface Fivetran pour l'étape suivante. Nous allons créer et configurer des connexions à plusieurs systèmes sources différents (consultez la documentation officielle pour une liste complète des connexions prises en charge). Dans notre exemple, nous avons trois sources de données : 1) les dossiers de polices stockés dans un data store opérationnel (ODS) ou un data warehouse d'entreprise (EDW), 2) les déclarations de sinistre, stockées dans une base de données opérationnelle et 3) les données externes acheminées dans le stockage blob. Nous devons donc configurer trois connexions différentes dans Fivetran. Pour chacune d'elles, nous allons simplement suivre le processus guidé de Fivetran pour établir une connexion avec le système source. Les figures 5 et 6 montrent comment configurer de nouvelles connexions à des sources de données.

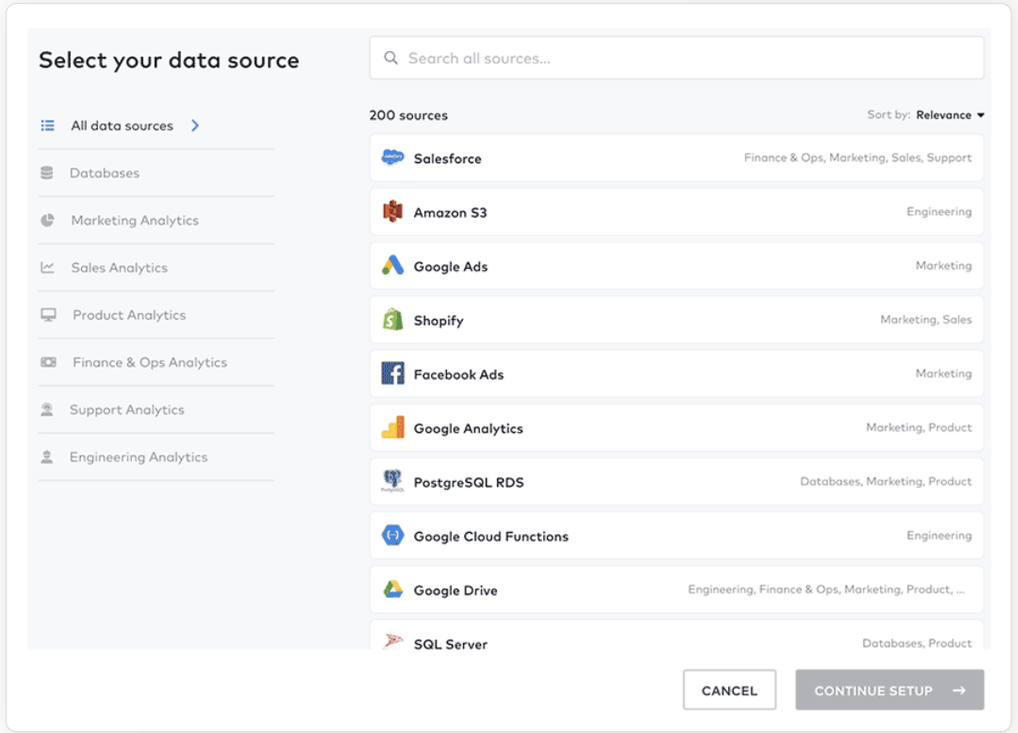


Figure 5 : Interface de Fivetran pour la sélection d'un type de source de données

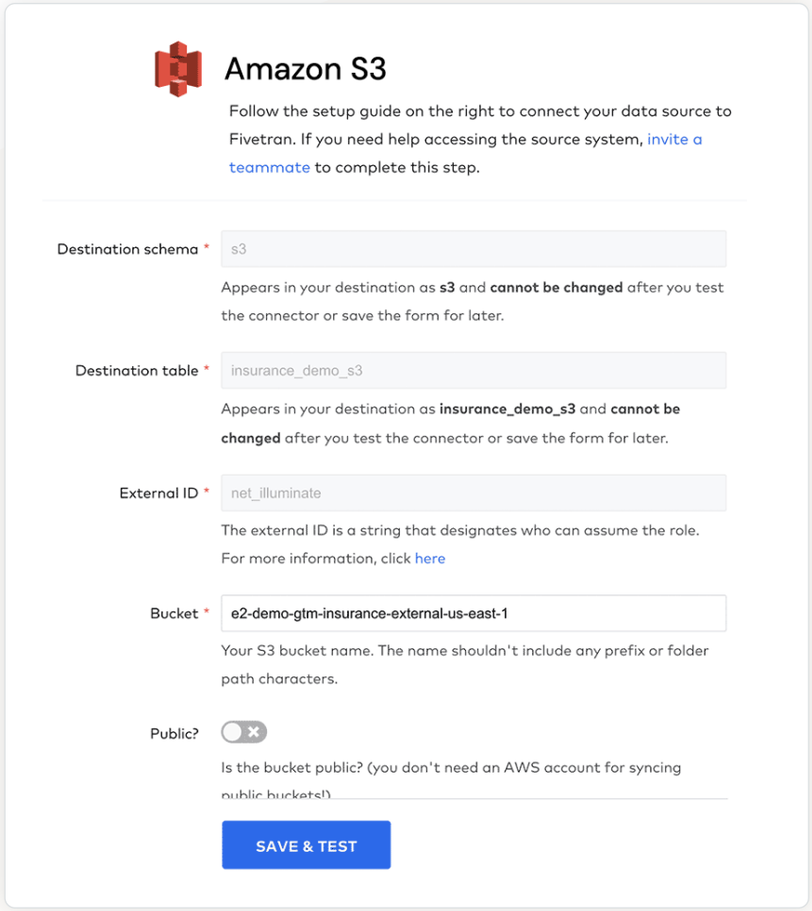


Figure 6 : Interface de Fivetran pour la configuration d'une connexion de source de données

Une fois les connexions validées, vous pouvez poursuivre leur configuration. Il est très important de définir la fréquence à laquelle Fivetran va interroger le système source pour récupérer les nouvelles données. La figure 7 nous montre à quel point Fivetran facilite le réglage de l'intervalle de synchronisation, de 5 minutes à 24 heures.

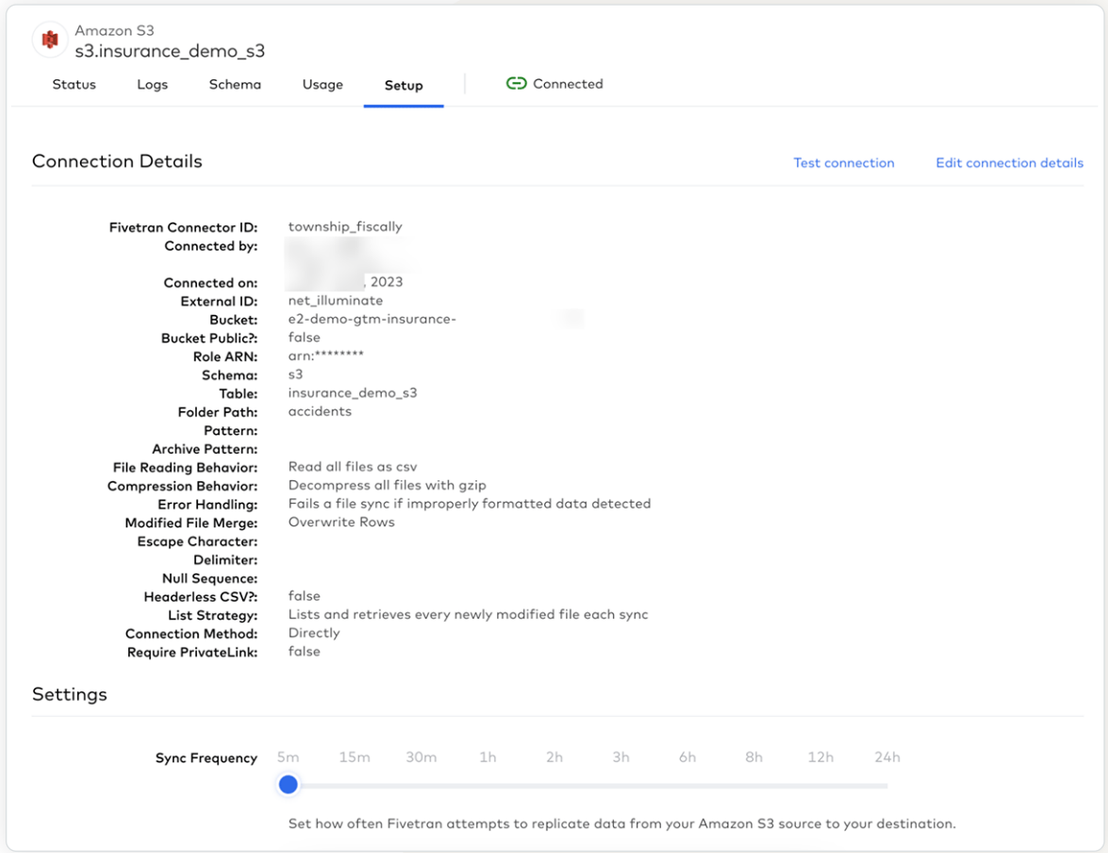


Figure 7 : Vue d'ensemble de la configuration d'un connecteur Fivetran.

Fivetran interroge et importe immédiatement les données des systèmes sources une fois la connexion validée. Les données sont stockées sous forme de tables Delta et peuvent être visualisées depuis Databricks via l'Explorateur de catalogue. Par défaut, Fivetran stocke toutes les données dans le metastore Hive. Chaque connexion entraîne la création d'un nouveau schéma contenant au moins deux tables : une pour les données, l'autre pour les journaux de chaque tentative de cycle d'ingestion (voir figure 8).

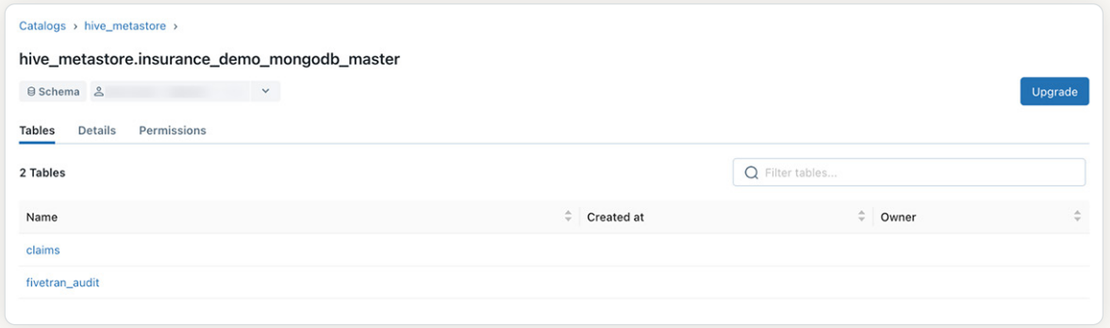


Figure 8 : Résumé des tables créées par Fivetran dans le warehouse Databricks pour un exemple de connexion.

Stocker les données dans des tables Delta présente des avantages. Delta Lake prend nativement en charge la gestion des versions à un niveau granulaire, ce qui permet de remonter dans les cycles d'importation (voir figure 9). Nous pouvons utiliser DB SQL pour interroger des versions spécifiques des données et analyser l'évolution des enregistrements sources.

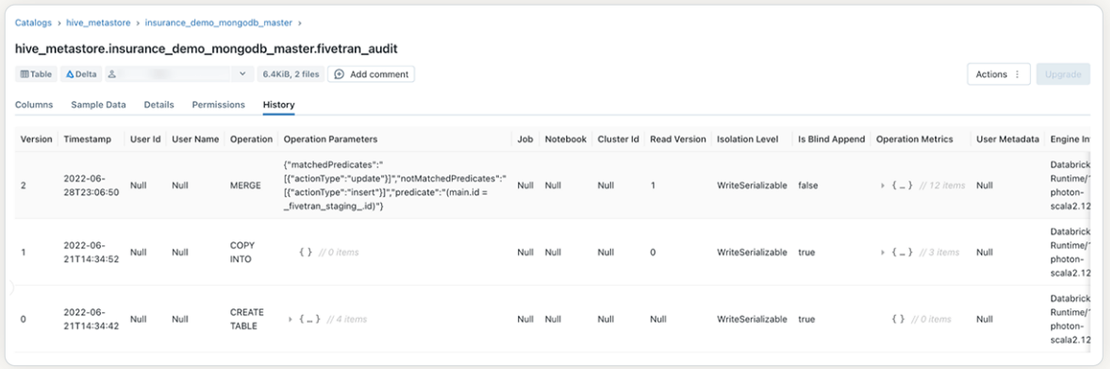


Figure 9 : Vue de l'historique montrant les modifications apportées à la table d'audit Fivetran.

Notez que si les données sources contiennent des valeurs semi-structurées ou non structurées, ces attributs seront aplatis lors du processus de conversion. Autrement dit, les résultats seront stockés dans des colonnes groupées de type texte, et ces entités devront être disséquées et décompressées avec DLT lors du processus d'organisation pour retrouver des attributs distincts.

Étape 2 : Automatisation du workflow

Une fois les données dans le lakehouse, les Delta Live Tables (DLT) permettent de créer un workflow de data engineering simple et automatisé. Les DLT fournissent un cadre déclaratif pour spécifier les étapes d'ingénierie de fonctionnalité de façon détaillée. Actuellement, DLT prend en charge des API pour Python et SQL. Dans cet exemple, nous allons créer notre workflow avec les API Python.

L'élément fondamental de DLT est la définition de la table. DLT interroge toutes les définitions de table pour créer un workflow complet de traitement des données. Par exemple, en Python, les tables sont créées à l'aide de définitions de fonction et du décorateur « dlt.table » (voir l'exemple de code Python ci-dessous). Ce décorateur permet de spécifier le nom de la table résultante, une description de sa finalité et un ensemble de propriétés.

```
@dlt.table(
   name             = "curated_claims",
   comment          = "Curated claim records",
   table_properties = {
       "layer": "silver",
       "pipelines.autoOptimize.managed": "true",
       "delta.autoOptimize.optimizeWrite": "true",
       "delta.autoOptimize.autoCompact": "true"
   }
)
def curate_claims():
   # Lire les enregistrements de sinistre préparés pour les mettre en mémoire
   staged_claims = dlt.read("staged_claims")
   # Extraire tous les attributs imbriqués pour créer une structure de table
aplatie
   curated_claims = unpack_nested(df = staged_claims, schema = schema_claims)

   …
```

Les instructions d'ingénierie de fonctionnalité sont définies dans le corps de la fonction à l'aide d'API PySpark standards et de commandes Python natives. L'exemple suivant montre comment PySpark associe les enregistrements de sinistre aux données de la table des polices pour produire une vue unique et organisée des sinistres.

```
   …

   # Lire les enregistrements de sinistre préparés pour les mettre en mémoire
   curated_policies = dlt.read("curated_policies")
   # Évaluer la validité du sinistre
   curated_claims = curated_claims \
       .alias("a") \
       .join(
           curated_policies.alias("b"),
           on  = F.col("a.policy_number") == F.col("b.policy_number"),
           how = "left"
       ) \
       .select([F.col(f"a.{c}") for c in curated_claims.columns] + [F.col(f"b.
{c}").alias(f"policy_{c}") for c in ("effective_date", "expiry_date")]) \
       .withColumn(
           # Calculer le nombre de mois écoulés entre le début de la couverture et
la déclaration de sinistre
           "months_since_covered", F.round(F.months_between(F.col("claim_date"),
F.col("policy_effective_date")))
       ) \
       .withColumn(
           # Vérifier si le sinistre a été déclaré avant l'entrée en vigueur de la
police
           "claim_before_covered", F.when(F.col("claim_date") < F.col("policy_
effective_date"), F.lit(1)).otherwise(F.lit(0))
       ) \
       .withColumn(
           # Calculer le nombre de jours écoulés entre le déroulement de
l'incident et la déclaration de sinistre
           "days_between_incident_and_claim", F.datediff(F.col("claim_date"),
F.col("incident_date"))
       )

   # Renvoyer le dataset organisé
   return curated_claims
```

L'un des grands avantages de DLT réside dans la possibilité de spécifier et appliquer des normes de qualité. On peut ainsi définir des attentes pour chaque table DLT et appliquer des contraintes détaillées de qualité aux données. Actuellement, DLT prend en charge les attentes dans trois scénarios différents :

Décorateur	Description
`expect`	Conserver les enregistrements qui ne répondent pas aux attentes
`expect_or_drop`	Supprimer les enregistrements qui ne répondent pas aux attentes
`expect_or_fail`	Suspendre l'exécution si un ou plusieurs enregistrements enfreignent des contraintes

Les attentes peuvent combiner plusieurs contraintes de qualité. Chaque contrainte doit comprendre une description et une expression Python ou SQL à évaluer. Il est possible de définir plusieurs contraintes à l'aide des décorateurs `expect_all`, `expect_all_or_drop` et `expect_all_or_fail`. Chaque décorateur attend un dictionnaire Python dont les clés sont les descriptions de contrainte et les valeurs, les expressions correspondantes. L'exemple ci-dessous montre plusieurs contraintes de qualité pour les scénarios de conservation et de suppression décrits ci-dessus.

```
@dlt.expect_all({
   "valid_driver_license": "driver_license_issue_date > (current_date() -
cast(cast(driver_age AS INT) AS INTERVAL YEAR))",
   "valid_claim_amount": "total_claim_amount > 0",
   "valid_coverage": "months_since_covered > 0",
   "valid_incident_before_claim": "days_between_incident_and_claim > 0"
})
@dlt.expect_all_or_drop({
   "valid_claim_number": "claim_number IS NOT NULL",
   "valid_policy_number": "policy_number IS NOT NULL",
   "valid_claim_date": "claim_date < current_date",
   "valid_incident_date": "incident_date < current_date",
   "valid_incident_hour": "incident_hour between 0 and 24",
   "valid_driver_age": "driver_age > 16",
   "valid_effective_date": "policy_effective_date < current_date()",
   "valid_expiry_date": "policy_expiry_date <= current_date()"
})
def curate_claims():
    …
```

Il est possible d'utiliser plusieurs notebooks Databricks pour déclarer des tables DLT. Si l'on suit l'architecture en médaillon, il est possible d'utiliser différents notebooks pour définir des tables comprenant les couches Bronze, Silver et Gold. Le cadre DLT peut condenser des instructions définies à l'aide de plusieurs notebooks en un seul workflow ; toutes les dépendances et relations entre les tables sont traitées et prises en compte automatiquement. La figure 10 montre le workflow complet de notre exemple de traitement des sinistres.

En partant des trois tables sources, DLT construit un pipeline complet qui produit treize tables utilisables par les fonctions métier.

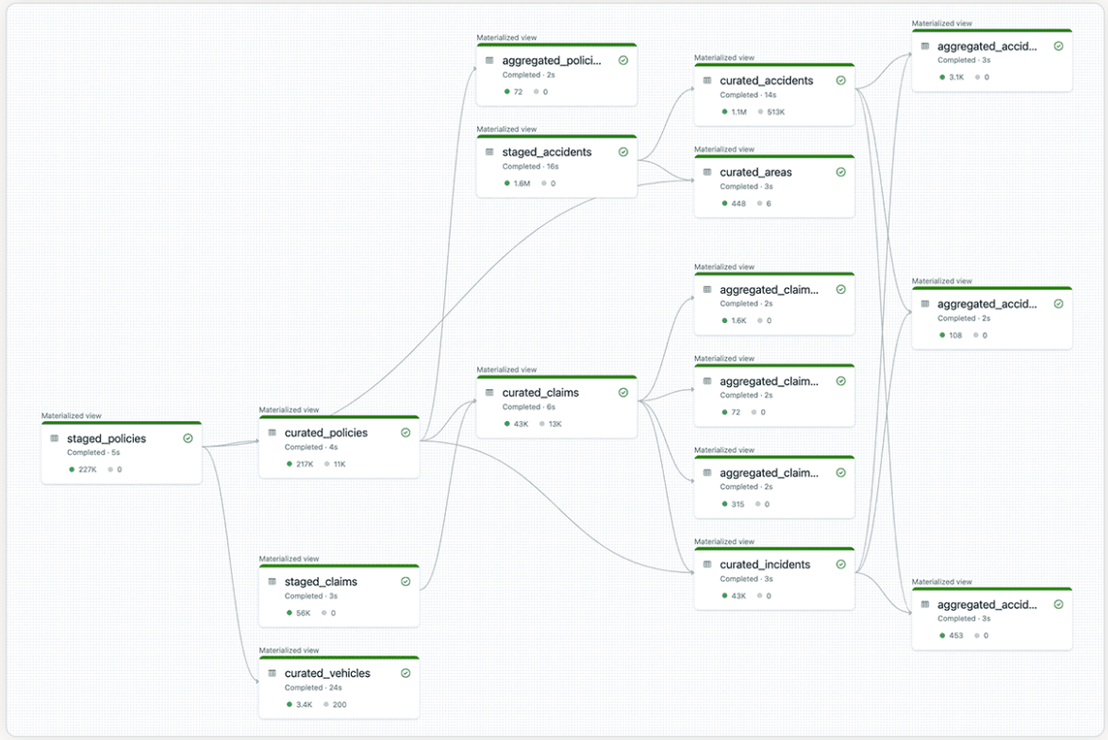


Figure 10 : Vue d'ensemble d'un workflow Delta Live Tables (DLT) complet.

Il est possible d'inspecter les résultats de chaque table en sélectionnant l'entité correspondante. La figure 11 offre un exemple de résultats de la table de sinistres organisée. DLT offre une vue d'ensemble de haut niveau des résultats du contrôle qualité :

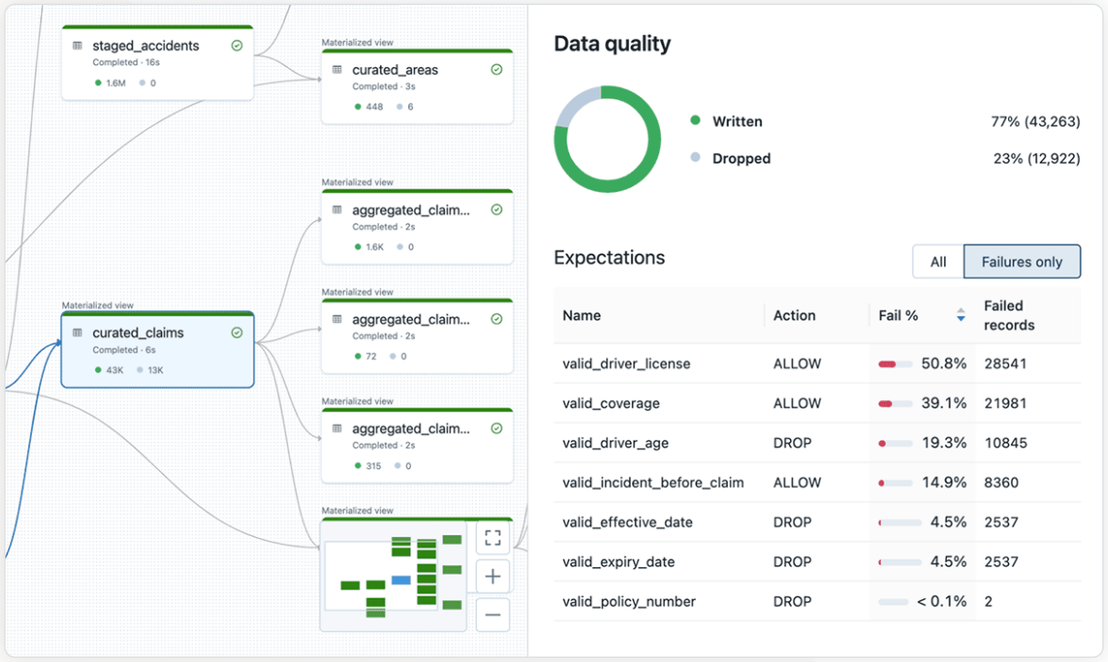
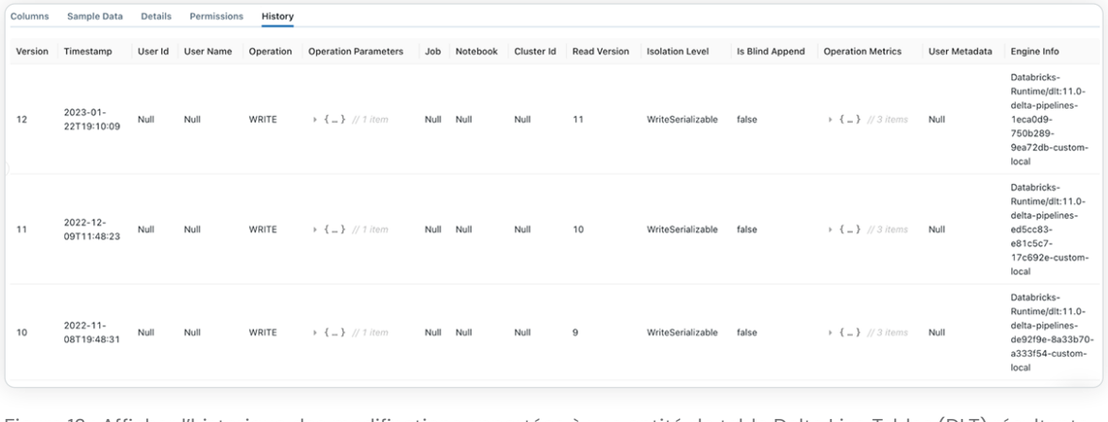


Figure 11 : Exemple de vue détaillée d'une entité de table Delta Live Tables (DLT) avec le rapport de qualité associé.

Il est possible d'approfondir l'analyse des résultats des attentes de qualité en interrogeant le journal d'événements. Il contient en effet des métriques détaillées sur toutes les attentes définies pour le pipeline du workflow. L'exemple de requête ci-dessous permet d'afficher les métriques clés de la dernière mise à jour du pipeline, et notamment le nombre d'enregistrements ayant répondu ou non aux attentes.

```
SELECT
  row_expectations.dataset AS dataset,
  row_expectations.name AS expectation,
  SUM(row_expectations.passed_records) AS passing_records,
  SUM(row_expectations.failed_records) AS failing_records
FROM
  (
    SELECT
      explode(
        from_json(
          details :flow_progress :data_quality :expectations,
          "array<struct<name: string, dataset: string, passed_records: int,
failed_records: int>>"
        )
      ) row_expectations
    FROM
      event_log_raw
    WHERE
      event_type = 'flow_progress'
      AND origin.update_id = '${latest_update.id}'
  )
GROUP BY
  row_expectations.dataset,
  row_expectations.name;
```

Une fois de plus, les journaux historiques Delta permettent de voir l'historique complet des modifications apportées à chaque table DLT (voir figure 12). Il est ainsi possible de voir l'évolution des tables au fil du temps et d'explorer des chaînes de mises à jour complètes en cas d'échec.



Columns | Sample Data | Details | Permissions | History

Version	Timestamp	User Id	User Name	Operation	Operation Parameters	Job	Notebook	Cluster Id	Read Version	Isolation Level	Is Blind Append	Operation Metrics	User Metadata	Engine Info
12	2023-01-22T19:10:09	Null	Null	WRITE	{ … } // 1 item	Null	Null	Null	11	WriteSerializable	false	{ … } // 3 items	Null	Databricks-Runtime/dlt:11.0-delta-pipelines-1eca0d9-750b289-9ea72db-custom-local
11	2022-12-09T11:48:23	Null	Null	WRITE	{ … } // 1 item	Null	Null	Null	10	WriteSerializable	false	{ … } // 3 items	Null	Databricks-Runtime/dlt:11.0-delta-pipelines-ed5cc83-e81c5c7-17c692e-custom-local
10	2022-11-08T19:48:31	Null	Null	WRITE	{ … } // 1 item	Null	Null	Null	9	WriteSerializable	false	{ … } // 3 items	Null	Databricks-Runtime/dlt:11.0-delta-pipelines-de92f9e-8a33b70-a333f54-custom-local

Figure 12 : Afficher l'historique des modifications apportées à une entité de table Delta Live Tables (DLT) résultante.

Il est ensuite possible d'utiliser les données CDC pour mettre à jour les tables en fonction des modifications des datasets sources. DLT CDC prend en charge la mise à jour des tables avec des dimensions à évolution lente (SCD) de type 1 et 2.

Il existe deux façons de faire en sorte que notre processus batch déclenche le pipeline DLT. Nous pouvons utiliser Auto Loader de Databricks pour traiter les nouvelles données de façon incrémentale, lorsqu'elles arrivent dans les tables sources, ou bien créer des tâches planifiées qui se déclencheront à des heures ou des intervalles définis. Ici, nous avons opté pour la deuxième approche : une tâche planifiée exécute le pipeline DLT toutes les cinq minutes.

Opérationnalisation des résultats

La capacité à traiter efficacement les données de manière incrémentale n'est que la moitié de l'équation. Les résultats du workflow DLT doivent être opérationnalisés et mis à disposition des utilisateurs métier. Dans notre exemple, les sorties du pipeline DLT sont exploitables via de l'analytique ad hoc ou sous forme d'insights prêts à l'emploi, mis à disposition dans un tableau de bord interactif.

Analytique ad hoc

Databricks SQL (ou DB SQL) fournit un data warehouse efficace et économique sur l'architecture lakehouse. Il permet d'exécuter des charges SQL directement sur les données sources avec un rapport prix-performance jusqu'à 12 fois supérieur à celui de ses alternatives.

Nous pouvons utiliser DB SQL pour effectuer des requêtes ad hoc spécifiques sur nos tables organisées et agrégées. Nous pourrions, par exemple, exécuter une requête sur la table des polices organisées afin de calculer l'exposition totale. L'éditeur de requêtes DB SQL propose une interface simple et intuitive pour créer et exécuter ce type de requêtes (voir l'exemple ci-dessous).

```
SELECT
  round(curr.total_exposure, 0) AS total_exposure,
  round(prev.total_exposure, 0) AS previous_exposure
FROM
  (
    SELECT
      sum(sum_insured) AS total_exposure
    FROM
      insurance_demo_lakehouse.curated_policies
    WHERE
      expiry_date > '{{ date.end }}'
      AND (effective_date <= '{{ date.start }}'
        OR (effective_date BETWEEN '{{ date.start }}' AND '{{ date.end }}'))
  ) curr
  JOIN
  (
    SELECT
      …
```

L'éditeur de requêtes DB SQL permet d'exécuter des requêtes sur différentes versions de nos tables Delta. Nous pouvons ainsi interroger une vue des déclarations de sinistres agrégés en fonction d'une date et d'une heure spécifiques (voir l'exemple ci-dessous). Nous pouvons également utiliser DB SQL pour comparer les résultats de différentes versions afin d'analyser uniquement les enregistrements modifiés entre ces états.

```
SELECT
  *
FROM
  insurance_demo_lakehouse.aggregated_claims_weekly TIMESTAMP AS OF '2022-06-
05T17:00:00';
```

DB SQL offre la possibilité d'utiliser un moteur de calcul serverless, ce qui évite d'avoir à configurer, gérer ou faire évoluer une infrastructure cloud et maintient les coûts au niveau le plus bas. Il s'intègre également aux autres consoles SQL (dont DataGrip) pour permettre aux analystes d'explorer les données et de générer des insights avec leurs outils habituels.

Insights commerciaux

Les requêtes DB SQL permettent enfin d'ajouter des visualisations riches aux résultats. Ces visualisations peuvent ensuite être présentées aux utilisateurs finaux via des tableaux de bord interactifs (voir figure 13).

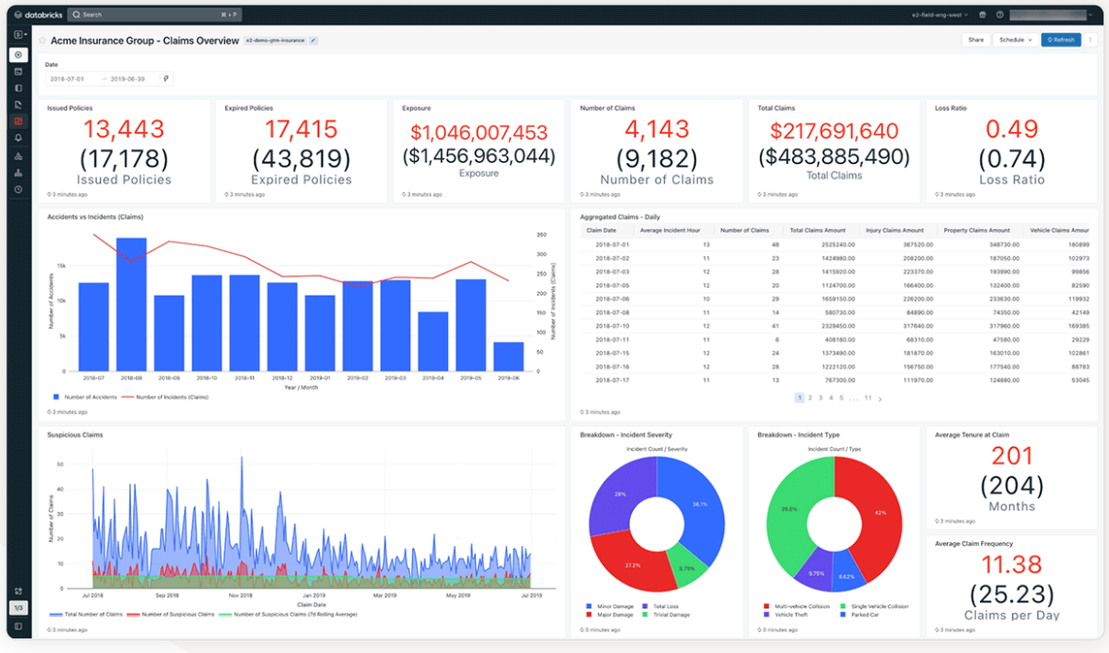


Figure 13 : Exemple de tableau de bord opérationnel reposant sur un ensemble d'entités de table Delta Live Tables (DLT).

Pour notre exemple, nous avons créé un tableau de bord qui rassemble des métriques clés, des calculs glissants, des ventilations de haut niveau et des vues agrégées. Ce tableau résume l'ensemble de notre processus de traitement des sinistres. Nous y avons ajouté une option permettant de spécifier une période. DB SQL prend en charge différents paramètres de requête qui seront remplacés par des valeurs au moment de l'exécution de la requête. Ces paramètres peuvent être définis au niveau du tableau de bord afin que toutes les requêtes associées soient actualisées en cas de changement.

DB SQL s'intègre à de nombreux outils tiers d'analytique et de BI, dont Power BI, Tableau et Looker. Comme dans le cas de Fivetran, Partner Connect peut aussi relier notre plateforme externe à DB SQL. Les analystes pourront ainsi créer et servir des tableaux de bord sur les plateformes habituelles de l'entreprise sans perdre les performances de DB SQL et de la Data Intelligence Platform.

Conclusion

Dans le monde hautement dynamique et volatile de la finance, le traitement batch reste un aspect essentiel de la pile de données moderne, y compris face aux fonctionnalités et aux avantages des services de streaming et en temps réel. Nous avons vu comment utiliser l'architecture lakehouse pour les services financiers et leur écosystème de partenaires afin de concevoir un cadre simple, évolutif et extensible capable d'accueillir des tâches complexes de traitement batch, en prenant l'exemple du traitement des sinistres dans le secteur de l'assurance. Delta Live Tables (DLT) et Databricks SQL (DB SQL) forment la base d'une plateforme de données possédant une architecture infiniment évolutive et extensible pour répondre durablement à l'évolution des besoins.

Pour en savoir plus sur l'exemple de pipeline décrit et sur l'installation et la configuration de l'infrastructure utilisée, explorez ce dépôt GitHub ou regardez cette vidéo.

SECTION

05

Success stories : des résultats concrets sur Databricks

SECTION 5.1

InMobi : Tisser des liens significatifs entre les marques et leurs clients

En matière de publicité, les consommateurs attendent du contenu pertinent et personnalisé, en particulier sur leurs appareils mobiles où le temps est précieux. Il faut donc savoir capter instantanément leur attention et susciter leur engagement. InMobi utilise les données client en temps réel pour proposer des publicités mobiles ciblées et des expériences sur l'écran de verrouillage. Mais quand le volume de données à traiter a dépassé les 20 To par heure, les coûts du data warehouse multicloud sont montés en flèche. D'autre part, les silos inhérents à sa conception entravaient la collaboration et le partage de données.

InMobi a donc abandonné son data warehouse multicloud pour Databricks afin d'unifier ses différentes charges (data warehousing, IA et analytique) et rationaliser ses opérations. Les ingénieurs, devenus plus agiles et efficaces, ont pu se consacrer à des tâches plus rentables. Pour l'entreprise, le lakehouse affiche d'abord un TCO très inférieur à celui du data warehouse multicloud. Mais il a surtout amélioré la productivité de l'ensemble de l'organisation et accéléré la mise sur le marché de nouveaux produits.

Databricks a parfaitement répondu à notre objectif d'optimiser le rapport prix-performance. Le lakehouse a réduit nos coûts sans dégrader les performances sur les charges de travail mixtes, nous permettant ainsi d'optimiser nos opérations de données et d'IA, aujourd'hui comme demain.

— MOHIT SAXENA

Cofondateur et CTO groupe,

InMobi

32 %
de réduction du TCO par rapport au data warehouse multicloud

15 %
d'accélération des requêtes par rapport au data warehouse multicloud

20 %
d'amélioration des performances du reporting fournisseurs

databricks

Une infrastructure complexe et un data warehouse multicloud difficile à gérer

Spécialiste de la publicité ciblée, InMobi aide les marques à atteindre et à impliquer les consommateurs de manière significative et rentable. Mais pour délivrer des publicités pertinentes, il faut des données, beaucoup de données. L'entreprise a progressivement étendu son système Hadoop local en y ajoutant des data warehouses cloud. Mais le volume de données à traiter augmentait de façon exponentielle, atteignant 20 To par heure. Cette approche d'expansion a fini par produire un data warehouse multicloud problématique à de nombreux égards. Trop complexe, souvent en panne et de plus en plus cher, il était fragmenté en silos qui limitaient le partage de données et la collaboration. Pour l'équipe d'InMobi, les choses étaient claires : poursuivre dans cette voie était un frein à sa capacité d'innovation et mobilisait inutilement ses ressources d'ingénierie.

> **« Notre infrastructure de données fonctionnait, mais sa complexité et sa maintenance nous détournaient de nos propres produits, » explique Madhan Sundaram, Directeur senior de l'ingénierie de plateforme chez InMobi. « Nous voulions que le talent de nos ingénieurs serve à créer plus de valeur pour nos clients, ce qui nécessitait un système plus simple et unifié. »**

InMobi souhaitait un système unique capable de résoudre plusieurs problèmes. Il lui fallait donc rassembler ses systèmes disjoints en une plateforme unique afin que ses ingénieurs puissent se consacrer à des tâches de valeur, comme le développement du ML et de grands modèles de langage. L'équipe a choisi la Data Intelligence Platform de Databricks pour unifier son data warehouse et ses charges d'IA sur une seule plateforme.

La migration vers le lakehouse unifie les données, l'analytique et l'IA

InMobi emploie une équipe d'ingénieurs très compétents, mais l'entreprise a beaucoup appris sur la productivité et l'agilité opérationnelle. « Nous passions trop de temps à maintenir notre environnement, et cela nous empêchait de collaborer utilement avec les fonctions métier, » explique M. Sundaram. « Il nous fallait une approche plus stratégique pour opérationnaliser les données plus efficacement. » Après avoir soigneusement évalué leur data warehouse multicloud et envisagé un développement interne, l'équipe a conclu que la Data Intelligence Platform de Databricks était la plus en phase avec ses objectifs : améliorer la productivité des développeurs en réduisant la complexité de l'infrastructure, avec un rapport prix-performance compétitif.

Une fois son choix arrêté, l'équipe a fait appel à Databricks pour entamer le processus de planification de la migration. Avec dix ans de code personnalisé et plus d'un pétaoctet de données, InMobi savait que la migration serait complexe. L'ETL impliquait à lui seul 150 pipelines, et sa migration depuis Apache Spark™ a mobilisé huit équipes. Il fallait également migrer environ 300 tableaux de bord pour garantir la continuité de la transmission des informations aux fournisseurs et aux clients. Databricks a collaboré étroitement avec InMobi – qui avait affecté deux ingénieurs internes par équipe – et son partenaire de mise en œuvre, Celebal Technologies, pour établir les optimisations indispensables à une migration fluide.

Cet accompagnement a permis à InMobi de passer sans douleur de son data warehouse multicloud au lakehouse. « Databricks nous donne un avantage technique de poids en nous permettant d'utiliser des fonctionnalités inédites, » affirme M. Sundaram. « Grâce à l'expertise évidente de l'équipe, nous avons pu optimiser nos ressources informatiques en termes de coût et de performance. Databricks a pris en main la réalisation des optimisations. Nous avons été impressionnés par sa transparence et son sens de la pédagogie. »

Présenter des publicités mobiles personnalisées aux clients du monde entier

Désormais, grâce à une architecture lakehouse unifiée et rationalisée, InMobi peut exploiter tout le potentiel de ses données clients pour proposer une publicité mobile plus intelligente et personnalisée. Différentes équipes utilisent les notebooks Databricks pour produire des analyses ad hoc, PowerBI pour les visualisations avec Databricks SQL (le data warehouse serverless du lakehouse) et MLflow pour créer une plateforme d'IA de nouvelle génération. Delta Live Tables s'est également avéré très utile pour la détection des anomalies, ce qui s'est traduit par une amélioration de 50 % des SLA et 80 % de réduction des coûts. InMobi a enfin résolu la question des silos et de la découverte des données avec Unity Catalog. L'équipe peut désormais gérer les accès au niveau des tables et des colonnes, tout en conservant un data lineage complet qui donne une visibilité sur l'origine des données et indique si elles sont obsolètes. Munie de cette plateforme conçue pour ses besoins d'analytique et d'IA, l'équipe InMobi explore de nouvelles technologies, notamment les grands modèles de langage (LLM), pour délivrer des insights toujours plus précis à ses clients. « Nous envisageons de mettre en œuvre des LLM pour permettre à nos utilisateurs finaux de poser plus facilement des questions et de trouver des informations. L'architecture lakehouse sera d'une aide précieuse, car les tâches s'exécuteront automatiquement en coulisses. Nos équipes, expertes ou non, n'auront qu'à poser leur question pour obtenir une réponse contextualisée, » détaille M. Sundaram.

Sur le plan commercial, l'équipe a observé des améliorations mesurables à tous les niveaux après la migration. Les coûts d'infrastructure ont baissé de 34 %, la vitesse des requêtes a augmenté de 15 % et le nombre d'échecs a diminué de 20 %. Le résultat : une amélioration de 20 % des performances de reporting et de création d'insights pour les utilisateurs finaux. Le TCO est inférieur de 32 % à celui du data warehouse multicloud, et les coûts d'exploitation des pipelines ETL ont également été réduits de 24 %. Sur un plan plus qualitatif, l'équipe constate une amélioration globale de la fiabilité et de la stabilité des systèmes, ce qui a rehaussé la réputation de la marque auprès de ses clients.

« Notre rythme d'expérimentation est beaucoup plus soutenu, » a déclaré Mohit Saxena, Cofondateur et directeur technique groupe chez InMobi. « Databricks a simplifié la collaboration et l'approche unifiée du lakehouse nous a fait gagner en efficacité, en productivité et en conformité dans la livraison de nouvelles fonctionnalités et de nouveaux produits. »

En misant sur l'unification avec la Data Intelligence Platform de Databricks, InMobi peut désormais innover dans la publicité mobile et offrir la personnalisation en temps réel. Cela génère de la valeur pour ses clients comme pour ses utilisateurs internes.

Lire d'autres success-stories

SECTION 5.2

Akamai : Analytique en temps réel et à grande échelle avec Delta Lake

Akamai exploite un vaste réseau de diffusion de contenu (CDN) hautement distribué. Reposant sur 345 000 serveurs dans plus de 135 pays et plus de 1 300 réseaux, ce CDN achemine le trafic Internet des plus grands noms des médias, du commerce, de la finance, du retail et de bien d'autres secteurs. Environ 30 % du trafic Internet transite par les serveurs d'Akamai. Akamai propose également des solutions de sécurité cloud.

En 2018, la société a lancé un outil d'analyse de sécurité web qui offre à ses clients une interface unifiée pour évaluer un large éventail d'événements de sécurité en streaming et les analyser. Cet outil d'analytique web permet aux clients d'Akamai de prendre des décisions éclairées en temps réel face aux événements de sécurité. Akamai parvient à diffuser ces volumes de données et à respecter les stricts SLA conclus avec ses clients en adossant son outil d'analytique sur Delta Lake et la Data Intelligence Platform de Databricks.

Delta Lake n'améliore pas seulement nos capacités de requête, il supporte aussi l'augmentation du volume de données. Il faut pouvoir évoluer à grande vitesse : le trafic et la quantité de données ont augmenté de 80 % l'année dernière.

— TOMER PATEL

Responsable de l'ingénierie,

Akamai

< 1

minute de temps d'importation, contre 15 minutes auparavant

> 85 %

des requêtes affichent un temps de réponse inférieur à 7 secondes

Importer et diffuser des quantités considérables de données

L'outil d'analyse de sécurité d'Akamai ingère environ 10 Go de données d'événements de sécurité par seconde. Ce volume peut augmenter considérablement lorsque des clients du secteur du retail connaissent des pics d'activité, notamment lors du Black Friday ou du Cyber Monday. L'outil d'analytique de sécurité stocke plusieurs pétaoctets de données à des fins d'analyse. Ces analyses ont pour but de protéger les clients d'Akamai et de leur permettre d'explorer et d'interroger eux-mêmes les événements de sécurité.

Cet outil reposait initialement sur une architecture Apache Spark™ sur Hadoop en local. Akamai garantit des SLA stricts à ses clients : 5 à 7 minutes maximum doivent s'écouler entre le moment où une attaque se produit et celui où elle s'affiche dans l'outil. L'entreprise voulait améliorer les vitesses d'importation et de requête pour tenir ces SLA. « Il faut s'approcher au maximum du temps réel afin que nos clients puissent voir ce qui les attaque, » déclare Tomer Patel, Responsable de l'ingénierie chez Akamai. « Ils doivent pouvoir interroger des données dans les meilleurs délais. Nous voulions abandonner notre système sur site pour améliorer nos performances et nos SLA, tout en réduisant la latence à quelques secondes. »

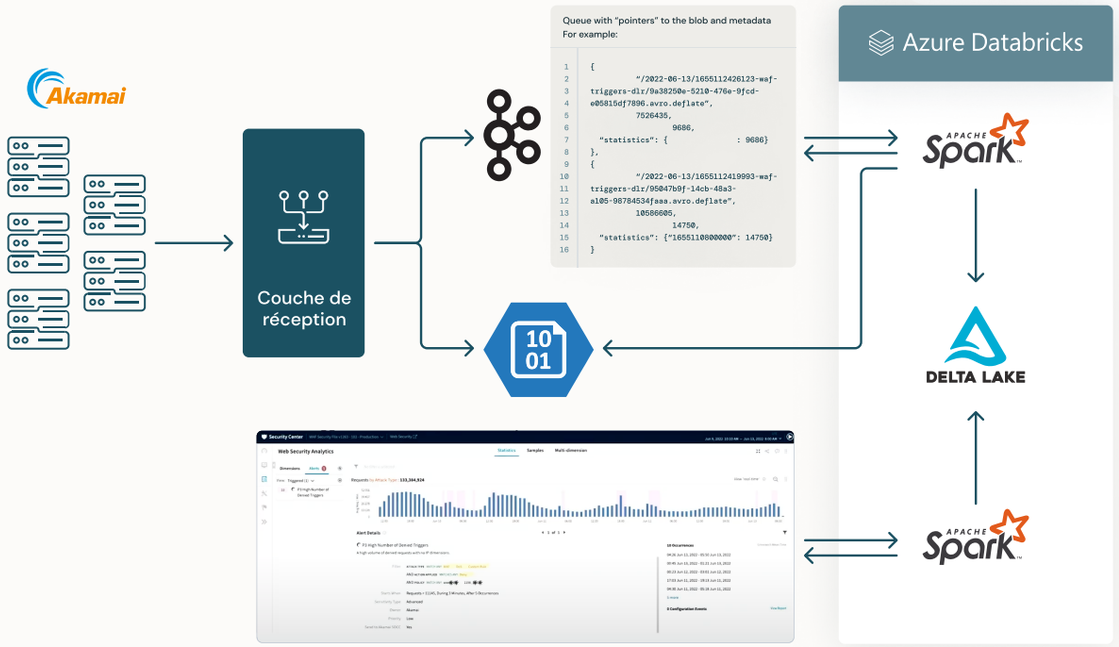
Après avoir réalisé des études de faisabilité auprès de divers fournisseurs, Akamai a choisi Spark et la Data Intelligence Platform de Databricks pour son architecture d'analytique en streaming. « Compte tenu de notre envergure et des exigences de nos SLA, nous avons conclu que Databricks était la meilleure solution, » explique T. Patel. « Si l'on considère l'optimisation du stockage et la mise en cache des données, aucune autre solution ne nous permettrait d'atteindre ce niveau de performances. »

Gagner en vitesse et réduire les coûts

Aujourd'hui, l'outil d'analytique de sécurité web ingère et transforme les données, les entrepose dans le cloud et envoie l'emplacement du fichier via Kafka. Il utilise ensuite une tâche Databricks comme application d'ingestion. Delta Lake, le format de stockage open source à la base de la Data Intelligence Platform de Databricks, permet d'effectuer des requêtes en temps réel sur les données d'analytique. Delta Lake est aussi un atout en termes d'évolutivité pour Akamai. « Delta Lake n'améliore pas seulement nos capacités de requête, il supporte aussi l'augmentation du volume de données, » affirme T. Patel. Il faut pouvoir évoluer à grande vitesse : le trafic et la quantité de données ont augmenté de 80 % l'année dernière. »

Akamai utilise également Databricks SQL (DBSQL) et Photon, qui offrent des performances de requête exceptionnelles. Selon T. Patel, Photon a considérablement amélioré les performances des requêtes. Dans l'ensemble, grâce à l'architecture de streaming de Databricks, combinée à DBSQL et Photon, Akamai réalise de l'analytique en temps réel, ce qui se traduit par des bénéfices commerciaux directs.

Et T. Patel apprécie le fait que Delta Lake soit open source, car il a pu s'appuyer sur le travail et les améliorations de la communauté d'utilisateurs. « Comme Delta Lake est open source et jouit du soutien d'une vaste communauté, nous n'avons pas besoin de tout mettre en œuvre nous-mêmes, » explique T. Patel. « Nous bénéficions des correctifs et des optimisations que d'autres équipes ont apportés au projet. » Akamai a collaboré avec Databricks afin que Delta Lake puisse répondre à ses exigences de mise à l'échelle et de performances. Ces améliorations ont été réinjectées dans le projet et, pour beaucoup, publiées dans le cadre de Delta Lake 2.0. De ce fait, tout utilisateur de Delta Lake profite désormais d'une technologie testée à très grande échelle dans un scénario de production réel.



Répondre à des exigences strictes d'évolutivité, de fiabilité et de performance

Grâce au streaming structuré de Spark sur la Data Intelligence Platform de Databricks, l'outil d'analytique de sécurité web diffuse de grands volumes de données et délivre aux clients d'Akamai des analyses en temps réel sous forme de service à faible latence. Akamai parvient ainsi à fournir les données d'événements de sécurité à ses clients dans le délai prescrit par le SLA, autrement dit dans les 5 à 7 minutes suivant l'apparition d'une attaque. « Notre objectif : la performance, encore et toujours la performance, » affirme T. Patel. « Tout repose sur la performance et l'évolutivité de la plateforme. »

Avec la Data Intelligence Platform de Databricks, il faut moins d'une minute pour importer les données d'événement. « Passer de 15 minutes à moins d'une minute est une amélioration considérable, » déclare T. Patel. « C'est un atout de poids pour nos clients qui peuvent consulter les données de sécurité plus rapidement et analyser la situation en filtrant les informations. »

La plus grande priorité d'Akamai est d'offrir à ses clients une expérience de qualité et des réponses rapides. À ce jour, Akamai a déplacé environ 70 % des données d'événements de sécurité vers Databricks, avec un impact direct sur les SLA liés aux vitesses de requête et de réponse. « Depuis le passage à Databricks, nos clients bénéficient d'un temps de réponse bien plus court : plus de 85 % des requêtes sont traitées en moins de 7 secondes. » En fournissant ces données en temps réel, Akamai aide ses clients à rester vigilants et à maintenir une position de sécurité optimale.

Lire d'autres success-stories

SECTION 5.3

Quartile : Devenir la plus grande plateforme publicitaire de l'e-commerce

Avec plus de 5 000 comptes publicitaires connectés sur plus de 10 canaux, Quartile est la plus grande plateforme de publicité multicanal pour l'e-commerce au monde. Bâtie sur six technologies de machine learning brevetées, elle automatise et optimise la publicité e-commerce sur Google, Facebook, Amazon, Instacart, Walmart et plus encore. Quartile réunit une technologie de pointe et des experts en marketing qui créent des stratégies sur mesure pour les objectifs commerciaux de leurs clients. Des milliers de vendeurs dans le monde ont choisi l'approche d'optimisation de l'entonnoir de conversion (funnel) de Quartile pour libérer leur potentiel commercial et publicitaire sur plusieurs canaux entièrement intégrés, à grande échelle.

80 %
de réduction du stockage des données par rapport aux bases de données traditionnelles

10 x plus rapide
45 minutes pour optimiser les enchères, contre 7,5 heures auparavant

La Data Intelligence Platform de Databricks permet à nos équipes technologiques d'accélérer sans cesse la commercialisation de nos nouvelles solutions et de ravir les clients avec des datasets de qualité. Devenir la plus grande plateforme publicitaire multicanal de l'e-commerce serait beaucoup plus difficile sans Databricks et l'architecture lakehouse.

— DANIEL KNIJNIK
CEO,
Quartile

databricks

Volume des données et impératifs de performance

Les difficultés de Quartile concernaient le stockage et le traitement des données de plusieurs canaux publicitaires, soumis à des impératifs de reporting et de consolidation des données, ce qui incluait plus de 60 jours d'attribution. L'architecture précédente n'était pas à la hauteur des volumes de données traités par Quartile. L'équipe traitait en batch plus de 10 To de données sur une seule tâche. Mais celle-ci appliquait toutes les transformations requises par le reporting, mobilisait entièrement le serveur et retardait la livraison des points de données. Ce processus chargé d'améliorer les performances des annonces présentait de graves problèmes de performances ; certaines tâches pouvaient prendre jusqu'à 7,5 heures par jour.

Évolution technologique : moderniser la pile de données

Au fil de son évolution, l'architecture de données de Quartile a franchi des étapes de maturité en troquant les bases de données SQL traditionnelles exécutées sur le cloud Azure contre une nouvelle solution reposant sur la Data Intelligence Platform de Databricks. Cette modernisation a eu un impact direct sur plusieurs domaines de l'entreprise, et les clients en ont récolté les fruits. Avec Delta Lake, les données sont plus fiables, les performances sont meilleures et les coûts diminuent. En passant d'une base de données SQL traditionnelle à Databricks, l'équipe a observé une réduction considérable du volume de données, principalement grâce aux optimisations de Delta avec contrôle de version et au compactage Parquet, qui a fait passer le stockage de 90 à 18 To environ.

L'architecture lakehouse soutient l'évolution de la pile de données de Quartile. Pour cela, plusieurs processus entrent en jeu. Databricks Auto Loader assure le traitement incrémentiel et efficace des fichiers de données qui arrivent dans le stockage cloud. La couche de stockage Delta Lake, avec son format ouvert, apporte de la fiabilité, de la sécurité et des performances au data lake. Databricks SQL facilite de son côté la collaboration entre ingénieurs et analystes avec ses outils simples de création de requêtes et de tableaux de bord. Enfin, Databricks Workflows relie toutes les pièces entre elles de manière stable et évolutive.

Pour offrir la meilleure expérience à ses clients, Quartile doit pouvoir compter sur des données précises. Pour relever ce défi de poids, les fonctions Spark définies par l'utilisateur permettent d'exploiter la puissance du parallélisme pour diviser le traitement autant que nécessaire. Pour assurer l'évolutivité de la solution, l'équipe utilise Terraform pour déployer les espaces de travail, créer facilement de nouveaux clusters et tâches et appliquer les normes appropriées dans toute l'entreprise.

Aider les partenaires à mieux diffuser leurs annonces

Les clients de Quartile ont besoin d'une solution centralisée pour analyser leurs ventes, leurs coûts et autres métriques de campagnes. Avec Quartile, ils peuvent compter sur un solide travail d'ingénierie des données et sur l'intégration de Databricks avec Power BI, qui incorpore directement les tableaux de bord au portail. Les clients disposent ainsi d'une interface unique pour configurer leurs campagnes sur plusieurs canaux et suivre l'évolution des performances. L'entreprise maintient par ailleurs une empreinte de stockage réduite et 80 % moins chère qu'avec les data warehouses traditionnels, grâce au format de fichier de Delta Lake sur le stockage d'objets. Combiner les données des différents canaux a déjà aidé plusieurs clients : SmartyPants, par exemple, a connu une croissance de plus de 100 % depuis son partenariat avec Quartile.

Mais ce n'est pas tout. Quartile a également breveté des algorithmes pour améliorer les performances des publicités, algorithmes implémentés via le persona machine learning dans Databricks. Pouvoir appuyer toute leur pile de données sur une architecture lakehouse centrale a considérablement simplifié la tâche des développeurs de Quartile. Ils peuvent désormais se concentrer sur la création de solutions innovantes et l'amélioration des résultats pour les clients. Autre exemple, OfficeSupply a obtenu d'excellents résultats dès sa première année de collaboration avec Quartile : ses revenus Google Ads ont augmenté de 67 % et les clics Google Shopping pour ses mots-clés déposés ont plus que doublé. Tout cela grâce à de meilleures performances sur des tâches désormais exécutées en 45 minutes (contre 7,5 heures auparavant) sur le lakehouse.

Quartile prévoit de continuer de faire appel à Databricks pour développer sa pile de données moderne, en intégrant et en testant de nouvelles solutions. L'entreprise pense notamment à Delta Live Tables pour renforcer les contrôles de qualité des données, à Delta Sharing pour envoyer aux clients leurs données, et à Data Marketplace pour faciliter la prise en main de sa solution. Quartile a pour ambition de développer le premier apprentissage multicanal pour mettre au point un algorithme d'optimisation des publicités dans cet espace. Databricks sera au centre de ces innovations.

À propos de Databricks

Databricks est une entreprise axée sur les données et l'intelligence artificielle. Plus de 10 000 entreprises internationales, parmi lesquelles figurent Comcast, Condé Nast, Grammarly et plus de 50 % des entreprises du Fortune 500, s'appuient sur la Data Intelligence Platform de Databricks pour unifier et démocratiser leurs données, leurs capacités d'analytique et d'intelligence artificielle. Databricks a son siège à San Francisco et des bureaux dans le monde entier. Elle a été fondée par les créateurs de Lakehouse, Apache Spark™, Delta Lake et MLflow.

Pour en savoir plus, suivez Databricks sur Twitter, LinkedIn et Facebook.

COMMENCEZ VOTRE ESSAI GRATUIT

Contactez-nous pour une démonstration personnalisée **databricks.com/company/contact**.

databricks