databricks

E-Book

Das Big Book of Data Warehousing and BI

Inhalt

Daten entscheiden über den Erfolg von Unternehmen. Je stärker Unternehmen auf Daten angewiesen sind, desto wichtiger wird eine effiziente Datenverwaltung und Governance. Die Data Intelligence Platform von Databricks ermöglicht eine effektive Datenverwaltung und -nutzung sowie den Zugriff auf Daten und KI. Sie setzt auf der Lakehouse-Architektur auf und bündelt die Vorteile von Data Lakes und Data Warehouses. Das spart Kosten und beschleunigt Daten- und KI-Initiativen. Die Plattform bietet eine einheitliche Governance für Daten und KI sowie eine flexible Abfrage-Engine für ETL, SQL, maschinelles Lernen und BI.

Einführung in die Data Intelligence Platform

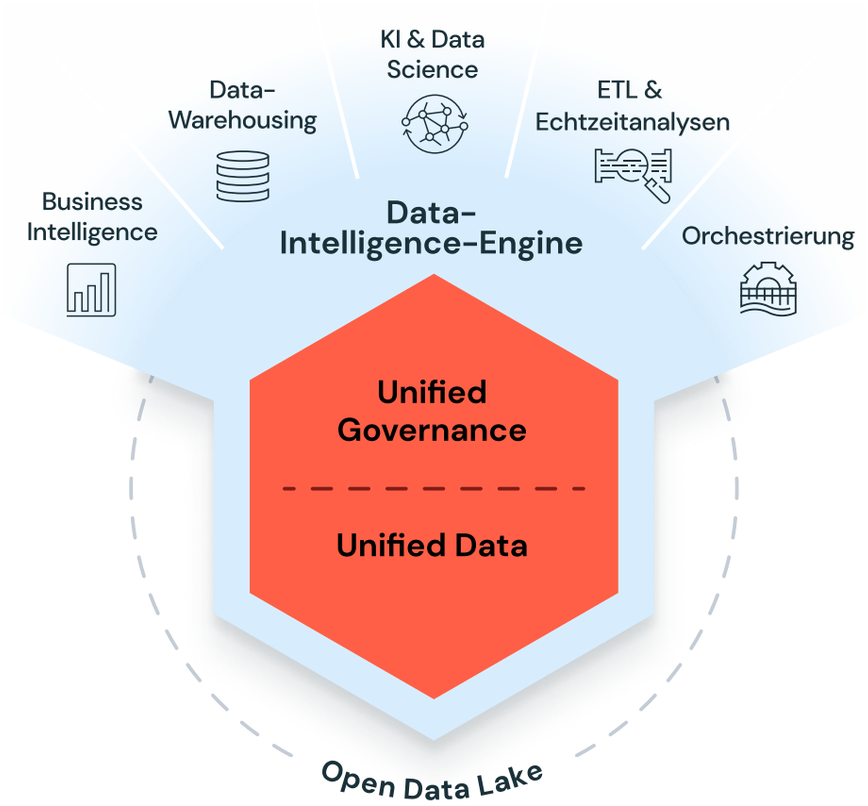
Über die Databricks Data Intelligence Platform können Unternehmen ihre Daten und KI effektiv verwalten, nutzen und darauf zugreifen. Die Plattform basiert auf der Lakehouse-Architektur mit einer einheitlichen Governance-Ebene für Daten und KI sowie einer zentralen Abfrage-Engine, die ETL, SQL, KI und BI umfasst. Sie kombiniert das Beste aus Data Lakes und Data Warehouses, um Kosten zu senken und Daten- und KI-Initiativen schneller umzusetzen.

Die Data Intelligence Platform verbindet generative KI und Lakehouse-Vereinheitlichung zu DatabricksIQ, einer Data-Intelligence-Engine, die die einzigartige Semantik Ihrer Unternehmensdaten versteht. DatabricksIQ analysiert automatisch alle Aspekte der Daten, einschließlich des Inhalts, der Metadaten und der Nutzungsmuster (z. B. Abfragen, Berichte und Herkunft). Durch diese umfassende Analyse lernt die Plattform kontinuierlich dazu, wird besser und integriert neue Funktionen. Das treibt die Optimierung von Datenverwaltung und KI-Anwendungen voran. Durch dieses umfassende Verständnis der Daten bietet die Data Intelligence Platform von Databricks viele Vorteile:

- Natürlichsprachlicher Datenzugriff: Mit Hilfe von KI-Modellen ermöglicht es die Data Intelligence Platform, in natürlicher Sprache mit Daten zu interagieren. Sie ist auf Jargon und Akronyme des jeweiligen Unternehmens zugeschnitten. Die Plattform analysiert die Datennutzung in vorhandenen Workloads, lernt die firmenspezifischen Termini und bietet eine individuell angepasste natürlichsprachliche Schnittstelle für alle Benutzer – vom Laien bis zum Data Engineer.
- Semantische Katalogisierung und Erkennung: Generative KI versteht Datenmodell, Metriken und KPIs eines jeden Unternehmens, bietet herausragende Erkennungsfunktionen oder deckt automatisch Diskrepanzen bei der Datennutzung auf.
- Automatisierte Verwaltung und Optimierung: KI-Modelle optimieren Datenlayout, Partitionierung und Indizierung basierend auf der Datennutzung, sodass der Bedarf an manueller Feinabstimmung und Parameterkonfiguration sinkt.

- Verbesserte Governance und besserer Datenschutz: Data-Intelligence-Plattformen erkennen, klassifizieren und verhindern automatisch Missbrauch sensibler Daten und vereinfachen die Verwaltung mithilfe natürlicher Sprache.

- Erstklassiger Support für KI-Workloads: Data-Intelligence-Plattformen optimieren jede Unternehmens-KI: Sie stellen eine Verknüpfung zu relevanten Geschäftsdaten her und nutzen die erlernte Semantik (Metriken, KPIs usw.), um korrekte Ergebnisse zu liefern. Entwickler von KI-Anwendungen-Intelligence müssen sich also nicht mehr durch sprödes Prompt-Engineering „hacken".



Die Databricks Platform vereinfacht auch die Entwicklung von KI-Anwendungen für Unternehmen. Die Entwicklung von KI-Anwendungen, die die Unternehmensdaten verstehen, wird damit zum Kinderspiel. Die Databricks Platform bietet zahlreiche Funktionen zur direkten Integration von Unternehmensdaten in KI-Systeme, darunter:

- Ende-zu-Ende-RAG (Retrieval Augmented Generation) zum Erstellen hochwertiger Dialogagenten auf der Grundlage Ihrer eigenen Daten
- Training kundenspezifischer Modelle entweder von Grund auf mit den Daten eines Unternehmens oder durch kontinuierliches Pre-Training bestehender Modelle wie MPT und Llama 2, um KI-Anwendungen durch ein intensives Verständnis eines Zielbereichs weiter zu verbessern
- Effiziente und sichere serverlose Inferenz Ihrer Unternehmensdaten, mit einheitlicher Governance und Qualitätsüberwachung
- Ende-zu-Ende-MLOps auf Basis des beliebten Open-Source-Projekts MLflow, wobei alle erzeugten Daten automatisch im Lakehouse verarbeitet, erfasst und kontrolliert werden können

Dieser praktische Leitfaden zeigt bewährte Data-Warehousing-Verfahren auf der Databricks Data Intelligence Platform anhand von End-to-End-Anwendungsfällen aus der Praxis. Erfahren Sie, wie die Plattform Unternehmen jeder Größe unterstützt, Rohdaten mittels SQL in nutzbare Informationen umzuwandeln – von Erfassung über Verarbeitung, KI und LLMs, bis zu Analyse und BI. Wir geben Ihnen Referenzarchitekturen und Code-Beispiele an die Hand, damit Sie alle Aspekte des Datenlebenszyklus auf der Databricks Intelligence Platform kennenlernen.

Weitere Informationen zu Databricks SQL finden Sie in unserem E-Book Warum das Data Lakehouse Ihr nächstes Data Warehouse sein wird.

KAPITEL

02

Data Warehousing mit Intelligence und Automatisierung

databricks

Der Erfolg von Lakehouse-basierten Daten- und KI-Initiativen hängt von einer vereinfachten Datenarchitektur, Datenqualität und Governance sowie von der Skalierbarkeit und der Performance ab. Diese Säulen sind das Fundament, auf dem Unternehmen ihre Datenstrategien aufbauen, die sie durch die Komplexität der modernen Datenverwaltung und -analyse navigieren.

Vereinfachte Datenarchitektur

Die Data-Lakehouse-Architektur löst die mit Datensilos einhergehenden Probleme und bietet gleichzeitig die Funktionen eines Data Warehouse. Am Anfang steht dabei das Zusammenführen der Daten in einem cloudbasierten Data Lake. Diese Basis, die von Delta Lake unterstützt wird, ermöglicht es, Analysen und KI-Anwendungsfälle auf einer einzigen Datenquelle zu betreiben. Dies senkt die Speicherkosten und rationalisiert das Data Engineering. Die Lakehouse-Architektur integriert eine einheitliche Governance- und Sicherheitsstruktur mit Unity Catalog, gewährleistet differenzierte Datenkontrolle und frühzeitigen Zugang für die jeweiligen Teams.

Der ganzheitliche Ansatz der Lakehouse-Architektur umfasst den gesamten Datenlebenszyklus, die Transformation und die Auswirkungen auf verschiedene Analyse- und KI-Workloads. Da alle Workloads dieselben Daten nutzen und dabei dieselben Sicherheits- und Governance-Richtlinien befolgen, können Sie sich auf die Richtigkeit der Daten verlassen. Funktionale Silos werden abgebaut, was den Weg für nahtlose Zusammenarbeit und folglich höhere Produktivität bei der Bereitstellung von Datenprodukten ebnet.

Nachfolgend aufgeführt sind einige weitere Vorteile einer vereinfachten Datenarchitektur:

- Ein auf Open Source und Standards basierendes Lakehouse beseitigt Datensilos, vereinfacht die Datenlandschaft und erleichtert den Daten- und KI-Betrieb.
- Der Einsatz einer einzigen Plattform fördert Integration, Speicherung, Verarbeitung, Governance, Weitergabe, Analyse und KI. Sie vermittelt einen einheitlichen Ansatz für den Umgang mit strukturierten und unstrukturierten Daten, eine vollständige Sicht auf Datenherkunft und -provenienz sowie ein konsolidiertes Toolset für Python und SQL, Notebooks, IDEs, Batch, Streaming und alle wichtigen Cloud-Anbieter.
- Die automatische Verbesserung von Performance und Speicherplatz sorgt für optimale TCO und setzt Leistungsmaßstäbe für Data Warehousing und KI. Dies schließt moderne Prozesse wie Large Language Models (LLMs) ein.

Data Governance und Datenqualität

Daten, die für Unternehmen von grundlegender Bedeutung sind, erfordern insbesondere angesichts der zunehmenden Menge und Vielfalt konsequenten Qualitätserhalt und Governance. Unternehmen, die das Potenzial ihres Lakehouse voll ausschöpfen wollen, müssen Fehlerfreiheit, Zuverlässigkeit und Compliance priorisieren. Eine unzureichende Datenqualität kann Erkenntnisse verfälschen, eine schwache Governance zu Regelungs- und Sicherheitslücken führen. Die Databricks Data Intelligence Platform mit ihrem Unity Catalog löst diese Probleme: Sie bietet ein integriertes Gerüst für die Verwaltung und Verbesserung der Datenqualität während des gesamten Lebenszyklus. Die Governance auf der Lakehouse-Architektur bietet Ihnen folgende Möglichkeiten:

- Entdecken, klassifizieren und konsolidieren Sie Daten und KI-Assets von verschiedenen Plattformen in einer beliebigen Cloud und verbessern Sie die Datenexploration und Gewinnung von Erkenntnissen mithilfe natürlicher Sprache – alles von einem zentralen Zugangspunkt aus.
- Vereinfachen Sie die Zugriffsverwaltung über eine einheitliche Oberfläche: Sorgen Sie cloud- und plattformübergreifend für einen konsistenten, sicheren Zugriff, mit verbesserter, fein abgestufter Kontrolle und skalierbaren Low-Code-Richtlinien.
- Automatisieren Sie mit KI das Monitoring von Daten und ML-Modellen, erhalten Sie proaktive Warnungen, optimieren Sie das Debugging und erreichen Sie durch integrierte Systemtabellen eine umfassende Überwachung Ihrer Lakehouse-Prozesse.
- Teilen Sie Daten und KI-Assets über Clouds, Regionen und Plattformen effizient über die Open-Source-Lösung Delta Sharing im Unity Catalog – für sichere Kooperation und Mehrwert ohne komplexe Abläufe oder teure Replikationen.

Skalierbarkeit und Performance

Angesichts wachsender Datenvolumina verteilt eine Lakehouse-Architektur die Rechenfunktionen unabhängig vom Speicherplatz und kann so eine konsistente Leistung bei optimalen Kosten gewährleisten. Die Data Intelligence Platform von Databricks ist auf Elastizität ausgelegt, sodass Unternehmen ihre Datenoperationen nach Bedarf skalieren können. Die Skalierbarkeit erstreckt sich dabei über verschiedene Dimensionen:

SERVERLOS

Die Databricks Platform nutzt serverlose cloudbasierte Computing-Ressourcen, wodurch Workloads auf der Basis der erforderlichen Rechenkapazität elastisch angepasst und skaliert werden können. Eine solch dynamische Ressourcenzuweisung garantiert selbst bei Höchstlast eine zügige Datenverarbeitung und -analyse.

NEBENLÄUFIGKEIT

Die Databricks Platform nutzt serverlose Berechnungen und KI-gesteuerte Optimierungen, um Datenverarbeitung und Abfragen simultan durchzuführen. So können auch mehrere Benutzer und Teams ohne Leistungseinbußen gleichzeitig Analyseaufgaben durchführen.

SPEICHER

Die Plattform integriert nahtlos in Data Lakes und ermöglicht so die kostengünstige Speicherung auch umfangreichster Datenmengen bei gleichzeitiger Gewährleistung von Datenverfügbarkeit und -zuverlässigkeit. Außerdem optimiert sie zur Performance-Steigerung die Datenspeicherung und senkt dadurch die Speicherkosten.

Skalierbarkeit ist zwar unverzichtbar, wird aber durch Leistung ergänzt. In dieser Hinsicht sticht die Databricks Data Intelligence Platform hervor, da sie eine Vielzahl KI-gestützter Optimierungen bietet:

OPTIMIERTE ABFRAGEVERARBEITUNG

Die Plattform nutzt Optimierungsverfahren für maschinelles Lernen, um die Ausführung von Abfragen zu beschleunigen. Sie verwendet eine automatische Indizierung, Caching und Predicate Pushdown, um sicherzustellen, dass Abfragen effizient verarbeitet werden, was zu schnellen Ergebnissen führt.

AUTOMATISCHE SKALIERUNG

Die Databricks Platform skaliert serverlose Ressourcen intelligent, um sie an Ihre Workloads anzupassen. So ist sichergestellt, dass Sie nur für die Rechenleistung zahlen, die Sie auch tatsächlich nutzen – und das bei optimaler Abfrageleistung.

PHOTON

Die neue native MPP-Engine (Massively Parallel Processing) auf der Databricks Platform ermöglicht eine extrem schnelle Abfrageperformance zu niedrigen Kosten – ob Datenerfassung, ETL, Streaming, Data Science oder interaktive Abfragen – direkt in Ihrem Data Lake. Photon ist kompatibel mit Apache Spark™-APIs: Weder sind Änderungen am Code erforderlich, noch binden Sie sich an einen einzigen Anbieter.

DELTA LAKE

Delta Lake mit Unity Catalog und Photon bietet standardmäßig das beste Preis-Leistungs-Verhältnis ohne manuelles Tuning. Die Databricks Platform setzt KI-Modelle ein, um typische Datenspeicherprobleme zu bewältigen. Sie profitieren von schnellerer Leistung, ohne Tabellen manuell verwalten zu müssen – auch dann, wenn sie sich im Laufe der Zeit ändern.

- Predictive I/O für Aktualisierungen optimiert Ihre Abfragepläne und das Datenlayout für Spitzenleistungen und sorgt für ein ausgewogenes Verhältnis zwischen Lese- und Schreibleistung. So können Sie mehr aus Ihren Daten herausholen, ohne sich auf Strategien wie Copy-on-Write oder Merge-on-Read festlegen zu müssen.
- Liquid Clustering bietet die Leistung einer optimal abgestimmten und wohlpartitionierten Tabelle ohne die herkömmlichen Probleme, die mit der Partitionierung einhergehen, wie z. B. die Frage der Partitionierbarkeit von Spalten mit hoher Kardinalität oder teure Rewrites, wenn Sie Partitionsspalten ändern. Das Ergebnis sind rasend schnelle, gut geclusterte Tabellen mit minimaler Konfiguration.
- Predictive Optimization sorgt automatisch für ein optimales Preis-Leistungs-Verhältnis Ihrer Daten. Sie lernt aus Ihren Datennutzungsmustern, plant die erforderlichen Optimierungen und führt diese dann auf einer hyperoptimierten serverlosen Infrastruktur aus.

Nachdem das Fundament der Lakehouse-Architektur aufgebaut wurde, bietet es sich an, die Bereitstellung von Data-Warehouse- und Analysefunktionen auf Databricks mit geeigneten Datenstrukturen und Verwaltungsfunktionen zu untersuchen. Diese werden mithilfe von Databricks SQL (DB SQL) realisiert.

Databricks SQL Serverless: Das beste Data Warehouse für eine Lakehouse-Architektur

Databricks SQL wurde entwickelt, um die Data-Warehousing-Funktionen zu verbessern und eine erstklassige SQL-Unterstützung auf der Databricks Data Intelligence Platform zu bieten. Hiermit werden SQL-basierte Datentransformationen, -explorationen und -analysen für Benutzer mit unterschiedlicher technischer Vorbildung optimiert. Ob BI-Analysten, Datenarchitekten oder Analytics Engineers: Die intuitive SQL-Oberfläche erleichtert Abfragen und komplexe Datenoperationen, ohne dass dafür eine besondere Programmierung erforderlich ist. Dieser erweiterte Datenzugang fördert eine unternehmenszentrierte Kultur und befähigt mehr Teams, auf Basis datengestützter Einsichten Entscheidungen zu treffen.

Databricks SQL zeichnet sich durch schnelle und effiziente Verarbeitung auch riesiger Datenmengen aus. Der Einsatz von Photon, der Next-Gen-Engine von Databricks mit KI-gesteuerten Optimierungen, sorgt für eine zügige Datenverarbeitung und -analyse und verkürzt vor allem die Ausführungsdauer von Abfragen. Eine hohe Leistung ist für Unternehmen mit Herausforderungen im Datenbereich entscheidend, um Erkenntnisse aus einer enormen Zahl von Datasets zu gewinnen. Darüber hinaus fördert Databricks SQL die Zusammenarbeit und bietet hierzu einen Arbeitsbereich, in dem Datenprofis Abfragen, Ergebnisse und Erkenntnisse direkt austauschen können. Diese gemeinsame Umgebung fördert den Wissensaustausch und beschleunigt die Lösungsfindung. So können Unternehmen von der kollektiven Intelligenz ihrer Teams profitieren.

Zudem bietet Databricks SQL fortschrittliche Funktionen für Data Governance, Sicherheit und Compliance, die es Unternehmen erlauben, die Datenqualität zu sichern, Zugriffsbeschränkungen einzurichten, Datenaktivitäten zu überwachen, sensible Daten zu schützen und rechtliche Vorgaben einzuhalten. Zusammenfassend bietet Databricks SQL Folgendes:

- Kürzere Time-to-Insights
  Verwenden Sie einfaches Englisch, um Daten abzurufen. Die entsprechenden SQL-Abfragen werden dann automatisch für Sie erstellt. So können Abfragen schneller verfeinert und allen Unternehmensangehörigen zugänglich gemacht werden.
- Bestes Preis-Leistungs-Verhältnis
  Serverloses Computing in Kombination mit KI-optimierter Verarbeitung führt zu erstklassiger Leistung und Skalierung bei geringeren Kosten – ganz ohne Verwaltung einer Cloud-Infrastruktur.
- Unified Governance
  Etablieren Sie eine einheitliche Governance-Ebene für alle Daten und KI-Assets, unabhängig vom Speicherort.
- Weniger Komplexität
  Führen Sie alle Daten, Analysen und die gesamte KI auf einer Plattform zusammen, die SQL und Python, Notebooks und IDEs, Batch und Streaming sowie alle großen Cloud-Anbieter unterstützt.
- Umfassendes Ökosystem
  Nutzen Sie SQL und beliebte Tools wie Power BI, Tableau, dbt und Fivetran in Kombination mit Databricks für BI, Datenerfassung und Transformation.

Fazit

Die Databricks Data Intelligence Platform stellt einen bedeutenden Fortschritt im Bereich Data Warehousing und Analytics dar. Sie adressiert den dringenden Bedarf an effizienten Data-Warehousing-Workloads in der heutigen datengesteuerten Unternehmenslandschaft. Die vereinfachte Datenarchitektur der Plattform zentralisiert Daten mit vollständigen End-to-End-Data-Warehouse-Funktionen. Das spart Kosten, beschleunigt die umfassende Umwandlung von Rohdaten in verwertbare Erkenntnisse und vereinheitlicht Batch und Streaming. Unity Catalog stellt darüber hinaus die Datenqualität und Governance sicher. Unternehmen können damit alle ihre Daten cloudübergreifend mit differenzierter Governance und Datenherkunft auffinden, schützen und verwalten.

Skalierbarkeit und Performance sind die wichtigsten Unterscheidungsmerkmale der Databricks Platform. Sie bietet serverloses Computing, Unterstützung für Nebenläufigkeit und optimierte Speicherstrategien. KI-gesteuerte Optimierungen verbessern die Abfrageverarbeitung, steigern die Leistung und optimieren die Daten für die bestmögliche Performance und kleines Geld. So wird die Datenanalyse schneller und kostengünstiger.

Databricks SQL erweitert die Möglichkeiten der Plattform durch erstklassigen SQL-Support, der die Transformation und Analyse von Daten für viele Benutzer noch leichter macht. Es fördert die Zusammenarbeit, die Data Governance und ein reichhaltiges Tool-Ökosystem, um Datensilos aufzubrechen und Ihr Unternehmen zu befähigen, das Potenzial Ihrer Daten vollständig auszuschöpfen. Werfen wir nun einen Blick auf einige Anwendungsfälle für die Ausführung Ihrer Data-Warehousing- und BI-Workloads auf der Databricks Platform.

WEITERE INFORMATIONEN

- → Warum das Data Lakehouse Ihr nächstes Data Warehouse sein wird. 2. Auflage
- → Was gibt es Neues bei Databricks SQL?
- → Einführung der Lakehouse Federation-Funktionen in Unity Catalog
- → Einführung in KI-Funktionen: Integration von Large Language Models in Databricks SQL

databricks

KAPITEL

03

Best Practices zu Data Warehousing im Lakehouse

KAPITEL 3.1

Techniken der Data-Warehousing-Modellierung und ihre Implementierung im Lakehouse

Einsatz von Data Vaults und Sternschemata im Lakehouse

von Soham Bhatt und Deepak Sekar

Das Lakehouse ist ein neues Paradigma für Datenplattformen, das die besten Eigenschaften von Data Lakes und Data Warehouses vereint. Es ist als groß angelegte Datenplattform für Unternehmen konzipiert, die eine Vielzahl von Anwendungsfällen und Datenprodukten unterstützt. Es kann als zentrales und einheitliches Unternehmensdaten-Repository für folgende Objekte in Ihrem Unternehmen fungieren:

- Datendomains
- Echtzeit-Streaming-Anwendungsfälle
- Data Marts
- verstreute Data Warehouses
- Data-Science-Feature-Stores und Data-Science-Sandboxen
- Self-Service-Analyse-Sandboxen auf Abteilungsebene

Angesichts der Vielfalt der Anwendungsfälle können in einem Lakehouse für verschiedene Projekte unterschiedlich Datenorganisationsprinzipien und Modellierungstechniken gelten. Technisch kann die Lakehouse-Architektur viele verschiedene Ausprägungen der Datenmodellierung unterstützen. In diesem Artikel erläutern wir die Umsetzung der Datenorganisationsprinzipien (Bronze, Silber und Gold) im Lakehouse und die Modellierungstechniken für die verschiedenen Schichten (oder Layer).

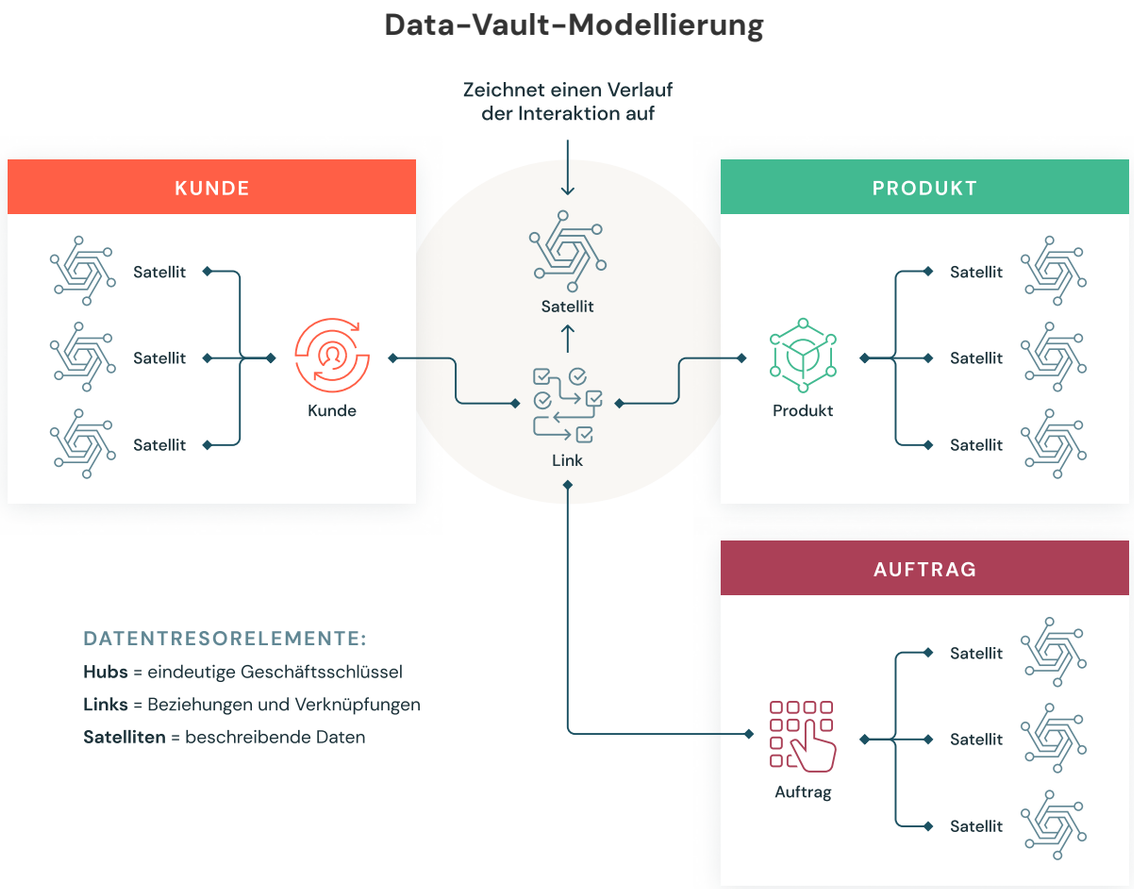
Was ist ein Data Vault?

Ein Data Vault ist ein neueres Muster für die Datenmodellierung, das zum Aufbau von Data Warehouses für Unternehmensanalysen verwendet wird und mit den Methoden von Kimball und Inmon vergleichbar ist.

Data Vaults ordnen Daten jeweils einem von drei Typen zu: Hubs, Links und Satelliten. Hubs stellen zentrale Geschäftseinheiten dar, Links stehen für Beziehungen zwischen Hubs, und Satelliten speichern Attribute zu Hubs oder Links.

Bei Data Vaults geht es vorrangig um die agile Data-Warehouse-Entwicklung: Im Vordergrund stehen Skalierbarkeit, Datenintegration/ETL und Entwicklungsgeschwindigkeit. Die meisten Kunden haben eine Landing-Zone (Anlandungszone), eine Vault-Zone und eine Data-Mart-Zone. Diese entsprechen den Databricks-Organisationsparadigmen der Bronze-, Silber- und Gold-Schichten. Der Modellierungsstil von Data Vault mit Hub-, Link- und Satellitentabellen passt sich in der Regel gut in die Silber-Schicht der Lakehouse-Architektur ein.

Weitere Informationen zur Data-Vault-Modellierung finden Sie bei der Data Vault Alliance.

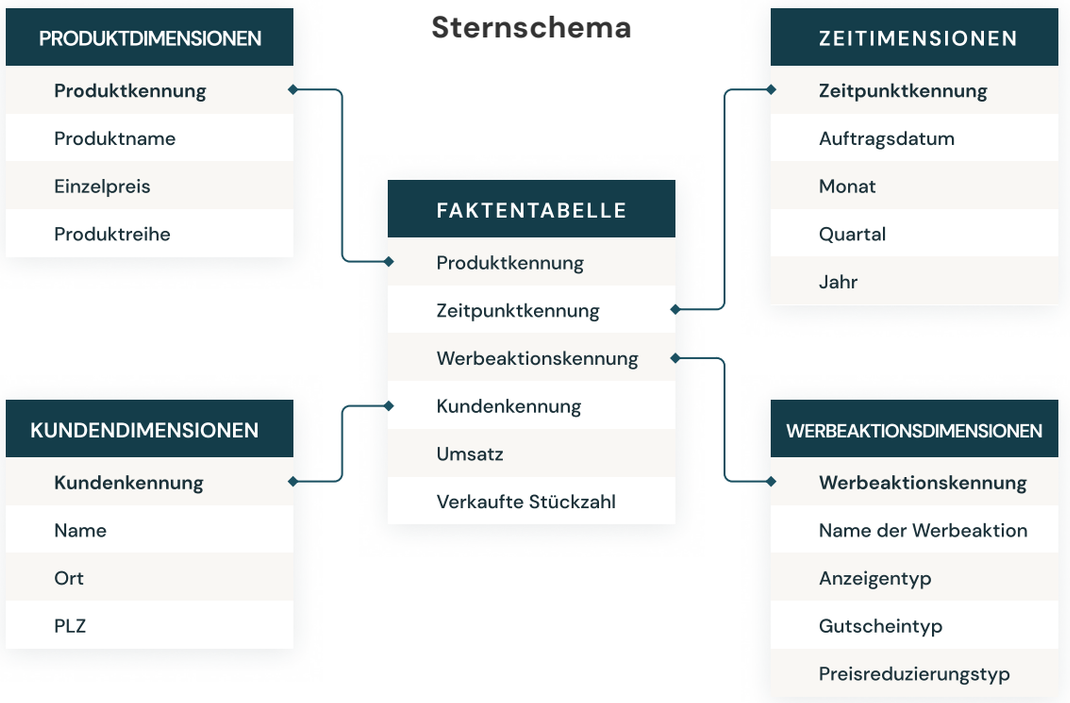


Data-Vault-Modellierung mit miteinander verbundenen Hubs, Links und Satelliten

Was versteht man unter dem Begriff Dimensionsmodellierung?

Die Dimensionsmodellierung ist ein Bottom-up-Ansatz zur Gestaltung von Data Warehouses, der auf die Optimierung für Analysen abzielt. Dimensionsmodelle dienen der Denormalisierung von Geschäftsdaten in Dimensionen (wie Zeit und Produkt) und Fakten (z. B. Transaktionen in Beträgen und Mengen) und verknüpfen verschiedene Themenbereiche über angepasste Dimensionen, um verschiedene Faktentabellen zu erstellen.

Die häufigste Form der Dimensionsmodellierung ist das Sternschema. Ein Sternschema ist ein mehrdimensionales Datenmodell, das Daten so organisiert, dass sie leicht verständlich und analysierbar sind und die Erstellung einfacher und intuitiver Berichte ermöglicht. Sternschemata oder Dimensionsmodelle im Kimball-Stil gelten als der Goldstandard für die Darstellungsschicht in Data Warehouses und Data Marts sowie für die Semantik- und Berichtsschicht. Das Sternschemadesign ist für die Abfrage großer Datenmengen optimiert.



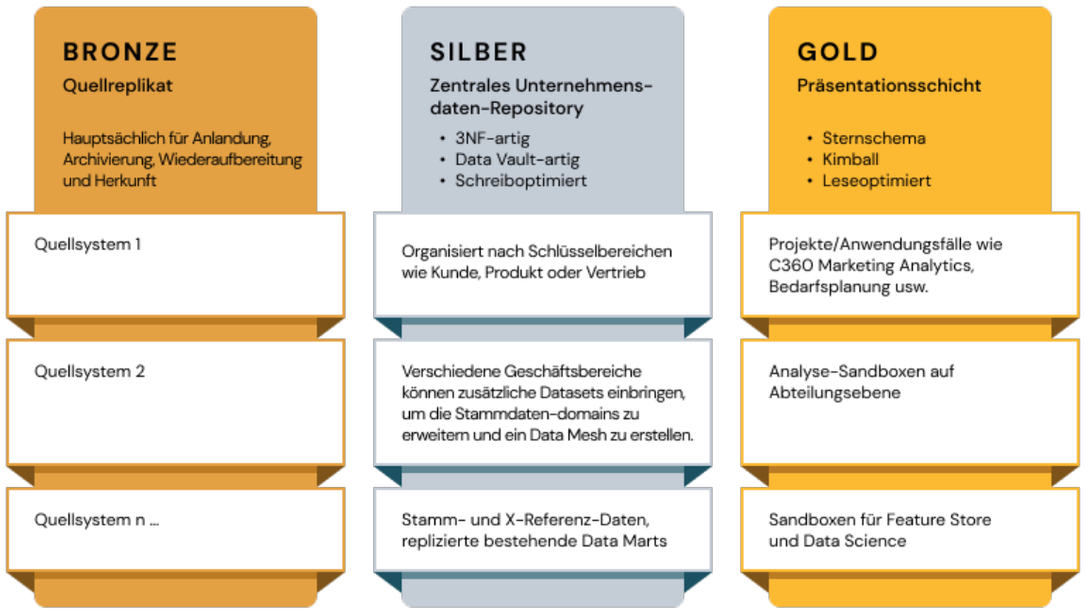
Beispiel für ein Sternschema

Sowohl normalisierte Data-Vault-Dimensionsmodelle (schreiboptimiert) als auch denormalisierte Dimensionsmodelle (leseoptimiert) haben ihren Platz auf der Databricks Data Intelligence Platform. Die Hubs und Satelliten des Data Vault in der Silber-Schicht werden zum Laden der Dimensionen im Sternschema verwendet, während die Linktabellen des Data Vault zentral für das Laden der Faktentabellen im Dimensionsmodell sind. Weitere Informationen zur Dimensionsmodellierung gibt es bei der Kimball Group.

Grundsätze der Datenorganisation in den einzelnen Schichten im Lakehouse

Ein modernes Lakehouse ist eine umfassende Datenplattform auf Unternehmensebene. Es ist hochgradig skalierbar, leistungsstark und eignet sich für sämtliche Anwendungsfälle (z. B. ETL, BI, Data Science und Streaming), die unterschiedliche Datenmodellierungsansätze erfordern können. Sehen wir uns an, wie ein typisches Lakehouse organisiert ist:

Die Data-Lakehouse-Architektur



Eigenschaften der Bronze-, Silber- und Gold-Schichten der Data-Lakehouse-Architektur

Bronze-Schicht: die Landing-Zone

In der Bronze-Schicht landen alle Daten aus den Quellsystemen. Die Tabellenstrukturen in dieser Schicht entsprechen dem As-Is-Zustand der Tabellenstrukturen des Quellsystems. Optional können Metadatenspalten hinzugefügt werden, um den Zeitpunkt des Ladens, die Prozess ID etc. zu erfassen. Die Schicht konzentriert sich auf Change Data Capture (CDC) und die Möglichkeit ein Verlaufsarchiv der Quelldaten (Cold Storage) sowie Infos zur Datenherkunft, Nachvollziehbarkeit und gegebenenfalls Wiederverarbeitung bereitzustellen, ohne dass die Daten erneut aus dem Quellsystem eingelesen werden müssen.

In den meisten Fällen empfiehlt es sich, die Daten in der Bronze-Schicht im Delta-Format zu halten, um die Leistungsfähigkeit späterer Lesevorgänge aus der Bronze-Schicht für ETL sicherzustellen. So können Sie Updates an Bronze-Daten vornehmen, um CDC-Änderungen zu schreiben. Manchmal, wenn Quelldaten im JSON- oder XML-Format eintreffen, sehen wir, dass Kunden sie im ursprünglichen Quelldatenformat anlanden und sie dann durch Konvertierung in das Delta-Format stagen. So kommt es vor, dass Kunden die logische Bronze-Schicht in eine physische Anlande- und Bereitstellungszone umwandeln.

Das Speichern von Rohdaten im ursprünglichen Quelldatenformat in einer Landing-Zone trägt auch zur Konsistenz bei, wenn Sie Daten mithilfe von Erfassungstools einlesen, die Delta nicht als native Senke unterstützen, oder wenn Quellsysteme Daten direkt in Objektspeichern ablegen. Dieses Muster passt auch gut zum Autoloader-Ingestion-Framework, bei dem Quellen die Daten in der Anlandungszone für Rohdateien abliefern und Databricks AutoLoader sie dann auf der Staging-Ebene in das Delta-Format konvertiert.

Silver-Schicht: das zentrale Unternehmens-Repository

In der Silver-Schicht der Lakehouse-Architektur werden die Daten aus der Bronze-Schicht abgeglichen, zusammengeführt, vereinheitlicht und (nach dem „Just-enough-Prinzip“) bereinigt. So kann die Silver-Schicht eine „Unternehmenssicht“ auf alle wichtigen Geschäftseinheiten, Konzepte und Transaktionen bieten. Das ist vergleichbar mit einem Enterprise ODS (Operational Data Store) oder dem Zentral-Repository oder den Datendomains eines Data Mesh (z. B. Stammkunden, Produkte, nicht-duplizierte Transaktionen und Querverweistabellen). Diese Unternehmenssicht führt die Daten aus verschiedenen Quellen zusammen und ermöglicht Self-Service-Analysen für Ad-hoc-Berichte, Advanced Analytics und ML. Zudem fungiert sie als Quelle, mit der abteilungsinterne Analysten, Data Engineers und Data Scientists weitere Datenprojekte und Analysen erstellen können, um geschäftliche Probleme mithilfe unternehmens- und abteilungsinterner Datenprojekte in der Gold-Schicht zu lösen.

Nach dem Data-Engineering-Paradigma des Lakehouse wird in der Regel die ELT-Methodik (Extract, Load, Transform) anstelle des traditionellen ETL-Prinzips (Extract, Transform, Load) befolgt. Beim ELT-Ansatz werden beim Laden der Silber-Schicht nur minimale oder „gerade ausreichende“ Transformationen und Datenbereinigungsregeln angewendet. Alle Regeln auf Unternehmensebene werden in der Silber-Schicht angewendet, die projektspezifischen Transformationsregeln hingegen in der Gold-Schicht. Hier liegt der Schwerpunkt auf Geschwindigkeit und Agilität bei der Erfassung und Bereitstellung der Daten im Lakehouse.

Aus Sicht der Datenmodellierung umfasst die Silver-Schicht mehr 3NF-ähnliche Datenmodelle. In dieser Schicht können Data Vault-ähnliche, für das Schreiben optimierte Datenarchitekturen und Datenmodelle verwendet werden. Wenn Sie eine Data-Vault-Methode verwenden, werden sowohl der Raw Data Vault als auch der Business Vault in die logische Silver-Schicht des Datenpools eingeordnet. Die PIT-Präsentationsansichten (Point-In-Time) oder Materialized Views werden dann in der Gold-Schicht dargestellt.

Gold-Schicht: die Darstellungsschicht

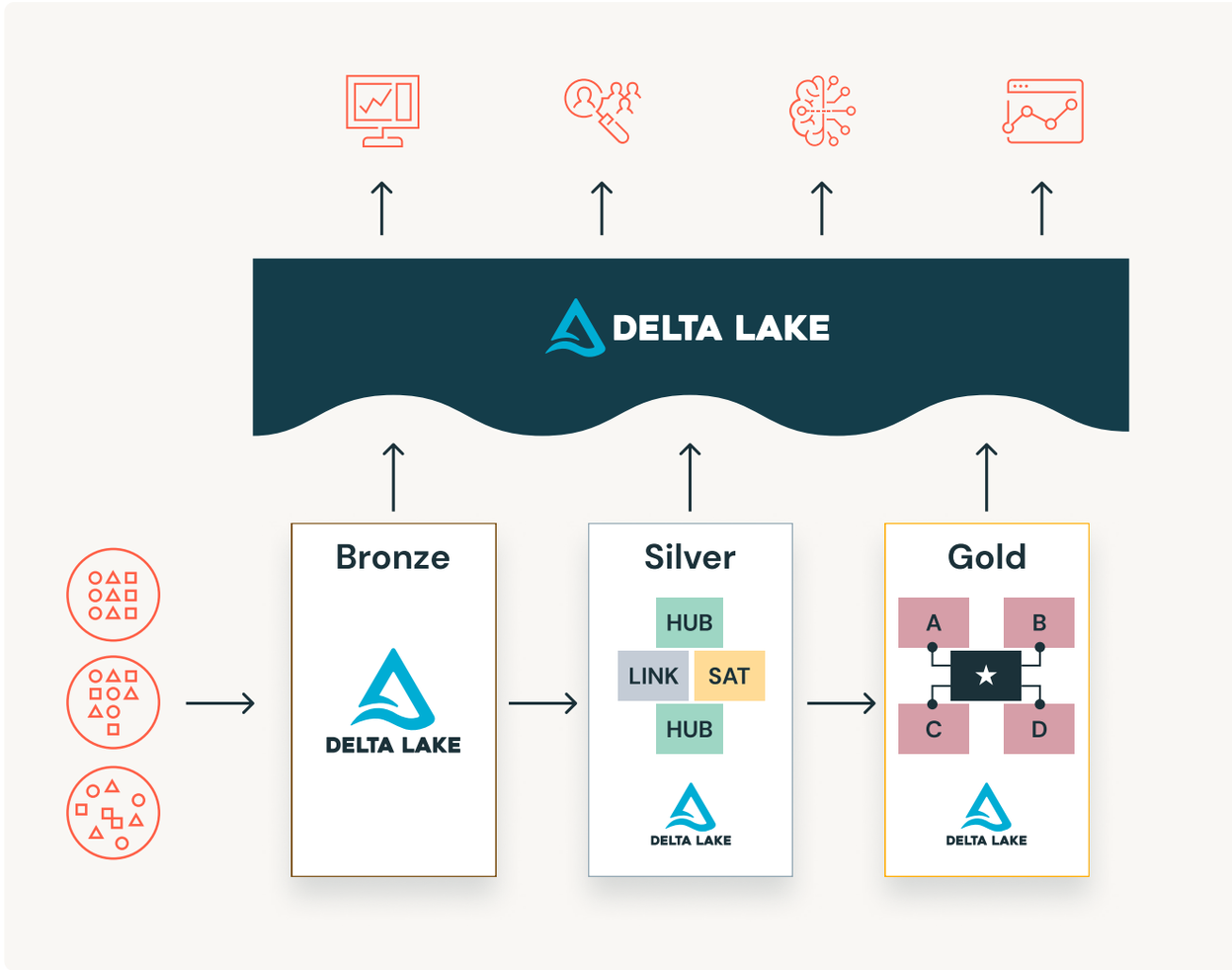
In der Gold-Schicht können mehrere Data Marts oder Warehouses gemäß den Prinzipien der Dimensionsmodellierung/Kimball-Methodik erstellt werden. Wie bereits erläutert, dient die Gold-Schicht der Berichterstattung und verwendet im Vergleich zur Silber-Schicht eher denormalisierte und leseoptimierte Datenmodelle mit weniger Joins. Manchmal können Tabellen in der Gold-Schicht vollständig denormalisiert werden. Typischerweise wird dies von Data Scientists so gewünscht, um die Algorithmen für das Feature Engineering zu bedienen.

ETL- und Datenqualitätsregeln, die „projektspezifisch“ sind, werden bei der Transformation von Daten aus der Silber- in die Gold-Schicht angewendet. Finale Präsentationsschichten wie Data Warehouses, Data Marts oder Datenprodukte wie Kundenanalysen, Produkt- und Qualitätsanalysen, Bestandsanalysen, Kundensegmentierung, Produktempfehlungen, Marketing- und Vertriebsanalysen usw. werden in dieser Schicht bereitgestellt. Sternschemabasierte Datenmodelle im Stil von Kimball oder Data Marts nach den Inmon-Prinzipien sind dieser Gold-Schicht des Lakehouse zuzuordnen. Gleiches gilt für Data-Science-Laboratorien und für Self-Service-Analysen implementierte Sandboxen auf Abteilungsebene.

Das Datenorganisationsparadigma beim Lakehouse

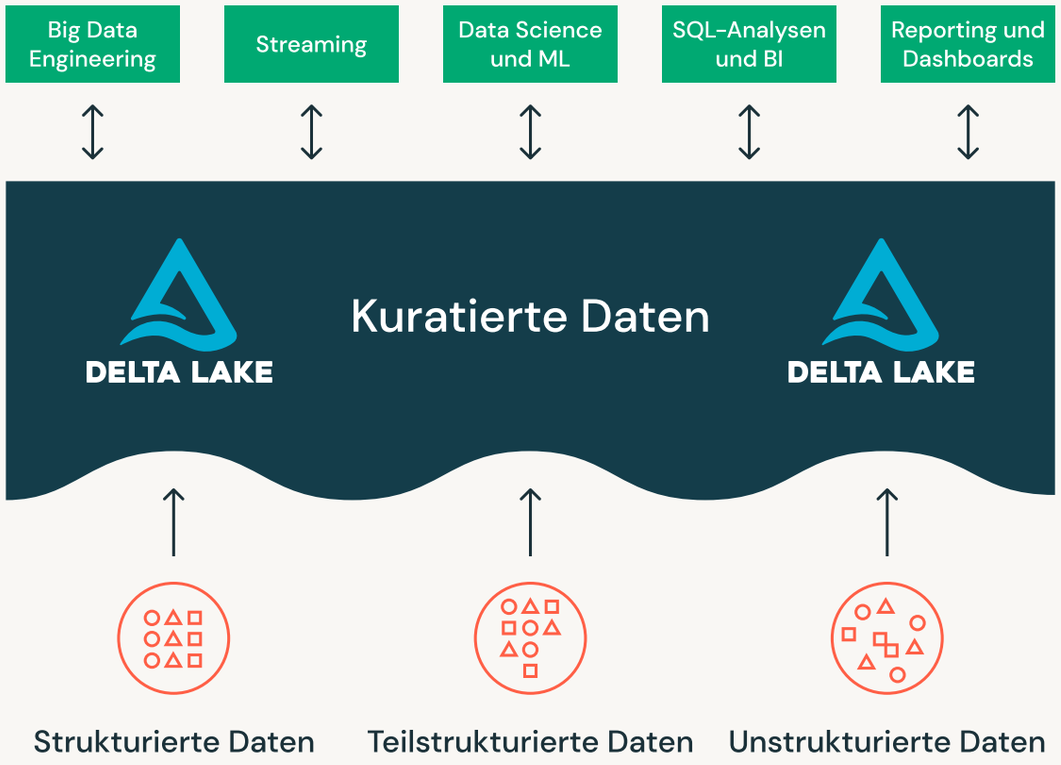
Zusammenfassend lässt sich sagen: Daten werden kuratiert, während sie die verschiedenen Schichten einer Data-Lakehouse-Architektur durchlaufen.

- Die Bronze-Schicht verwendet die Datenmodelle der Quellsysteme. Wenn die Daten in den Rohformaten hier anlanden, werden sie in das Delta Lake-Format konvertiert.
- Die Silver-Schicht führt die Daten aus verschiedenen Quellen erstmals zusammen und passt sie an, um eine Unternehmenssicht der Daten zu erstellen. Dabei kommen in der Regel stärker normalisierte, schreiboptimierte Datenmodelle zum Einsatz, die meist 3NF oder Data Vault ähneln.
- Die Gold-Schicht ist die Darstellungsschicht. Hier gibt es im Vergleich zur Silber-Schicht stärker denormalisierte oder reduzierte Datenmodelle, die meist Dimensionsmodelle im Kimball-Stil oder Sternschemata verwenden. Die Gold-Schicht beherbergt auch Sandboxes für Abteilungen und Data Science, um unternehmensweit Self-Service-Analysen und Data Science zu ermöglichen. Diese Sandboxen und ihre separaten Rechencluster verhindern, dass Geschäftsteams eigene Datenkopien außerhalb des Lakehouse erstellen.



Der Lakehouse-Datenorganisationsansatz bricht Datensilos auf, fördert die Teamarbeit und ermöglicht den Einsatz von ETL, Streaming, BI und KI auf einer zentralen Plattform mit angemessener Governance. Zentrale Datenteams agieren als Innovationsmotoren im Unternehmen und beschleunigen das Onboarding neuer Self-Service-Nutzer sowie die parallele Entwicklung zahlreicher Datenprojekte beschleunigen, damit der Datenmodellierungsprozess nicht zum Flaschenhals wird. Der Databricks Unity Catalog bietet Funktionen für Suche und Ermittlung, Governance und Herkunft im Lakehouse, um eine ordnungsgemäße Data-Governance-Abfolge zu gewährleisten.

Jetzt Data Vaults und Data Warehouses mit Sternschema entwerfen – mit Databricks SQL. →



Kuratierung von Daten auf ihrem Weg durch die verschiedenen Schichten der Lakehouse-Architektur

WEITERE INFORMATIONEN

- → Fünf einfache Schritte zur Implementierung eines Sternschemas in Databricks mit Delta Lake
- → Best Practices zur Implementierung eines Data-Vault-Modells in Databricks Lakehouse
- → Best Practices zur Dimensionsmodellierung und -implementierung im modernen Lakehouse
- → Jetzt auch in einem Lakehouse in Ihrer Nähe: Identitätsspalten zur Generierung von Surrogatschlüsseln!
- → Laden eines EDW-Dimensionsmodells in Echtzeit mit Databricks Lakehouse

KAPITEL 3.2

Best Practices zur Dimensionsmodellierung und -implementierung in einer modernen Lakehouse-Architektur

von Leo Mao, Abhishek Dey, Justin Breese und Soham Bhatt

Viele unserer Kunden migrieren ihre alten Data Warehouses auf Databricks Lakehouse, da sie damit nicht nur ihr Data Warehouse modernisieren, sondern auch direkt Zugang zu einer ausgereiften Streaming- und Advanced-Analytics-Plattform erhalten. Lakehouse kann das alles; es eine zentrale Plattform für Ihre gesamten Streaming-, ETL-, BI- und KI-Anforderungen. Es ermöglicht Geschäfts- und Datenteams zudem, auf einer gemeinsamen Plattform zusammenzuarbeiten. In der Praxis erleben wir, dass viele Kunden nach Best Practices für die richtige Datenmodellierung und die Implementierung physischer Datenmodelle in Databricks suchen.

In diesem Artikel tauchen wir deshalb tiefer in die Best Practices der Dimensionsmodellierung auf der Databricks Data Intelligence Platform ein und liefern ein Livebeispiel für die Implementierung eines physischen Datenmodells unter Verwendung unserer Best Practices für die Tabellenerstellung und DDL.

In diesem Artikel gehen wir auf folgende Kernthemen ein:

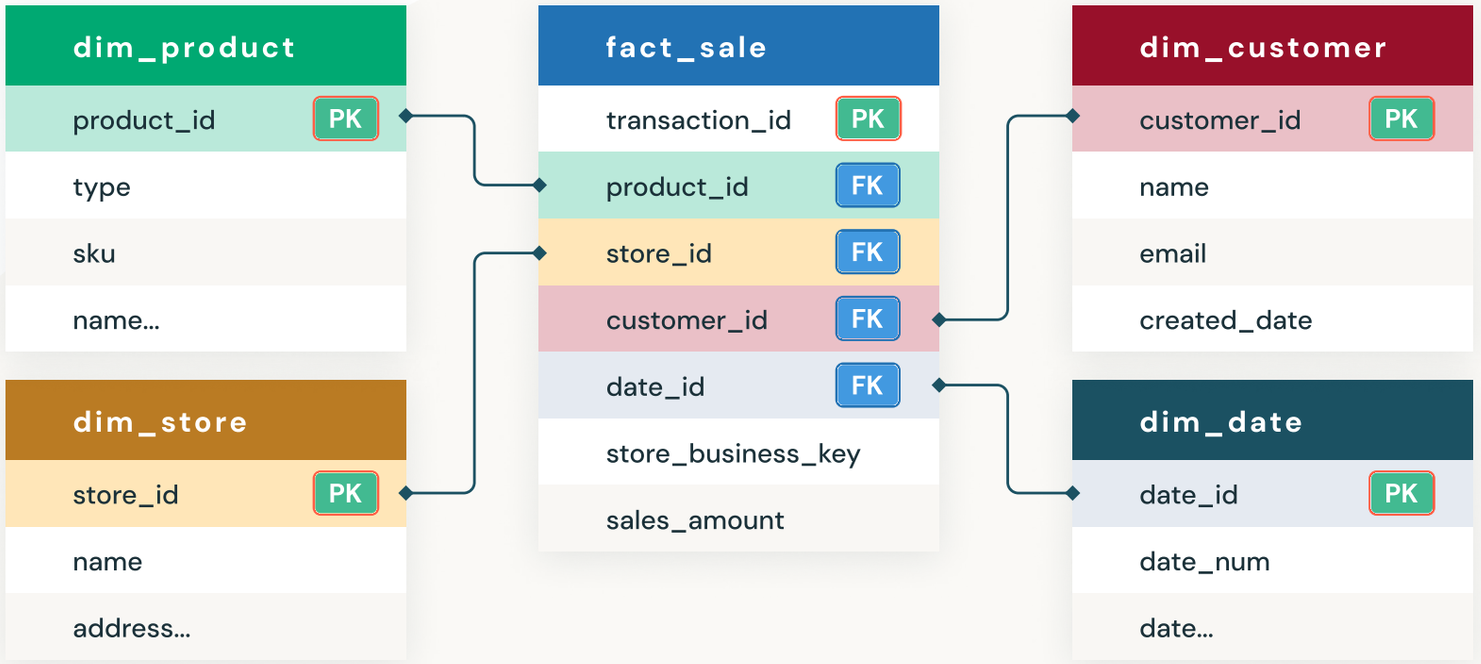
- Die Bedeutung der Datenmodellierung
- Gängige Datenmodellierungstechniken
- DDL-Implementierung der Data-Warehouse-Modellierung
- Best Practices und Empfehlungen für die Datenmodellierung im Lakehouse

Die Bedeutung der Datenmodellierung für Data Warehouse

Datenmodelle stehen beim Aufbau eines Data Warehouse im Mittelpunkt. Üblicherweise beginnt der Prozess mit der Definition des semantischen Geschäftsinformationsmodells, gefolgt von einem logischen Datenmodell und schließlich dem physischen Datenmodell (PDM). Los geht es mit einer umfassenden Analyse- und Entwurfsphase: Zuerst werden ein Geschäftsinformationsmodell und Prozessabläufe erstellt. Dabei erfasst man die wesentlichen Geschäftseinheiten und -attribute sowie ihre Interaktionen gemäß den Unternehmensprozessen. Anschließend wird das logische Datenmodell erstellt. Es zeigt die Beziehungen zwischen den Einheiten auf. Dabei bleiben die verwendeten Technologien unberücksichtigt. Am Ende entwickelt man ein PDM, das sich nach der eingesetzten Technologie richtet und effiziente Schreib- und Leseoperationen sicherstellt. Wie wir alle wissen, sind beim Data Warehousing Analytics-freundliche Modellierungsstile wie Sternschema und Data Vault sehr beliebt.

Best Practices für die Erstellung eines physischen Datenmodells in Databricks

Ziel der Datenmodellierung, ausgehend vom Geschäftsproblem, ist eine einfache Darstellung der Daten, die Wiederverwendbarkeit, Flexibilität und Skalierbarkeit gewährleistet. Hier sehen Sie ein typisches Datenmodell nach dem Sternschema. Es umfasst eine Faktentabelle „Sales“, die alle Transaktionen beinhaltet, und diverse Dimensionstabellen für Kunden, Produkte, Filialen, Datum etc., nach denen Sie die Daten aufschlüsseln. Die Dimensionen lassen sich per Join mit der Faktentabelle verknüpfen, um spezifische Geschäftsfragen zu klären, wie etwa die Beliebtheit von Produkten in einem bestimmten Monat oder die umsatzstärksten Filialen in einem Quartal. Wir wollen uns einmal ansehen, wie das in Databricks implementiert werden kann.



Dimensionsmodell in der Lakehouse-Architektur

Beachten Sie, dass jede Dimensionstabelle die Spalten **__START_AT** und **__END_AT** enthält, um SCD-Type-2 zu unterstützen. Aus Platzgründen werden diese hier nicht gezeigt.

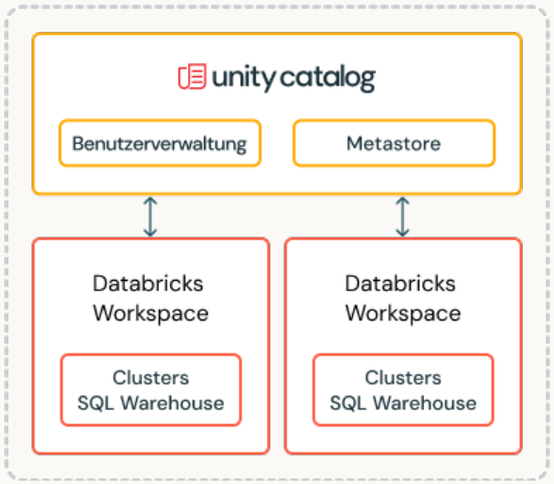
DDL-Implementierung der Data-Warehouse-Modellierung

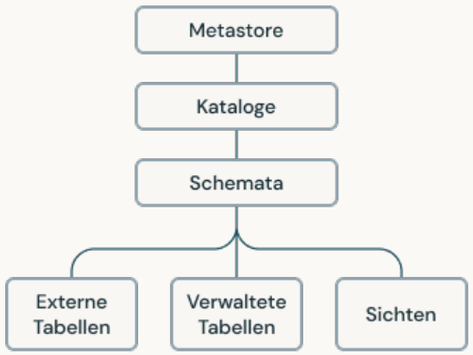
In den nächsten Abschnitten werden wir Folgendes anhand von Beispielen demonstrieren.

- Erstellen von Katalog, Datenbank und Tabelle auf drei Ebenen
- Definitionen für Primär- und Fremdschlüssel
- Identitätsspalten für Surrogatschlüssel
- Spalten-Constraints für Datenqualität
- Indizieren, Optimieren und Analysieren
- Fortgeschrittene Techniken

1. Unity Catalog: Namespace mit drei Ebenen

Unity Catalog ist bei Databricks eine Governance-Schicht, die es es Administratoren und Data Stewards erlaubt, Benutzerzugriffe über einen Metastore zentral für alle Arbeitsbereiche eines Databricks-Kontos zu steuern. Benutzer aus verschiedenen Arbeitsbereichen können auf die gleichen Daten zugreifen, jedoch abhängig von den im Unity Catalog zentral vergebenen Berechtigungen. Unity Catalog organisiert Ihre Daten in einem Namespace mit drei Ebenen: catalog.schema(database).table. Weitere Informationen über Unity Catalog finden Sie hier.





Hier erfahren Sie, wie wir den Katalog und das Schema einrichten, bevor wir Tabellen in der Datenbank erstellen. In unserem Beispiel erstellen wir nachfolgend einen Katalog US_Stores und ein Schema (d. h. eine Datenbank) Sales_DW. Wie wir diese nutzen, wird im weiteren Verlauf dieses Abschnitts erläutert.

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

Einrichten von Katalog und Datenbank

Hier sehen wir ein Beispiel dafür, wie die Tabelle fact_sales bei einem Namespace mit drei Ebenen abgefragt wird.

```
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.

Beispiel für Tabellenabfrage mit catalog.database.tablename

2. Definitionen für Primär- und Fremdschlüssel

Primär- und Fremdschlüsseldefinitionen sind beim Erstellen eines Datenmodells sehr wichtig. Die Möglichkeit, die Primär- und Fremdschlüsseldefinition zu unterstützen, macht die Definition des Datenmodells in Databricks zum reinsten Vergnügen. Außerdem hilft dies Analysten, die Join-Beziehungen in Databricks SQL Warehouse schnell zu durchschauen, um effektiv Abfragen schreiben zu können. Wie bei den meisten anderen MPP- (Massively-Parallel-Processing-), EDW- und Cloud Data Warehouses haben die Beschränkungen für Primär- und Fremdschlüssel nur informativen Charakter. Databricks unterstützt die Erzwingung der Primär- und Fremdschlüsselbeziehung nicht, bietet aber die Möglichkeit, diese zu definieren, um die Erstellung eines semantischen Datenmodells zu vereinfachen.

Hier sehen Sie ein Beispiel für das Erstellen der Tabelle dim_store mit der Identitätsspalte store_id, die zudem als Primärschlüssel definiert ist.

```sql
-- Store dimension
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
);
```

DDL-Implementierung zum Erstellen der store-Dimension mit Primärschlüsseldefinitionen

Nach dem Erstellen der Tabelle sehen wir, dass der Primärschlüssel (store_id) als Constraint in der Tabellendefinition unten erstellt wurde.

Describe Dim Store table information

```sql
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.

Primärschlüssel store_id erscheint als Tabellen-Constraint

Hier sehen wir ein Beispiel für das Erstellen der Tabelle fact_sales mit transaction_id als Primärschlüssel sowie Fremdschlüsseln, die die Dimensionstabellen referenzieren.

```sql
-- 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
);
```

DDL-Implementierung zum Erstellen von fact_sales mit Fremdschlüsseldefinitionen

Nach dem Erstellen der Faktentabelle sehen wir, dass der Primärschlüssel (transaction_id) und die Fremdschlüssel als Constraints in der unten stehenden Tabellendefinition erstellt werden.

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.

Definition der Faktentabelle mit Primärschlüssel sowie die Dimensionen referenzierenden Fremdschlüsseln

3. Identitätsspalten für Surrogatschlüssel

Eine Identitätsspalte ist eine Spalte in einer Datenbank, die automatisch eine eindeutige Kennung (ID) für jede neue Datenzeile erzeugt. Diese werden in der Regel zur Erstellung von Surrogatschlüsseln in Data Warehouses verwendet. Surrogatschlüssel sind systemseitig erzeugte Schlüssel ohne sachliche Bedeutung. Sie sind nur vorhanden, damit wir uns nicht auf verschiedene natürliche Primärschlüssel und Verkettungen mehrerer Felder verlassen müssen, um eine Zeile eindeutig zu identifizieren. Normalerweise werden diese Surrogatschlüssel als Primär- und Fremdschlüssel in Data Warehouses verwendet. Einzelheiten zu Identitätsspalten werden in diesem Blogpost behandelt. Hier ist ein Beispiel für das Erstellen einer Identitätsspalte customer_id, der beginnend mit 1 automatisch Werte zugewiesen werden, die jeweils um 1 erhöht werden.

```
1  -- Customer dimension
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 );
```

DDL-Implementierung für das Erstellen einer Kundendimension mit Identitätsspalte

4. Spalten-Constraints für Datenqualität

Databricks unterstützt neben den üblichen Informations-Constraints für Primär- und Fremdschlüssel auch Constraints für die Datenqualitätsprüfung auf Spaltenebene. Diese sollen die Qualität und Integrität der zu einer Tabelle hinzugefügten Daten sicherstellen. Die Constraints werden automatisch getestet. Gute Beispiele hierfür sind NOT NULL-Constraints und Constraints für Spaltenwerte. Im Gegensatz zu anderen Cloud Data Warehouses geht Databricks hier noch einen Schritt weiter und bietet Constraints für die Prüfung von Spaltenwerten. Diese sind sehr nützlich, um die Datenqualität einer bestimmten Spalte sicherzustellen. Wie wir nachstehend sehen, sorgt der Constraint valid_sales_amount dafür, dass alle vorhandenen Zeilen die Anforderung (also sales amount > 0) erfüllen, bevor sie zur Tabelle hinzugefügt werden. Weitere Informationen finden Sie hier.

Nachstehend sind Beispiele für das Hinzufügen von Constraints für dim_store und fact_sales aufgeführt, mit denen sichergestellt werden soll, dass store_id und sales_amount gültige Werte haben.

```
-- Add constraint to dim_store to make sure column store_id is between 1 and 9998
ALTER TABLE US_Stores.Sales_DW.dim_store ADD CONSTRAINT valid_store_id CHECK
(store_id > 0 and store_id < 9999);

-- Add constraint to fact_sales to make sure column sales_amount has a valid value
ALTER TABLE US_Stores.Sales_DW.fact_sales ADD CONSTRAINT valid_sales_amount CHECK
(sales_amount > 0);
```

Hinzufügen eines Spalten-Constraints zu bestehenden Tabellen, um die Datenqualität zu gewährleisten

5. Indizieren, Optimieren und Analysieren

Herkömmliche Datenbanken verfügen über B-Baum- und Bitmap-Indizes. Databricks bietet eine deutlich fortschrittlichere Form der Indizierung: die mehrdimensionale Z-Order-Clustered-Indizierung. Außerdem unterstützen wir die Bloomfilter-Indizierung. Zunächst einmal verwendet das Delta-Dateiformat das Parquet-Format. Dies ist ein spaltenkomprimiertes Dateiformat, d. h., es zeigt bereits eine hohe Effizienz bei der Spaltenreduzierung. Darüber hinaus können Sie mit der Z-Order-Indizierung Daten im Petabyte-Bereich in Sekundenschnelle durchforsten. Sowohl die Z-Order- als auch die Bloomfilter-Indizierung reduzieren die zu durchsuchende Datenmenge erheblich, um hochgradig selektive Abfragen für große Delta-Tabellen durchzuführen. Dies führt in der Regel zu Laufzeitverbesserungen und Kosteneinsparungen um mehrere Größenordnungen. Verwenden Sie Z-Order für diejenigen Primär- und Fremdschlüssel, die für die häufigsten Joins verwendet werden. Die Bloomfilter-Indizierung kann bei Bedarf ergänzend genutzt werden.

```
-- Optimise fact_sales table by customer_id and product_id for better query and
join performance
OPTIMIZE US_Stores.Sales_DW.fact_sales
ZORDER BY (customer_id, product_id);
```

Optimize fact_sales on customer_id and product_id for better performance

```
-- Create a bloomfilter index to enable data skipping on store_business_key
CREATE BLOOMFILTER INDEX
ON TABLE US_Stores.Sales_DW.fact_sales
FOR COLUMNS(store_business_key)
```

Erstellen eines Bloomfilter-Index, um das Überspringen von Daten in einer bestimmten Spalte zu ermöglichen

Wie bei anderen Data Warehouses können Sie mit ANALYZE TABLE die Statistiken aktualisieren, um sicherzustellen, dass der Abfrageoptimierer die besten Statistiken für die Erstellung des optimalen Abfrageplans verwendet.

```
-- collect stats for all columns for better performance
ANALYZE TABLE US_Stores.Sales_DW.fact_sales COMPUTE STATISTICS FOR ALL COLUMNS;
```

Erfassen von Statistiken zu allen Spalten für einen besseren Abfrageausführungsplan

6. Fortgeschrittene Techniken

Databricks unterstützt fortgeschrittene Techniken wie die Tabellenpartitionierung. Doch sollen Sie diese sparsam einsetzen und nur, wenn Sie viele Terabyte komprimierter Daten haben. In den meisten Fällen nämlich werden unsere OPTIMIZE- und Z-ORDER-Indizes die bestmögliche Datei- und Datenbereinigung bieten, sodass die Partitionierung einer Tabelle nach Datum oder Monat fast schon als schlechte Praxis zu bezeichnen ist. Eine gute Praxis besteht dagegen darin, dafür zu sorgen, dass für Ihre Tabellen-DDLs automatische Optimierung und automatische Komprimierung aktiviert sind. So ist gewährleistet, dass Ihre häufig in kleine Dateien geschriebenen Daten zu größeren, spaltenweise komprimierten Delta-Formaten zusammengefasst werden.

Möchten Sie ein Tool zur visuellen Datenmodellierung einsetzen? Mit unserem Partner erwin Data Modeler von Quest können Sie mit wenigen Klicks Sternschemata, Data Vaults und andere branchenübliche Datenmodelle in Databricks zurückentwickeln, erstellen und implementieren.

Beispiel für ein Databricks-Notebook

Mit der Databricks Platform können Sie problemlos verschiedene Datenmodelle entwerfen und implementieren. Um alle oben genannten Beispiele in einem vollständigen Workflow zu sehen, betrachten Sie bitte dieses Beispiel.

Wir empfehlen außerdem die Lektüre des zugehörigen Blogposts
Five Simple Steps for Implementing a Star Schema in Databricks mit Delta Lake →

KAPITEL 3.3

Laden eines Data Warehouse-Datenmodells in Echtzeit mit der Databricks Data Intelligence Platform

Implementierung der Dimensionsmodellierung für das moderne Lakehouse mit Delta Live Tables

von Leo Mao, Soham Bhatt und Abhishek Dey

Die Dimensionsmodellierung ist eine der beliebtesten Datenmodellierungstechniken für die Erstellung eines modernen Data Warehouse. Sie ermöglicht es Kunden, Fakten und Dimensionen schnell gemäß den geschäftlichen Anforderungen eines Unternehmens zu entwickeln. Im direkten Kundenkontakt haben wir festgestellt, dass viele nach Best Practices und einer Implementierungsreferenzarchitektur von Databricks suchen.

In diesem Artikel möchten wir tiefer in die Best Practices der Dimensionsmodellierung in der Lakehouse-Architektur einsteigen und ein Livebeispiel für das Laden eines EDW-Dimensionsmodells in Echtzeit mit Delta Live Tables geben.

Hier sind die allgemeinen Schritte, die wir in diesem Blogpost behandeln werden:

- Definieren eines geschäftlichen Problems
- Entwerfen eines Dimensionsmodells
- Best Practices und Empfehlungen für die Dimensionsmodellierung
- Implementieren eines Dimensionsmodells auf einer Lakehouse-Architektur
- Fazit

Definieren eines geschäftlichen Problems

Die Dimensionsmodellierung ist betriebswirtschaftlich orientiert: Am Anfang steht immer ein geschäftliches Problem. Bevor wir ein Dimensionsmodell erstellen, müssen wir das zugrunde liegende Geschäftsproblem verstehen. Es bestimmt, wie das Datenmodell gestaltet und von den Endbenutzern genutzt werden soll. Wir müssen das Datenmodell so gestalten, dass es niederschwelligere und schnellere Abfragen unterstützt.

Die Business Matrix ist ein grundlegendes Konzept in der Dimensionsmodellierung. Unten sehen Sie ein Beispiel für eine Business Matrix, bei der die Spalten gemeinsame Dimensionen und die Zeilen Geschäftsprozesse darstellen. Das definierte Geschäftsproblem bestimmt die Struktur der Faktendaten und die erforderlichen Dimensionen. Der Kerngedanke dabei ist, dass wir auf der Grundlage der Business Matrix und ihrer gemeinsamen oder angepassten Dimensionen problemlos schrittweise zusätzliche Daten-Assets aufbauen können.

Gemeinsame Dimensionen →

Geschäftsprozesse ↓	Datum	Kunde	Produkt	Anbieter	Werbeaktion	Händler	Absatzgebiet	Mitarbeiter	Account	Organisation
Internetverkäufe	✓	✓	✓	✓	✓		✓			
Händlerverkäufe	✓		✓		✓	✓	✓	✓		
Hauptbuch	✓								✓	✓
Vertriebsplan	✓		✓				✓			
Bestand	✓		✓	✓					✓	
Kundenbefragungen	✓	✓								
Anrufe beim Kundendienst	✓	✓	✓					✓		

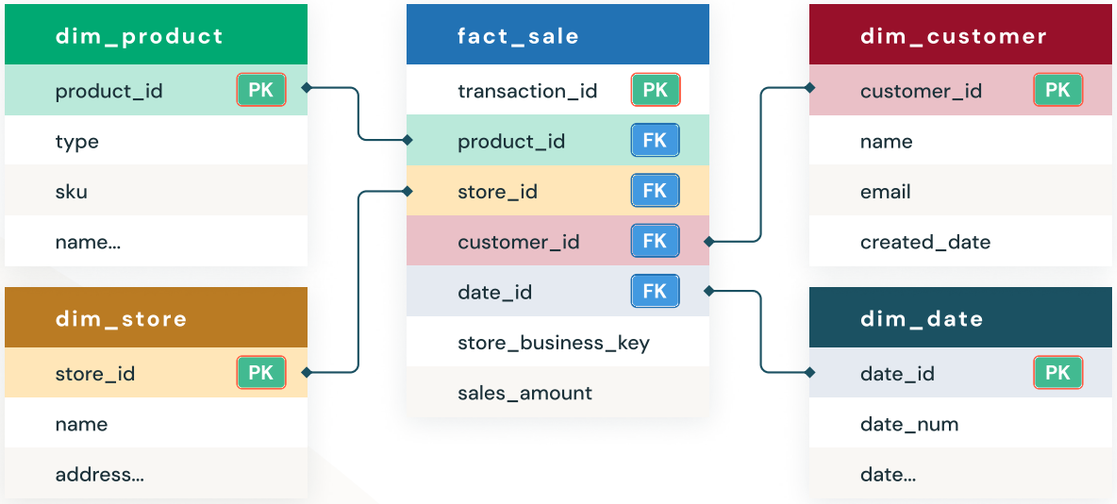
Business Matrix mit gemeinsamen Dimensionen und Geschäftsprozessen

Hier gehen wir davon aus, dass der Auftraggeber vom Team einen Bericht erstellen lassen möchte, der folgende Fragen beantwortet:

- Was sind die meistverkauften Produkte? So lässt sich die Beliebtheit von Produkten nachvollziehen.
- Welche Niederlassungen bringen die beste Performance? Daraus lassen sich gute Betriebspraktiken ableiten.

Entwerfen eines Dimensionsmodells

Ausgehend vom definierten Geschäftsproblem zielt die Datenmodellierung darauf ab, die Daten effizient darzustellen, um Wiederverwendbarkeit, Flexibilität und Skalierbarkeit zu gewährleisten. Hier ist eine allgemeine Version des Datenmodells für die oben genannten geschäftlichen Fragen.



Dimensionsmodell in der Lakehouse-Architektur

Beachten Sie, dass jede Dimensionstabelle die Spalten **__START_AT** und **__END_AT** enthält, um SCD-Type-2 zu unterstützen. Aus Platzgründen werden diese hier nicht gezeigt.

Das Modell sollte leicht verständlich und effizient sein und verschiedene Abfragemuster für die Daten enthalten. Anhand des Modells haben wir die Tabelle fact_sale entworfen, um unsere geschäftlichen Fragen zu beantworten. Wie Sie sehen können, enthält sie außer den Fremdschlüsseln zu den Dimensionen nur die numerischen Metriken, die zur Messung des Geschäfts verwendet werden, z. B. sales_amount.

Wir haben auch Dimensionstabellen wie Produkt, Store, Customer und Date erstellt, die Kontextinformationen zu den Faktendaten liefern. Dimensionstabellen werden in der Regel per Join mit Faktentabellen verknüpft, um bestimmte geschäftliche Fragen zu beantworten, z. B. nach den beliebtesten Produkten in einem bestimmten Monat, den umsatzstärksten Filialen des Quartalsl etc.

Best Practices und Empfehlungen für die Dimensionsmodellierung

Mit der Databricks Data Intelligence Platform können Sie ganz einfach Dimensionsmodelle entwerfen und implementieren und die Fakten und Dimensionen für den gegebenen Sachbereich leicht erstellen.

Im Folgenden finden Sie einige der Best Practices, die für die Implementierung eines Dimensionsmodells empfohlen werden:

- Die Dimensionstabellen sollten Sie denormalisieren. Dimensionstabellen nutzen kein 3NF- oder Snowflake-Modell, sondern sind in der Regel stark denormalisiert. Dadurch entstehen reduzierte n:1-Beziehungen innerhalb einer einzelnen Dimensionstabelle.
- Verwenden Sie angepasste Dimensionstabellen, wenn Attribute in verschiedenen Dimensionstabellen die gleichen Spaltennamen und Domäneninhalte haben. Der Vorteil: Daten aus verschiedenen Faktentabellen können in einem einzigen Bericht zusammengeführt werden, indem den einzelnen Faktentabellen angepasste Dimensionsattribute zugeordnet werden.
- Ein üblicher Trend bei Dimensionstabellen ist die Nachverfolgung von Änderungen an Dimensionen im zeitlichen Verlauf zur Unterstützung von As-Is- oder As-Was-Berichten. Sie können problemlos die folgenden grundlegenden Techniken für den Umgang mit Dimensionen je nach den verschiedenen Anforderungen anwenden.
  - Die Type-1-Technik überschreibt den Ausgangswert des Dimensionsattributs.
  - Mit der Type-2-Technik – der häufigsten SCD-Technik – können Sie eine korrekte Änderungsverfolgung im zeitlichen Verlauf durchführen.

  Mit der Delta Live Tables-Implementierung ist dies problemlos und ohne weiteren Aufwand möglich.
  - Sie können SCD-Type-1 oder SCD-Type-2 mit Delta Live Tables ganz einfach durchführen. Hierzu verwenden Sie APPLY CHANGES INTO.
- Primär- und Fremdschlüssel-Constraints ermöglichen es Endbenutzern wie Ihnen, Beziehungen zwischen Tabellen nachzuvollziehen.
- Bei Nutzung von Identitätsspalten werden automatisch eindeutige ganzzahlige Werte erzeugt, wenn neue Zeilen hinzugefügt werden. Identitätsspalten sind eine Form von Surrogatschlüsseln. Ausführliche Informationen finden Sie in diesem Blogpost.
- **Erzwungene CHECK-Constraints** sorgen dafür, dass Sie sich niemals Gedanken über die Qualität oder Richtigkeit Ihrer Daten machen müssen.

Implementieren eines Dimensionsmodells auf einer Lakehouse-Architektur

Betrachten wir nun ein Beispiel für die Implementierung von Delta Live Tables auf Basis der Dimensionsmodellierung:

Der folgende Beispielcode zeigt uns, wie wir eine Dimensionstabelle (dim_store) mit SCD-Type-2 erstellen, wobei die Änderungsdaten aus dem Quellsystem erfasst werden.

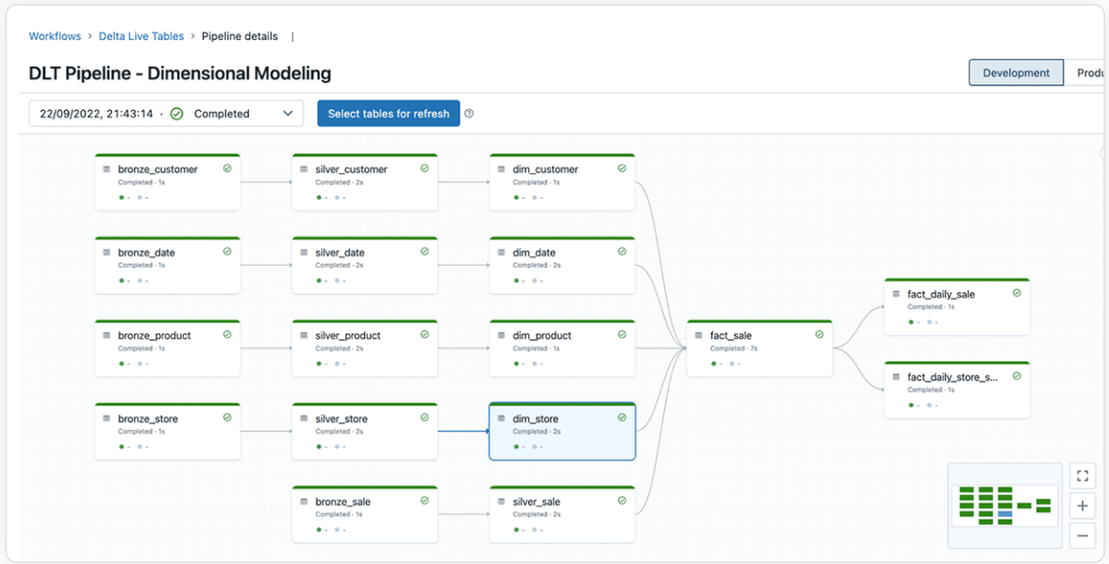
```
-- create the gold table
CREATE INCREMENTAL LIVE TABLE dim_store
TBLPROPERTIES (“quality” = “gold”)
COMMENT “Slowly Changing Dimension Type 2 for store dimension in the gold layer”;

-- store all changes as 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;
```

Der Beispielcode unten zeigt uns, wie wir eine Faktentabelle (fact_sale) erstellen. Mit dem Constraint valid_product_id können wir sicherstellen, dass alle geladenen Faktendatensätze mit einem gültigen Produkt verknüpft sind.

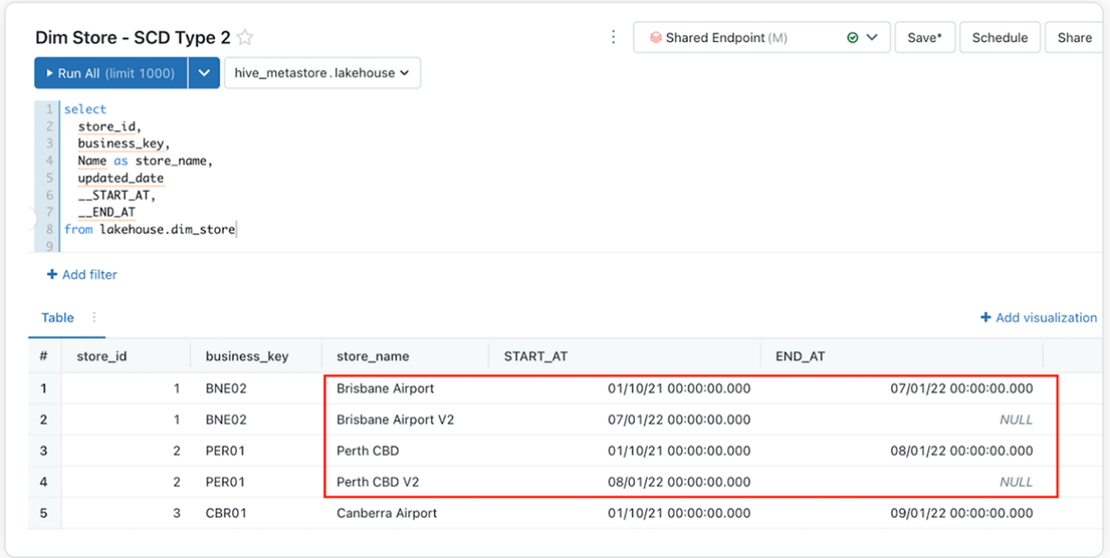
```
-- create the fact table for sales in gold layer
CREATE STREAMING LIVE TABLE fact_sale (
  CONSTRAINT valid_store_business_key EXPECT (store_business_key IS NOT NULL) ON
VIOLATION DROP ROW,
  CONSTRAINT valid_product_id EXPECT (product_id IS NOT NULL) ON VIOLATION DROP
ROW
)
TBLPROPERTIES (“quality” = “gold”, “ignoreChanges” = “true”)
COMMENT “sales fact table in the gold layer” 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’)
  -- only join with the active customers
  INNER JOIN (SELECT * FROM live.dim_customer WHERE __END_AT IS NULL) customer
  ON sale.customer_id = customer.customer_id
  -- only join with the active products
  INNER JOIN (SELECT * FROM live.dim_product WHERE __END_AT IS NULL) product
  ON sale.product = product.SKU
  -- only join with the active stores
  INNER JOIN (SELECT * FROM live.dim_store WHERE __END_AT IS NULL) store
  ON sale.store = store.business_key
```

Das Delta Live Tables-Pipelinebeispiel finden Sie hier. Informationen zum Erstellen einer Delta Live Tables-Pipeline entnehmen Sie der Schnellstartanleitung zu Delta Live Tables. Wie nachstehend zu sehen ist, bietet DLT vollständigen Einblick in die ETL-Pipeline und die Abhängigkeiten zwischen verschiedenen Objekten in der Bronze-, Silber- und Gold-Schicht und orientiert sich dabei an der Lakehouse-Medaillon-Architektur.



Ende-zu-Ende-DLT-Pipeline

Hier ist ein Beispiel dafür, wie die Dimensionstabelle dim_store auf der Grundlage der eingehenden Änderungen aktualisiert wird. Unten wurde Store Brisbane Airport in Brisbane Airport V2 aktualisiert, und mit der standardmäßig vorhandenen SCD-Type-2-Unterstützung endete der ursprüngliche Datensatz am 7. Januar 2022. Gleichzeitig wurde ein neuer Datensatz erstellt, der am selben Tag mit einem offenen Enddatum (NULL) beginnt; dies ist der neueste Datensatz für den Flughafen 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 für Store-Dimension

Wenn Sie mehr zur Implementierung erfahren möchten, sehen Sie sich das vollständige Beispiel-Notebook an, das Sie hier finden.

Fazit

In diesem Beitrag haben wir die Konzepte der Dimensionsmodellierung im Detail kennengelernt und erfahren, welche Best Practices es gibt und wie man sie mit Delta Live Tables implementiert.

Weitere Informationen zur Dimensionsmodellierung finden Sie bei Kimball Technology.

KAPITEL 3.4

Was gibt es Neues bei Databricks SQL?

KI-gesteuerte Optimierungen, Lakehouse Federation und mehr für unternehmenstaugliche BI

von Alex Lichen, Miranda Luna, Can Efeoglu und Cyrielle Simeone

Auf dem diesjährigen Data + AI Summit hat Databricks SQL die Erwartungen an ein Data Warehouse erneut übertroffen. Nun wird KI produktoberflächenübergreifend eingesetzt, um unsere Führungsposition in Sachen Leistung und Effizienz auszubauen, die Benutzerfreundlichkeit zu verbessern und neue Möglichkeiten für unsere Kunden zu erschließen. Parallel dazu verbessern wir kontinuierlich unserer zentralen Data-Warehousing-Funktionen, um Ihren gesamten Daten-Stack unter der Lakehouse-Architektur der Databricks Data Intelligence Platform zusammenzuführen.

In diesem Blogpost stellen wir Ihnen die Highlights der neuen und kommenden Features in Databricks SQL vor:

- KI-gesteuerte Leistungsoptimierungen wie Predictive I/O, die Spitzenleistungen und Kosteneinsparungen ermöglichen, ohne manuelle Abstimmung
- Neue Möglichkeiten für Benutzer durch KI-Funktionen, SQL Warehouses in Notebooks, neue Dashboards und benutzerdefinierte Python-Funktionen
- Umfangreiche Unterstützung externer Datenquellen mit Lakehouse Federation
- Neue Wege für den Zugriff auf Ihre Daten mit der SQL Statement Execution-API
- Einfache und effiziente Datenverarbeitung durch Integration von Streaming Tables, Materialized Views und Workflows
- Intelligente Unterstützung durch DatabricksIQ, unsere Knowledge Engine
- Verbesserte Verwaltungstools mit Databricks SQL System Tables und Live Query Profile
- Zusätzliche Funktionen für unsere Partnerintegrationen mit Fivetran, dbt labs und Power BI

Das KI-optimierte Warehouse: einsatzbereit für alle Ihre Workloads – ganz ohne Tuning

Wir sind überzeugt, dass das beste Data Warehouse ein Lakehouse ist. Deshalb bauen wir unsere Führungsposition bei ETL-Workloads weiter aus und setzen auch künftig auf das Potenzial der KI. Zudem bietet Databricks SQL jetzt auch branchenführende Leistung für Ihre EDA- und BI-Workloads und sorgt gleichzeitig für Kosteneinsparungen – ganz ohne manuelles Tuning.

Vergessen Sie das manuelle Erstellen von Indizes. Mit Predictive I/O für Lesevorgänge (allgemein verfügbar) und Aktualisierungen (in der öffentlichen Vorschau) analysiert Databricks SQL jetzt historische Lese- und Schreibmuster, um intelligent Indizes zu erstellen und Arbeitslasten zu optimieren. Erste Kunden profitieren bereits von einer bemerkenswerten Effizienzsteigerung bei der Punktsuche (um den Faktor 35) sowie beeindruckenden Leistungssteigerungen um das 2- bis 6-fache bei MERGE-Operationen und das 2- bis 10-fache bei DELETE-Operationen.

Mit Predictive Optimizations (in der öffentlichen Vorschau) führt Databricks OPTIMIZE-, VACUUM-, ANALYZE- und CLUSTERING-Befehle aus, um die Dateigrößen und Clustering nahtlos für Sie zu optimieren. Mit dieser Funktion konnte Anker Innovations die Abfrageleistung um das 2,2-fache steigern und gleichzeitig 50 % der Speicherkosten einsparen.

> **Die Predictive Optimizations von Databricks haben unseren Unity Catalog-Speicher sinnvoll optimiert. Dadurch konnten wir 50 % der jährlichen Speicherkosten einsparen und gleichzeitig unsere Abfragen um mehr als das Doppelte beschleunigen. Er hat gelernt, unsere größten und meistgenutzten Tabellen zu priorisieren. Und all dies geschah automatisch, was unserem Team wertvolle Zeit sparte.**
>
> – ANKER INNOVATIONS

Sind Sie es leid, verschiedene Warehouses für kleinere und größere Workloads zu verwalten oder Skalierungsparameter abzustimmen? Intelligent Workload Management ist eine Suite mit Funktionen, die Abfragen beschleunigen und gleichzeitig die Kosten niedrig halten. Intelligentes Workload Management analysiert Echtzeitmuster, um sicherzustellen, dass Ihre Workloads über optimale Rechenkapazität verfügen und eingehende SQL-Anweisungen ohne Beeinträchtigung bereits laufender Abfragen ausgeführt werden können.

Dank KI-gestützter Optimierungen bietet Databricks SQL branchenweit führende TCO und Performance für Workloads aller Art, ohne dass ein manuelles Tuning erforderlich wäre. Wenn Sie mehr über die Optimierungen erfahren möchten, die in der Vorschau verfügbar sind, sehen Sie sich die Keynote von Reynold Xin und Databricks SQL Serverless Under the Hood: How We Use ML to Get the Best Price/Performance vom Data + AI Summit an.

Aufbrechen von Datensilos mit Lakehouse Federation

Unternehmen stehen heute vor der Aufgabe, verstreute Datenquellen in fragmentierten Systemen finden, verwalten und abfragen zu müssen. Mit Lakehouse Federation können Datenteams Databricks SQL nutzen, um Daten auf externen Plattformen wie MySQL, PostgreSQL, Amazon Redshift, Snowflake, Azure SQL Database, Azure Synapse, Googles BigQuery (in Kürze) und weiteren zu erkennen, abzufragen und zu verwalten.

Außerdem lässt sich Lakehouse Federation nahtlos in die erweiterten Funktionen von Unity Catalog integrieren, wenn Sie aus Databricks heraus auf externe Datenquellen zugreifen. Sie können Sicherheit auf Zeilen- und Spaltenebene erzwingen, um den Zugriff auf sensible Informationen zu beschränken. Ermitteln Sie die Datenherkunft, um die Datenqualität und Compliance sicherzustellen. Zum Organisieren und Verwalten von Daten-Assets können Sie föderierte Katalog-Assets ganz einfach taggen und Daten so unkompliziert entdecken.

Schließlich unterstützt Lakehouse Federation Materialized Views, um komplexe Transformationen oder Cross-Joins bei föderierten Quellen zu beschleunigen und die Abfragezeiten zu verkürzen.

Nähere Einzelheiten finden Sie in unserer speziellen Data+AI Summit Session Lakehouse Federation: Access and Governance of External Data Sources from Unity Catalog.

Entwickeln auf der Lakehouse-Architektur mit der SQL Statement Execution-API

Die SQL Statement Execution-API ermöglicht den Zugriff auf Ihr Databricks SQL Warehouse über eine REST-API, um Abfragen zu stellen und Ergebnisse abzurufen. Da HTTP-Frameworks für fast alle Programmiersprachen zur Verfügung stehen, können Sie problemlos eine Vielzahl von Anwendungen und Plattformen direkt mit einem Databricks SQL Warehouse verbinden.

Die Databricks SQL Statement Execution-API ist im Umfang der Tarife Databricks Premium und Databricks Enterprise enthalten. Wenn Sie mehr erfahren möchten, sehen Sie sich unsere Session an, folgen Sie unserem Tutorial (AWS | Azure), lesen Sie die Dokumentation (AWS | Azure) oder besuchen Sie unser Repository mit Codebeispielen.

Optimieren Ihrer Datenverarbeitung mit Streaming Tables, Materialized Views und DB SQL in Workflows

Mit Streaming Tables, Materialized Views und DB SQL in Workflows kann jetzt jeder SQL-Benutzer bei der Verarbeitung von Daten Best Practices für das Data Engineering anwenden. Erfassen, transformieren, orchestrieren und analysieren Sie Daten effizient mit nur wenigen SQL-Zeilen.

Streaming Tables sind der ideale Weg, um Daten in Bronze-Tabellen zu übertragen. Mit nur einer einzigen SQL-Anweisung können Sie Daten aus verschiedenen Quellen wie Cloud-Speicher (S3, ADLS, GCS), Message Busses (EventHub, Kafka, Kinesis) und weiteren skalierbar erfassen. Die Erfassung erfolgt dabei inkrementell und ermöglicht so kostengünstige Pipelines mit geringer Latenz, ohne dass Sie eine komplexe Infrastruktur verwalten müssten.

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

Materialisierte Ansichten reduzieren Kosten und verbessern die Abfragelatenz, indem sie langsame Abfragen und häufig verwendete Berechnungen vorab berechnen und inkrementell aktualisieren. Im Data-Engineering-Kontext werden sie für die Transformation von Daten verwendet. Materialisierte Ansichten sind für Analystenteams in einem Data-Warehouse-Kontext wertvoll. Sie beschleunigen Endbenutzerabfragen (1) und BI-Dashboards und ermöglichen eine sichere Datenfreigabe (2). In nur vier Codezeilen kann jeder Benutzer eine Materialized View für leistungsstarke Datenverarbeitung erstellen.

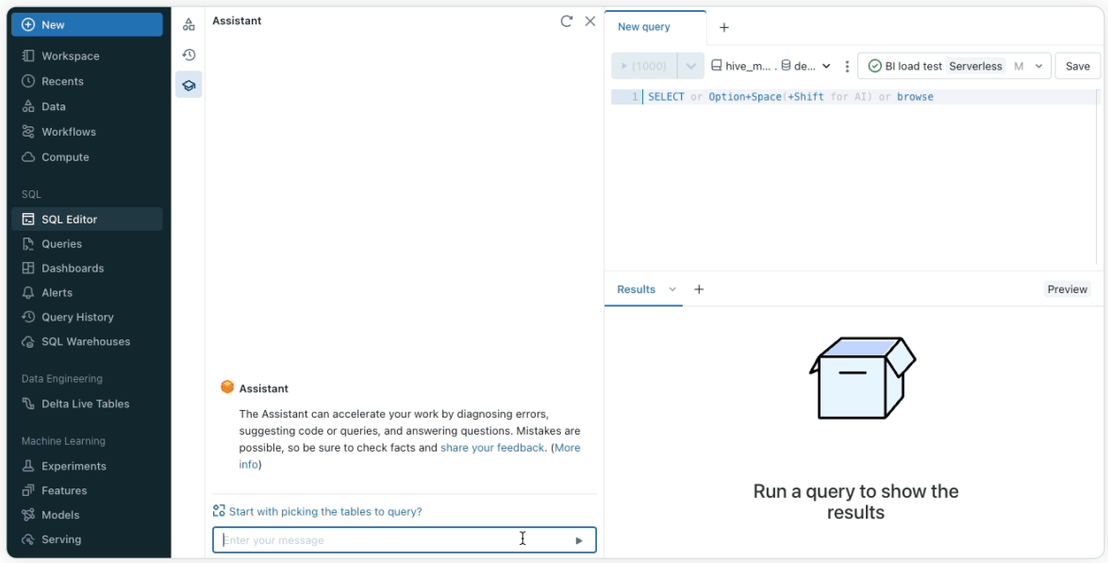
```
CREATE MATERIALIZED VIEW customer_orders
AS
SELECT
  customers.name,
  sum(orders.amount),
  orders.orderdate
FROM orders
  LEFT JOIN customers ON
    orders.custkey = customers.c_custkey
GROUP BY
  name,
  orderdate;
```

Brauchen Sie eine Orchestrierung bei DB SQL? Mit Workflows können Sie jetzt SQL- Abfragen, Dashboards und Alerts planen. Verwalten Sie mühelos komplexe Abhängigkeiten zwischen Tasks und beobachten Sie vergangene Jobausführungen mit der intuitiven Workflows-UI oder per API.

Streaming Tables und Materialized Views sind jetzt in der öffentlichen Vorschau. Mehr erfahren Sie in unserem Blogpost zum Thema. Registrieren Sie sich für die öffentliche Vorschau beider Funktionen: Preview-Registrierung. Workflows in DB SQL ist inzwischen allgemein verfügbar. Weitere Informationen finden Sie in der Dokumentation (AWS | Azure).

Databricks Assistant: schneller besseren SQL-Code schreiben mit natürlicher Sprache

Databricks Assistant ist ein kontextsensitiver KI-Assistent, der in Databricks Notebooks und den SQL-Editor eingebettet ist. Databricks Assistant nimmt eine Frage in natürlicher Sprache entgegen und schlägt zu deren Beantwortung eine SQL-Abfrage vor. Wenn Sie eine komplexe Abfrage verstehen möchten, können Sie den Assistenten bitten, sie in natürlicher Sprache zu erklären. So verstehen Sie die Logik hinter den Ergebnissen besser.



Databricks Assistant nutzt verschiedene Signale, um präzisere und relevantere Ergebnisse zu liefern. Mithilfe des Kontexts von Codezellen, Bibliotheken, beliebten Tabellen, Unity Catalog-Schemata und Tags setzt er in natürlicher Sprache gestellte Fragen in Abfragen und Code um.

Für die Zukunft planen wir eine Integration mit DatabricksIQ, um noch mehr Kontext für Ihre Anfragen bereitzustellen.

Souveränes Verwalten Ihres Data Warehouse

Admins und IT-Teams benötigen geeignete Tools, um die Nutzung eines Data Warehouse zu verstehen. Mit System Tables, Live Query Profile und Statement Timeouts können sie Abläufe überwachen, Probleme sofort lösen und die Effizienz des Data Warehouse sichern.

System Tables vermitteln einen besseren Einblick in Ihre SQL-Umgebung. Diese von Databricks bereitgestellten Tabellen enthalten die Informationen über frühere Anweisungsausführungen, Kosten, Herkunft und mehr. Analysieren Sie Metadaten und Nutzungsstatistiken, um Fragen zu klären wie: „Welche Befehle wurden von wem ausgeführt?“, „Wann und wie skalierte meine Warehouses?“ oder „Was wurde mir in Rechnung gestellt?“ Da System Tables in Databricks integriert sind, haben Sie Zugriff auf native Funktionen wie SQL-Alerts und SQL-Dashboards, um Überwachungs- und Alarmierungsprozesse zu automatisieren.

Zurzeit sind drei System Tables in der öffentlichen Vorschau: Audit Logs, Billable Usage System Table und Lineage System Table (AWS | Azure). Weitere System Tables für Warehouse-Ereignisse und den Anweisungsverlauf werden in Kürze folgen.

Wenn Sie z. B. die monatlich verbrauchten DBUs pro Artikelnummer berechnen möchten, können Sie Billable Usage System Tables abfragen.

```
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
```

Mit Live Query Profile erhalten Sie in Echtzeit Informationen über die Abfrageleistung und können Ihre Workloads direkt optimieren. Visualisieren Sie Pläne zur Abfrageausführung und bewerten Sie Live-Ausführungen von Abfragetasks, um häufige SQL-Fehler wie Exploding Joins oder vollständige Tabellenscans zu beheben. Live Query Profile hilft Ihnen, Abfragen in Ihrem Data Warehouse zu optimieren und effizient auszuführen. Weitere Informationen finden Sie in der Dokumentation (AWS | Azure).

Suchen Sie nach automatisierten Kontrollen? Mit Statement Timeouts legen Sie benutzerdefinierte Timeouts auf Arbeitsbereichs- oder Abfrageebene fest. Überschreitet eine Abfrage die Timeoutschwelle, wird sie automatisch gestoppt. Weitere Informationen finden Sie in der Dokumentation (AWS | Azure).

Überzeugende neue Nutzungsmöglichkeiten in DBSQL

Im letzten Jahr haben wir Databricks SQL intensiv um innovative Funktionen erweitert. Wir freuen uns, neue Features ankündigen zu können, die SQL-Benutzern die Möglichkeiten der künstlichen Intelligenz eröffnen. Dazu zählen SQL Warehouses auf der Databricks Plattform, eine neue Generation von SQL-Dashboards und der Einsatz von Python in Databricks SQL.

Demokratisieren der Analyse unstrukturierter Daten mit KI-Funktionen

Mit AI Functions erweitert DB SQL das SQL Warehouse um die Leistungsfähigkeit der KI. Schöpfen Sie das Potenzial unstrukturierter Daten mühelos aus – mit Funktionen für Stimmungsanalyse, Textklassifizierung, Zusammenfassung, Übersetzung und mehr. Datenanalysten können KI-Modelle per Self-Service nutzen und Data Engineers eigenständig KI-fähige Pipelines erstellen.

Die Verwendung von KI-Funktionen ist ganz einfach. Nehmen wir zum Beispiel ein Szenario, in dem einige Artikel mit den Stimmungen „Frustrated“ (Frustriert), „Happy“ (Glücklich), „Neutral“ oder „Satisfied“ (Zufrieden) verknüpft werden sollen.

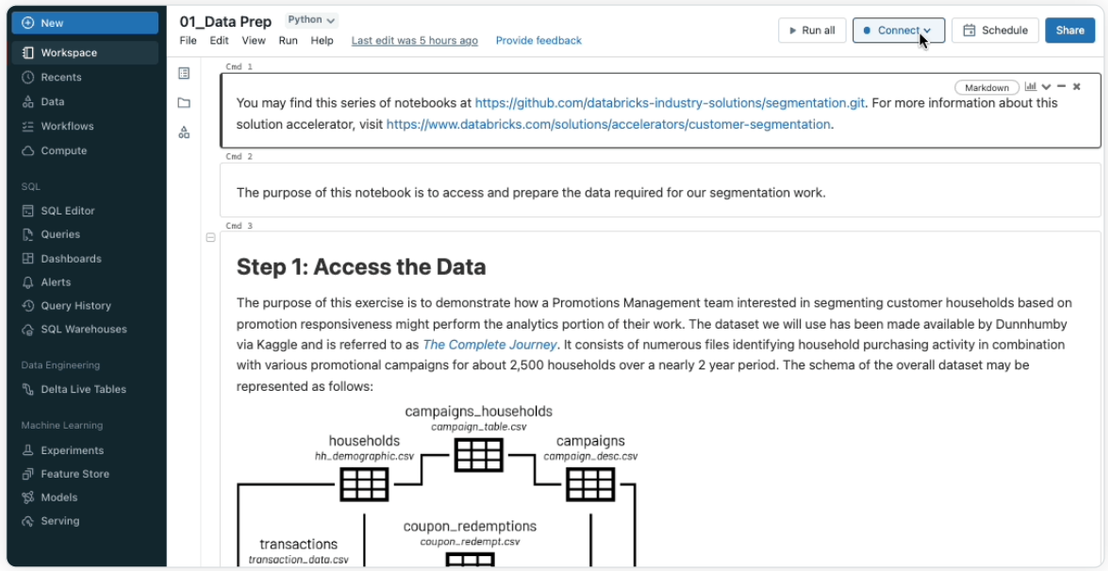
```
-- create a udf for sentiment classification
CREATE FUNCTION classify_sentiment(text STRING)
  RETURNS STRING
  RETURN ai_query(
    'Dolly', -- the name of the model serving endpoint
    named_struct(
      'prompt',
      CONCAT('Classify the following text into one of four categories [Frustrated,
Happy, Neutral, Satisfied]:\n',
        text),
      'temperature', 0.5),
    'returnType', 'STRING');
```

```
-- use the udf
SELECT classify_sentiment(text) AS sentiment
FROM reviews;
```

AI Functions ist jetzt in der öffentlichen Vorschau. Zum Registrieren für die Vorschau füllen Sie bitte dieses Formular aus. Wenn Sie mehr erfahren möchten, lesen Sie unseren ausführlichen Blogpost oder die Dokumentation (AWS | Azure).

Ausstatten von Notebooks mit der Leistungsfähigkeit von SQL Warehouses

Databricks SQL-Warehouses sind jetzt für Notebooks in der öffentlichen Vorschau. Hierdurch wird die Flexibilität von Notebooks mit der Performance und den TCO von Databricks SQL Serverless- und Pro-Warehouses kombiniert. Um SQL-Warehouses in Notebooks zu aktivieren, wählen Sie einfach ein verfügbares Warehouse aus dem Compute-Dropdown-Menü für Notebooks aus.



Herstellen einer Verbindung aus Databricks-Notebooks mit Serverless SQL-Warehouses

Finden und Teilen von Erkenntnissen mithilfe einer neuen Dashboard-Generation

Erleben Sie eine ganz neue Dashboard-Nutzung direkt in der Lakehouse-Architektur. Benutzer können ein gewünschtes Dataset einfach auswählen und per SQL-Option beeindruckende Visualisierungen erstellen. Machen Sie Schluss mit der Verwaltung separater Abfragen und Dashboard-Objekte: Ein einheitliches Inhaltsmodell vereinfacht den Genehmigungs- und Verwaltungsprozess. Veröffentlichen Sie Ihr Dashboard unternehmensweit, sodass jeder authentifizierte Nutzer Ihres Identitätsanbieter über einen sicheren Weblink Zugriff darauf hat, selbst ohne Databricks-Zugang.

Neue Databricks SQL-Dashboards sind gegenwärtig in der privaten Vorschau. Kontaktieren Sie Ihr Kundenbetreuungsteam, um mehr zu erfahren.

Nutzen der Flexibilität von Python in SQL

Nutzen Sie die Flexibilität von Python in Databricks SQL – mit benutzerdefinierten Python-Funktionen (User-Defined Functions, UDFs). Rufen Sie benutzerdefinierte Python-Funktionen direkt aus Ihrer SQL-Abfrage heraus auf, um ML-Modelle zu integrieren oder benutzerdefinierte Bearbeitungslogik auf Datenverarbeitung und -analyse anzuwenden. UDFs sind wiederverwendbare Funktionen, die eine einheitliche Verarbeitung Ihrer Datenpipelines und Analysen ermöglichen.

Wenn Sie beispielsweise E-Mail-Adressen und Telefonnummern in einer Datei unkenntlich machen möchten, verwenden Sie die folgende CREATE FUNCTION-Anweisung.

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

Mehr zur Registrierung für die private Vorschau

databricks

Integrationen mit Ihrem Datenökosystem

Beim Data + AI Summit kündigte Databricks SQL neue Integrationen an, um eine reibungslose Nutzung von Tools Ihrer Wahl zu ermöglichen.

Databricks + Fivetran

Wir freuen uns, die allgemeine Verfügbarkeit von Fivetran-Zugang in Partner Connect für alle Nutzer (inklusive Nicht-Admins mit ausreichenden Katalogrechten) bekanntzugeben. Diese Innovation macht es für alle Benutzer zehnmal einfacher, Daten mit Fivetran nach Databricks zu importieren. Ein großer Fortschritt für Databricks-Kunden: Sie können nun Daten von Hunderten Fivetran-Connectors, etwa für Salesforce und PostgreSQL, in die Lakehouse-Architektur einbringen. Und: Fivetran unterstützt jetzt auch serverlose Warehouses vollständig!

Weitere Informationen finden Sie in diesem Blogpost.

Databricks + dbt Labs

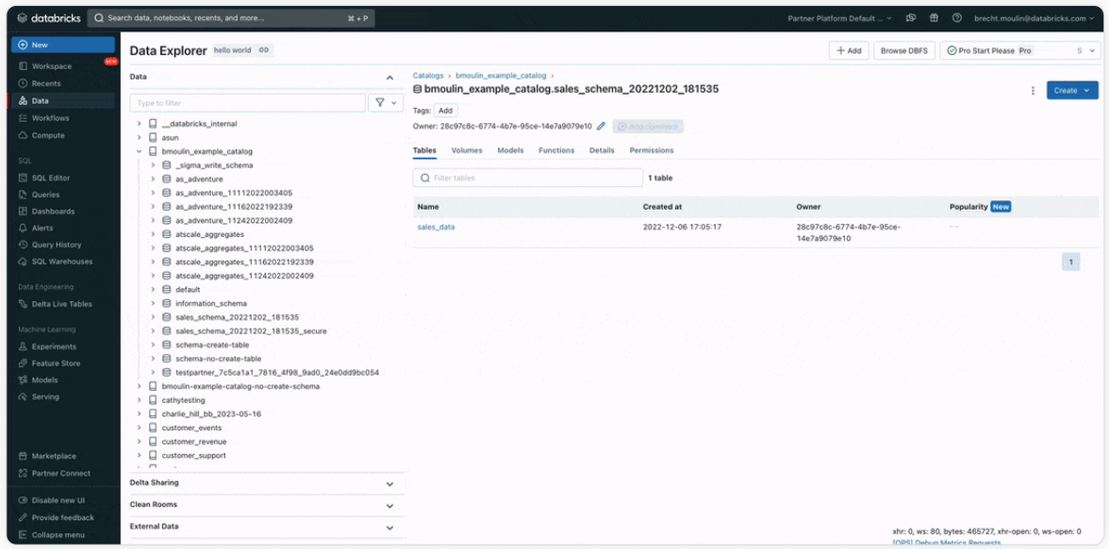
Vereinfachen Sie die Entwicklung von Echtzeitanalysen auf der Lakehouse-Architektur mit Databricks und dbt Labs. Die Kombination des äußerst beliebten Analytics-Engineering-Frameworks von dbt mit der Databricks Platform bietet leistungsstarke Funktionen:

- dbt + Streaming Tables: Die Erfassung von Streaming-Daten aus beliebigen Quellen ist jetzt in dbt-Projekte integriert. Mit SQL können Analytics Engineers Cloud-/Streaming-Daten direkt in ihren dbt-Pipelines definieren und aufnehmen.
- dbt + Materialized Views: Die Entwicklung effizienter Pipelines wird mit dbt einfacher, da Sie nun die leistungsstarken inkrementellen Neuladefunktionen von Databricks nutzen können. Mit dbt lassen sich Pipelines erstellen und ausführen, die auf MVs basieren. Benutzer profitieren von effizienter, inkrementeller Berechnung und senken so ihre Infrastrukturkosten.

Weitere Informationen finden Sie in diesem Blogpost.

Databricks + Power BI: Veröffentlichen in Power BI-Arbeitsbereichen

Veröffentlichen Sie Datasets aus Ihrem Databricks-Arbeitsbereich mit nur wenigen Klicks im Power BI-Onlinearbeitsbereich! Nie wieder ODBC- und JDBC-Verbindungen verwalten: Wählen Sie einfach das Dataset aus, das Sie veröffentlichen möchten. Sie wählen einfach die Datasets oder Schemata aus, die Sie veröffentlichen möchten, und dann Ihren PBI-Arbeitsbereich. So wird es für BI-Administratoren und Berichtersteller noch einfacher, Power BI-Workspaces zu unterstützen, ohne auch Power BI Desktop verwenden zu müssen.



Erste Schritte mit Databricks SQL

Sie wollen jetzt in Databricks SQL einsteigen? Dann befolgen Sie die Anweisungen im Leitfaden (AWS | Azure | GCP) zur Einrichtung eines SQL-Warehouse. Databricks SQL Serverless ist derzeit mit einem Aktionsrabatt von über 20 % erhältlich. Weitere Informationen finden Sie auf unserer Preise-Seite.

Einen vollständigen Überblick erhalten Sie auch über folgende Ressourcen: **Databricks SQL: Why the Best Serverless Data Warehouse is a Lakehouse** und **What's New in Databricks SQL — With Live Demos**.

KAPITEL 3.5

Verteilte Data Governance und isolierte Umgebungen mit Unity Catalog

von Max Nienu, Zeashan Pappa, Paul Roome und Sachin Thakur

Wirksame Data Governance ist für jedes Unternehmen unerlässlich, das sich bei seiner Arbeit auf Daten, Analysen und KI stützt. In vielen Unternehmen wächst die Erkenntnis, dass zentralisierte Data Governance einen echten Mehrwert bietet. Doch selbst mit den besten Absichten kann die Umsetzung zentralisierter Governance ohne die entsprechenden organisatorischen Prozesse und Ressourcen eine Herausforderung darstellen. Die Rolle des Chief Data Officer (CDO) ist in vielen Unternehmen noch im Entstehen begriffen. So stellt sich die Frage, wer die Data-Governance-Richtlinien im Unternehmen definieren und umsetzen soll.

Daher ist die Verantwortung für die Definition und Umsetzung von Data-Governance-Richtlinien im Unternehmen oft nicht zentralisiert: Unterschiedliche Geschäftsbereiche, Abteilungen und Unterabteilungen haben unterschiedliche Richtlinien oder unterstehen jeweils anderen Führungsgremien. Der Einfachheit halber bezeichnen wir dieses Muster als verteilte Governance: Es besteht Einigkeit über die Unterschiede zwischen den Verwaltungseinheiten, es gibt jedoch oft keine zentrale Data-Governance-Funktion.

In diesem Blogpost befassen wir uns mit der Implementierung eines verteilten Governance-Modells unter Verwendung von Databricks Unity Catalog. Unity Catalog bietet eine einheitliche Governance-Lösung für Daten, Analysen und KI im Lakehouse.

Entwicklung von Data Governance in Databricks

Vor Einführung von Unity Catalog war das Konzept eines Arbeitsbereichs monolithisch: Jeder Arbeitsbereich hatte einen eigenen Metastore, eine eigene Benutzerverwaltung und einen eigenen Tabellen-ACL-Speicher. Dies führte zur Isolation der Daten und der Governance zwischen den einzelnen Arbeitsbereichen und zu doppeltem Aufwand, um die Konsistenz zu gewährleisten.

Um dieses Problem anzugehen, verwendeten einige Kunden Pipelines oder Code zur Synchronisation ihrer Metastores und ACLs; andere richteten selbstverwaltete Metastores ein, die sie arbeitsbereichsübergreifend nutzten. Diese Lösungen verursachten jedoch zusätzlichen Aufwand und Wartungskosten, da sie vorausschauende Architekturentscheidungen zur Datenpartitionierung im Unternehmen erzwangen. Dabei entstanden Datensilos.

Data Governance mit Unity Catalog

Zur Überwindung dieser Beschränkungen wurde von Databricks Unity Catalog entwickelt. Unity Catalog soll die Implementierung von Data Governance vereinfachen und gleichzeitig die Möglichkeiten zur Zusammenarbeit und zur Datenfreigabe maximieren. Der erste Schritt dazu war ein gemeinsamer Namespace, der Zugriff auf alle Daten in einer Organisation ermöglicht.

Dieser Ansatz mag als Herausforderung für das bereits erwähnte verteilte Governance-Muster erscheinen, aber Unity Catalog bietet neue Isolationsmechanismen innerhalb des Namespace, die Unternehmen früher mit mehreren Hive-Metastores lösten. Diese Isolationsmechanismen erlauben es Gruppen, unabhängig voneinander zu operieren – mit minimaler oder keiner Interaktion – und bieten auch Isolation in anderen Szenarien, etwa zwischen Produktions- und Entwicklungsumgebungen.

Hive-Metastore im Vergleich zu Unity Catalog in Databricks

Bei Hive war ein Metastore eine Dienstgrenze, d. h., verschiedene Metastores erforderten separat gehostete zugrunde liegende Hive-Services und separate zugrunde liegende Datenbanken. Unity Catalog dagegen ist ein Plattformservice innerhalb der Databricks Data Intelligence Platform, sodass es keine Servicegrenzen zu beachten gilt.

Unity Catalog stellt einen gemeinsamen Namespace zur Verfügung, mit dem Sie Ihre Daten an einer zentralen Stelle verwalten und überwachen können.

Beim Einsatz von Hive nutzte man oft mehrere Metastores mit eigenen Namespaces, um Entwicklungs- und Produktionsumgebungen zu isolieren oder Daten zwischen Geschäftseinheiten zu trennen.

Unity Catalog löst diese Anforderungen mit dynamischen Isolierungsmechanismen für Namespaces, die gemeinsame Datennutzung und Zusammenarbeit nicht einschränken und keine vorgefassten, starren Architekturentscheidungen erfordern

Arbeiten in verschiedenen Teams und Umgebungen

Bei Verwendung einer Datenplattform besteht oft die Notwendigkeit, Umgebungen wie Entwicklung und Produktion oder aber Geschäftsgruppen, Teams oder operative Einheiten Ihres Unternehmens voneinander zu trennen.

Definieren wir zunächst einmal die Isolationsgrenzen bei einer Datenplattform wie Databricks:

- Benutzer dürfen nur entsprechend den vereinbarten Zugriffsregeln Zugang zu Daten erhalten.
- Daten können von ausgewählten Personen oder Teams verwaltet werden.
- Daten müssen bei der Speicherung physisch voneinander getrennt werden.
- Der Zugriff auf Daten darf nur in dafür vorgesehenen Umgebungen erfolgen.

Benutzer dürfen nur entsprechend den vereinbarten Zugriffsregeln Zugang zu Daten erhalten

Unternehmen haben oft strikte Anforderungen an den Datenzugriff, basierend auf organisatorischen oder gesetzlichen Vorgaben, die für die Datensicherheit essenziell sind. Typische Beispiele sind Gehaltsinformationen von Mitarbeitern oder Informationen über Kreditkartenzahlungen.

Der Zugriff auf solche Informationen ist streng reglementiert und unterliegt regelmäßigen Kontrollen. Unity Catalog bietet Unternehmen eine granulare Kontrolle über die Daten-Assets im Katalog, um diese Branchenstandards zu erfüllen. Mit den Kontrollen, die Unity Catalog zur Verfügung stellt, können Benutzer nur die Daten sehen und abfragen, zu deren Anzeige und Abfrage sie auch berechtigt sind.

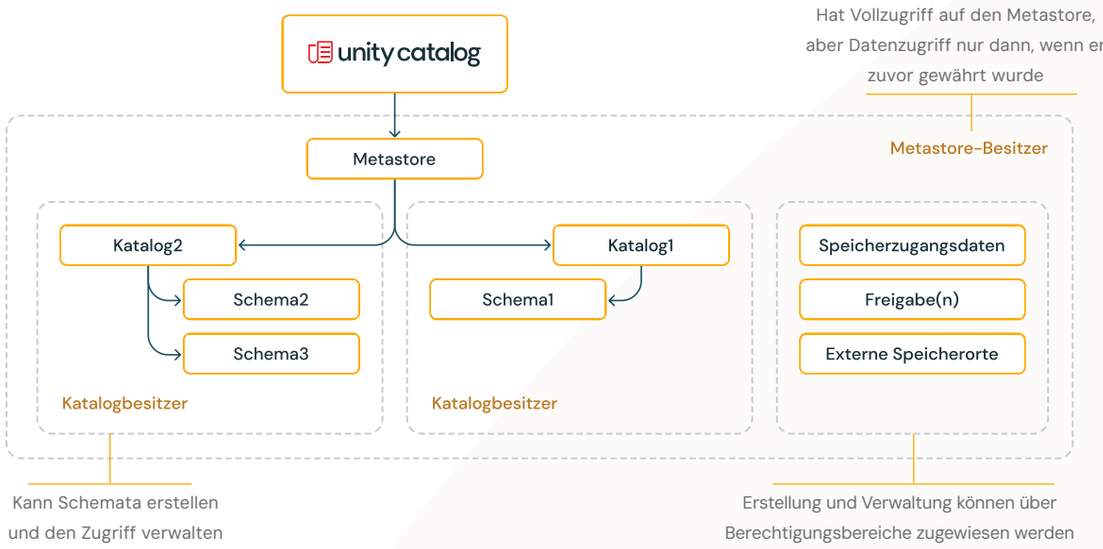
Daten können von ausgewählten Personen oder Teams verwaltet werden

Unity Catalog erlaubt es Ihnen, zwischen zentraler Governance und verteilten Governance-Modellen zu wählen.

Beim zentralisierten Governance-Modell sind Ihre Governance-Administratoren Inhaber des Metastore und können die Rechte an jedem Objekt übernehmen sowie ACLs und Richtlinien festlegen.

In einem verteilten Governance-Modell würde man dagegen einen oder mehrere Kataloge als Datendomäne betrachten. Der Inhaber dieses Katalogs kann alle Assets erstellen und besitzen und die Governance innerhalb dieser Domäne verwalten. Daher können die Inhaber von Domains unabhängig von anderen Inhabern anderer Domains agieren.

Wir empfehlen dringend, für beide Optionen eine Gruppe als Inhaber oder Service Principal festzulegen, sofern die Verwaltung über das Tooling erfolgt.



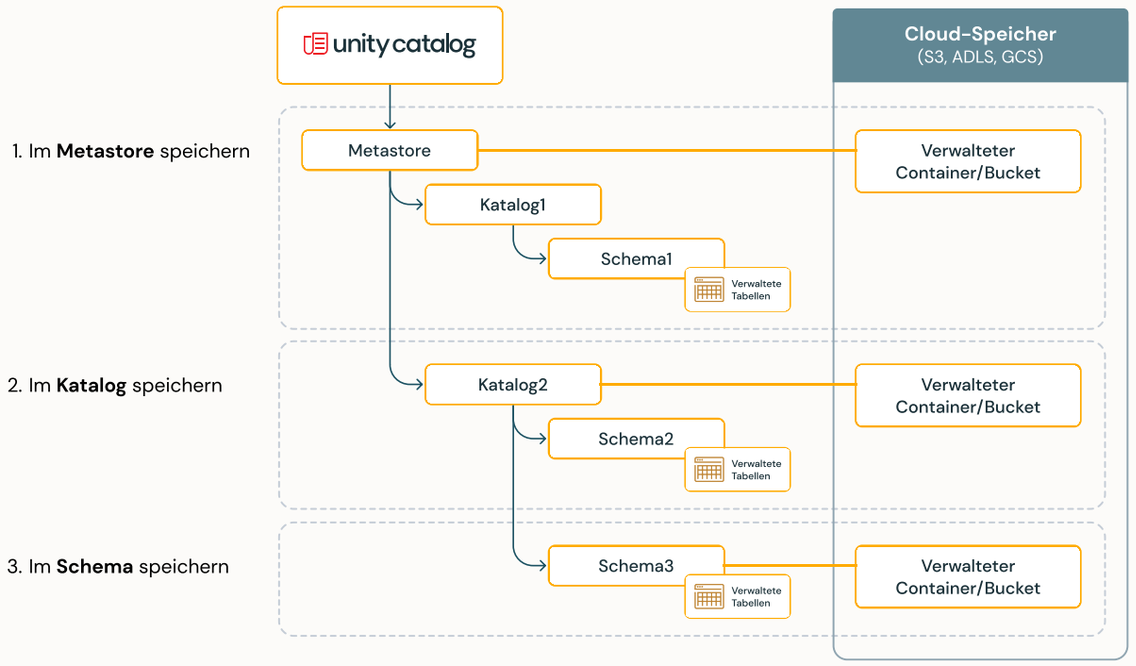
Daten müssen bei der Speicherung physisch voneinander getrennt werden

Beim Erstellen eines UC-Metastores legt der Databricks Account Admin standardmäßig genau einen Cloud-Speicherort mit Zugangsdaten als Default für verwaltete Tabellen fest.

Unternehmen, die aus regulatorischen Gründen oder etwa für SDLC-bereichsübergreifende Zwecke, zwischen Geschäftseinheiten oder für die Kostenverrechnung eine physische Datentrennung benötigen, sollten Funktionen verwalteter Datenquellen auf Katalogebene und Schemaebene in Betracht ziehen.

Bei Unity Catalog können Sie die Vorgaben für die Trennung der Daten im Speicher auswählen. Standardmäßig werden alle Daten im Metastore gespeichert. Mit Unterstützung für verwaltete Datenquellen in Katalogen und Schemata können Sie Datenspeicherung und -zugriff physisch isolieren. Das erleichtert es Ihrem Unternehmen, Governance- und Datenmanagementanforderungen zu erfüllen.

Beim Erstellen verwalteter Tabellen speichert Unity Catalog die Daten zuerst im Schemaspeicher (falls vorhanden), dann im Katalogspeicher (falls vorhanden). Der Metastore-Speicher kommt zum Einsatz, wenn die anderen beiden Speicherorte nicht festgelegt wurde.

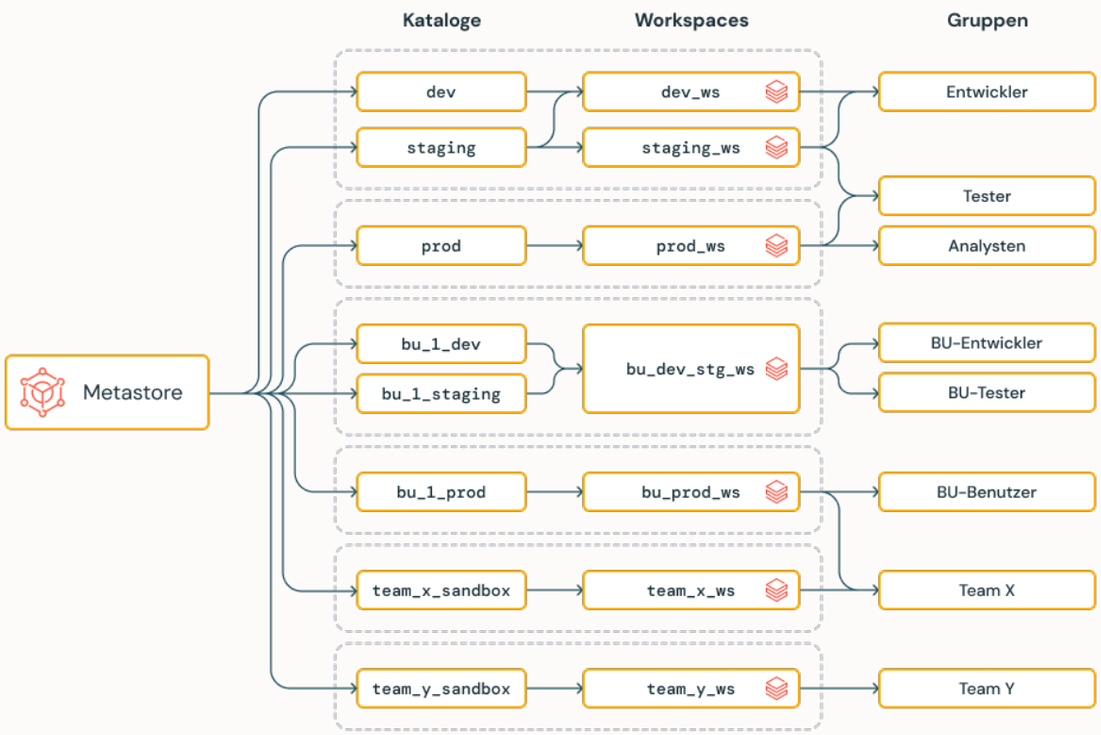


databricks

Der Datenzugriff darf je nach Zweck der Daten nur in bestimmten Umgebungen erfolgen

Häufig fordern organisatorische und Compliance-Anforderungen, dass bestimmte Daten nur in spezifischen Umgebungen zugänglich sein dürfen, in anderen jedoch nicht. Beispiele sind Entwicklungs- und Produktionsumgebungen oder HIPAA- bzw. PII-Umgebungen mit personenbezogenen Daten zur Analyse. Hier gelten spezielle Zugriffsregeln, die festlegen, wer auf die Daten zugreifen darf und in welchen Umgebungen das möglich ist. Manchmal schreiben die Anforderungen vor, dass bestimmte Datasets oder Datenbereiche nicht miteinander gekreuzt oder kombiniert werden dürfen.

In Databricks betrachten wir einen Arbeitsbereich als eine Umgebung. Unity Catalog verfügt über eine Funktion, mit der Sie Kataloge an Arbeitsbereiche „binden“ können. Mit diesen umgebungsbezogenen ACLs können Sie sicherstellen, dass nur bestimmte Kataloge innerhalb eines Arbeitsbereichs verfügbar sind – unabhängig von den individuellen ACLs eines Benutzers. Das bedeutet, der Metastore-Administrator oder der Katalogbesitzer kann die Arbeitsbereiche festlegen, von denen aus auf einen Datenkatalog zugegriffen werden kann. Dies kann über unsere Benutzeroberfläche oder über API/Terraform gesteuert werden, um Integrationen zu vereinfachen. Vor kurzem haben wir sogar einen Blogpost darüber veröffentlicht, wie sich Unity Catalog über Terraform steuern lässt, damit es Ihrem individuellen Governance-Modell gerecht wird.



Zugriff und Verfügbarkeit von Daten können über Arbeitsbereiche und Gruppen hinweg isoliert werden. Benutzer können nur auf bestimmte Kataloge in bestimmten Umgebungen zugreifen.

Fazit

Mit Unity Catalog im Zentrum Ihrer Lakehouse-Architektur können Sie eine flexible und skalierbare Governance-Implementierung erreichen, ohne Ihre Fähigkeit zur effektiven Verwaltung und Freigabe von Daten opfern zu müssen. Unity Catalog ermöglicht es Ihnen, die Grenzen und Beschränkungen Ihres bestehenden Hive-Metastores zu überwinden, sodass Sie Daten gemäß Ihren spezifischen Geschäftsanforderungen einfacher isolieren und gemeinsam bearbeiten können. Wie der Einstieg gelingt, beschreiben die Unity Catalog-Leitfäden (AWS, Azure). Laden Sie dieses kostenlose E-Book über Daten, Analysen und KI-Governance herunter und erfahren Sie mehr über Best Practices zum Aufbau einer effektiven Governance-Strategie für Ihr Data Lakehouse.

KAPITEL 3.6

Per Anhalter durch Datenberechtigungsmodell und Zugriffssteuerung in Unity Catalog

Wie Sie innerhalb des Unity Catalog-Berechtigungsmodells Konzepte auf einfache und verständliche Art entwickeln, um verschiedene Zugriffsanforderungen und -muster zu unterstützen

von Som Natarajan und Vuong Nguyen

Mit dem Zunahme von Datenvolumen, -geschwindigkeit und -vielfalt setzen Unternehmen zunehmend auf solide Data-Governance-Praktiken, um sicherzustellen, dass ihre wichtigsten Geschäftsziele angemessen erreicht werden. Unity Catalog ist eine präzise Governance-Lösung für Daten und KI, die auf der Databricks Data Intelligence Platform aufsetzt. Unity Catalog vereinfacht Sicherheit und Governance Ihrer Unternehmensdaten durch Bereitstellung eines zentralisierten Mechanismus zur Verwaltung und Prüfung des Datenzugriffs.

Denken wir einmal etwas zurück: Bevor Unity Catalog das Berechtigungsmodell für Dateien und Tabellen vereinheitlichte und die Unterstützung für alle Sprachen hinzufügte, mussten Kunden bei Databricks eine differenzierte Datenzugriffskontrolle unter Verwendung der alten Tabellen-ACLs (TACLs) auf Arbeitsplatzebene implementieren. Diese waren im Wesentlichen auf bestimmte Clusterkonfigurationen beschränkt und funktionierten nur für Python und SQL. Sowohl mit Unity Catalog als auch mit TACLs können Sie den Zugriff auf sicherheitsrelevante Objekte wie Kataloge, Schemata (Datenbanken), Tabellen und Sichten kontrollieren. Trotzdem sind einige Nuancen in der Funktionsweise der einzelnen Zugriffsmodelle zu beachten.

Ein gutes Verständnis des Objektzugriffsmodells ist für eine Implementierung von Data Governance im großen Stil mit Unity Catalog unerlässlich. Das gilt umso mehr, wenn Sie das Tabellen-ACL-Modell bereits implementiert haben und nun ein Upgrade auf Unity Catalog durchführen möchten, um von den neuesten Funktionen wie der Unterstützung mehrerer Sprachen, der zentralisierten Zugriffskontrolle und Datenherkunft zu profitieren.

Die Axiome des Unity Catalog-Zugriffsmodells

- Unity Catalog-Berechtigungen werden im Metastore definiert: Sie beziehen sich immer auf Identitäten auf Kontoebene – TACL-Berechtigungen, die im Katalog „hive_metastore“ definiert sind, immer auf die lokalen Identitäten im Arbeitsbereich.
- Berechtigungsvererbung. Objekte in Unity Catalog sind hierarchisch und die Berechtigungen werden von oben nach unten vererbt. Das höchste Objekt, von dem Berechtigungen geerbt werden, ist der Katalog.
- Objektbesitz ist wichtig. Berechtigungen können nur von einem Metastore-Administrator, dem Besitzer eines Objekts oder dem Besitzer des Katalogs oder Schemas, das dieses enthält, gewährt werden. Nur der Besitzer eines Objekts oder aber der Besitzer des Katalogs oder Schemas, das es enthält, kann das Objekt löschen.
- USE-Berechtigungen für Grenzen: Für die Interaktion mit Objekten in einem Katalog oder Schema ist USE CATALOG/SCHEMA erforderlich. Allerdings erlaubt es die USE-Berechtigung nicht, durch die Objektmetadaten zu navigieren, die sich im Katalog bzw. Schema befinden.
- Berechtigungen für abgeleitete Objekte werden vereinfacht: In Unity Catalog muss der Besitzer einer Sicht nur über die SELECT-Berechtigung verfügen, zusätzlich zu USE SCHEMA für das übergeordnete Schema oder USE CATALOG für den übergeordneten Katalog. Bei TACL dagegen muss der Besitzer einer Sicht auch Besitzer aller referenzierten Tabellen und Sichten sein.

Einige etwas komplexere Axiome

- Secure by Default: Nur Cluster mit Unity Catalog-spezifischen Zugriffsmodi (freigegeben oder Einzelbenutzer) können auf Unity Catalog-Daten zugreifen. Bei TACL dagegen haben alle Benutzer Zugriff auf alle Daten auf nicht freigegebenen Clustern.
- Beschränkung von Einzelbenutzer-Clustern: Einzelbenutzer-Cluster unterstützen keine dynamischen Sichten. Benutzer brauchen die SELECT-Berechtigung für alle referenzierten Tabellen und Sichten, um aus einer Sicht lesen zu können.
- Keine Unterstützung für ANY FILE oder ANONYMOUS FUNCTION: Unity Catalog unterstützt diese Berechtigungen nicht, da sie zur Umgehung von Zugriffsbeschränkungen verwendet werden könnten: Ein unbefugter Benutzer könnte hiermit privilegierten Code ausführen.

Interessante Muster

Es gibt viele Governance-Muster, die mit dem Unity Catalog-Zugriffsmodell umgesetzt werden können.

Beispiel 1: Arbeitsbereichsübergreifend konsistente Berechtigungen

Axiom 1 ermöglicht es dem Produktteam, in seinem eigenen Arbeitsbereich Berechtigungen für sein Datenprodukt zu definieren. Diese gelten dann in allen anderen Arbeitsbereichen, egal woher die Nutzer kommen.

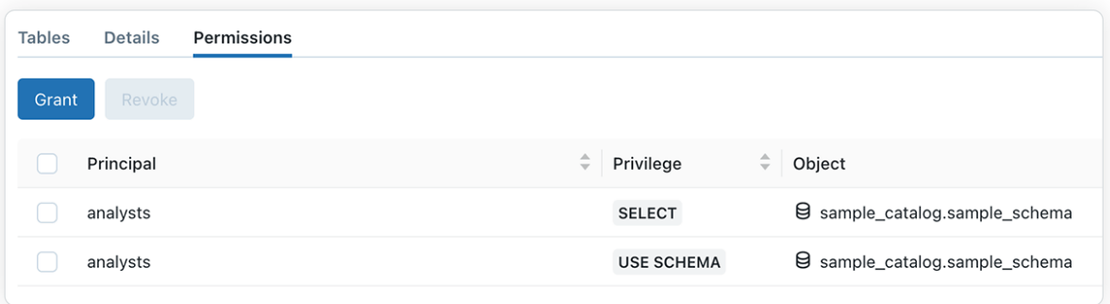
Beispiel 2: Festlegen von Grenzen für den Datenaustausch

Dank Axiom 2 können Besitzer von Katalogen oder Schemata Standardzugriffsregeln für ihre Daten einrichten. Mit den folgenden Befehlen kann das Machine-Learning-Team beispielsweise Tabellen in einem Schema erstellen und die Tabellen der jeweils anderen lesen:

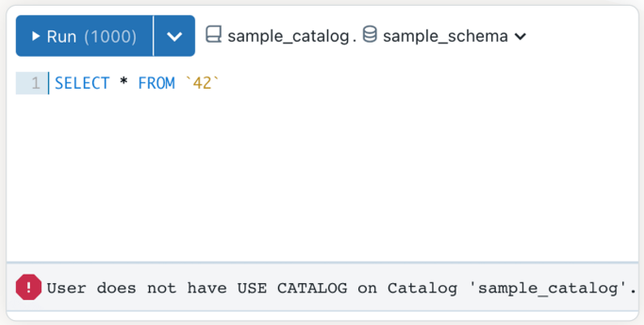
```
CREATE CATALOG ml;
CREATE SCHEMA ml.sandbox;
GRANT USE_CATALOG ON CATALOG ml TO ml_users;
GRANT USE_SCHEMA ON SCHEMA ml.sandbox TO ml_users;
GRANT CREATE TABLE ON SCHEMA ml.sandbox TO ml_users;
GRANT SELECT ON SCHEMA ml.sandbox TO ml_users;
```

Axiom 4 geht noch einen Schritt weiter: Es ermöglicht Besitzern von Katalogen bzw. Schemata, die Weitergabe der von einzelnen Schema-/Tabellenbesitzern erstellten Daten einzuschränken. Auch wenn ein Tabellenbesitzer einem anderen Nutzer das SELECT-Recht gibt, kann dieser nur dann Lesezugriff auf die Tabelle erhalten, wenn er auch USE CATALOG-Rechte für den übergeordneten Katalog und USE SCHEMA-Rechte für das übergeordnete Schema hat.

Im folgenden Beispiel gehört sample_catalog Benutzer A, während Benutzer B das Schema sample_schema und Tabelle 42 erstellt hat. Obwohl dem Analystenteam die Berechtigungen USE SCHEMA und SELECT erteilt wurden, können sie die Tabelle aufgrund der von Benutzer A festgelegten Berechtigungsgrenze nicht abfragen.



Die Berechtigungsseite zeigt, dass die Analystengruppe die Berechtigungen SELECT und USE SCHEMA für sample_catalog.sample_schema besitzt.

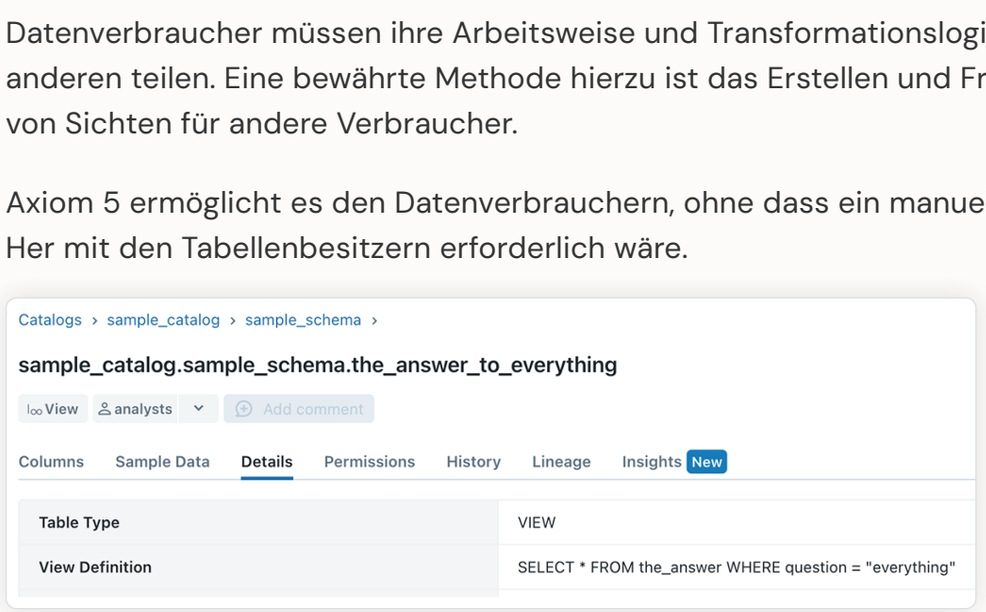


Abfrageseite mit Fehlermeldung

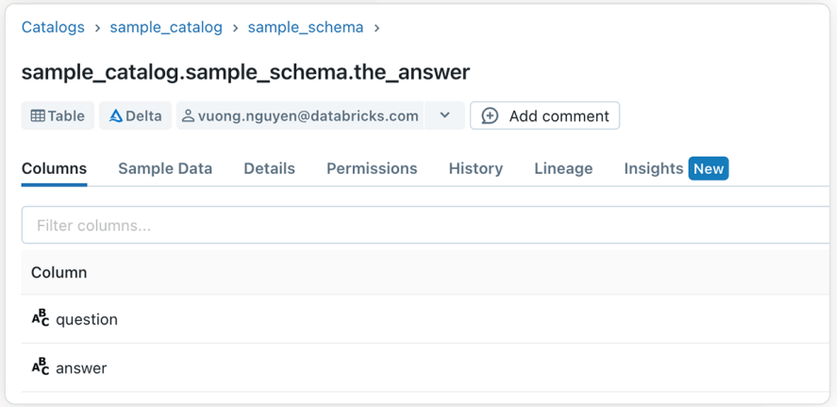
Beispiel 3: Einfachere Freigabe von Geschäftslogik

Datenverbraucher müssen ihre Arbeitsweise und Transformationslogik mit anderen teilen. Eine bewährte Methode hierzu ist das Erstellen und Freigeben von Sichten für andere Verbraucher.

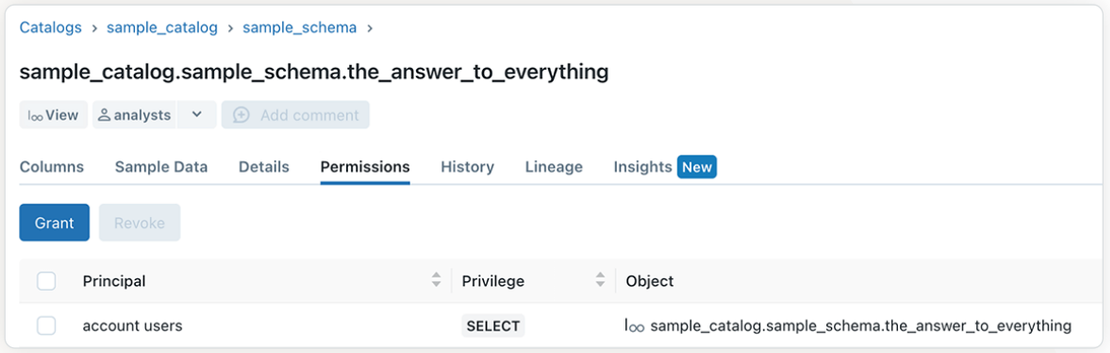
Axiom 5 ermöglicht es den Datenverbrauchern, ohne dass ein manuelles Hin und Her mit den Tabellenbesitzern erforderlich wäre.



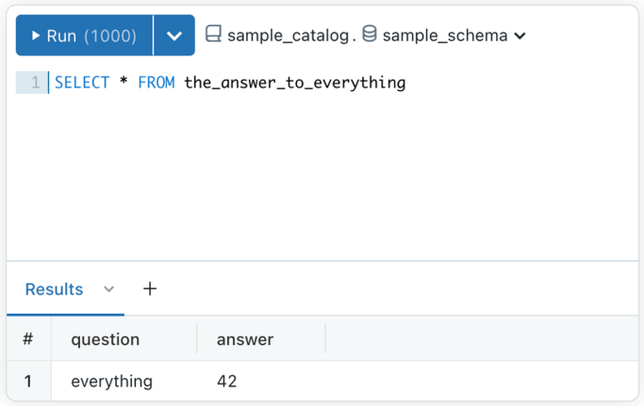
Definition der Sicht



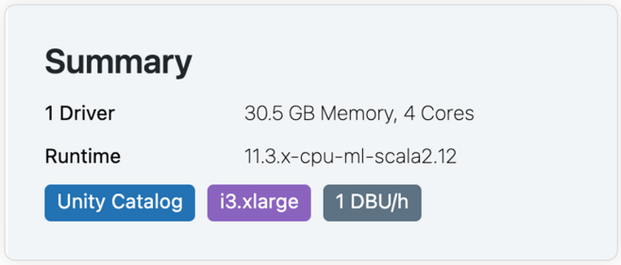
Besitz der Tabelle



Berechtigungsseite mit einer Sicht im Besitz der Analystengruppe sowie einer Kontonutzergruppe mit SELECT-Berechtigung



Abfrageseite, die das Ergebnis der Antwort auf alles anzeigt



Clusterübersicht mit Angabe der Unity Catalog-Unterstützung

Beispiel 4: Nie wieder Datenverluste

Axiom 6 gibt Datenbesitzern die Sicherheit, dass Fehlkonfigurationen des Clusters keinen unbefugten Datenzugriff ermöglichen. Cluster, auf denen nicht der richtige Zugriffsmodus konfiguriert ist, können nicht auf die Daten in Unity Catalog zugreifen.

Benutzer können mithilfe dieses praktischen Tooltips auf der Seite „Create Clusters“ (Cluster erstellen) überprüfen, ob ihre Cluster auf die Daten in Unity Catalog zugreifen können.

Da Datenbesitzer das Datenberechtigungsmodell und die Zugriffskontrolle jetzt nachvollziehen können, können sie die Verwaltung von Zugriffsrichtlinien mithilfe von Unity Catalog umfassend vereinfachen.

Es sind weitere Funktionen geplant, die es Datenadministratoren und -besitzern ermöglichen werden, noch komplexere Zugriffsrichtlinien zu erstellen:

- Zeilenfilterung und Spaltenmaskierung: Mithilfe von SQL-Standardfunktionen lassen sich Zeilenfilter und Spaltenmasken definieren, die differenzierte Zugriffskontrollen für Zeilen und Spalten ermöglichen.
- Attributbasierte Zugriffskontrollen: Definieren Sie Zugriffsrichtlinien auf der Grundlage von Tags (Attributen) in Ihren Daten-Assets.

KAPITEL

04

Analyseanwendungsfälle auf Databricks

KAPITEL 4.1

Wie man mit Fivetran und dbt auf Databricks eine Marketinganalyselösung entwickelt

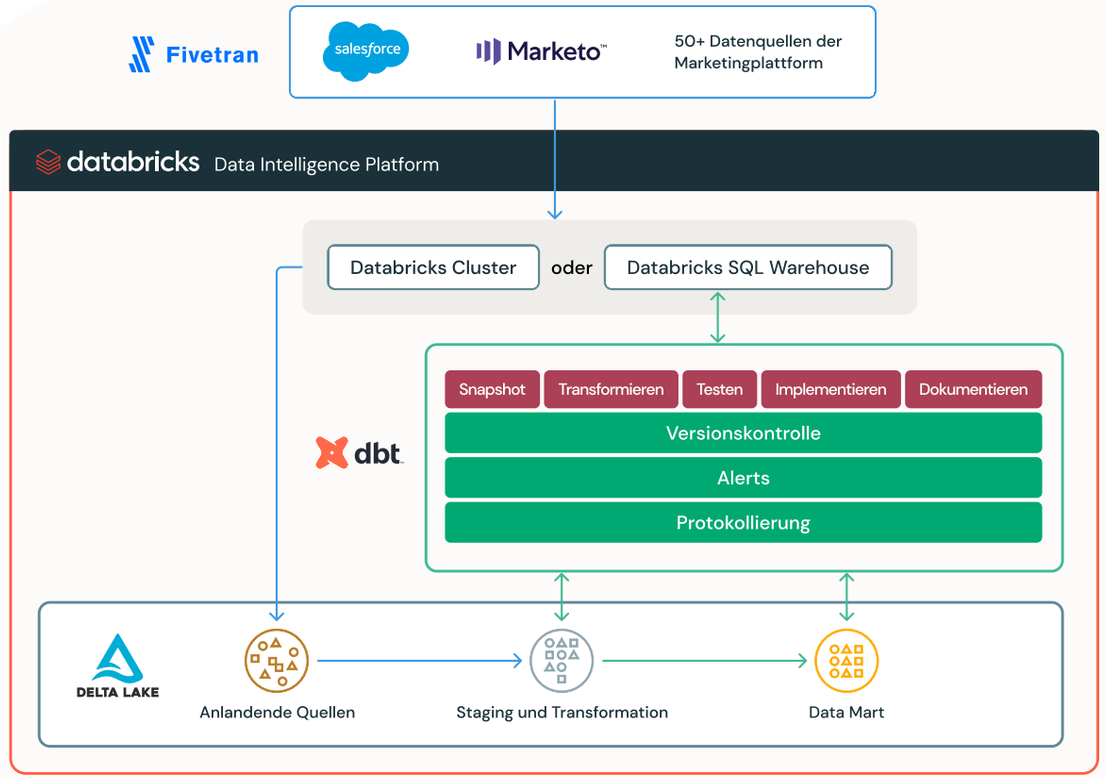
von Tahir Fayyaz, Bilal Aslam und Robert Saxby

Marketingteams nutzen für ihre Marketing- und Vertriebskampagnen viele verschiedene Plattformen. Das führt zu einer beträchtlichen Menge an wertvollen, jedoch nicht miteinander verbundenen Daten. Das Zusammenführen dieser Daten kann eine hohe Rendite bringen, wie das Beispiel der Publicis Groupe zeigt, die ihre Kampagneneinnahmen um bis zu 50 % steigern konnte.

Databricks vereint Data Warehousing und KI auf einer Plattform. Es ist damit ideal für die Entwicklung einer Marketinganalyse-Lösung: Wir pflegen eine Single-Source-of-Truth und erschließen KI- und ML-Anwendungsfälle. Ferner nutzen wir zwei Partnerlösungen von Databricks – Fivetran und dbt –, um eine breite Palette von Marketinganalyseanwendungen zu erschließen. Dazu gehören Abwanderungs- und Lifetime-Value-Analysen, Kundensegmentierung und Werbeeffizienz.

Mit Fivetran können Sie ganz einfach Daten von über 50 Marketingplattformen in Delta Lake einspeisen, ohne komplexe Pipelines erstellen und pflegen zu müssen. Sollte sich eine der Marketingplattform-APIs ändern oder ausfallen, kümmert sich Fivetran um die Aktualisierung und Reparatur der Integrationen, damit Sie Ihre Marketingdaten auch künftig nutzen können.

dbt ist ein beliebtes Open-Source-Framework, mit dem Lakehouse-Benutzer mithilfe von einfachem SQL Datenpipelines erstellen können. Alles ist im Plain-Text-Format in Verzeichnissen organisiert, was Versionskontrolle, Bereitstellung und Testmöglichkeiten vereinfacht. Sobald die Daten in Delta Lake eingelesen sind, verwenden wir dbt, um sie zu transformieren, zu testen und zu dokumentieren. Der Data Mart mit den transformierten Marketinganalysedaten



Fivetran und dbt können über einen Databricks-Cluster oder ein Databricks SQL-Warehouse in Delta Lake lesen und schreiben.

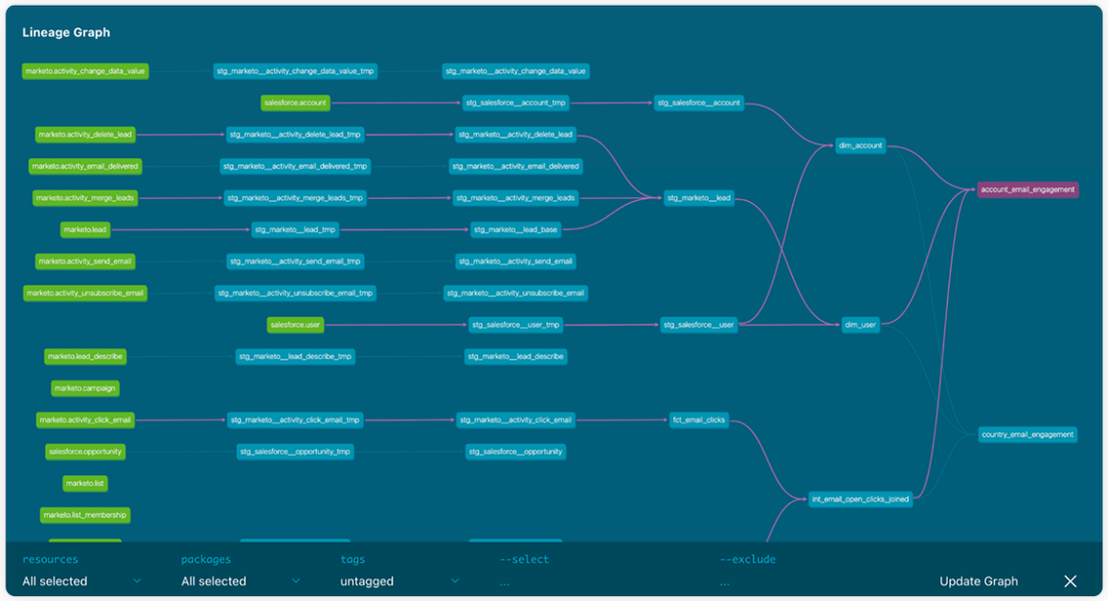
setzt auf den eingelesenen Daten auf und kann dann seinerseits verwendet werden, um neue Marketingkampagnen und -initiativen zu unterstützen.

Sowohl Fivetran als auch dbt sind Teil von Databricks Partner Connect, einem zentralen Portal, über das Sie Daten, Analyse- und KI-Tools direkt auf der Databricks Platform kennenlernen und sicher vernetzen können. Mit nur wenigen Klicks können Sie diese Tools (wie auch viele weitere) direkt in Ihrem Databricks-Arbeitsbereich konfigurieren und verbinden.

So entwickeln Sie eine Marketinganalyselösung

In dieser praktischen Demo zeigen wir Ihnen, wie Sie Marketo- und Salesforce-Daten mit Fivetran in Databricks einlesen und dann dbt verwenden, um Ihr Datenmodell für die Marketinganalyse zu transformieren, zu testen und zu dokumentieren.

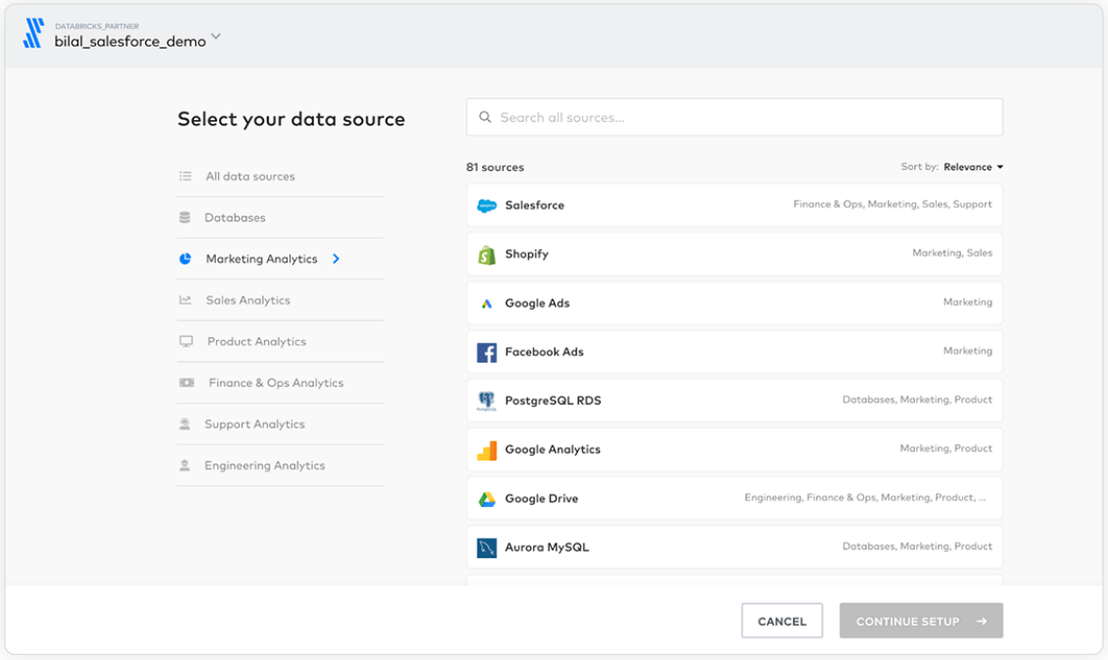
Der gesamte Code für die Demo ist auf GitHub im Repository workflows-examples verfügbar.



dbt-Herkunftsdiagramm mit Datenquellen und Modellen

Das fertige dbt-Modellherkunftsdiagramm wird wie folgt aussehen. Die Fivetran-Quelltabellen sind auf der linken Seite grün dargestellt, die endgültigen Marketinganalysemodelle sind rechts abgebildet. Wenn Sie ein Modell auswählen, sehen Sie die entsprechenden Abhängigkeiten, wobei die verschiedenen Modelle lila hervorgehoben sind.

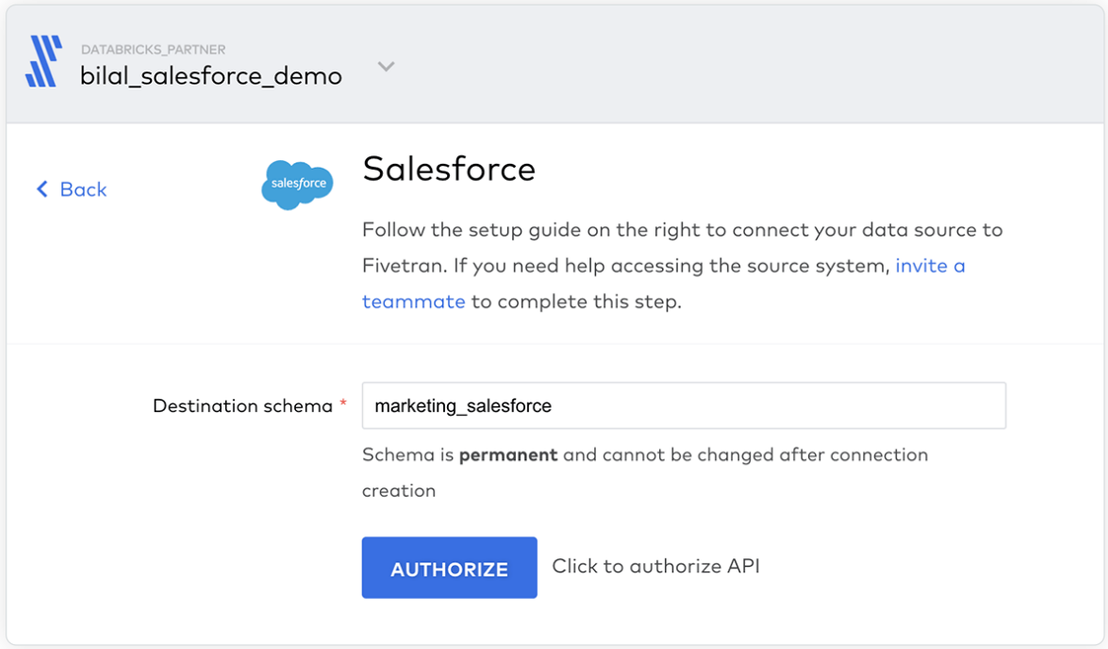
Datenerfassung mit Fivetran



Fivetran bietet zahlreiche Connectors für Marketinganalysedatenquellen.

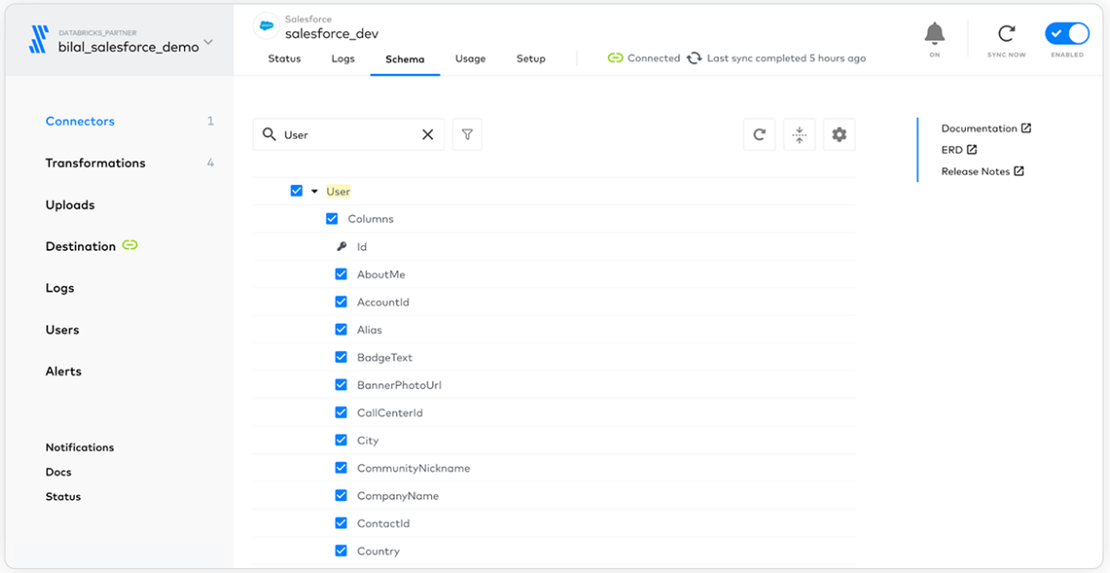
databricks

Erstellen Sie neue Salesforce- und Marketo-Verbindungen in Fivetran, um Marketingdaten in Delta Lake einzulesen Beim Anlegen der Verbindungen erstellt und verwaltet Fivetran auch automatisch ein Schema für jede Datenquelle in Delta Lake. Später werden wir dbt zum Transformieren, Bereinigen und Aggregieren dieser Daten verwenden.



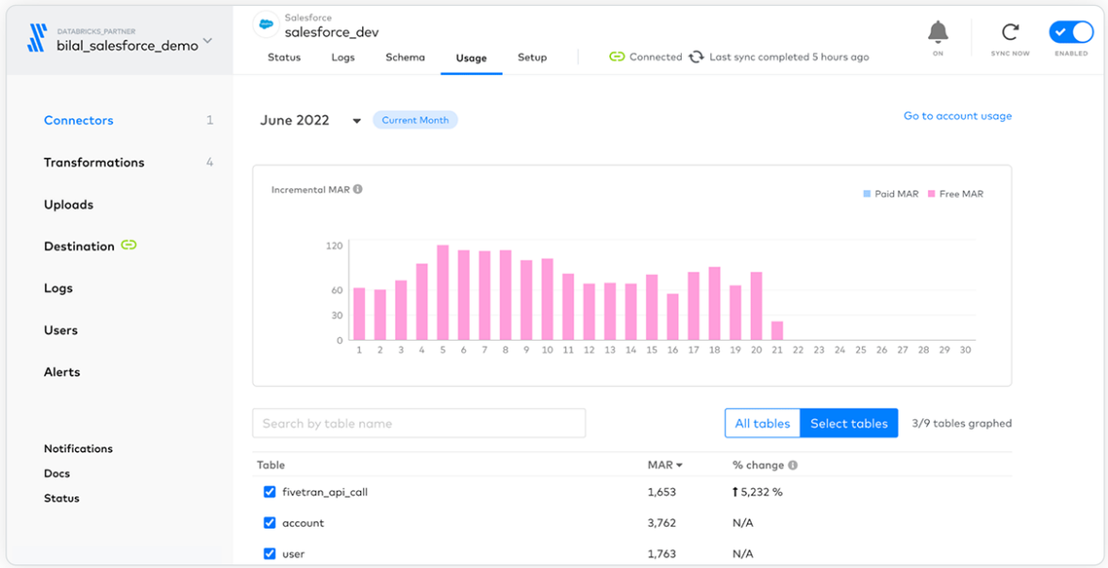
Definieren eines Zielschemas für die Salesforce-Datenquelle in Delta Lake

Für die Demo benennen Sie die Schemata, die in Delta Lake erstellt werden, „marketing_salesforce“ bzw. „marketing_marketo“. Wenn die Schemata noch nicht vorhanden sind, erstellt Fivetran sie im Zuge des ersten Einlesens.



Auswählen, welche Datenquellenobjekte als Delta Lake-Tabellen synchronisiert werden sollen

Danach können Sie auswählen, welche Objekte mit Delta Lake synchronisiert werden sollen, wobei jedes Objekt als einzelne Tabelle gespeichert wird. Fivetran erleichtert auch das Verwalten und Anzeigen der Spalten, die für jede Tabelle synchronisiert werden:



Monitoring-Dashboard in Fivetran zur Überwachung der synchronisierten monatlich aktiven Zeilen

Darüber hinaus bietet Fivetran ein Monitoring-Dashboard, mit dem Sie analysieren können, wie viele monatlich aktive Datenzeilen in jeder Tabelle pro Tag bzw. pro Monat synchronisiert werden, und weiteren nützlichen Statistiken und Logs.

Datenmodellierung mit dbt

Nachdem sich nun alle Marketingdaten in Delta Lake befinden, können Sie mithilfe von dbt Ihr Datenmodell erstellen. Führen Sie dazu die folgenden Schritte aus:

dbt-Projekt lokal einrichten und mit Databricks SQL vernetzen

Befolgen Sie die Anweisungen zum Einrichten von dbt Core und dbt-databricks, um Ihre lokale dbt-Entwicklungsumgebung in der IDE Ihrer Wahl einzurichten.

Erstellen Sie ein neues dbt-Projekt und stellen Sie mit dbt init eine Verbindung mit einem Databricks SQL-Warehouse her. Dabei werden die folgenden Informationen abgefragt.

```
$ 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]:
```

Wenn Sie das Profil konfiguriert haben, können Sie die Verbindung wie folgt testen:

```
$ dbt debug
```

Installieren der Fivetran dbt-Modellpakete für das Staging

Der erste Schritt bei der Verwendung der Marketo- und Salesforce-Daten: die Tabellen als Quellen für unser Modell erstellen. Glücklicherweise hat Fivetran mit seinen vorkonfigurierten Fivetran-dbt-Modellpaketen dafür gesorgt, dass das leicht zu realisieren ist. Für diese Demo verwenden wir die Pakete marketo_source und `salesforce_source`.

Zum Installieren der Pakete fügen Sie einfach eine **`packages.yml`**-Datei in das Stammverzeichnis Ihres dbt-Projekts ein und die Pakete `marketo-source`, `salesforce-source` und `fivetran-utils` hinzu:

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

To download and use the packages run
```

```
$ dbt deps
```

Sie sollten nun die installierten Fivetran-Pakete im Ordner packages sehen.

Aktualisieren von dbt_project.yml für Fivetran-dbt-Modelle

Es gibt einige paar Konfigurationseinstellungen in der Datei `dbt_project.yml`, die Sie ändern müssen, damit die Fivetran-Pakete korrekt mit Databricks funktionieren.

Sie finden die Datei `dbt_project.yml` im Stammordner Ihres dbt-Projekts.

spark_utils überschreibt dbt_utils-Makros

Die Fivetran-dbt-Modelle verwenden Makros aus dem dbt_utils-Paket. Einige dieser Makros müssen jedoch für die Nutzung mit Databricks modifiziert werden, was mit dem spark_utils-Paket leicht möglich ist.

Zu diesem Zweck werden Parameter für bestimmte `dbt_utils`-Makros angegeben, die Sie über dispatch config in der Datei `dbt_project.yml` einstellen können. Damit sucht dbt bei der Auflösung von Makros aus dem `dbt_utils`-Namespace zunächst nach Makros im `spark_utils`-Paket.

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

Variablen für die Schemata marketo_source und salesforce_source

Für die Fivetran-Pakete müssen Sie den Katalog (in dbt als Datenbank bezeichnet) und das Schema definieren, in dem die Daten gespeichert werden, wenn sie von Fivetran eingelesen werden.

Fügen Sie diese Variablen der Datei `dbt_project.yml` hinzu und geben Sie dabei die korrekten Namen für Katalog und Schema an. Der Standardkatalog heißt `hive_metastore`. Er wird verwendet, wenn `_database` freigelassen wurde. Die Schemanamen sind diejenigen, die Sie beim Erstellen der Verbindungen in Fivetran definiert haben.

```
vars:
 marketo_source:
   marketo_database: # leave blank to use the default hive_metastore catalog
   marketo_schema: marketing_marketo
 salesforce_source:
   salesforce_database: # leave blank to use the default hive_metastore catalog
   salesforce_schema: marketing_salesforce
```

Zielschema für Fivetran-Staging-Modelle

Um zu vermeiden, dass Staging-Tabellen der Fivetran-Quellmodelle im Standard-Zielschema landen, ist es ratsam, ein separates Staging-Schema festzulegen.

Fügen Sie in der Datei `dbt_project.yml` den Namen des Staging-Schemas hinzu. Dieses wird dann als Suffix an den Namen des Standardschemas angehängt.

```
models:
 marketo_source:
   +schema: your_staging_name # leave blank to use the default target_schema
 salesforce_source:
   +schema: your_staging_name # leave blank to use the default target_schema
```

Wenn Sie wie oben beschrieben vorgehen und Ihr in `profiles.yml` definiertes Zielschema **`mkt_analytics`** heißt, heißt das für die Tabellen `marketo_source` und `salesforce_source` verwendete Schema **`mkt_analytics_your_staging_name`**.

Deaktivieren fehlender Tabellen

An dieser Stelle können Sie die Fivetran-Modellpakete ausführen, um zu testen, ob sie korrekt funktionieren.

```
dbt run -select marketo_source
```

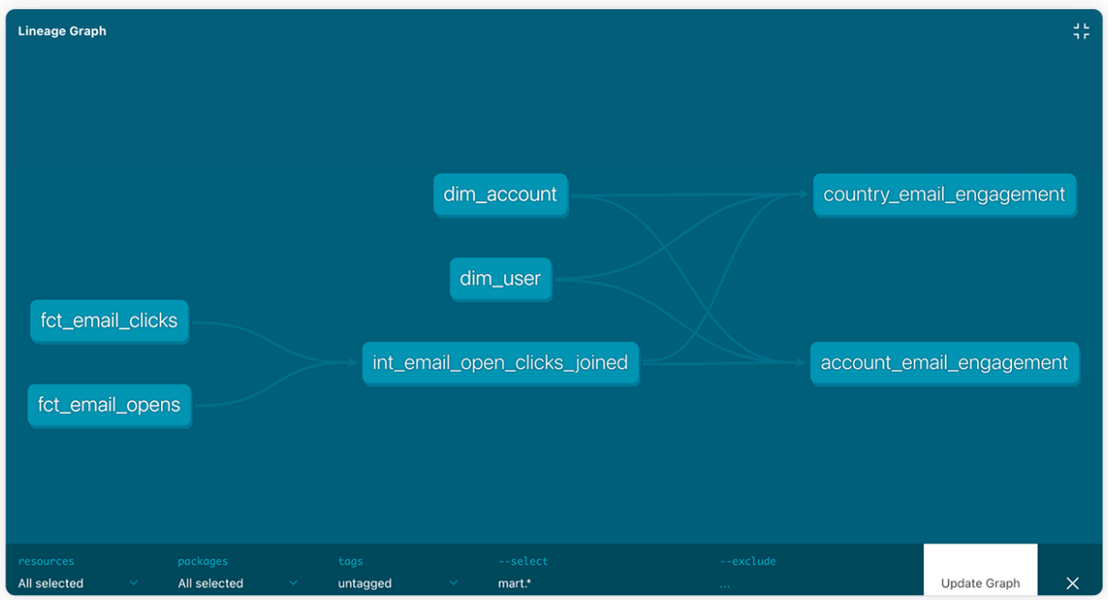
```
dbt run -select marketo_source
```

Wenn Modelle wegen nicht synchronisierter Tabellen in Fivetran fehlschlagen, können Sie diese im Quellschema deaktivieren, indem Sie die Datei **`dbt_project.yml`** entsprechend ändern.

Fehlen etwa die Tabellen für abgewiesene E-Mails und E-Mail-Vorlagen im Marketo-Quellschema, können Sie die Modelle für diese Tabellen deaktivieren, indem Sie unter der Modellkonfiguration Folgendes hinzufügen:

```
models:
 marketo_source:
   +schema: your_staging_name
   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
```

Entwickeln der Marketinganalysemodelle



dbt-Herkunftsdiagramm, das das Sternschema und das Datenmodell der aggregierten Tabellen zeigt

Nachdem die Fivetran-Pakete das Erstellen und Testen der Staging-Modelle erledigt haben, können Sie nun mit der Entwicklung der Datenmodelle für Ihre Marketinganalyseanwendungen beginnen. Bei diesen handelt es sich um ein Sternschema-Datenmodell und materialisierte aggregierte Tabellen.

Für das erste Marketinganalyse-Dashboard möchten Sie beispielsweise herausfinden, wie aktiv bestimmte Unternehmen und Vertriebsregionen sind, und zwar anhand der Anzahl der E-Mail-Kampagnen, die sie geöffnet und angeklickt haben.

Dazu können Sie die Salesforce- und Marketo-Tabellen über Salesforce-Benutzer-E-Mail, Salesforce-`account_id` und Marketo `lead_id` per Join verknüpfen.

Die Modelle werden unter dem Ordner mart folgendermaßen strukturiert.

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

Sie können den Code für alle Modelle auf Github im Verzeichnis /models/mart einsehen. Im Folgenden wird beschrieben, was sich in den einzelnen Ordnern befindet, zusammen mit einem Beispiel.

Kernmodelle

Die Kernmodelle sind die Fakten- und Dimensionstabellen, auf denen alle nachgelagerten Modelle aufbauen.

Der dbt-SQL-Code für das Modell `dim_user` sieht wie folgt aus:

```
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
```

Sie können auch Dokumentation und Tests für die Modelle hinzufügen, indem Sie eine YAML-Datei im Ordner ablegen.

In der Datei `core.yml` wurden zwei einfache Tests hinzugefügt.

```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
```

Zwischenmodelle

Einige der endgültigen nachgelagerten Modelle stützen sich möglicherweise auf dieselben berechneten Metriken. Um die Wiederholung von SQL-Code zu vermeiden, können Sie Zwischenmodelle erstellen, die wiederverwendet werden können.

Hier der dbt-SQL-Code für das Modell `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
```

Marketinganalysemodelle

Das sind die fertigen Marketinganalysemodelle, die für die Dashboards und Berichte der Marketing- und Vertriebsteams verwendet werden.

Hier der dbt-SQL-Code für das Modell `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
```

dbt-Modelle ausführen und testen

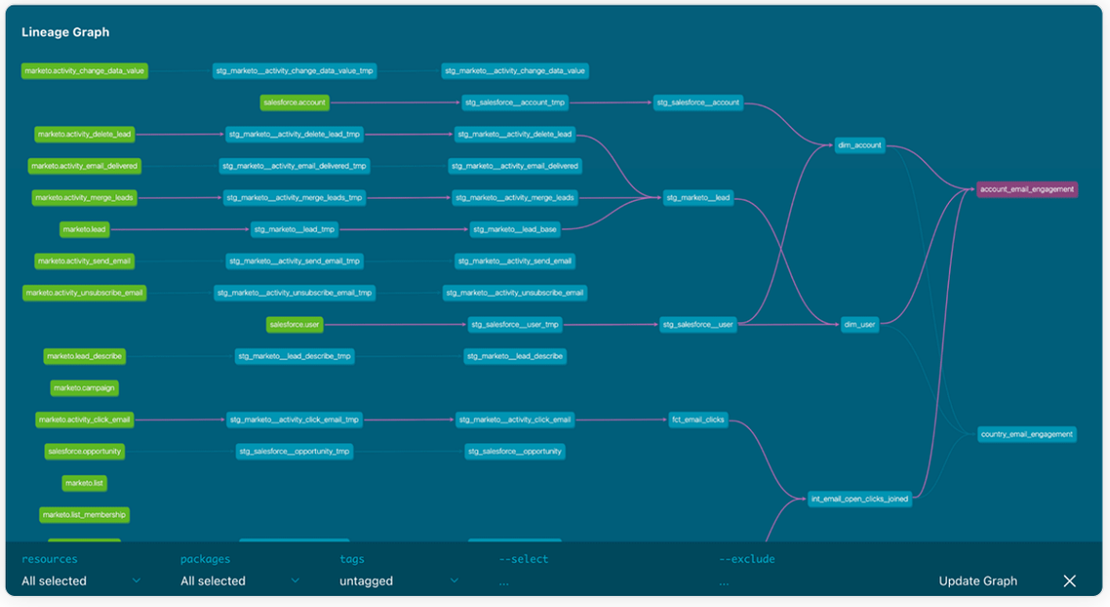
Da Ihr Modell nun fertig ist, können Sie alle Modelle wie folgt ausführen:

```
dbt run
```

Danach führen Sie die Tests wie folgt aus:

```
dbt test
```

dbt-Dokumente und Herkunftsdiagramm anzeigen



dbt-Herkunftsdiagramm für das Marketinganalysemodell

Wenn Ihre Modelle erfolgreich ausgeführt wurden, können Sie die Dokumente und das Herkunftsdiagramm wie folgt erstellen:

```
$ dbt docs generate
```

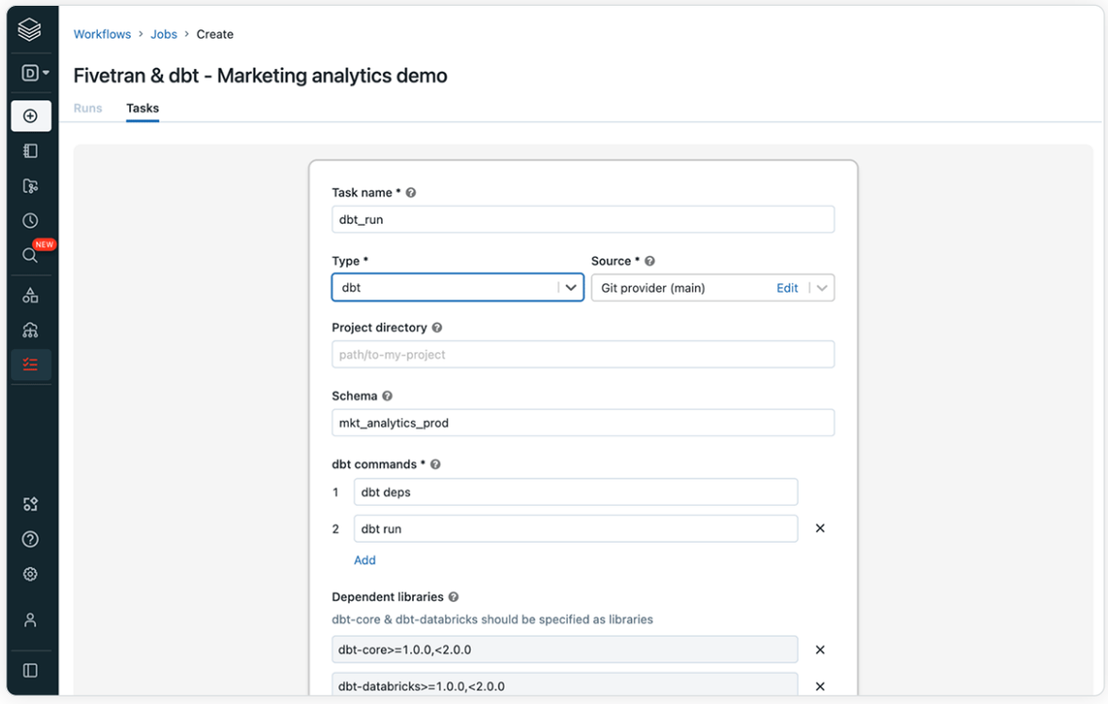
Danach führen Sie Folgendes aus, um sie lokal anzuzeigen:

```
$ dbt docs serve
```

dbt-Modelle in die Produktion überführen

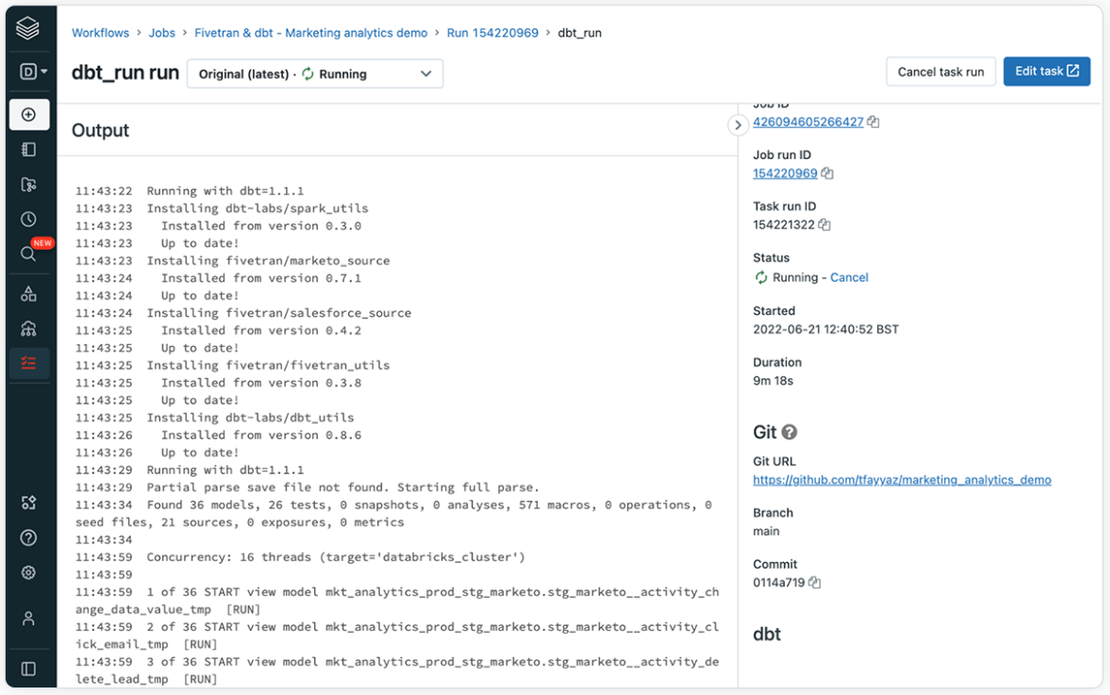
Sobald Sie Ihr dbt-Modell lokal entwickelt und getestet haben, stehen Ihnen mehrere Optionen für die Bereitstellung in der Produktion zur Verfügung. Eine davon ist der neue dbt-Tasktyp in Databricks Workflows (in der privaten Vorschau).

Ihr dbt-Projekt sollte in einem Git-Repository verwaltet und versioniert werden. Sie können einen dbt-Task in Ihrem Databricks Workflows-Job erstellen, der auf das Git-Repository verweist.



Verwenden eines dbt-Tasktyps in Databricks Workflows zur Orchestrierung von dbt

Da Sie Pakete in Ihrem dbt-Projekt verwenden, sollte der erste Befehl dbt deps sein, gefolgt von dbt run für den ersten Task und anschließend dbt test für den nächsten Task.



Einsehen der dbt-Logs für jede dbt-Ausführung

Für jede Ausführung können Sie die Logs für den jeweiligen dbt-Befehl anzeigen. Dies hilft bei der Fehlersuche und der Behebung von Problemen.

Leistungsstarke Marketinganalysen mit Fivetran und dbt

Wie hier gezeigt, können Sie mit Fivetran und dbt in Verbindung mit der Databricks Data Intelligence Platform problemlos eine leistungsstarke Marketinganalyselösung aufbauen, die einfach einzurichten und zu verwalten sowie flexibel genug ist, um alle Ihre Datenmodellierungsanforderungen zu erfüllen.

Für den schnellen Einstieg in die Erstellung Ihrer eigenen Lösung besuchen Sie die Dokumentation zur Integration von Fivetran und dbt in Databricks und verwenden das Beispielprojekt marketing_analytics_demo.

Der dbt-Tasktyp in Databricks Workflows befindet sich in der privaten Vorschau. Zum Ausprobieren des dbt-Tasktyps wenden Sie sich bitte an Ihren Databricks-Kundenbetreuer.

KAPITEL 4.2

Forderungsautomatisierung mit Databricks

Smart Claims steigert die Effizienz durch die Automatisierung aller Aspekte der Schadensbearbeitung – von der Erfassung über die Analyse bis hin zur Entscheidungsfindung.

von Anindita Mahapatra und Marzi Rasooli

Laut den jüngsten Berichten der globalen Beratungsfirma EY wird die Zukunft des Versicherungswesens zunehmend datengesteuert und analytisch sein. Die jüngste Fokussierung auf die Cloud hat den Zugang zu fortschrittlicher technologischer Infrastruktur verbessert. Trotzdem brauchen die meisten Unternehmen noch immer Hilfe bei der Implementierung und Nutzung dieser Funktionen. Es wird Zeit, den Schwerpunkt auf die Operationalisierung von Services zu verlagern, um Mehrwert zu erzielen.

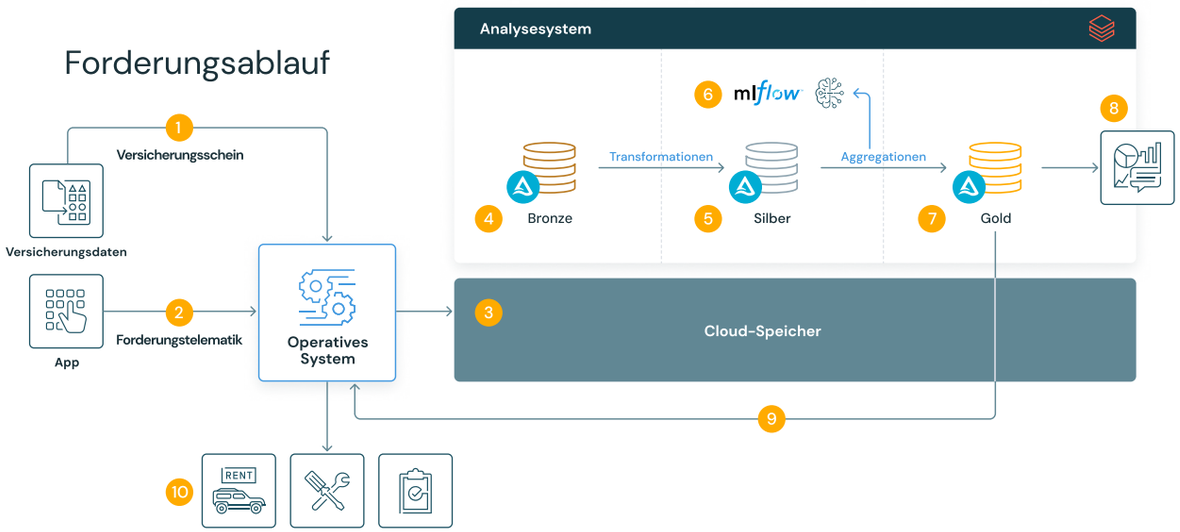
In der aktuellen Wirtschaftslage stehen Versicherungsunternehmen vor immer neuen Herausforderungen. Sie fühlen sich gezwungen, Daten zu ihrem Vorteil zu nutzen und Innovationen schneller voranzutreiben. Für Privatkundenversicherer im Bereich Sach- und Unfallversicherung bedeutet dies eine verstärkte Konzentration auf Personalisierung und Kundenbindung. Die Markentreue ist auf einem historischen Tiefststand, denn die Kunden suchen ständig nach günstigeren Tarifen und einem besseren Gesamtangebot. Das erhöht das Abwanderungsrisiko. Gleichzeitig schmälert die Zunahme beim Versicherungsbetrug die Margen. Versicherer müssen neue Wege finden, Kosten zu senken und Risiken besser zu bewältigen.

Durch Automatisierung und Optimierung der Schadensabwicklung lassen sich Kosten merklich senken: Versicherer sparen Zeit und benötigen weniger Humankapital. Zusätzlich können die gezielte Auswertung von Daten und der Einsatz fortschrittlicher Analysemethoden das Gesamtrisiko deutlich mindern.

Die Motivation für den „Smart Claims"-Lösungsbeschleuniger liegt auf der Hand: mit dem Lakehouse-Ansatz den Prozess der Schadensbearbeitung so verbessern, dass eine zügigere Abwicklung möglich ist, Bearbeitungskosten sinken und rascher Einblick in möglicherweise betrügerische Forderungen gewonnen wird. Die Implementierung des Lakehouse-Paradigmas vereinfacht die aktuelle Architekturlandschaft und schafft die Voraussetzungen für eine zukünftige Erweiterung des Unternehmens. Die zugehörigen Assets finden Sie hier.

Referenzarchitektur und Workflow

Ein typischer Workflow in der Schadensbearbeitung beinhaltet ein gewisses Maß an Orchestrierung zwischen operativen Systemen wie Guidewire und Analysesystemen wie Databricks. Das nachstehende Diagramm zeigt ein Beispiel für einen solchen Workflow bei einem Kfz-Versicherer.



Smart Claims-Referenzarchitektur und -Workflow

Die Automatisierung und Optimierung der Schadensbearbeitung erfordert ein tiefes Verständnis der Kundeninteraktion mit den operativen Systemen und den verschiedenen Informationsquellen, die für die Analyse zur Verfügung stehen.

In diesem Beispiel gehen wir davon aus, dass die Kunden hauptsächlich über eine mobile Anwendung interagieren und von dort aus Schäden einreichen und den Status bestehender Fälle überwachen können. Dieser Touchpoint bietet wichtige Informationen über das Kundenverhalten. Eine weitere wichtige Informationsquelle sind IoT-Geräte, die in Kundenfahrzeugen installiert sind. Telematikdaten können an die Betriebs- und Analysesysteme weitergeleitet werden und liefern so wertvolle Einblicke in das Fahrverhalten und die Fahrmuster der Kunden. Weitere externe Datenquellen können Wetter- und Straßendaten sein, die die traditionellen Datenkategorien wie Fahrzeugmerkmale (Marke, Modell, Baujahr), Fahrereigenschaften und Versicherungsschutz (Höchstbeträge, Selbstbeteiligungen) ergänzen.

Besonders wenn Daten aus herkömmlichen Quellen wie Kreditauskunfteien fehlen, gewinnt der Zugriff auf weitere Datenquellen zunehmend an Bedeutung. Kreditscores von Auskunfteien dienen üblicherweise als Basis für die Risikomodellierung, mit der das Risiko für Autofahrer bewertet wird, was wiederum ihre Versicherungsprämien beeinflusst. Andererseits liefern Daten aus mobilen Anwendungen und IoT-Geräten eine individuellere Sicht auf das Kundenverhalten. Sie könnten dazu verwendet werden, einen genaueren Risikoindikator für eine bestimmte Partei zu entwickeln. Dieser alternative, verhaltensbasierte Ansatz zur Risikomodellierung und Preisgestaltung ist für die Bereitstellung eines hyperpersonalisierten Kundenerlebnisses unverzichtbar.

Die auf der Databricks Data Intelligence Platform aufsetzende Lakehouse-Architektur ist die einzige Plattform, die alle erforderlichen Funktionen und Services zur Unterstützung einer zukunftssicheren Schadensbearbeitung vereint. Von Streaming bis hin zu maschinellem Lernen und Reporting bietet Databricks die beste Plattform für den Aufbau einer Komplettlösung für die Versicherungsbranche von morgen.

Die folgenden Schritte beschreiben den gesamten Ablauf:

- Die Versicherungsdaten werden erfasst.
- Telematikdaten werden kontinuierlich von IoT-Sensoren eingelesen. Ein Antragsteller reicht seine Antragsdaten über eine mobile App ein.
- Alle operativen Daten werden in den Cloud-Speicher aufgenommen.
- Diese Daten werden schrittweise als „Rohdaten“ in Delta Bronze-Tabellen geladen.
- Die Daten werden über verschiedene Datentransformationen aufbereitet und verfeinert.
- Die Daten werden anhand des trainierten Modells bewertet.
- Die Prognosen werden in eine Gold-Tabelle geladen.
- Das Forderungs-Dashboard wird für die Visualisierung neu geladen.
- Die gewonnenen Erkenntnisse werden an das operative System zurückgegeben. Das schafft eine Feedbackschleife: Es werden Daten von Guidewire gezogen und die „nächste beste Maßnahme“ in Echtzeit an Guidewire zurückgespielt, um zu ermitteln, welche Forderungen priorisiert behandelt werden sollten.
- Die Entscheidungsworkflows verwenden diese generierten Erkenntnisse, um den Fall wie erforderlich weiterzuleiten (z. B. Reparaturkosten oder die Erstattung einer Fahrzeugmiete zu genehmigen oder die Behörden zu benachrichtigen).

Wie das Lakehouse-Paradigma Smart Claims unterstützt

Die Lakehouse-Architektur ermöglicht es allen Datenpersonas (Data Engineers, Data Scientists, Analytic Engineers und BI-Analysten), auf einer einzigen Plattform zusammenzuarbeiten. Die Unterstützung aller Big-Data-Workloads und -Paradigmen (z. B. Batch-Verarbeitung, Streaming, DataOps, ML, MLOps und BI) auf einer einzigen kollaborativen Plattform vereinfacht die Gesamtarchitektur erheblich, erhöht die Stabilität und senkt die Kosten beträchtlich.

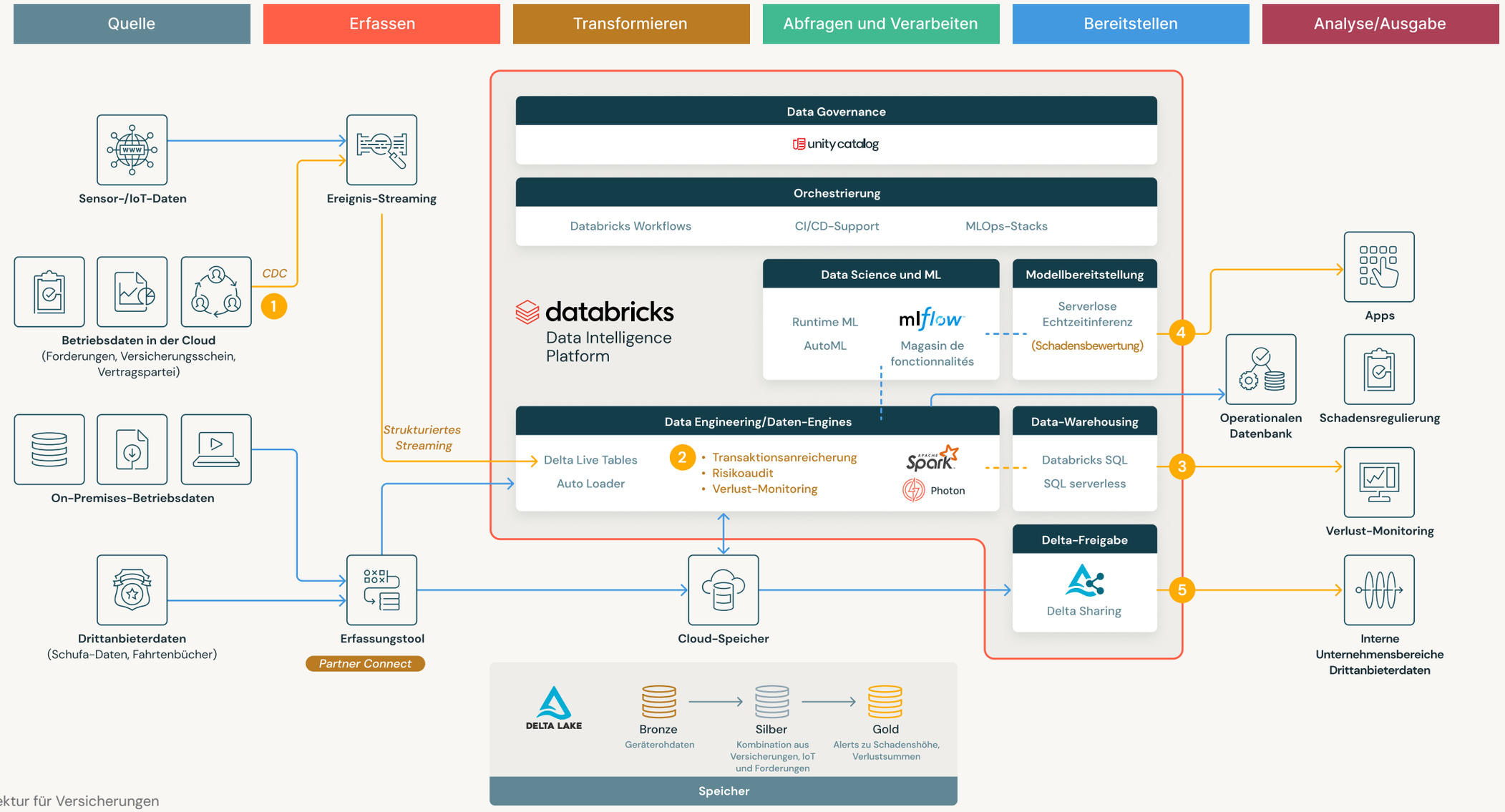
Databricks DLT-Pipelines (Delta Live Tables) bieten ein einfaches, deklaratives Framework zur schnellen Entwicklung und Implementierung von Workloads. Außerdem implementiert es native Unterstützung für das Datenqualitätsmanagement mit differenzierten Einschränkungen, um die Integrität der Ausgabe zu gewährleisten.

ML- und KI-Workloads können mit MLflow einfach erstellt und verwaltet werden, um Reproduzierbarkeit und Nachvollziehbarkeit zu gewährleisten. MLFlow vereinfacht den gesamten Modelllebenszyklus vom Experimentieren über die Modellbereitstellung bis hin zur Bereitstellung und Archivierung. ML kann für alle Arten von Daten angewendet werden, einschließlich unstrukturierter Nicht-Text-Daten (d. h. Bilder, Audio, Video usw.). Bei dieser Lösung nutzen wir die Möglichkeiten des maschinellen Sehens, um den Schaden am Fahrzeug zu beurteilen.

Zu guter Letzt bietet Databricks SQL eine schnelle und effiziente Engine zur Abfrage kuratierter und aggregierter Daten. Diese Erkenntnisse können dann innerhalb von Minuten über interaktive Dashboards aufbereitet und ausgeliefert werden.

Unity Catalog ist eine zentralisierte Governance-Lösung für alle Daten und KI-Assets wie Dateien, Tabellen, Machine-Learning-Modelle und Dashboards mit integrierter Suche, Erkennung und automatisierter Workload-Hierarchie.

Das folgende Diagramm zeigt eine Referenzarchitektur für das Lakehouse im Kontext typischer Anwendungsfälle im Versicherungswesen:



Referenzarchitektur für Versicherungen

Datenerfassung mit DLT und Multitasking-Workflows

Der erste Schritt zur Automatisierung der Schadensbearbeitung ist die Optimierung des Erfassungs- und Datenverarbeitungs-Workflows. Die folgende Abbildung zeigt eine Übersicht über die typischen Datenquellen, darunter strukturierte, halbstrukturierte und unstrukturierte. Einige Quellen sind langsamer, während andere schneller aktualisiert werden. Darüber hinaus können einige Quellen additiv sein, d. h. sie müssen angehängt werden. Andere bieten inkrementelle Updates und müssen daher als sich langsam verändernde Dimensionen behandelt werden.

Volumen	Schnelligkeit	Varietät
Hoch • Telematikdaten	**Annähernd in Echtzeit** • Telematikdaten	**Strukturiert** • Versicherungsschein, Daten zum Versicherungsnehmer, Telematikdaten
Moderat • Forderungsdaten	**Streaming** • Forderungsdaten, Unfalldaten	**Teilstrukturiert** • Forderungsdaten
Niedrig • Versicherungsschein, Daten zum Versiche • Unfalldaten	**Batch** • Versicherungsschein, Daten zum Versiche	**Unstrukturiert** • Unfalldaten • Bilder, Videos und Texte

In der Schadensbearbeitung verwendete Beispieldatensätze

DLT kann die Datenverarbeitungspipeline vereinfachen und operationalisieren. Das Framework unterstützt den Auto Loader zur einfachen Erfassung von Streaming-Quellen, bietet effizientes Auto-Scaling für abrupte Änderung des Datenvolumens und garantiert Ausfallsicherheit durch Task-Neustart bei Fehlschlägen.

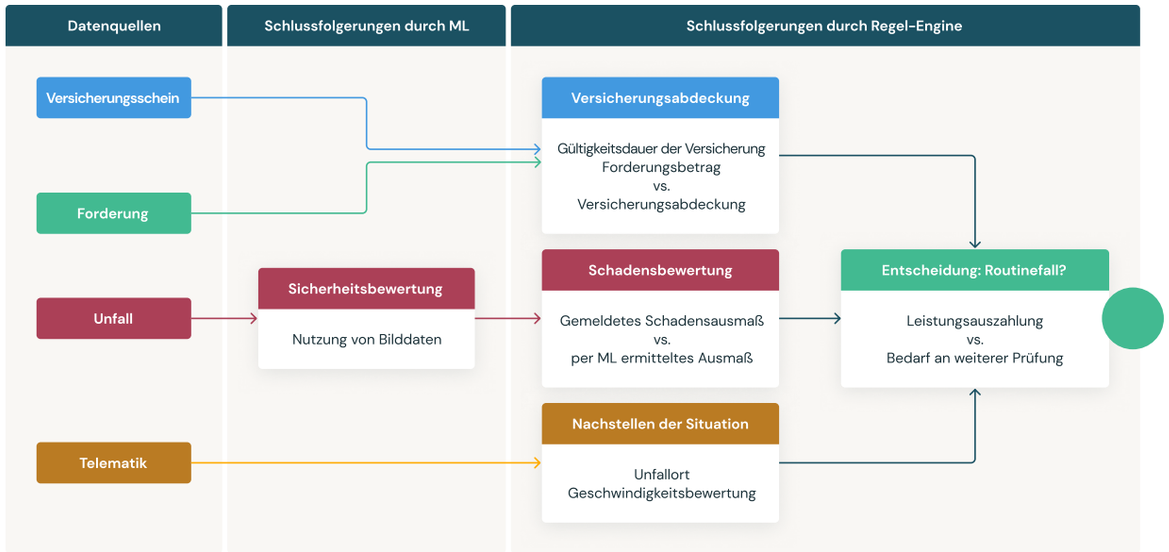
Databricks Workflows können mehrere Tasks und Workloads (z. B. Notebooks, DLT, ML, SQL) aufnehmen. Workflows unterstützen Repair-and-Run und die gemeinsame Nutzung von Rechenkapazität für verschiedene Aufgaben. So werden Workloads robust, skalierbar und kostengünstig. Außerdem kann ein Workflow ganz einfach durch Zeitpläne oder programmgesteuerte Aufrufe über REST-APIs automatisiert werden.

Erkenntnisgewinnung mit ML und der Dynamic Rules Engine

Die Nutzung von ML ist unerlässlich, um bislang unbekannte Muster aufzudecken, neue Erkenntnisse zu gewinnen und verdächtige Aktivitäten zu erkennen. Allerdings kann die Kombination aus ML und traditionellen regelbasierten Ansätzen noch wirkungsvoller sein.

Im Rahmen der Schadensbearbeitung kann ML für verschiedene Anwendungsfälle eingesetzt werden. Ein Beispiel wäre der Einsatz von maschinellem Sehen und ML zur Beurteilung und Bewertung von Bildern, die bei Kfz-Versicherungsansprüchen eingereicht werden. Modelle können so trainiert werden, dass sie den Schwerpunkt auf Relevanz und Schweregrad von Schäden legen. Hier kann MLFlow mit seinen End-to-End-MLOps-Fähigkeiten entscheidend zur Vereinfachung des Modelltrainings und der Bereitstellung von Modellen beitragen. MLFlow bietet eine serverlose Modellbereitstellung über REST-APIs. Trainierte Modelle können mit einem Mausklick operationalisiert und in die Produktion überführt werden.

Andererseits bieten Regel-Engines flexible Möglichkeiten zur Definition bekannter betrieblicher Merkmale und statistischer Prüfungen, die automatisiert und ohne menschliche Interaktion angewendet werden können. Wenn die Daten nicht den vorgegebenen Erwartungen entsprechen, werden sie markiert und zur Überprüfung und Untersuchung durch einen Menschen weitergeleitet. Die Einbindung eines solchen Ansatzes in ML-basierte Workflows bietet zusätzliche Transparenz und reduziert deutlich den Zeitaufwand für die Untersuchung und Überprüfung markierter Fälle durch Schadensermittler.



Schlussfolgerungen durch ML und Regel-Engine

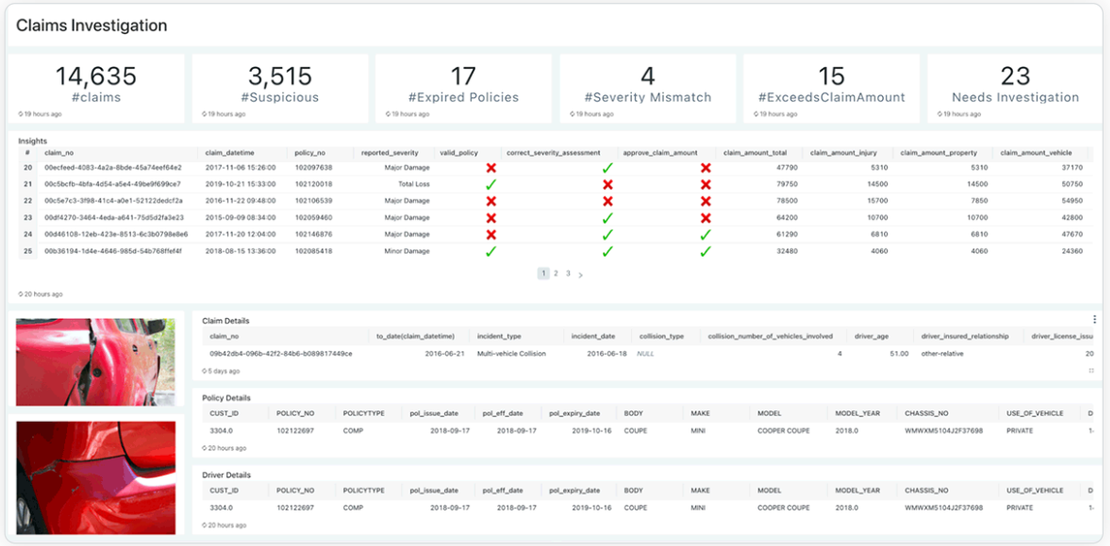
Visualisierung von Erkenntnissen mithilfe von Dashboards

In diesem Beispiel haben wir zwei Dashboards erstellt, um wichtige geschäftliche Erkenntnisse zu gewinnen. Konkret sind das:

- Ein Dashboard Loss Summary (Verlustübersicht) für einen Überblick über den gesamten Geschäftsbetrieb
- Ein Dashboard Claims Investigation (Schadensuntersuchung) mit einer detaillierten Ansicht der Forderungsdetails, um die Besonderheiten eines bestimmten Falles zu verstehen

Die Analyse aktueller Trends kann bei der Überprüfung ähnlicher Fälle einen Beitrag leisten. Hier ein paar Beispiele:

- Der Schadenskoeffizient errechnet sich aus den gezahlten Versicherungsansprüchen plus Schadenregulierungskosten geteilt durch die verdienten Gesamtprämien. So sollte der typische durchschnittliche Schadenskoeffizient (alle Deckungen zusammen, Personen- und Sachschäden) für Privatfahrzeuge etwa 65 % betragen.
- Die zusammenfassende Visualisierung erfasst die Anzahl der Falltypen nach Schadensschwere.
- Trendlinien für verschiedene Merkmale/Dimensionen
- Geografische Verteilung der Verträge



Dashboard „Claim Investigation“

Das „Claims Investigation“-Dashboard ermöglicht eine schnellere Schadensermittlung, indem es alle wichtigen Informationen zu einem Schadensfall bereitstellt. Ermittler können einen Fall detailliert betrachten, einschließlich Bilder des beschädigten Fahrzeugs, Schadensdetails, Informationen zum Versicherungsvertrag und zum Fahrer. Telematikdaten zeigen die Route des Fahrzeugs, und gemeldete Daten werden mit bewerteten Daten verglichen.

Gibt aktuelle Forderungen an, die in der Pipeline mithilfe von ML-Inferencing und einer Regel-Engine automatisch bewertet werden.

- Ein grünes Häkchen zeigt an, dass die automatische Bewertung der Forderungsbeschreibung entspricht.
- Ein rotes Kreuz weist auf eine Unstimmigkeit hin, die eine weitere manuelle Untersuchung erfordert.

Fazit

Innovation und Personalisierung sind für Versicherer unerlässlich, um sich von der Konkurrenz abzuheben. Databricks bietet ihnen eine Data Intelligence Platform an, die mit einer offenen, sicheren und erweiterbaren Architektur, die sich leicht in Tools und Services von Drittanbietern integrieren lässt, Innovation ermöglicht und beschleunigt. Dieser Lösungsbeschleuniger zeigt, wie das Paradigma auf die Schadensbearbeitung angewendet werden kann. Darüber hinaus bietet das Databricks-Ökosystem eine Reihe von Funktionen, die es Datenteams und geschäftlichen Akteuren ermöglichen, zusammenzuarbeiten und Erkenntnisse zu generieren und auszutauschen, die geschäftliche Entscheidungen unterstützen und unterm Strich einen spürbaren Mehrwert generieren.

Die technischen Ressourcen, einschließlich der in diesem Beispiel verwendeten Pipeline-Konfigurationen, Modelle und Beispieldaten, können Sie hier oder direkt auf Git abrufen.

KAPITEL 4.3

Entwurfsmuster für die Batch-Verarbeitung bei Finanzdienstleistern

Schaffung einer Grundlage für die Automatisierung von Arbeitsabläufen

von Eon Retief

Einführung

Finanzdienstleister weltweit sehen sich mit ungeahnten Herausforderungen konfrontiert, die von Marktvolatilität und politischer Unsicherheit bis hin zu immer neuen und geänderten Gesetzen und Vorschriften reichen. Unternehmen sind gezwungen, Programme zur digitalen Transformation zu beschleunigen und kritische Prozesse zu automatisieren, um Betriebskosten zu senken und Reaktionszeiten zu verbessern. Da die Daten jedoch in der Regel über mehrere Systeme verstreut sind, ist der Zugriff auf die Informationen, die für die Durchführung dieser Initiativen erforderlich sind, leichter gesagt als getan.

Die Erstellung eines Ökosystems mit Services, die die Vielzahl von datengesteuerten Anwendungsfällen in dieser digital transformierten Branche unterstützen können, kann jedoch eine unmöglich umzusetzende Aufgabe sein. In diesem Blogpost widmen wir uns einem entscheidenden Aspekt des modernen Daten-Stacks: der Batch-Verarbeitung. Auch wenn dieses Paradigma scheinbar veraltet ist, werden wir sehen, warum die Batch-Verarbeitung nach wie vor eine unverzichtbare und äußerst nützliche Komponente der Datenarchitektur ist. Außerdem werden wir erfahren, wie Databricks Finanzdienstleistern dabei helfen kann, einige der wichtigsten Herausforderungen zu meistern, die beim Aufbau einer Infrastruktur zur Unterstützung dieser geplanten oder periodischen Arbeitsabläufe auftreten können.

Warum die Batch-Erfassung wichtig ist

In den letzten zwei Jahrzehnten zwang der Wandel zur Instant-Gesellschaft Unternehmen, ihre Betriebs- und Interaktionsmodelle zu überdenken. Eine Digital-First-Strategie ist nicht mehr optional, sondern überlebenswichtig.Die Kundenbedürfnisse und -anforderungen entwickeln sich schneller als je zuvor. Der Wunsch nach sofortiger Befriedigung hat zu einem verstärkten Fokus auf Echtzeit-Verarbeitung und Entscheidungsfindung geführt. Doch stellt sich die Frage, ob Batch-Verarbeitung in dieser dynamischen Welt noch relevant ist.

Echtzeitsysteme und Streaming-Dienste sind zwar agil, decken aber oft nicht alle Back-Office-Bedürfnisse ab. Die meisten Geschäftsentscheidungen erfordern strategische Überlegungen und eine systematische Prüfung von langfristig gesammelten aggregierten Daten. In diesem Kontext ist die Batch-Verarbeitung immer noch die effizienteste und kostengünstigste Methode zur Verarbeitung großer, aggregierter Datenmengen, da sie eine systematische Überprüfung von über einen längeren Zeitraum gesammelten Daten ermöglicht. Außerdem kann sie offline durchgeführt werden. Das senkt Betriebskosten und ermöglicht eine bessere Kontrolle über den gesamten Vorgang.

Die Finanzwelt ist im Wandel, doch sowohl etablierte Unternehmen als auch Startups verlassen sich weiterhin stark auf Batch-Verarbeitung für ihre Kernfunktionen. Sei es für Berichterstellung und Risikomanagement oder zur Erkennung und Überwachung von Anomalien: Finanzdienstleister brauchen die Batch-Verarbeitung, um menschliche Fehler zu reduzieren, die Bereitstellung zu beschleunigen und die Betriebskosten zu senken.

Erste Schritte

Aus der Vogelperspektive betrachtet, verfügen die meisten Finanzdienstleister über eine Vielzahl von Datenquellen, die auf lokale Systeme, cloudbasierte Services und sogar Anwendungen von Drittanbietern verstreut sind. Der Aufbau eines Frameworks für die Batch-Erfassung, das all diese Verbindungen unterstützt, erfordert eine komplexe Technik und kann die Wartungsteams schnell überlasten. Und dabei ist von Faktoren wie Change Data Capture (CDC), Zeitplanung und Schemaevolution noch gar nicht die Rede. In diesem Abschnitt zeigen wir, wie die Lakehouse-Architektur für Finanzdienstleister und ihr Partnerökosystem eingesetzt werden können, um zentrale Herausforderungen anzugehen und die Gesamtarchitektur deutlich zu vereinfachen.

Die Lakehouse-Architektur wurde entwickelt, um eine einheitliche Plattform bereitzustellen, die alle analytischen und wissenschaftlichen Daten-Workloads unterstützt. Abbildung 1 zeigt die Referenzarchitektur für ein entkoppeltes Design, das eine einfache Integration mit anderen Plattformen ermöglicht, die das moderne Datenökosystem unterstützen. Mit dem Lakehouse ist es ganz einfach, Erfassungs- und Bereitstellungsschichten zu erstellen, die unabhängig von Quelle, Volumen, Geschwindigkeit und Ziel der Daten funktionieren.

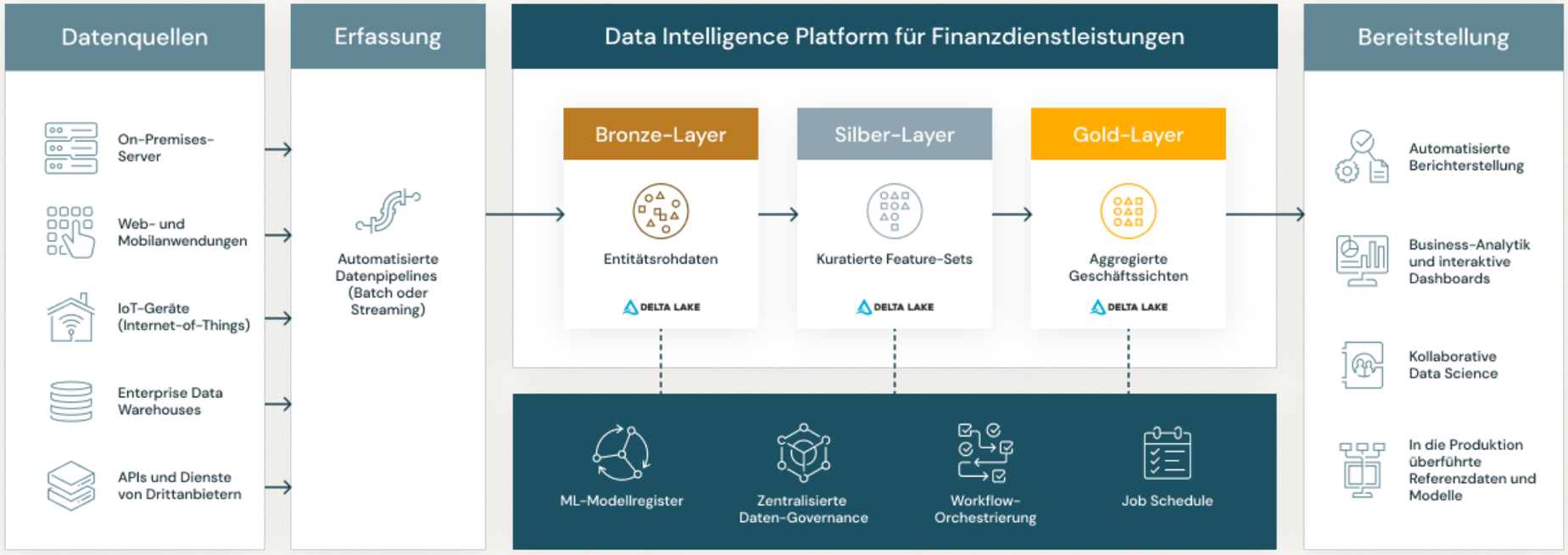


Abbildung 1: Referenzarchitektur des Lakehouse für Finanzdienstleistungen

Um die Leistungsfähigkeit und Effizienz von LFS zu demonstrieren, verwenden wir ein Beispiel aus der Welt der Versicherungen. Betrachten wir die grundlegenden Berichtsanforderungen bei einem typischen Schadensfall-Workflow. In diesem Szenario könnte das Unternehmen an den Schlüsselkennzahlen interessiert sein, die durch Schadensprozesse bestimmt werden. Zum Beispiel:

- Anzahl der aktiven Versicherungsverträge
- Anzahl der Forderungen
- Wert der Forderungen
- Gesamtbelastung
- Schadensquote

Außerdem benötigt das Unternehmen vielleicht einen Überblick über potenziell verdächtige Schadensfälle und eine Aufschlüsselung nach Ereignistyp und -schwere. Alle diese Kennzahlen lassen sich anhand von zwei wichtigen Datenquellen leicht berechnen: der Übersicht über bestehende Verträge und den von Kunden eingereichten Forderungen. Die Vertrags- und Schadensdaten sind in der Regel in einer Kombination aus Enterprise Data Warehouses (EDWs) und operativen Datenbanken gespeichert. Die größte Herausforderung besteht darin, sich mit diesen Quellen zu verbinden und Daten in unser Lakehouse zu importieren, um dort die Stärken von Databricks für die Berechnung der gewünschten Ergebnisse zu nutzen.

Zum Glück macht es das flexible Design des LFS ganz einfach, zum Erledigen bestimmter Aufgaben erstklassige Produkte aus einer Reihe von SaaS-Technologien und -Tools zu nutzen. Eine mögliche Lösung für unseren Anwendungsfall der Schadensanalyse wäre die Verwendung von Fivetran für die Batch-Erfassungsebene. Fivetran bietet eine einfache und sichere Plattform für die Verbindung zu zahlreichen Datenquellen und die direkte Übermittlung von Daten an die Databricks Data Intelligence Platform. Außerdem unterstützt es CDC, Schemaentwicklung und Workload-Planung nativ. In Abbildung 2 zeigen wir die technische Architektur einer praktischen Lösung für diesen Anwendungsfall.

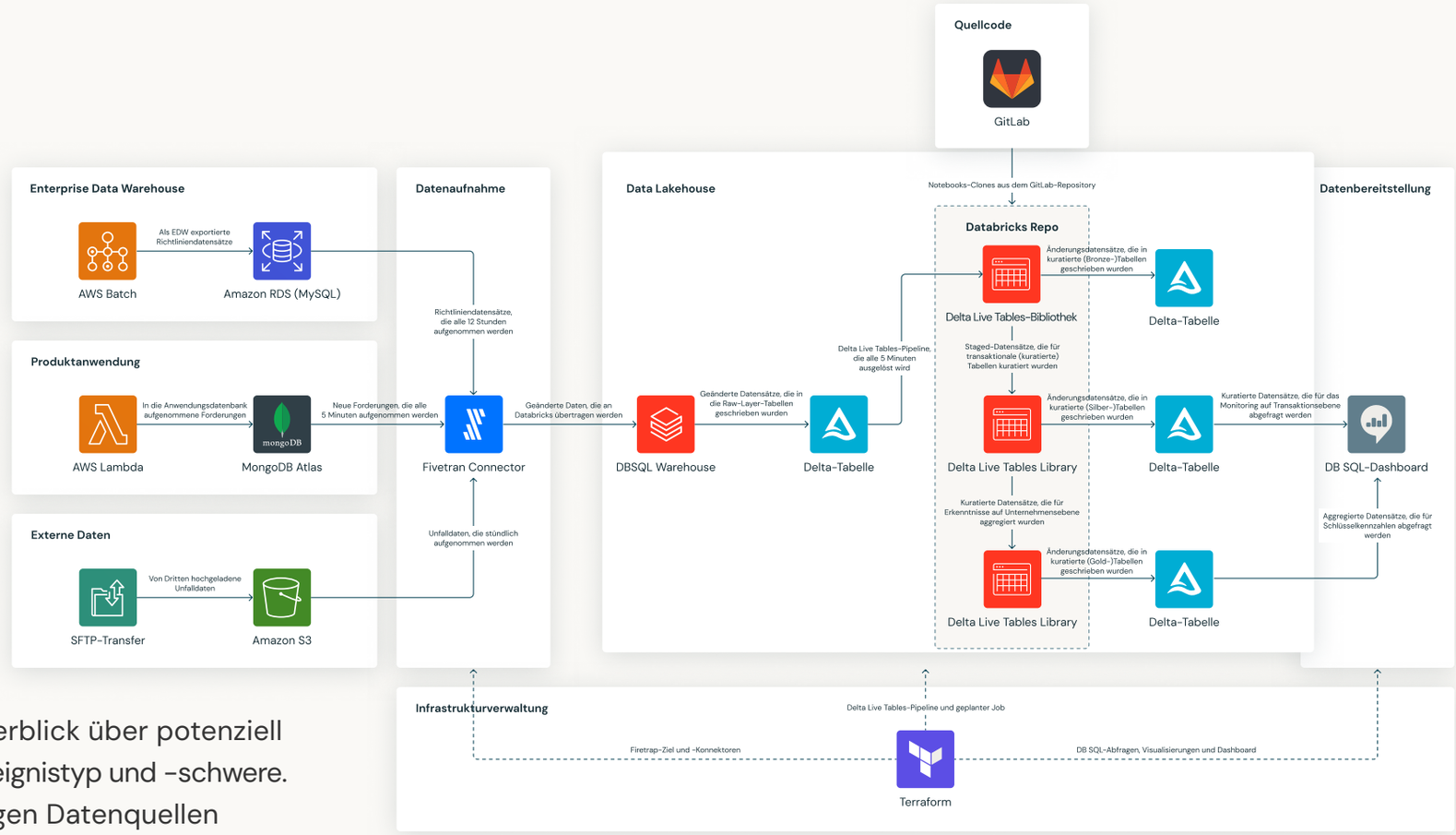


Abbildung 2: Technische Architektur für einen einfachen Workflow für die Schadensregulierung bei einer Versicherung

Sobald die Daten eingelesen und an das LFS übergeben wurden, können wir Delta Live Tables (DLT) für den gesamten Engineering-Workflow verwenden. DLT bietet ein einfaches, skalierbares deklaratives Framework für die Automatisierung komplexer Workflows und die Durchsetzung von Datenqualitätskontrollen. Die Ergebnisse unseres DLT-Workflows – unsere kuratierten und aggregierten Assets – können dann mit Databricks SQL (DB SQL) abgefragt werden. DB SQL erweitert das LFS um Data Warehousing, um geschäftskritische Analyse-Workloads zu unterstützen. Die Ergebnisse von DB SQL-Abfragen können in einfach zu nutzende Dashboards gepackt und Geschäftskunden zur Verfügung gestellt werden.

Schritt 1: Erstellen der Erfassungsschicht

Für das Einrichten einer Erfassungsschicht sind bei Fivetran zwei Schritte erforderlich: erstens die Konfiguration eines sogenannten Ziels, an das die Daten übermittelt werden sollen, und zweitens die Herstellung einer oder mehrerer Verbindungen mit den Quellsystemen. Mit Partner Connect wird der erste Schritt mithilfe einer einfachen Oberfläche mit Benutzerführung umgesetzt, um Fivetran mit einem Databricks Warehouse zu verbinden. Fivetran nutzt das Warehouse dann, um Rohdaten in Delta-Tabellen umzuwandeln und die Ergebnisse auf der Databricks Data Intelligence Platform zu speichern. Die Abbildungen 3 und 4 zeigen die Schritte zur Konfiguration eines neuen Ziels mithilfe der Partner Connect- und Fivetran-Oberflächen.

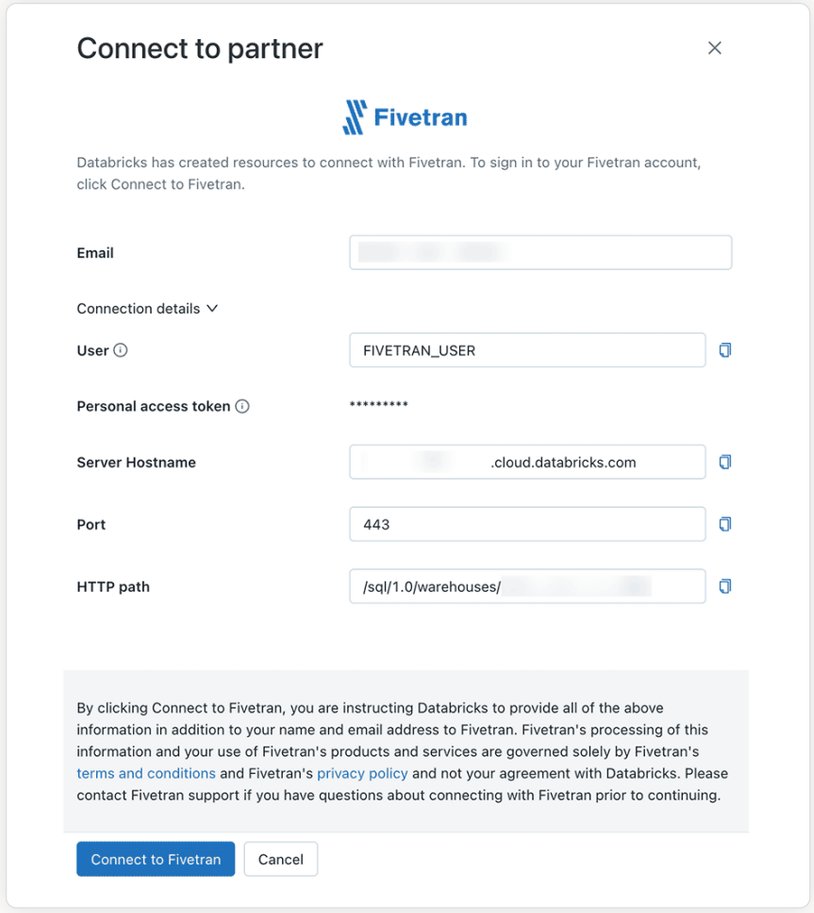


Abbildung 3: Databricks Partner Connect-Oberfläche zum Erstellen einer neuen Verbindung

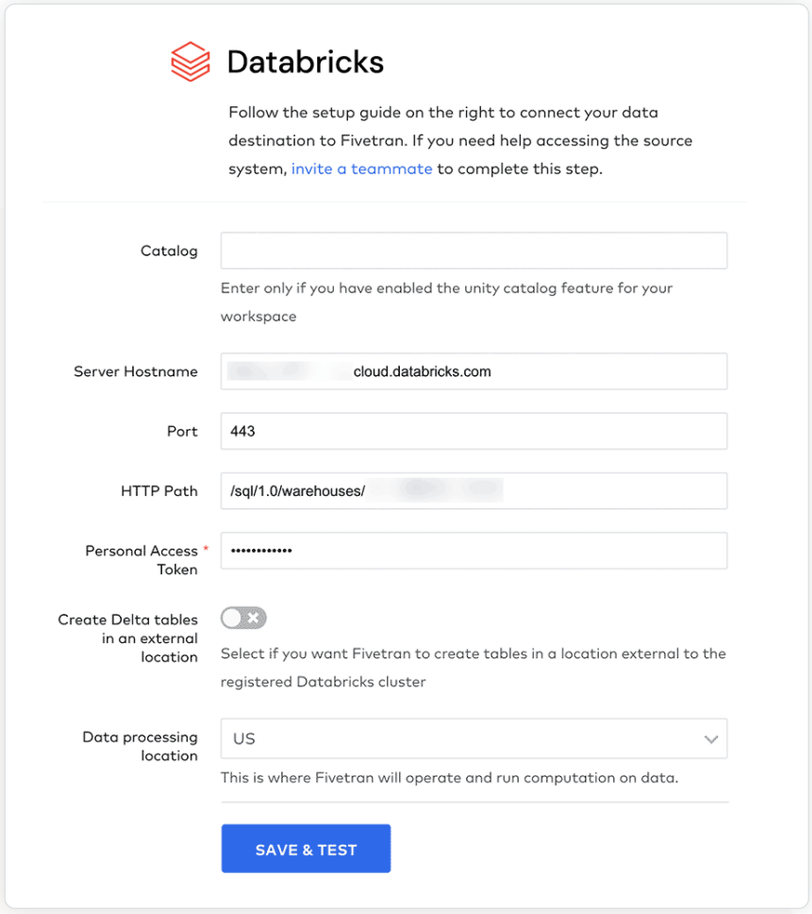


Abbildung 4: Fivetran-Oberfläche zum Bestätigen eines neuen Ziels

Für den nächsten Schritt wechseln wir zur Fivetran-Oberfläche. Hier können wir ganz einfach Verbindungen zu mehreren verschiedenen Quellsystemen erstellen und konfigurieren (eine vollständige Liste aller unterstützten Verbindungen finden Sie in der offiziellen Dokumentation). In unserem Beispiel betrachten wir drei Datenquellen: zum einen in einem Operational Data Store (ODS) oder Enterprise Data Warehouse (EDW) gespeicherte Vertragsunterlagen, zum zweiten Schadensdatensätze, die in einer operativen Datenbank gespeichert sind, und zum dritten externe Daten, die an einen Blob-Speicher geliefert werden. Folglich benötigen wir drei verschiedene Verbindungen, die in Fivetran konfiguriert werden müssen. In allen drei Fällen können wir den einfachen, geführten Prozess in Fivetran befolgen, um eine Verbindung mit dem jeweiligen Quellsystem einzurichten. Die Abbildungen 5 und 6 zeigen, wie neue Verbindungen zu Datenquellen konfiguriert werden.

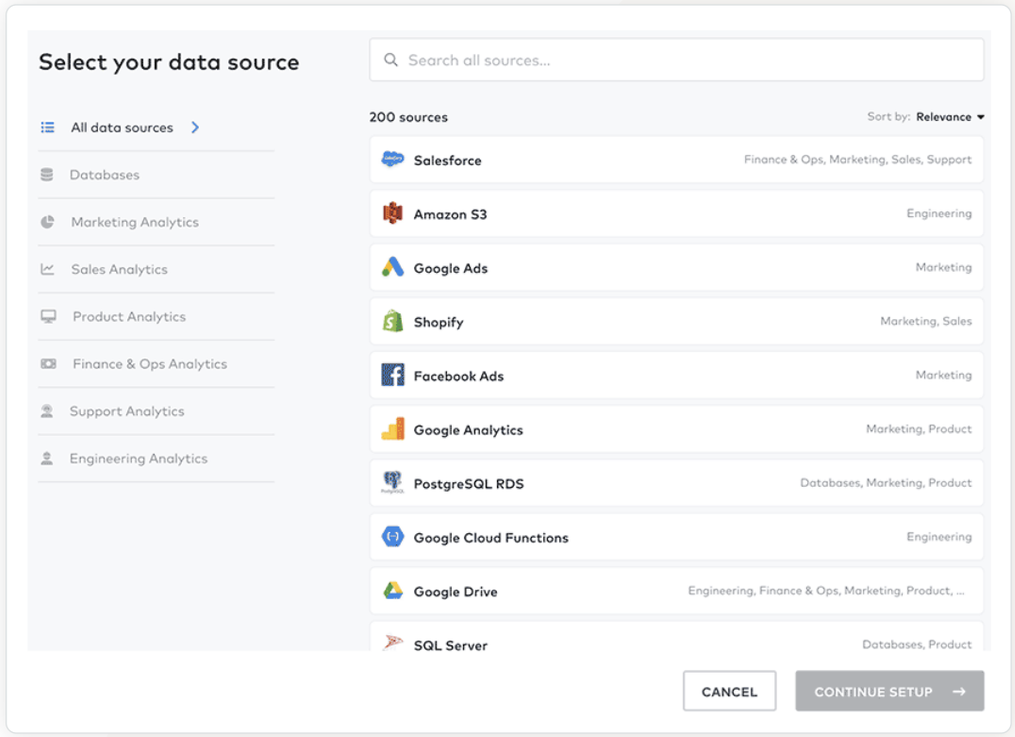


Abbildung 5: Fivetran-Oberfläche zur Auswahl eines Datenquellentyps

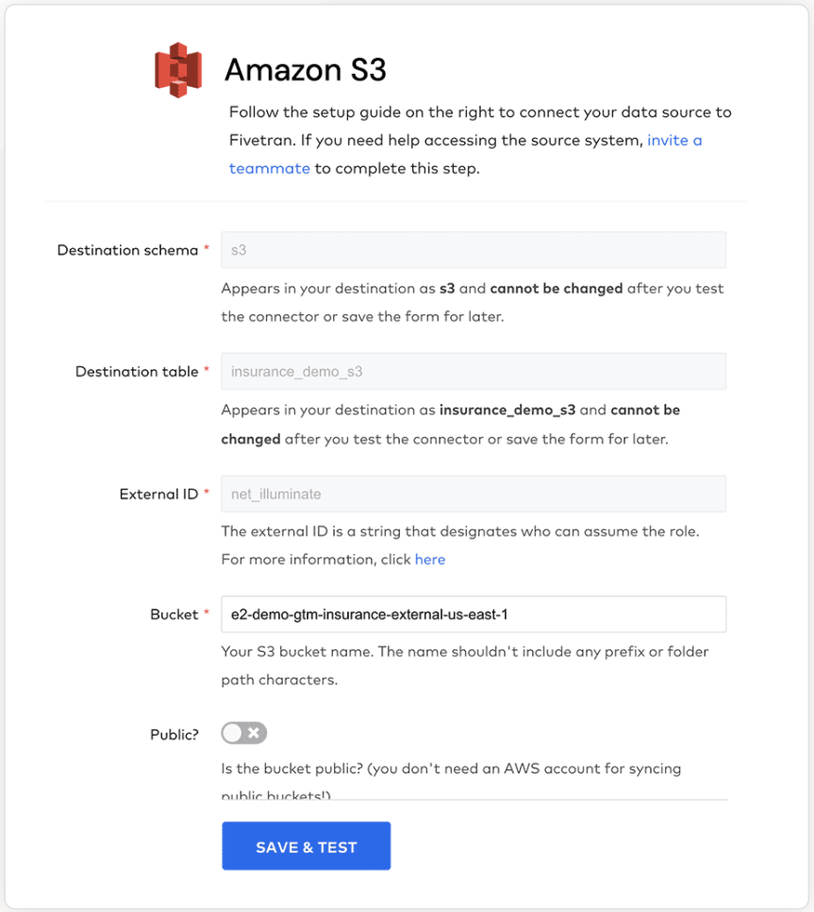


Abbildung 6: Fivetran-Oberfläche zum Konfigurieren einer Datenquellenverbindung

Verbindungen können weiter konfiguriert werden, nachdem sie validiert wurden. Eine wichtige Option, die Sie einstellen können, ist die Häufigkeit, mit der Fivetran das Quellsystem nach neuen Daten abfragt. In Abbildung 7 sehen Sie, wie einfach es bei Fivetran ist, für die Synchronisierungsfrequenz ein Intervall zwischen 5 Minuten und 24 Stunden festzulegen.

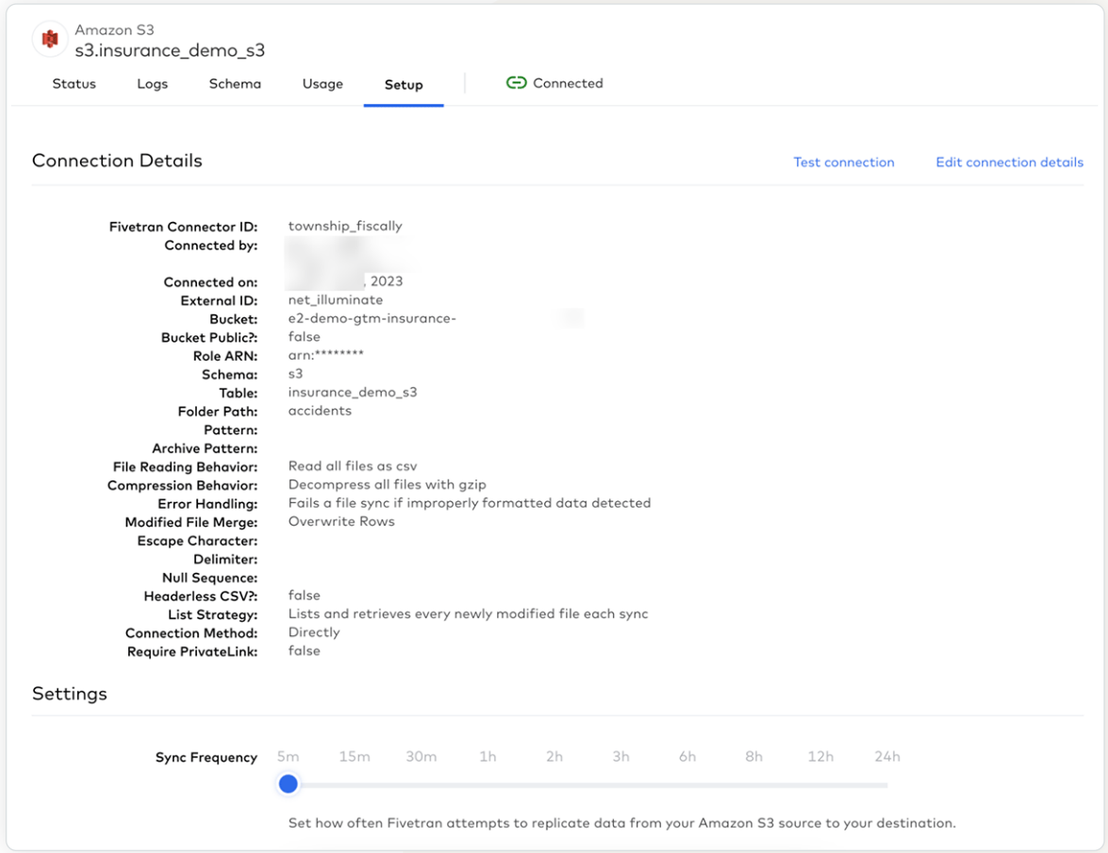


Abbildung 7: Konfiguration eines Fivetran-Connectors im Überblick

Fivetran fragt die Daten aus den Quellsystemen sofort ab und liest sie ein, sobald eine Verbindung validiert wurde. Die Daten werden als Delta-Tabellen gespeichert und können innerhalb von Databricks über den Catalog Explorer angezeigt werden. Standardmäßig speichert Fivetran alle Daten unter dem Hive-Metastore. Für jede neue Verbindung wird ein neues Schema erstellt, und jedes Schema enthält mindestens zwei Tabellen: eine mit den Daten und eine weitere mit den Logs von jedem versuchten Erfassungszyklus (Abb. 8).

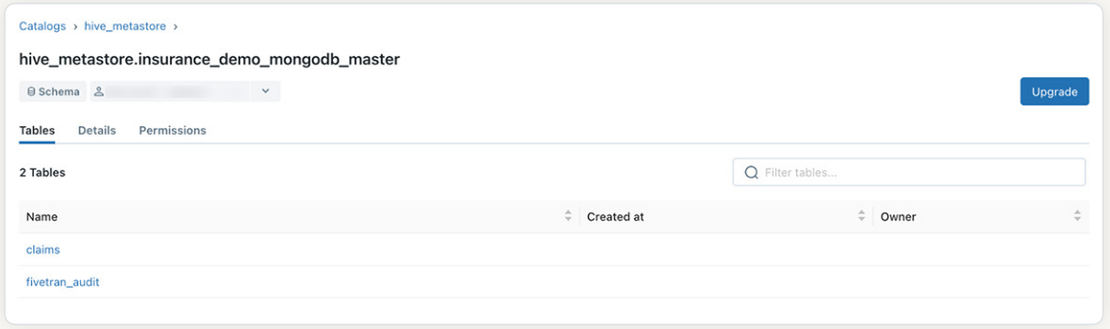


Abbildung 8: Zusammenfassung der von Fivetran im Databricks Warehouse erstellten Tabellen für eine Beispielverbindung

Das Speichern der Daten in Delta-Tabellen ist ein großer Vorteil. Delta Lake unterstützt von Haus aus eine granulare Datenversionierung, d. h., wir können uns den Verlauf Erfassungszyklus für Erfassungszyklus ansehen (Abb. 9). Wir können mit DB SQL bestimmte Versionen der Daten abfragen, um zu analysieren, wie sich die Quelldatensätze entwickelt haben.

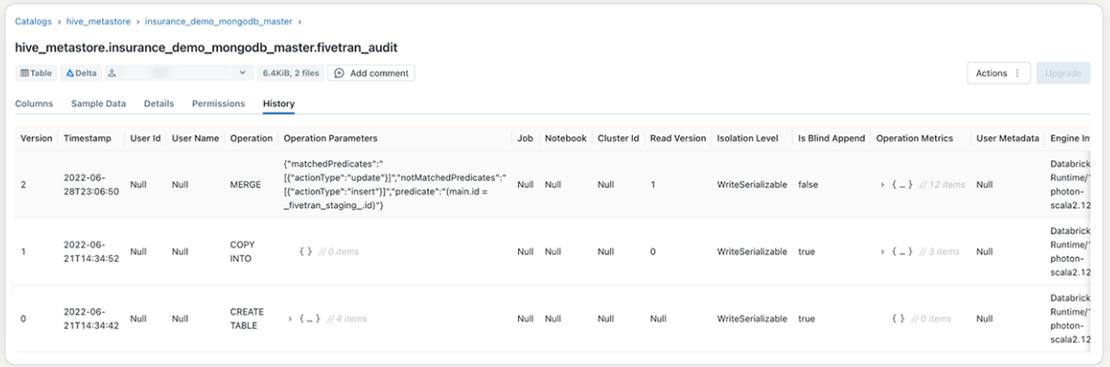


Abbildung 9: Verlaufsansicht der Änderungen, die an der Fivetran-Audit-Tabelle vorgenommen wurden

databricks

Wichtig ist, dass, wenn die Quelldaten halb- oder unstrukturierte Werte enthalten, diese Attribute während des Konvertierungsprozesses reduziert werden. Das bedeutet, dass die Ergebnisse in gruppierten Textspalten gespeichert werden und dass diese Entitäten im Kuratierungsprozess mit DLT zerlegt und entpackt werden müssen, um separate Attribute zu erstellen.

Schritt 2: Automatisieren des Workflows

Sobald die Daten im Lakehouse sind, können wir mit Delta Live Tables (DLT) einen einfachen, automatisierten Data-Engineering-Workflow erstellen. DLT bietet ein deklaratives Framework für die Spezifikation detaillierter Feature-Engineering-Schritte. Derzeit unterstützt DLT APIs sowohl für Python als auch für SQL. In diesem Beispiel verwenden wir Python-APIs zum Erstellen unseres Workflows.

Das grundlegendste Konstrukt in DLT ist die Definition einer Tabelle. DLT fragt alle Tabellendefinitionen ab, um einen umfassenden Workflow zu erstellen, der die Verarbeitung der Daten beschreibt. In Python zum Beispiel werden Tabellen mithilfe von Funktionsdefinitionen und dem Dekorator „dlt.table“ erstellt (ein Beispiel für den Python-Code sehen Sie nachstehend). Mit dem Dekorator geben Sie den Namen der resultierenden Tabelle, einen beschreibenden Kommentar, der den Zweck der Tabelle erläutert, und eine Sammlung von Tabelleneigenschaften an.

```
@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():
   # Read the staged claim records into memory
   staged_claims = dlt.read(“staged_claims”)
   # Unpack all nested attributes to create a flattened table structure
   curated_claims = unpack_nested(df = staged_claims, schema = schema_claims)

   ...
```

Die Anweisungen für das Feature Engineering werden innerhalb des Funktionskörpers mithilfe von PySpark-Standard-APIs und nativen Python-Befehlen definiert. Das folgende Beispiel zeigt, wie PySpark Schadensdatensätze mit Daten aus der Tabelle mit den Versicherungsverträgen verknüpft, um eine einzige, kuratierte Ansicht der Forderungen zu erstellen.

```
    ...

    # Read the staged claim records into memory
    curated_policies = dlt.read("curated_policies")
    # Evaluate the validity of the claim
    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(
            # Calculate the number of months between coverage starting and the
claim being filed
            "months_since_covered", F.round(F.months_between(F.col("claim_date"),
F.col("policy_effective_date")))
        ) \
        .withColumn(
            # Check if the claim was filed before the policy came into effect
            "claim_before_covered", F.when(F.col("claim_date") < F.col("policy_
effective_date"), F.lit(1)).otherwise(F.lit(0))
        ) \
        .withColumn(
            # Calculate the number of days between the incident occurring and the
claim being filed
            "days_between_incident_and_claim", F.datediff(F.col("claim_date"),
F.col("incident_date"))
        )

    # Return the curated dataset
    return curated_claims
```

Ein wesentlicher Vorteil von DLT ist die Möglichkeit, Standards für die Datenqualität festzulegen und durchzusetzen. Wir können für jede DLT-Tabelle Erwartungen mit detaillierten Datenqualitätsbeschränkungen festlegen, die auf den Inhalt der Tabelle angewendet werden sollen. Gegenwärtig unterstützt DLT Erwartungen für drei verschiedene Szenarien:

Dekorator	Beschreibung
`expect`	Datensätze behalten, die gegen die Erwartungen verstoßen
`expect_or_drop`	Datensätze verwerfen, die gegen die Erwartungen verstoßen
`expect_or_fail`	Ausführung anhalten, wenn mindestens ein Datensatz gegen Constraints verstößt

Erwartungen können wahlweise mit einem oder mit mehreren Datenqualitäts-Constraints definiert werden. Jeder Constraint erfordert eine Beschreibung und einen auszuwertenden Python- oder SQL-Ausdruck. Mit den Dekoratoren `expect_all`, `expect_all_or_drop` und `expect_all_or_fail` können mehrere Constraints definiert werden. Jeder Dekorator erwartet ein Python-Wörterbuch, bei dem die Schlüssel die Beschreibungen der Constraints und die Werte die jeweiligen Ausdrücke sind. Das folgende Beispiel zeigt mehrere Datenqualitäts-Constraints für die oben beschriebenen Szenarien „Behalten“ und „Verwerfen“.

```
@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():
    ...
```

Wir können zum Deklarieren unserer DLT-Tabellen auch mehrere Databricks-Notebooks verwenden. Wenn wir uns an die Medaillon-Architektur halten, können wir zum Beispiel verschiedene Notebooks verwenden, um Tabellen mit den Bronze-, Silber- und Gold-Schichten zu definieren. Das DLT-Framework kann Anweisungen verarbeiten, die über mehrere Notebooks hinweg definiert wurden, um einen einzigen Workflow zu erstellen. Dabei werden alle Abhängigkeiten und Beziehungen zwischen den Tabellen automatisch verarbeitet und berücksichtigt. Abbildung 10 zeigt den vollständigen Workflow unseres Forderungsbeispiels. Ausgehend von drei Quelltabellen erstellt DLT eine umfassende Pipeline, die dreizehn Tabellen für den Geschäftsfall liefert.

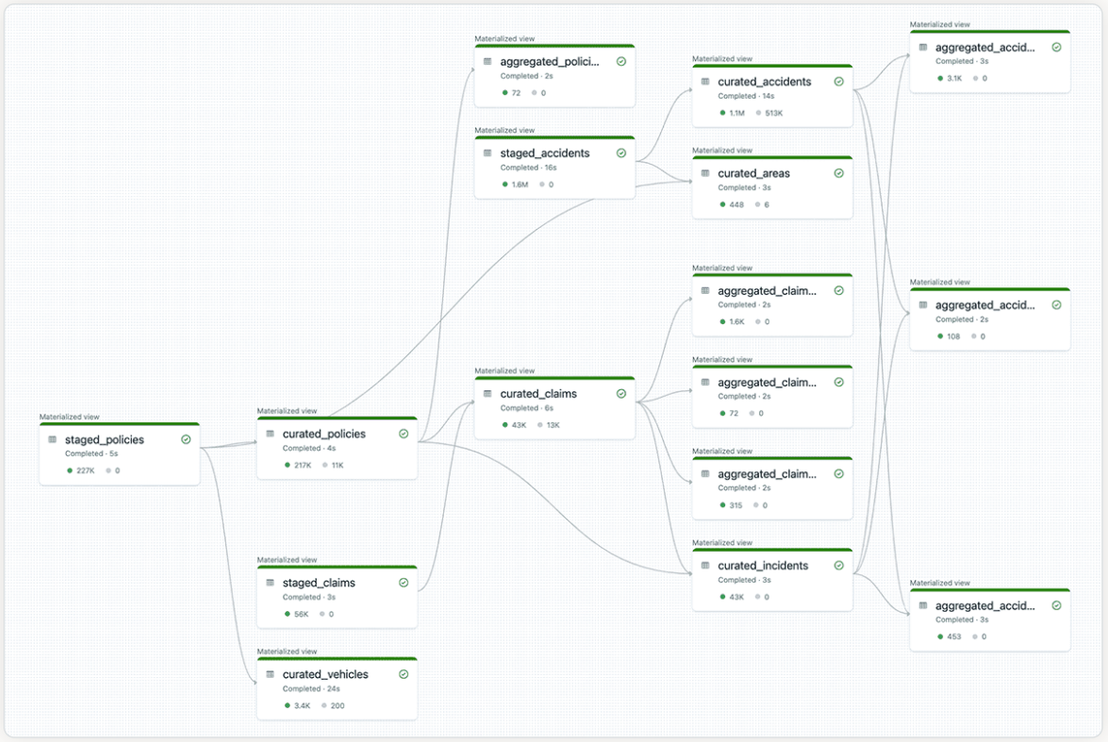


Abbildung 10: Überblick über einen vollständigen DLT-Workflow (Delta Live Tables)

Sie können die Ergebnisse für jede Tabelle einsehen, indem Sie die gewünschte Entität auswählen. Abbildung 11 zeigt ein Beispiel für die Ergebnisse der kuratierten Forderungstabelle. DLT bietet einen differenzierten Überblick über die Ergebnisse der Datenqualitätskontrollen:

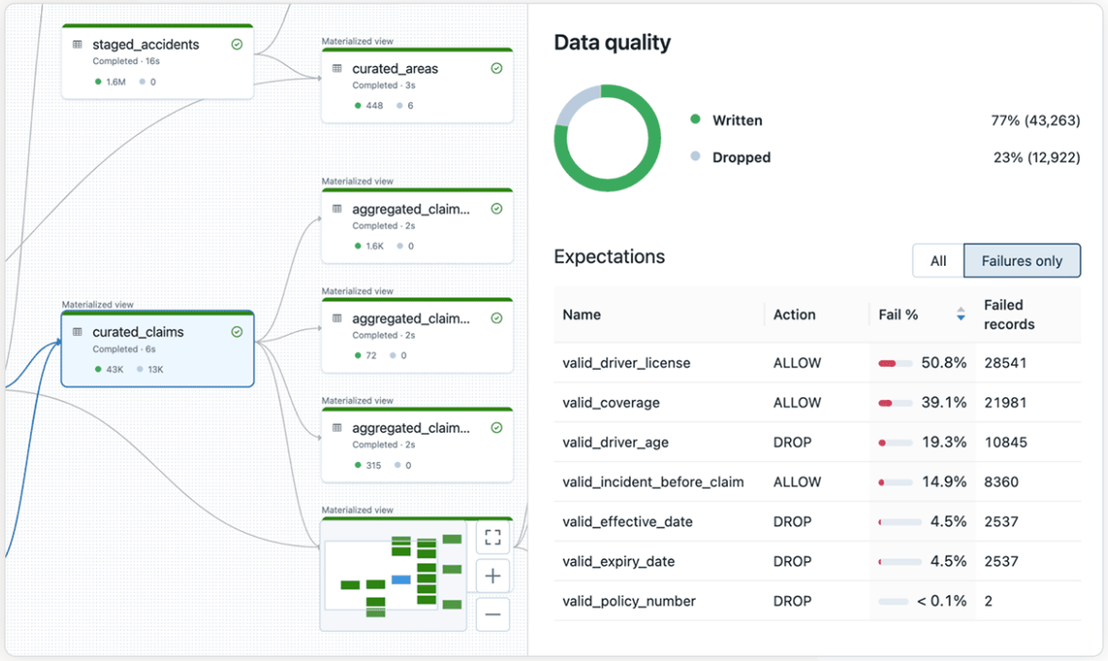
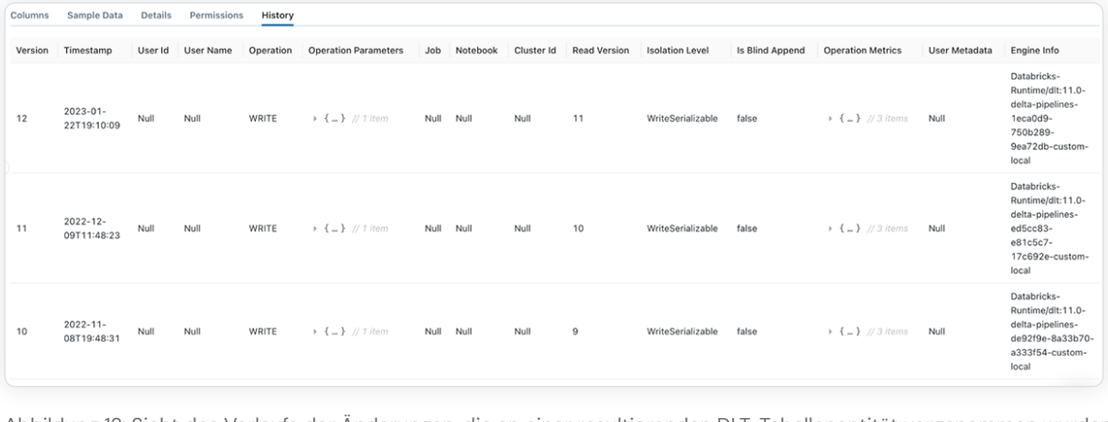


Abbildung 11: Beispiel einer detaillierten Sicht für eine DLT-Tabellenentität mit dem zugehörigen Datenqualitätsbericht

Die Ergebnisse der Datenqualitätserwartungen können durch Abfragen des Ereignislogs weiter analysiert werden. Das Ereignislog enthält detaillierte Kennzahlen zu allen für die Workflow-Pipeline definierten Erwartungen. Die folgende Abfrage ist ein Beispiel für die Darstellung der wichtigsten Metriken nach der letzten Pipeline-Aktualisierung, wobei auch die Anzahl für die Datensätze angegeben wird, die die Erwartungen erfüllt bzw. nicht erfüllt haben:

```
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;
```

Auch hier können wir den Delta-Verlaufslogs den vollständigen Verlauf der an jeder DLT-Tabelle vorgenommenen Änderungen entnehmen (Abb. 12). So können wir nachvollziehen, wie sich die Tabellen im Laufe der Zeit verändern, und komplette Update-Threads untersuchen, wenn eine Pipeline ausfällt.



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

Abbildung 12: Sicht des Verlaufs der Änderungen, die an einer resultierenden DLT-Tabellenentität vorgenommen wurden

Wir können Change Data Capture (CDC) nutzen, um Tabellen anhand der Änderungen in den Quell-Datasets zu aktualisieren. CDC unterstützt in DLT die Aktualisierung von Tabellen mit den SCD-Typen 1 und 2.

Hier können wir eine von zwei Optionen für unseren Batch-Prozess nutzen, um die DLT-Pipeline auszulösen: Wir können entweder den Databricks Auto Loader verwenden, um neue Daten inkrementell zu verarbeiten, sobald sie in den Quelltabellen ankommen, oder wir erstellen geplante Aufträge, die zu bestimmten Zeiten oder in bestimmten Intervallen ausgelöst werden. In diesem Beispiel haben wir uns für Letzteres entschieden: ein geplanter Auftrag, der die DLT-Pipeline alle fünf Minuten ausführt.

Operationalisieren der Ergebnisse

Die Möglichkeit, Daten inkrementell und effizient zu verarbeiten, ist nur die eine Seite der Medaille. Die Ergebnisse des DLT-Workflows müssen operationalisiert und den Geschäftsanwendern zur Verfügung gestellt werden. In unserem Beispiel können wir die Ergebnisse der DLT-Pipeline über Ad-hoc-Analysen oder ein interaktives Dashboard zur Verfügung gestellter Erkenntnisse nutzen.

Ad-hoc-Analysen

Databricks SQL stellt ein effizientes, kostengünstiges Data Warehouse auf der Grundlage der Lakehouse-Architektur bereit. Es ermöglicht uns, unsere SQL-Workloads direkt an den Quelldaten auszuführen – und das mit einem bis zu 12-fach besseren Preis-Leistungs-Verhältnis als Alternativen.

Wir können DB SQL nutzen, um spezifische Ad-hoc-Abfragen an unseren kuratierten und aggregierten Tabellen durchzuführen. Wir könnten zum Beispiel eine Abfrage über die Tabelle mit den kuratierten Versicherungsverträgen ausführen, die das Gesamtrisiko berechnet. Der DB SQL-Abfrage-Editor bietet eine einfache und benutzerfreundliche Oberfläche zur Erstellung und Ausführung solcher Abfragen (siehe Beispiel unten).

```
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
      ...
```

Der DB SQL-Abfrage-Editor ermöglicht es, Abfragen für unterschiedliche Versionen von Delta-Tabellen auszuführen. Zum Beispiel lässt sich eine Ansicht aggregierter Forderungsdaten für ein bestimmtes Datum und Uhrzeit abfragen (siehe Beispiel unten). Mit DB SQL können wir Ergebnisse verschiedener Versionen vergleichen, um ausschließlich die Veränderungen zwischen den Zuständen zu analysieren.

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

DB SQL ermöglicht die Nutzung einer serverlosen Compute-Engine, wodurch sich Konfiguration, Verwaltung und Skalierung der Cloud-Infrastruktur erübrigen und die Kosten minimiert werden. Es integriert sich ebenso in alternative SQL-Workbenches (z. B. DataGrip) und ermöglicht Analysten, ihre bevorzugten Tools für Datenanalyse und Erkenntnisgewinnung zu nutzen.

Geschäftliche Erkenntnisse

Schließlich können wir DB-SQL-Abfragen auch verwenden, um auf der Grundlage unserer Abfrageergebnisse ansprechende Visualisierungen zu erstellen. Diese Visualisierungen können dann gepackt und den Endbenutzern über interaktive Dashboards präsentiert werden (Abb. 13).

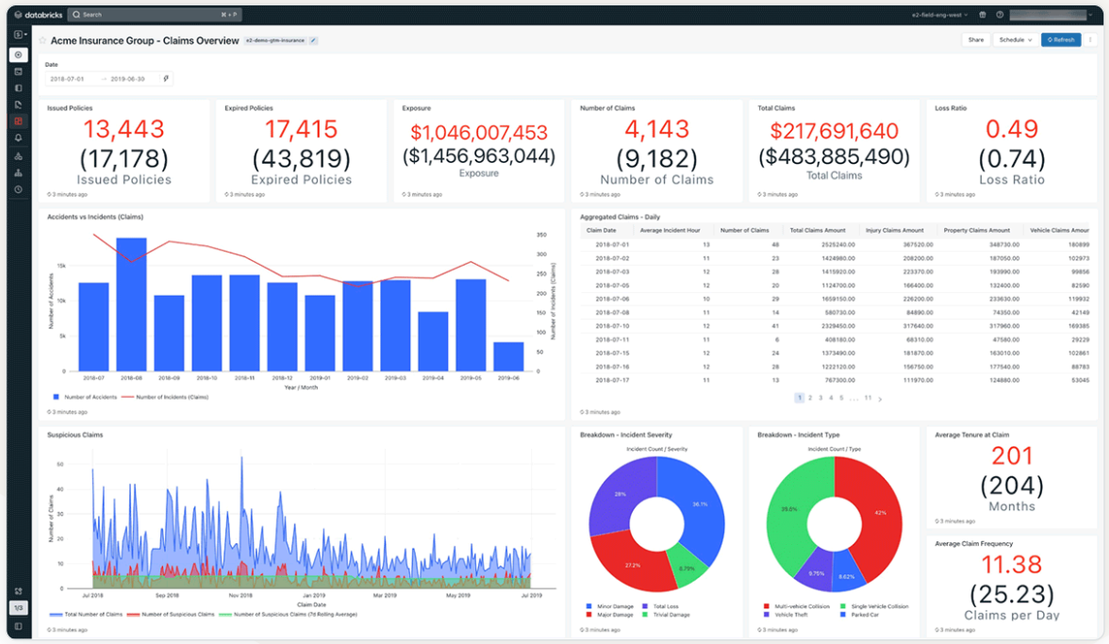


Abbildung 13: Beispiel für ein operatives Dashboard, das auf einer Reihe von DLT-Tabellenentitäten mit Ergebnissen basiert

Für unseren Anwendungsfall haben wir ein Dashboard mit einer Reihe von Schlüsselmetriken, rollierenden Berechnungen, detaillierten Aufschlüsselungen und aggregierten Sichten erstellt. Das Dashboard bietet auf einen Blick eine vollständige Übersicht über unseren Schadensprozess. Außerdem haben wir die Möglichkeit hinzugefügt, bestimmte Datumsbereiche anzugeben. DB SQL unterstützt eine Reihe von Abfrageparametern, die Werte in einer Abfrage zur Laufzeit ersetzen können. Diese Abfrageparameter können auf Dashboard-Ebene definiert werden, um alle damit verbundenen Abfragen entsprechend zu aktualisieren.

DB SQL lässt sich in zahlreiche Analyse- und BI-Tools von Drittanbietern wie Power BI, Tableau und Looker integrieren. Wie bei Fivetran können wir auch Partner Connect nutzen, um unsere externe Plattform mit DB SQL zu verbinden. So können Analysten Dashboards auf den vom Unternehmen bevorzugten Plattformen anlegen und bereitstellen, ohne auf die Leistung von DB SQL und der Databricks Data Intelligence Platform verzichten zu müssen.

Fazit

In der schnelllebigen, unberechenbaren Finanzwelt bleibt die Batch-Verarbeitung ein wichtiger Bestandteil des modernen Daten-Stacks, der mit Funktionen und Vorteilen von Streaming- und Echtzeitservices mithalten kann. Wir haben gezeigt, wie die Lakehouse-Architektur für Finanzdienstleistungen und ihr Partnernetzwerk genutzt werden können, um ein einfaches, skalierbares und erweiterbares Framework zu entwickeln, das komplexe Batch-Workloads unterstützt. Vorgestellt haben wir ein praktisches Beispiel aus der Schadensbearbeitung von Versicherungen. Mit Delta Live Tables (DLT) und Databricks SQL (DB SQL) können wir eine unbegrenzt skalierbare Datenplattform aufbauen, die sich einfach erweitern lässt und auch sich ändernden Anforderungen und Gegebenheiten standhält.

Für weitere Details zur Einrichtung und Konfiguration der Infrastruktur, besuchen Sie bitte dieses GitHub-Repository oder sehen Sie sich dieses Demovideo an.

KAPITEL

05

Kundenreferenzen: Echte Erfolge mit Databricks

KAPITEL 5.1

InMobi: Sinnvolle Vernetzungen zwischen Kunden und Marken schaffen

Bei Werbung bevorzugen Verbraucher relevante und personalisierte Inhalte, besonders auf mobilen Geräten, wo Zeit kostbar ist. Daher ist es entscheidend, sofort ihre Aufmerksamkeit zu gewinnen und ihr Interesse zu wecken. InMobi nutzt dazu Echtzeitdaten seiner Kunden, um gezielt Werbung auf Mobilgeräten und Sperrbildschirmen zu schalten. Mit dem Anstieg der Datenverarbeitungsanforderungen auf über 20 Terabyte pro Stunde explodierten die Kosten für das Betreiben des Multicloud-Datenlagers. Die proprietäre Beschaffenheit des Systems begünstigte zudem Silostrukturen, die die Zusammenarbeit und den Datenaustausch erschwerten.

InMobi migrierte daher von seinem Multicloud Data Warehouse zu Databricks, um seine verschiedenen Workloads (Data Warehousing, KI und Analysen) zu konsolidieren, die Abläufe zu rationalisieren und dafür zu sorgen, dass die Engineers mehr Zeit für wertvollere Aufgaben haben, um so eine bessere betriebliche Agilität und Effizienz zu erreichen. Seit dem Wechsel zum Lakehouse-Modell hat das Unternehmen nicht nur die Gesamtbetriebskosten im Vergleich zur Nutzung eines Multicloud-Datenlagers deutlich reduziert, sondern auch die Produktivität im gesamten Unternehmen gesteigert. Das führte zu einer beschleunigten Markteinführung neuer Produkte.

32 %
niedrigere TCO im Vergleich zum vorherigen Multicloud Data Warehouse

15 %
schnellere Abfragen im Vergleich zum vorherigen Multicloud Data Warehouse

20 %
höhere Leistung bei der Lieferantenberichterstattung

Unser Ziel, das Preis-Leistungs-Verhältnis zu optimieren, wurde von Databricks von Anfang an erfüllt. Das Lakehouse hat uns geholfen, Kosten zu senken, ohne die Leistung gemischter Workloads zu beeinträchtigen. So können wir den Daten- und KI-Betrieb heute und in Zukunft optimieren.

– MOHIT SAXENA
Mitbegründer und Group CTO,
InMobi

databricks

Komplexe Legacy-Infrastruktur und ein schwerfälliges Multicloud Data Warehouse

Bei InMobi dreht sich alles um zielgerichtete Werbung: Das Unternehmen hilft Marken dabei, Verbraucher auf effektiv und kostengünstig anzusprechen und mit ihnen zu interagieren. Aber um relevante Anzeigen zu schalten, braucht man Daten. Und zwar sehr viele. Im Laufe der Zeit erweiterte InMobi sein lokales Hadoop-System durch Hinzufügen mehrerer Cloud Data Warehouses, um verschiedene Probleme zu lösen. Da jedoch die Datenmenge, die das Unternehmen verarbeiten musste, exponentiell anstieg (auf 20 TB pro Stunde), baute InMobi weiter auf seinem Altsystem auf. Am Ende stand ein Multicloud Data Warehouse, das eine Reihe von Herausforderungen mit sich brachte: Es war übermäßig komplex, verursachte Ausfälle, war in der Skalierung extrem kostspielig und führte zur Entstehung von Datensilos, die den Datenaustausch und die Zusammenarbeit einschränkten. Das Team von InMobi erkannte, dass die Beibehaltung dieses Systems die Innovationsfähigkeit einschränken und wertvolle technische Ressourcen im Wartungsmodus binden würde.

„Die Dateninfrastruktur, die wir aufgebaut hatten, funktionierte zwar, aber sie verursachte eine erhebliche Komplexität und einen Overhead, der unseren Fokus von unseren eigenen Kernprodukten ablenkte“, so Madhan Sundaram, Senior Director of Platform Engineering bei InMobi. „Wir brauchten unsere talentierten Engineers aber, um Mehrwert für unsere Kunden zu schaffen, und dafür benötigten wir ein einfacheres und stärker vereinheitlichtes System.“

Was InMobi wollte, war ein zentrales System, mit dem gleich mehrere Probleme gelöst werden konnten. Um das Ziel zu erreichen, musste das Unternehmen seine getrennten Systeme auf einer Plattform konsolidieren, damit die Ingenieure sich auf wertschöpfende Aufgaben wie die Entwicklung von ML- und Large-Language-Modellen konzentrieren konnten. So entschied sich das Team für die Databricks Data Intelligence Platform, um ihre Data Warehouse- und KI-Workloads auf einer einzigen Plattform zusammenzuführen.

Die Umstellung auf Lakehouse führt zu vereinheitlichten Daten, Analysen und KI

Trotz eines Teams talentierter Engineers hat InMobi eine wichtige Lektion über Produktivität und betriebliche Flexibilität gelernt. Sundaram erklärt: „Wir haben erkannt, dass wir zu viel Zeit für die Wartung unserer Umgebung aufwenden mussten, was unsere Fähigkeit beeinträchtigte, geschäftliche Anwender zu unterstützen und mit ihnen zusammenzuarbeiten. Wir wollten strategischer vorgehen und effizientere Wege finden, Daten zu nutzen.“ Nach eingehender Prüfung des aktuellen Multicloud-Data-Warehouses und der Optionen für den Eigenbau entschied das Unternehmen, dass die Databricks Data Intelligence Platform am besten passt: mehr Entwicklerproduktivität durch eine einfachere Infrastruktur und das beste Preis-Leistungs-Verhältnis.

Also begann das Team, in Zusammenarbeit mit Databricks die Migration zu planen. Mit über einem Jahrzehnt an Anpassungen auf bestehenden Systemen und einem Datenbestand von über 1 Petabyte wusste InMobi, dass die Migration eine komplexe Aufgabe sein würde. Allein auf der ETL-Seite waren 150 Pipelines und acht Teams mit der Migration der Open-Source-Software Apache Spark™ beschäftigt. Außerdem mussten etwa 300 Berichts-Dashboards migriert werden, um sicherzustellen, dass die Bereitstellung von Informationen für Lieferanten und Kunden nicht unterbrochen wird. Zusätzlich waren etwa 300 Berichts-Dashboards zu migrieren, um die kontinuierliche Informationsbereitstellung für Lieferanten und Kunden sicherzustellen. Dazu arbeitete InMobi (es wurden zwei Engineers je Team abgestellt) und mit Databricks‘ Implementierungspartner Celebal Technologies an den Optimierungen für eine reibungslose Migration.

So konnte Databricks InMobi bei der problemlosen Umstellung vom Multicloud Data Warehouse zum Lakehouse helfen. „Databricks verschafft uns den Vorteil, dass wir branchenweit einmalige Funktionen nutzen können, die uns ein technisches Alleinstellungsmerkmal vermitteln“, so Sundaram. „Ihre Expertise kam während der Migration zum Tragen: Sie halfen, unsere Compute-Ressourcen mit Blick auf Kosten und Leistung zu optimieren. Bemerkenswert waren die Eigeninitiative von Databricks bei Umsetzung der Optimierungen sowie die Transparenz und Aussagekraft.“

databricks

Ausliefern stärker personalisierter Werbung an mobile Kunden überall

Mit einer einheitlichen, optimierten Lakehouse-Architektur ist InMobi nun in der Lage, seine robusten Kundendaten voll auszuschöpfen, um intelligentere und stärker personalisierte mobile Werbung auszuliefern. Verschiedene Teams verwenden Databricks Notebooks für Ad-hoc-Analysen, Power BI für Visualisierungen mit Databricks SQL als serverlosem Data Warehouse auf der Lakehouse-Architektur und MLflow für den Aufbau ihrer nächsten KI-Plattform. Auch bei der Erkennung von Anomalien arbeiten sie mit Delta Live Tables sehr erfolgreich: Die Einhaltung der SLAs konnte um 50 % verbessert und die Kosten um 80 % gesenkt werden. Auch Datensilos und Probleme mit der Auffindbarkeit von Daten sind dank Unity Catalog kein Thema mehr. Mit Unity Catalog lässt sich der Zugriff auf Tabellen- und Spaltenebene steuern und gleichzeitig sicherstellen, dass die komplette Datenhistorie erfasst wird. So wissen die Mitarbeiter immer, woher die Daten stammen und ob sie veraltet sind oder nicht. Mit einer Plattform, die auf die Bedürfnisse von Analyse und KI zugeschnitten ist, setzt das InMobi-Team auf neue Technologien wie Large Language Models (LLMs), um seinen Kunden effizientere Erkenntnisse zu vermitteln. „Wir haben damit begonnen, LLMs zu implementieren, damit es für unsere Endnutzer leichter wird, eine Frage zu stellen und benötigte Informationen zu erhalten. Die Lakehouse-Architektur macht diese Arbeit auf Dauer einfacher, da die Jobs automatisch im Hintergrund ausgeführt werden. Dadurch bekommen unsere Teams die Möglichkeit, ohne Fachwissen einfach eine Frage zu stellen und die gewünschten kontextbezogenen Antworten auf Knopfdruck zu erhalten“, erklärt Sundaram.

Was die Auswirkungen auf das Geschäft angeht, gab es nach der Umstellung eine Reihe messbarer Verbesserungen in allen Bereichen. Die Infrastrukturkosten sanken um 34 %, während die Abfragegeschwindigkeit um 15 % zunahm und es 20 % weniger Jobausfälle gab als zuvor. Insgesamt führte dies zu einer um 20 % verbesserten Leistung bei der Bereitstellung von Berichten und Erkenntnissen für Endnutzer. Die Gesamtbetriebskosten (TCO) sind um 32 % niedriger als bei der Verwendung eines Multicloud Data Warehouse, und auch die Kosten für den Betrieb der ETL-Pipelines konnten um 24 % gesenkt werden. Qualitativ gesehen ist das Team insgesamt zuverlässiger, auch weil die Systeme stabiler sind als je zuvor. Daher konnte das Team bei den Kunden einen positiven Imagezuwachs verzeichnen.

„Unsere Experimentierquote hat sich enorm verbessert“, sagt Mohit Saxena, Mitbegründer und Group CTO bei InMobi. „Databricks hat die Zusammenarbeit vereinfacht, und der einheitliche Ansatz des Lakehouse ermöglicht es uns, bei der Bereitstellung neuer Funktionen und Produkte effizienter und produktiver vorzugehen und gleichzeitig alle gesetzlichen Vorschriften einzuhalten.“

Durch die Umstellung auf eine vereinheitlichte Plattform auf der Databricks Data Intelligence Platform kann sich InMobi nun ganz den Innovationen im Bereich der mobilen Werbung widmen und eine Echtzeitpersonalisierung anbieten, die sowohl für die Kunden von InMobi als auch für die internen Endnutzer einen handfesten Mehrwert schafft.

Weitere Kundenreferenzen lesen

KAPITEL 5.2

Akamai: Mit Delta Lake Echtzeitanalysen im großen Stil liefern

Akamai betreibt ein flächendeckendes, hochgradig verteiltes Content Delivery Network (CDN). Dieses nutzt etwa 345.000 Server in mehr als 135 Ländern und über 1.300 Netzwerke weltweit, um den Internetverkehr für einige der größten Unternehmen in den Bereichen Medien, Commerce, Finanzen, Handel und vielen anderen Branchen zu routen. Etwa 30 % des gesamten Internetverkehrs fließt über Server von Akamai. Außerdem bietet Akamai Cloud-Sicherheitslösungen an.

2018 brachte das Unternehmen ein Web Security Analytics Tool auf den Markt, das Akamai-Kunden eine zentrale, einheitliche Oberfläche zur Bewertung und Analyse einer Vielzahl von Streaming-Sicherheitsereignissen bietet. Das Tool hilft Akamai-Kunden dabei, bei Auftreten von Sicherheitsvorfällen in Echtzeit informierte Maßnahmen zu ergreifen. Durch den Einsatz von Delta Lake und der Databricks Data Intelligence Platform für das Webanalysetool ist Akamai in der Lage, enorme Datenmengen zu streamen und gleichzeitig die strengen SLAs einzuhalten, die es seinen Kunden anbietet.

Delta Lake ermöglicht es uns nicht nur, die Daten besser abzufragen, sondern auch ein größeres Datenvolumen zu erfassen. Wir haben allein im letzten Jahr einen Anstieg des Datenverkehrs und der Datenmenge um 80 % erlebt. Daher ist es für uns entscheidend, schnell skalieren zu können.

– TOMER PATEL
Engineering Manager,
Akamai

<1 Minute
Erfassungszeit (von vormals 15 Minuten)

>85 %
der Abfragen haben eine Reaktionszeit von maximal 7 Sekunden

Erfassen und Streamen enormer Datenmengen

Das Web Security Analytics Tool von Akamai erfasst pro Sekunde etwa 10 GB an Daten zu Sicherheitsereignissen. Das Datenvolumen kann zudem erheblich ansteigen, wenn Einzelhandelskunden eine große Anzahl von Verkäufen tätigen, aber auch an den wichtigen Einkaufstagen wie dem Black Friday oder Cyber Monday. Das Web Security Analytics Tool speichert mehrere Petabyte an Daten für Analysezwecke. Diese Analysen werden durchgeführt, um die Kunden von Akamai zu schützen und ihnen die Möglichkeit zur eigenständigen Untersuchung und Abfrage von Sicherheitsereignissen zu geben.

Ursprünglich basierte das Tool auf einer On-Premises-Architektur mit Apache Spark™ auf Hadoop. Akamai bietet seinen Kunden strikte Service Level Agreements (SLAs) von 5 bis 7 Minuten ab dem Zeitpunkt, an dem ein Angriff stattfindet, bis zur Anzeige im Tool. Das Unternehmen wollte die Erfassungs- und Abfragegeschwindigkeit verbessern, um diese SLAs zu erfüllen. „Die Daten müssen so nah an der Echtzeit sein wie möglich, damit die Kunden sehen können, was sie angreift“, sagt Tomer Patel, Engineering Manager bei Akamai. „Daher ist es wichtig, den Kunden schnell abfragbare Daten zur Verfügung zu stellen. Wir wollten weg von On-Premises, um die Leistung und unsere SLAs zu verbessern, damit die Latenzzeit nur noch Sekunden statt Minuten beträgt.“

Nachdem Akamai mit mehreren Unternehmen Machbarkeitsstudien durchgeführt hatte, entschied man sich dafür, seine Streaming-Analytics-Architektur auf Spark und der Databricks Data Intelligence Platform aufzubauen. „Aufgrund unserer Größe und der Anforderungen unseres SLA haben wir uns für Databricks entschieden“, sagt Patel. „Wenn wir die Speicheroptimierung und das Caching von Daten einbeziehen, könnten wir mit einer anderen Lösung nicht das gleiche Leistungsniveau erreichen.“

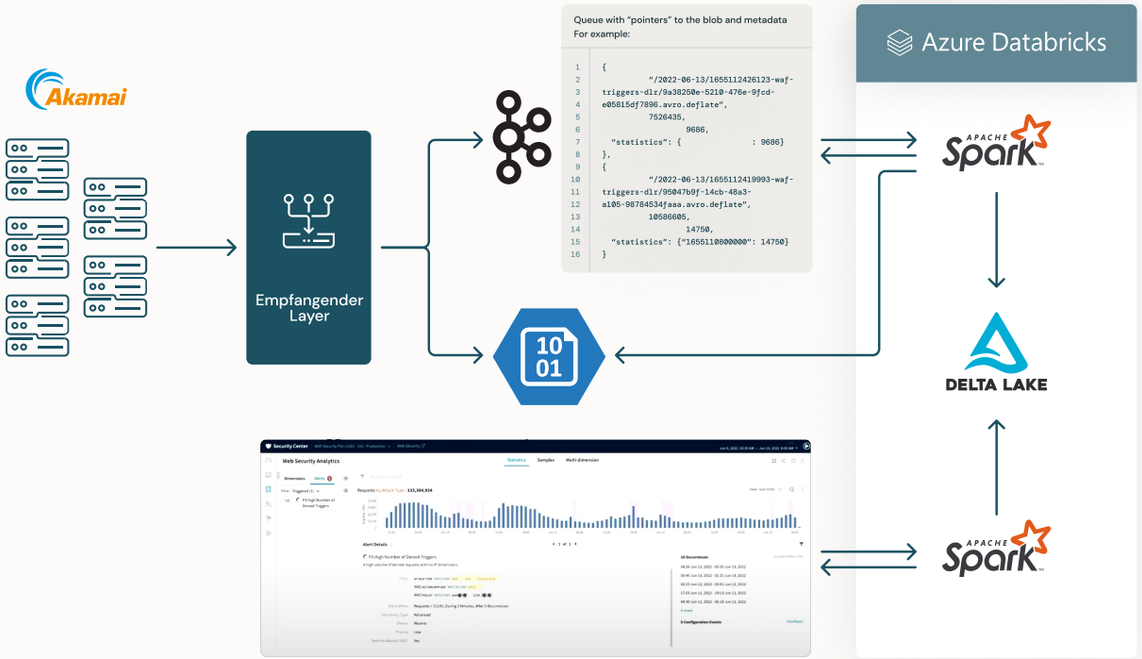
Geschwindigkeit erhöhen und Kosten senken

Heute erfasst und transformiert das Web Security Analytics Tool Daten, speichert sie in einem Cloud-Speicher und sendet den Speicherort der Datei über Kafka. Dann verwendet es einen Databricks-Job als Erfassungsanwendung. Delta Lake, das Open-Source-Speicherformat, das der Data Intelligence Platform von Databricks zugrunde liegt, unterstützt die Echtzeitabfrage von Websicherheitsanalysedaten. Außerdem ermöglicht Delta Lake Akamai ein schnelles Skalieren. „Mit Delta Lake können wir nicht nur die Daten besser abfragen, sondern auch ein größeres Datenvolumen erfassen“, sagt Patel. „Wir haben allein im letzten Jahr einen Anstieg des Datenverkehrs und der Datenmenge um 80 % erlebt, daher ist es wichtig, dass wir schnell skalieren können.“

Ferner setzt Akamai Databricks SQL (DBSQL) und Photon ein, die für eine extrem schnelle Abfrageleistung sorgen. Patel fügt hinzu, dass Photon die Abfrageleistung wesentlich verbessert habe. Insgesamt ermöglicht die Streaming-Architektur von Databricks in Kombination mit DBSQL und Photon Akamai Echtzeitanalysen, was sich wiederum in geschäftlichen Vorteilen in Echtzeit niederschlägt.

Patel sagt, ihm gefalle, dass Delta Lake Open Source ist, da das Unternehmen von einer Gemeinschaft von Nutzern profitiere, die an der Verbesserung des Produkts arbeiten. „Die Tatsache, dass Delta Lake Open Source ist und eine große Community dahinter steht, bedeutet, dass wir nicht alles selbst implementieren müssen“, sagt Patel. „Wir profitieren von behobenen Fehlern, die andere gefunden haben, und von Optimierungen, die in das Projekt eingebracht werden.“ Akamai hat eng mit Databricks zusammengearbeitet, um sicherzustellen, dass Delta Lake die von Akamai festgelegten Skalierungs- und Leistungsanforderungen erfüllen kann. Diese Verbesserungen wurden wieder in das Projekt eingebracht (wobei sie zum großen Teil als Bestandteil von Delta Lake 2.0 zur Verfügung gestellt wurden), sodass jeder Nutzer, der Delta Lake einsetzt, jetzt von der Technologie profitiert, die in einem so großen Maßstab in einem realen Produktionsszenario getestet wurde.

databricks



Erfüllen hoher Anforderungen an Umfang, Zuverlässigkeit und Leistung

Der Einsatz von Spark Structured Streaming auf der Databricks Data Intelligence Platform ermöglicht es dem Web Security Analytics Tool, riesige Datenmengen zu streamen und den Kunden von Akamai Analytics-as-a-Service in Echtzeit und mit geringer Latenz anzubieten. Auf diese Weise ist Akamai in der Lage, den Kunden innerhalb der SLA-Vorgaben von 5 bis 7 Minuten nach dem Auftreten eines Angriffs Sicherheitsereignisdaten zur Verfügung stellen zu können. „Unser Fokus ist Leistung, Leistung und nochmals Leistung", so Patel. „Die Performance und Skalierbarkeit der Plattform sind ursächlich für unseren Erfolg."

Mit der Data Intelligence Platform von Databricks dauert es jetzt noch nicht einmal mehr eine Minute, um die Sicherheitsereignisdaten aufzunehmen. „Die Verringerung der Erfassungszeit von 15 Minuten auf unter eine Minute ist eine enorme Verbesserung", sagt Patel. „Unsere Kunden profitieren davon, weil ihnen die Daten zu Sicherheitsereignissen schneller vorliegen. So erhalten sie einen Überblick darüber, was genau passiert, und die Möglichkeit, alles zu filtern."

Akamai legt größten Wert darauf, seinen Kunden ein gutes Erlebnis und schnelle Reaktionszeiten zu bieten. Bislang hat Akamai etwa 70 % der Sicherheitsereignisdaten von seiner On-Premises-Architektur auf Databricks verlagert, und das SLA für die Abfrage- und Antwortzeiten der Kunden hat sich dadurch erheblich verbessert. „Mit der Umstellung auf Databricks erleben unsere Kunden jetzt eine viel bessere Reaktionszeit, denn über 85 % der Abfragen sind bereits nach weniger als 7 Sekunden vollständig bearbeitet." Durch die Bereitstellung solcher Echtzeitdaten kann Akamai seinen Kunden helfen, aufmerksam zu bleiben und eine optimale Sicherheitskonfiguration aufrechtzuerhalten.

Weitere Kundenreferenzen lesen

KAPITEL 5.3

Quartile: Wie man zur größten E-Commerce-Werbeplattform wird

Quartile ist die weltweit größte kanalübergreifende E-Commerce-Werbeplattform mit über 5.000 angeschlossenen Werbekunden aus mehr als zehn Kanälen. Die Plattform setzt auf sechs patentierte Technologien für maschinelles Lernen auf und automatisiert und optimiert E-Commerce-Werbung bei Google, Facebook, Amazon, Instacart, Walmart und anderen. Quartile bringt führende Technologie mit Marketingexperten zusammen, die Strategien entwickeln, die genau auf die Geschäftsziele ihrer Kunden zugeschnitten sind. Tausende von Anbietern auf der ganzen Welt vertrauen auf den Ansatz der Trichteroptimierung von Quartile, um das volle Potenzial ihres Vertriebs und ihrer Werbung kanalübergreifend, vollständig integriert und in großem Umfang auszuschöpfen.

80 %
weniger Datenspeicherbedarf im Vergleich zu herkömmlichen Datenbanken

10 × schneller
45 Minuten für die Gebotsoptimierung (von vormals 7,5 Stunden)

Mit der Databricks Data Intelligence Platform können unsere Technologieteams neue Lösungen schneller auf den Markt bringen und die Kunden mit hochwertigen Datasets begeistern. Ohne Databricks und die Lakehouse-Architektur wäre es deutlich schwieriger gewesen, zur größten kanalübergreifenden E-Commerce-Anzeigenplattform zu werden.

– DANIEL KNIJNIK
CEO,
Quartile

Größe der Daten und erforderliche Leistung zur Skalierung

Quartile stand vor der Herausforderung, die Daten für mehrere Werbekanäle speichern und verarbeiten zu müssen, da die Berichterstattung und Konsolidierung der Verkaufsdaten, einschließlich der Attribution von über 60 Tagen, entscheidend war. Die bisherige Architektur konnte die Datenmenge, die Quartile verarbeiten musste, nicht bewältigen. Das Team verarbeitete über 10 TB Daten in einem einzigen Batch-Job, wobei alle notwendigen Transformationen für die Berichterstattung durchgeführt wurden. Dies führte zu Serverausfällen und verspäteter Bereitstellung der Datenpunkte. Allein dieser Prozess, der für die Verbesserung der Anzeigenleistung zuständig ist, litt unter gravierenden Leistungsproblemen. So dauerten einzelne Jobs Tag für Tag bis zu 7,5 Stunden!

Technologieentwicklung – von Legacy zum modernen Daten-Stack

Im Zuge der Weiterentwicklung des Unternehmens hat die Datenarchitektur von Quartile einen neuen Reifegrad erreicht und den Schritt von traditionellen SQL-Datenbanken, die in der Azure-Cloud ausgeführt werden, zu einer neuen Lösung mit der Databricks Data Intelligence Platform als Grundlage vollzogen. Diese Modernisierung hatte unmittelbare Auswirkungen auf mehrere Bereiche des Unternehmens und bot den Kunden von Quartile klare Vorteile: Mit Delta Lake gab es mehr Datenzuverlässigkeit und schnellere Performance bei reduzierten Kosten. Die Migration von einer traditionellen SQL-Datenbank auf Databricks brachte eine deutliche Reduktion des Datenvolumens. Dank Delta-Optimierungen, inklusive Versionskontrolle und Parquet-Komprimierung, sank der Speicherplatzbedarf von 90 TB auf rund 18 TB.

Dank der Lakehouse-Architektur konnte sich der Daten-Stack von Quartile weiterentwickeln. Dies wurde durch mehrere Prozesse erreicht: durch den Einsatz von Databricks Auto Loader für die inkrementelle und effiziente Verarbeitung neuer Datendateien, sobald sie im Cloud-Speicher ankommen, durch den Einsatz von Delta Lake für seine formatoffene Speicherschicht, die für Zuverlässigkeit, Sicherheit und Leistung im Data Lake sorgt, durch die Nutzung von Databricks SQL, das Data Engineers und Datenanalysten mit einfach zu erstellenden Abfragen und Dashboards noch näher zusammenbringt, und durch die Verwendung von Databricks Workflows, das alle Teile stabil und skalierbar miteinander verknüpft.

Um seinen Kunden optimale Erlebnisse zu bieten, muss Quartile fehlerfreie Daten abrufen können. Dies ist eine wichtige Herausforderung, die mit den benutzerdefinierten Funktionen von Spark leichter zu bewältigen ist. Auf diese Weise nutzt das Unternehmen die Kraft der Parallelität, um die Verarbeitung in so viele Teile wie nötig zu zerlegen. Um ihre Lösung skalierbar zu machen, nutzen sie Terraform zur Bereitstellung aller Arbeitsbereiche, zum einfachen Hochfahren neuer Cluster und Jobs sowie zur Gewährleistung der Einhaltung der richtigen Standards im gesamten Unternehmen.

Den Partnern helfen, die Anzeigenschaltung zu optimieren

Es ist wichtig, dass die Kunden von Quartile eine zentrale Lösung nutzen können, um ihre Umsätze, Kosten und sonstigen Metriken im Zusammenhang mit ihren Kampagnen zu analysieren. Bei Quartile macht man sich solides Data Engineering zunutze und integriert Databricks mit Power BI, um die Dashboards direkt in das Portal einzubetten. So können Kunden an einem zentralen Ort sowohl Marketingkampagnen kanalübergreifend konfigurieren als auch Leistungsänderungen nachverfolgen, während sie gleichzeitig durch die Nutzung des Delta Lake-Dateiformats auf Objektspeicher einen im Vergleich zu herkömmlichen Data Warehouses um 80 % geringeren Datenspeicherbedarf haben. Die Möglichkeit, Daten aus all den verschiedenen Kanälen zu kombinieren, hat bereits einigen ihrer Kunden geholfen. SmartyPants beispielsweise ist seit Beginn der Zusammenarbeit mit Quartile um über 100 % gewachsen.

Aber das ist noch nicht alles. Quartile hat auch Algorithmen zur Verbesserung der Leistung von Anzeigen zum Patent angemeldet, die mithilfe der Machine-Learning-Persona in Databricks umgesetzt werden. Die Möglichkeit, eine zentrale Lakehouse-Architektur als Grundlage für den gesamten Daten-Stack zu nutzen, hat das Leben der Quartile-Entwickler viel einfacher gemacht: Jetzt können sie den Schwerpunkt auf die Entwicklung innovativer Lösungen legen und immer bessere Ergebnisse für ihre Kunden erzielen. Ein weiteres Beispiel: OfficeSupply hat im ersten Jahr der Zusammenarbeit mit Quartile hervorragende Ergebnisse erzielt: Die Google Ads-Einnahmen stiegen um 67 % und die Google Shopping-Klicks für Markenbegriffe um 103 %, weil die Leistung der einzelnen Aufträge verbessert wurde: Früher dauerten sie 7,5 Stunden, jetzt dank der Lakehouse-Architektur nur noch 45 Minuten.

Auch in Zukunft wird Quartile mit Databricks zusammenarbeiten, um den modernen Daten-Stack zu erstellen und neue Lösungen zu integrieren und zu testen. Dazu gehören Delta Live Tables für bessere Datenqualitätsprüfungen, Delta Sharing, um Kunden ihre eigenen Daten senden zu können, und Data Marketplace, mit dem Kunden einen schnellen Einstieg finden. Quartile hat sich das ehrgeizige Ziel gesetzt, den ersten kanalübergreifenden Lernalgorithmus für die Anzeigenoptimierung in diesem Bereich zu entwickeln, und Databricks wird im Mittelpunkt dieser Innovationen stehen.

Infos zu Databricks

Databricks ist das Unternehmen für Daten und KI. Mehr als 10.000 Unternehmen weltweit – darunter Comcast, Condé Nast, Grammarly und mehr als 50 Prozent der Fortune 500 – setzen bei der Konsolidierung und Demokratisierung ihrer Daten, Analysen und KI auf die Databricks Data Intelligence Platform. Databricks wurde von den Erfindern von Lakehouse, Apache Spark™, Delta Lake und MLflow gegründet und hat seinen Hauptsitz in San Francisco mit Niederlassungen auf der ganzen Welt.

Wenn Sie mehr erfahren möchten, folgen Sie Databricks auf Twitter, LinkedIn und Facebook.

JETZT KOSTENLOS TESTEN

Kontaktieren Sie uns für eine personalisierte Demo:
databricks.com/company/contact

databricks