databricks

eBook

Il grande libro del Data Warehousing e della BI

Sommario

I dati hanno un ruolo essenziale nel successo di qualsiasi azienda. Mentre le organizzazioni si affidano sempre di più ai dati, la necessità di soluzioni efficienti per la loro gestione si fa più pressante. Per rispondere a questa esigenza, la Databricks Data Intelligence Platform permette di gestire, utilizzare e accedere in maniera efficace a dati e AI. Costruita sull'architettura lakehouse, combina le migliori funzionalità di data lake e data warehouse, riducendo i costi e accelerando le iniziative di dati e AI. La piattaforma offre governance unificata per dati e AI oltre a un versatile motore di query per ETL, SQL, machine learning e BI.

Introduzione alla Data Intelligence Platform

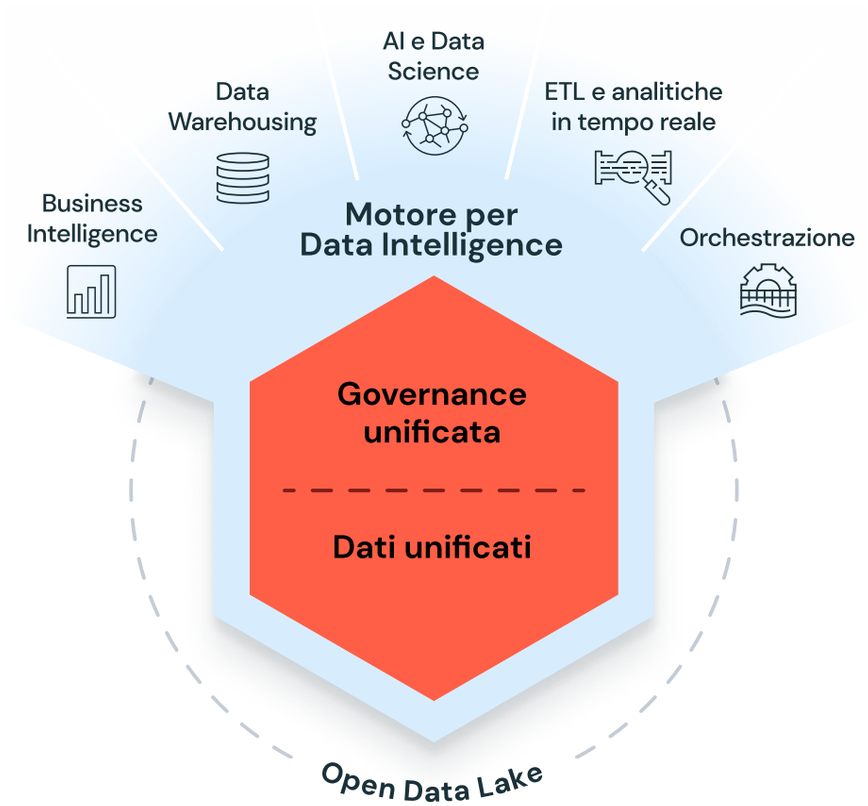
La Databricks Data Intelligence Platform consente alle organizzazioni di gestire, utilizzare e accedere in maniera efficace a tutti i dati e AI. La piattaforma, costruita sull'architettura lakehouse con un livello di governance unificato per dati e AI e un singolo motore di query che copre ETL, SQL, AI e BI, combina il meglio di data lake e data warehouse per ridurre i costi e accelerare le iniziative di dati e AI.

Coniugando l'AI generativa con i vantaggi in termini di unificazione offerti da un lakehouse, la Data Intelligence Platform offre un motore di Data Intelligence chiamato DatabricksIQ in grado di comprendere la semantica unica dei dati di un'azienda. DatabricksIQ analizza automaticamente tutti gli aspetti dei dati, inclusi contenuto, metadati e modelli di utilizzo (come query, reportistica e provenienza). Questa analisi completa permette alla piattaforma di apprendere, migliorarsi e aggiungere continuamente nuove funzionalità, ottimizzando la gestione dei dati e le applicazioni di AI. Grazie a questa profonda comprensione dei dati, la Databricks Data Intelligence Platform consente:

- **Accesso con linguaggio naturale.** Sfruttando modelli di AI, la Data Intelligence Platform permette di lavorare con i dati usando un linguaggio naturale personalizzato in base al gergo settoriale e agli acronimi di ogni organizzazione. La piattaforma osserva il modo in cui vengono usati i dati nei carichi di lavoro esistenti per apprendere il lessico specifico dell'organizzazione e offrire un'interfaccia in linguaggio naturale personalizzata a tutti gli utenti, dai non addetti ai lavori ai data engineer.
- **Catalogazione e scoperta semantiche.** L'AI generativa comprende modelli di dati, metriche e KPI di ciascuna organizzazione per offrire impareggiabili funzionalità di scoperta o identificare automaticamente eventuali discrepanze nel modo in cui i dati vengono usati.
- **Gestione e ottimizzazione automatizzate.** I modelli AI possono ottimizzare layout, ripartizione e indicizzazione dei dati in base al loro uso, riducendo la necessità di configurazione e ottimizzazione manuale.

Governance e privacy migliorate. Le piattaforme di Data Intelligence possono individuare, classificare e salvaguardare automaticamente i dati sensibili, rendendone al contempo più semplice la gestione tramite il linguaggio naturale.

Supporto di prima classe per carichi di lavoro AI. Le piattaforme di Data Intelligence possono migliorare qualsiasi applicazione AI di un'azienda consentendole di connettersi ai dati aziendali pertinenti e di appoggiarsi alla semantica appresa dalla piattaforma (metriche, KPI, ecc.) per fornire risultati accurati. Gli sviluppatori di applicazioni AI non avranno più bisogno di affidarsi alla fragile ingegneria dei prompt per ricavare faticosamente informazioni dai dati.



La Databricks Platform, inoltre, semplifica lo sviluppo di applicazioni AI aziendali. Grazie alla piattaforma, infatti, le aziende possono costruire facilmente applicazioni AI che capiscano i loro dati. La Databricks Platform offre molteplici funzionalità per integrare direttamente i dati aziendali nei sistemi AI, tra cui:

- RAG (retrieval augmented generation) a 360 gradi per costruire agenti conversazionali di qualità partendo dai dati dell'organizzazione.
- Addestramento di modelli personalizzati da zero usando i dati di un'organizzazione o continuando il preaddestramento di modelli esistenti quali MPT e Llama 2, in modo da potenziare ulteriormente le applicazioni AI con la profonda comprensione del settore specifico.
- Inferenza serverless efficiente e sicura sui dati aziendali, con funzionalità di governance unificata e monitoraggio della qualità.
- MLOps complete basate sul popolare progetto open source MLflow, dove tutti i dati prodotti sono automaticamente fruibili, tracciabili e monitorabili nel lakehouse.

Questa guida illustra le best practice del data warehousing sulla Databricks Data Intelligence Platform attraverso un'ampia gamma di casi d'uso tratti da situazioni reali. Scopri come la piattaforma aiuta le aziende di ogni dimensione a tradurre dati grezzi in informazioni fruibili usando SQL: dall'acquisizione dei dati alla loro elaborazione, da AI e LLM ad analisi e BI. Ti metteremo a disposizione architetture di riferimento ed esempi di codice per aiutarti a scoprire tutti gli aspetti del ciclo di vita dei dati sulla Data Intelligence Platform.

Per maggiori informazioni su Databricks SQL, leggi il nostro eBook Why the Data Lakehouse Is Your Next Data Warehouse.

CAPITOLO

02

Data warehousing con funzioni di Data Intelligence e automazione

databricks

Il successo delle iniziative per dati e AI basate su lakehouse si fonda su un'architettura dei dati semplificata, qualità e governance dei dati, scalabilità e prestazioni. Su queste fondamenta, le organizzazioni costruiscono una strategia che li accompagnerà nell'intricato labirinto della gestione e dell'analisi dei dati.

Architettura dei dati semplificata

L'architettura lakehouse risolve il problema dei silos incorporando al tempo stesso le funzionalità di un data warehouse. La prima fase di questo approccio è la centralizzazione dei dati in un data lake basato su cloud. Questa solida base, supportata da **Delta Lake**, permette ai casi d'uso di analisi e AI di operare su una singola sorgente di dati, riducendo i costi di archiviazione e snellendo il data engineering. L'architettura lakehouse integra una struttura unificata per la governance e la sicurezza con **Unity Catalog**, assicurando un controllo granulare e un accesso tempestivo ai dati ai rispettivi team.

L'approccio olistico dell'architettura lakehouse copre l'intero ciclo di vita dei dati, la loro trasformazione e il loro impatto su vari carichi di lavoro di analisi e AI. Il fatto che tutti i carichi di lavoro condividano gli stessi dati e aderiscano a policy di sicurezza e governance comuni garantisce di poter contare sempre su informazioni accurate e affidabili. Le barriere tra silos funzionali si abbattono, aprendo la strada a una collaborazione più fluida e, di conseguenza, a una maggiore produttività nella fornitura di prodotti di dati.

Un'architettura dei dati semplificata offre inoltre ulteriori vantaggi, tra cui:

- Eliminazione dei silos. Basandosi su open source e standard, il lakehouse semplifica il paesaggio dei dati e agevola le operazioni su dati e AI.
- Un'unica piattaforma per integrazione, archiviazione, elaborazione, governance, condivisione, analisi e AI. Offre un approccio unificato alla gestione di dati strutturati e non strutturati, una vista completa sull'origine e la provenienza dei dati e un set di strumenti consolidato per Python e SQL, notebook, IDE, batch, streaming e tutti i principali fornitori di servizi cloud.
- Ottimizzazione automatizzata di prestazioni e archiviazione, che garantisce costi operativi ottimali (TCO) impostando benchmark di prestazioni per data warehousing e AI, inclusi processi avanzati come modelli linguistici di grandi dimensioni (LLM).

Governance e qualità dei dati

Data l'importanza fondamentale che rivestono per le organizzazioni, i dati devono soddisfare severi requisiti di qualità e governance, soprattutto all'aumentare dei volumi e della varietà. Le organizzazioni devono dare priorità ad accuratezza, affidabilità e conformità per massimizzare il potenziale del loro lakehouse. Dati di scarsa qualità possono fornire informazioni distorte, mentre una governance debole può portare a inadempienze normative e falle nella sicurezza. La Databricks Data Intelligence Platform risponde a questi problemi con il suo Unity Catalog, un framework integrato per gestire e migliorare la qualità dei dati durante tutto il loro ciclo di vita. La funzionalità di governance sull'architettura lakehouse rende possibile:

- Scoprire, classificare e consolidare risorse dati e AI provenienti da varie piattaforme su qualsiasi cloud, migliorando la consultazione e l'estrazione di informazioni mediante il linguaggio naturale, il tutto da un unico punto di accesso.
- Semplificare la gestione degli accessi attraverso un'interfaccia unificata che assicura un accesso costante e sicuro su più cloud e piattaforme, con controllo granulare migliorato e policy low-code scalabili.
- Sfruttare la potenza dell'AI per automatizzare il monitoraggio di modelli di dati e ML, ricevere notifiche proattive in caso di problemi, semplificare il debugging e avere una panoramica completa delle operazioni del lakehouse attraverso tabelle integrate nel sistema.
- Condividere in modo efficiente risorse di dati e AI tra più cloud, aree e piattaforme usando Delta Sharing in Unity Catalog, rendendo possibile una collaborazione sicura e la creazione di valore senza processi complicati o repliche costose.

Scalabilità e prestazioni

All'aumentare del volume dei dati, un'architettura lakehouse distribuisce le funzionalità di calcolo, indipendenti da quelle di archiviazione, con l'obiettivo di mantenere prestazioni costanti a costi ottimali. Progettata per essere elastica, la Databricks Data Intelligence Platform consente alle organizzazioni di scalare le operazioni sui dati secondo necessità. La scalabilità si estende su più dimensioni:

SERVERLESS

La Databricks Platform utilizza risorse di calcolo serverless basate su cloud, permettendo ai carichi di lavoro di adattarsi e scalare elasticamente in base alla capacità computazionale richiesta. Questa allocazione dinamica delle risorse garantisce rapidità nell'elaborazione e nell'analisi dei dati anche durante i picchi nella domanda.

SIMULTANEITÀ

Sfruttando le funzionalità di calcolo serverless e le ottimizzazioni basate sull'AI, la Databricks Platform facilita l'elaborazione dei dati e l'esecuzione di query in concomitanza. Questo fa sì che molteplici utenti e team possano svolgere attività di analisi simultaneamente senza limitare le prestazioni.

ARCHIVIAZIONE

La piattaforma si integra direttamente con i data lake, facilitando l'archiviazione a basso costo di ampi volumi di dati e assicurando al contempo la disponibilità e l'affidabilità dei dati. Ottimizza inoltre l'archiviazione dei dati per migliorare le prestazioni, riducendo i costi di storage.

Per quanto essenziale, la scalabilità non può essere disgiunta dalle prestazioni. Da questo punto di vista, la Databricks Data Intelligence Platform non teme rivali, offrendo un'ampia gamma di ottimizzazioni basate sull'AI.

ELABORAZIONE OTTIMIZZATA DELLE QUERY

La piattaforma utilizza tecniche di ottimizzazione del machine learning per accelerare l'esecuzione delle query. Sfrutta indicizzazione automatica, memorizzazione nella cache e pushdown dei predicati per garantire un'elaborazione efficiente delle query con generazione più rapida di informazioni.

AUTOSCALABILITÀ

La Databricks Platform scala in modo intelligente risorse serverless in base ai carichi di lavoro; in questo modo si ha la certezza di pagare solo per la potenza di calcolo utilizzata, mantenendo al contempo una prestazione ottimale delle query.

PHOTON

Il nuovo motore MPP (elaborazione a elevata parallelizzazione) nativo sulla Databricks Platform offre prestazioni di query estremamente rapide a costi contenuti per acquisizione di dati, ETL, streaming, data science e query interattive, direttamente sul proprio data lake. Photon è compatibile con le API Apache Spark™, senza modificare il codice e senza vincoli.

DELTA LAKE

Delta Lake con Unity Catalog e Photon offre il miglior rapporto prezzo/prestazioni con una soluzione pronta all'uso, senza necessità di ottimizzazione manuale. La Databricks Platform utilizza modelli AI per risolvere i problemi comunemente associati all'archiviazione dei dati, così da offrire prestazioni più veloci senza l'onere di gestire manualmente le tabelle, neanche quando cambiano nel tempo.

- La Predictive I/O per gli aggiornamenti ottimizza i piani di query e layout dei dati per prestazioni ottimali, bilanciando in modo intelligente le prestazioni in lettura e in scrittura. Potrai così ottenere più valore dai tuoi dati senza dover scegliere tra strategie alternative come copy-on-write o merge-on-read.
- Il clustering liquido assicura le prestazioni di una tabella ben ottimizzata e partizionata senza le preoccupazioni tipicamente associate al partizionamento, come la possibilità o meno di partizionare colonne ad alta cardinalità o l'esigenza di costose riscritture quando si cambiano le colonne di partizione. Il risultato sono tabelle velocissime, ben clusterizzate, che richiedono una configurazione minima.
- L'ottimizzazione predittiva perfeziona automaticamente i dati per garantire le migliori prestazioni, al miglior prezzo. Apprende dai modelli di utilizzo dei dati, pianifica le ottimizzazioni da implementare e poi le esegue su un'infrastruttura serverless iper-ottimizzata.

Una volta gettate le basi dell'architettura lakehouse, si possono valutare la distribuzione di funzionalità di data warehouse e analisi su Databricks con appropriate strutture di dati e funzionalità di gestione facilitate da Databricks SQL (DB SQL).

Databricks SQL Serverless: il miglior data warehouse per un'architettura lakehouse

Databricks SQL è stato introdotto per migliorare le funzionalità di data warehousing e offrire un supporto SQL completo sulla Databricks Data Intelligence Platform. Semplifica la trasformazione, la consultazione e l'analisi dei dati basate su SQL, rispondendo alle esigenze di utenti con diverso background tecnico. Che si tratti di analisti BI, data architect o analytics engineer, la sua intuitiva interfaccia SQL facilita query e complesse operazioni sui dati senza richiedere competenze di programmazione specialistiche. Un più ampio accesso ai dati favorisce lo sviluppo di una cultura che mette al centro l'organizzazione nel suo complesso, dando a più team la possibilità di prendere decisioni sulla base di insight.

Databricks SQL si distingue per la capacità di gestire enormi set di dati in modo veloce ed efficiente. L'utilizzo di Photon, il motore di nuova generazione di Databricks con ottimizzazioni basate sull'AI, assicura rapidità nell'elaborazione e nell'analisi dei dati, riducendo notevolmente il tempo necessario per l'esecuzione delle query. Una garanzia di prestazioni elevate è essenziale alle organizzazioni per rispondere alle sfide poste dai dati ed estrarre informazioni approfondite da un'ampia varietà di data set. Databricks SQL inoltre promuove la collaborazione, fornendo uno spazio di lavoro in cui i professionisti dei dati possono condividere istantaneamente query, risultati e deduzioni. Questo ambiente condiviso favorisce lo scambio di conoscenze e velocizza i processi decisionali, consentendo alle organizzazioni di sfruttare l'intelligenza collettiva dei team.

Databricks SQL incorpora inoltre strumenti avanzati per la governance, la sicurezza e la conformità dei dati, permettendo alle organizzazioni di mantenere un'alta qualità dei dati, implementare restrizioni all'accesso, supervisionare le attività, proteggere le informazioni sensibili e soddisfare gli standard normativi. Riepilogando, Databricks SQL offre:

- **Insight in tempi più rapidi**
  Accedi ai dati usando il linguaggio naturale: Databricks SQL creerà automaticamente le query SQL al posto tuo. Potrai così affinare le query più rapidamente e i dati saranno accessibili a tutti i componenti della tua organizzazione.
- **Miglior rapporto prezzo/prestazioni**
  Il calcolo serverless combinato con l'elaborazione ottimizzata dall'AI consente di ottenere prestazioni di livello superiore e di scalare a costi inferiori, senza la necessità di gestire l'infrastruttura cloud.
- **Governance unificata**
  Stabilisci un livello di governance unificato per tutte le risorse di dati e AI, ovunque si trovino.
- **Riduzione della complessità**
  Unifica dati, analisi e AI in una sola piattaforma che supporta SQL e Python, notebook e IDE, batch e streaming e tutti i principali fornitori di servizi cloud.
- **Ricco ecosistema**
  Utilizza SQL e i tuoi strumenti preferiti come Power BI, Tableau, dbt e Fivetran con Databricks per BI, acquisizione e trasformazione dei dati.

Conclusioni

La Databricks Data Intelligence Platform rappresenta un significativo passo avanti nel campo del data warehousing e dell'analisi. Risponde all'esigenza di carichi di lavoro di data warehousing efficienti in un panorama aziendale sempre più guidato dai dati. L'architettura semplificata della piattaforma centralizza i dati con funzionalità di data warehouse a 360 gradi, consentendo di ridurre i costi e il tempo necessario per trasformare i dati grezzi in informazioni fruibili su larga scala e per unificare batch e streaming. Inoltre, garantisce qualità e governance dei dati tramite lo Unity Catalog, permettendo alle organizzazioni di scoprire, proteggere e gestire facilmente tutti i loro dati con una governance granulare che traccia la provenienza dei dati su tutti i cloud.

Scalabilità e prestazioni sono gli elementi distintivi della Databricks Platform, insieme a calcolo serverless, simultaneità degli accessi e strategie di archiviazione ottimizzate. Le ottimizzazioni basate sull'AI migliorano l'elaborazione delle query, aumentano le prestazioni e ottimizzano i dati per offrire il miglior rapporto tra prestazioni e prezzo, rendendo l'analisi più veloce e conveniente.

Databricks SQL amplia ulteriormente le funzionalità della piattaforma offrendo un supporto SQL completo che facilita la trasformazione e l'analisi dei dati per molte categorie di utenti. Promuove la collaborazione, la governance e un ricco ecosistema di strumenti, abbattendo i silos e consentendo alle organizzazioni di sfruttare appieno il potenziale dei loro dati. Esaminiamo ora alcuni casi d'uso per carichi di lavoro di data warehousing e BI sulla Databricks Platform.

MAGGIORI INFORMAZIONI

- → Why the Data Lakehouse Is Your Next Data Warehouse: 2nd Edition
- → What's new in Databricks SQL?
- → Introducing Lakehouse Federation Capabilities in Unity Catalog
- → Introducing AI Functions: Integrating Large Language Models with Databricks SQL

CAPITOLO

03

Best practice di data warehousing nel lakehouse

CAPITOLO 3.1

Le tecniche di modellazione del data warehousing e la loro implementazione nel lakehouse

Usare data vault e schemi a stella nel lakehouse

di Soham Bhatt e Deepak Sekar

Il lakehouse è un nuovo paradigma architettonico che unisce il meglio dei data lake e dei data warehouse. È progettato per essere una piattaforma dati di livello aziendale in grado di gestire molti casi d'uso e prodotti di dati. Può agire da singolo repository aziendale unificato per:

- domini di dati
- casi d'uso di streaming in tempo reale
- data mart
- data warehouse eterogenei
- ambienti di prova per negozio di funzionalità e data science
- ambienti di prova per analisi di reparto

Data la varietà dei casi d'uso, a ogni progetto su un lakehouse possono essere applicati diversi principi di organizzazione dei dati e tecniche di modellazione. Da un punto di vista tecnico, l'architettura lakehouse può supportare molti stili di modellazione di dati diversi. In questa sezione, illustreremo l'implementazione dei principi per l'organizzazione dei dati sul lakehouse e vedremo come le diverse tecniche di modellazione si applichino ai livelli Bronze, Silver e Gold.

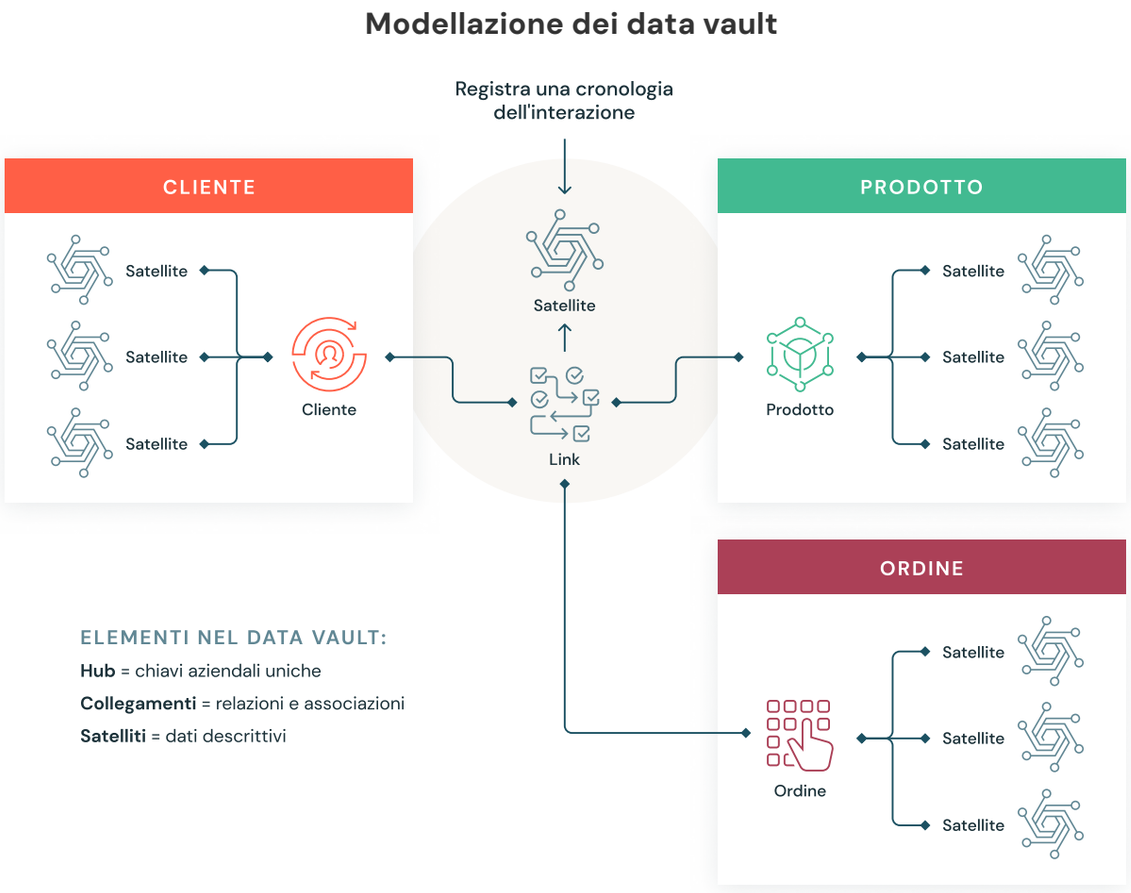
Che cos'è un data vault?

Il data vault è uno schema di modellazione dei dati utilizzato per costruire data warehouse per analisi su scala aziendale, più recente rispetto alle metodologie Kimball e Inmon.

I data vault organizzano i dati in tre tipi diversi: hub, collegamenti e satelliti. Gli hub rappresentano concetti aziendali chiave, i collegamenti rappresentano le relazioni tra gli hub e i satelliti contengono attributi degli hub o dei collegamenti.

Il data vault dà la priorità allo sviluppo di un data warehouse agile in cui scalabilità, integrazione dei dati/ETL e velocità di sviluppo sono aspetti essenziali. La maggior parte dei clienti ha una zona di landing, una zona vault e un data mart che corrispondono ai paradigmi organizzativi Databricks dei livelli Bronze, Silver e Gold. Solitamente, lo stile di modellazione composto da hub, collegamenti e satelliti del data vault si inserisce bene nel livello Silver dell'architettura lakehouse.

Per maggiori informazioni sulla modellazione dei data vault, vedi Data Vault Alliance.

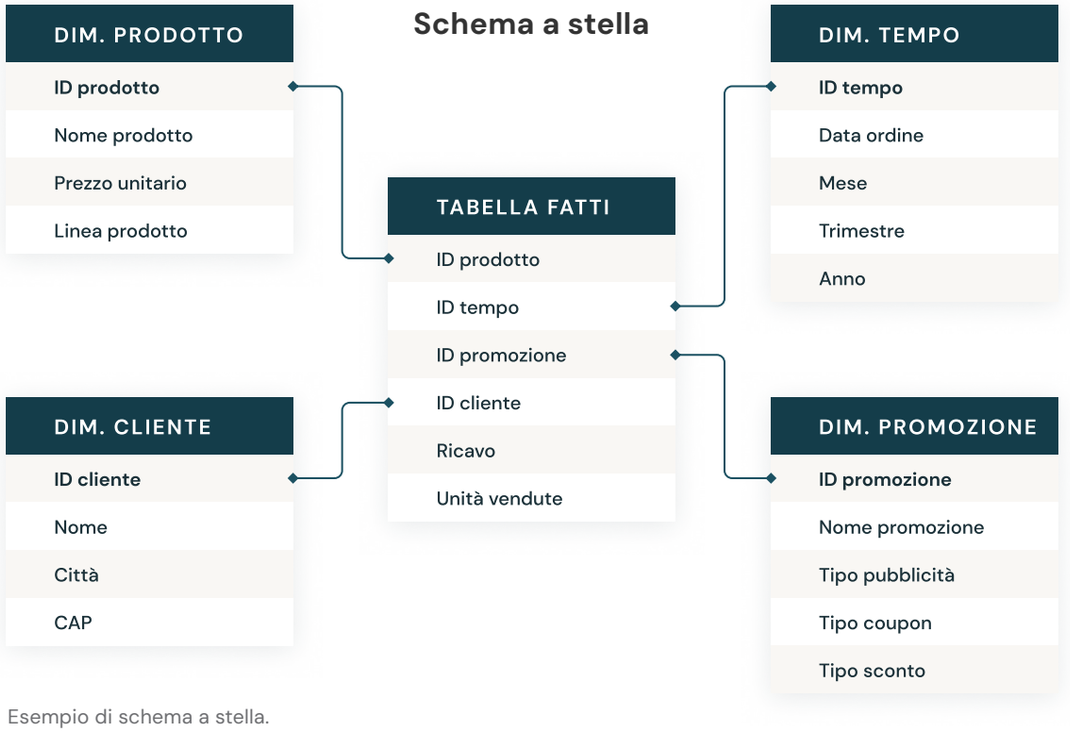


Il diagramma mostra come funziona la modellazione dei data vault, con hub, collegamenti e satelliti connessi tra loro.

Che cos'è la modellazione dimensionale?

La modellazione dimensionale è un approccio bottom-up alla progettazione di data warehouse ottimizzati per l'analisi. I modelli dimensionali sono utilizzati per denormalizzare i dati aziendali in dimensioni (come tempo e prodotto) e fatti (come importi e quantità di transazioni). Le diverse aree tematiche sono connesse tramite dimensioni conformi a diverse tabelle di fatti.

La forma più comune di modellazione dimensionale è lo schema a stella. Uno schema a stella è un modello multidimensionale utilizzato per organizzare i dati in modo che siano facili da comprendere, analizzare e usare per la generazione di report. Gli schemi a stella in stile Kimball, o modelli dimensionali, sono praticamente lo standard aureo per il livello di presentazione nei data warehouse e data mart e anche nei livelli semantici e di reporting. La conformazione dello schema a stella è ottimizzata per l'interrogazione di set di dati di grandi dimensioni.



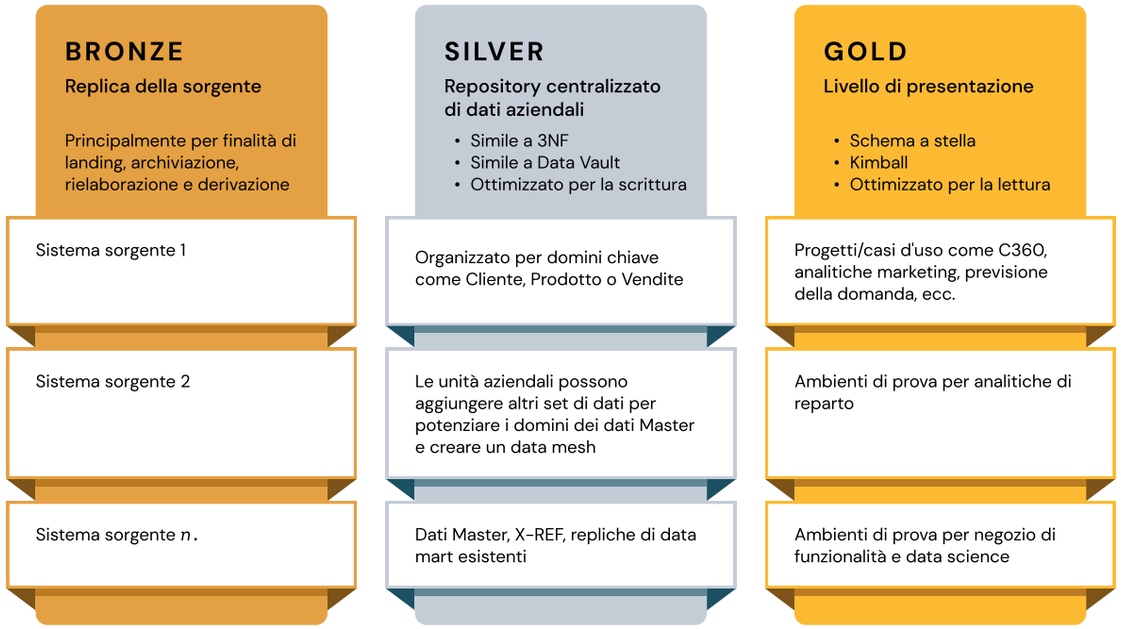
Esempio di schema a stella.

Entrambi gli stili di modellazione dei dati – i data vault normalizzati (ottimizzati per la scrittura) e i modelli dimensionali denormalizzati (ottimizzati per la lettura) – trovano posto nella Databricks Data Intelligence Platform. Gli hub e i satelliti del data vault nel livello Silver sono usati per caricare le dimensioni nello schema a stella e le tabelle dei collegamenti del data vault diventano la base per caricare le tabelle dei fatti nel modello dimensionale. Per maggiori sulla modellazione dimensionale, vedi Kimball Group.

Principi di organizzazione dei dati in ciascun livello del lakehouse

Un lakehouse moderno è una piattaforma dati di livello enterprise completa. È altamente scalabile e performante per casi d'uso di ogni tipo, inclusi ETL, BI, data science e streaming che potrebbero richiedere approcci diversi alla modellazione dei dati. Vediamo com'è organizzato un tipico lakehouse:

Architettura del data lakehouse



Il diagramma mostra le caratteristiche dei livelli Bronze, Silver e Gold dell'architettura del data lakehouse.

Livello Bronze — la landing zone (zona di "atterraggio")

Il livello Bronze è quello in cui "atterrano" tutti i dati provenienti da sorgenti esterne. Le strutture delle tabelle in questo livello coincidono esattamente con le strutture delle tabelle nel sistema sorgente, ad eccezione delle colonne opzionali di metadati che contengono data/ora di caricamento, ID del processo ecc. Lo scopo principale di questo livello è l'acquisizione dei dati cambiati (CDC) per avere un archivio storico della sorgente (cold storage), della provenienza dei dati e della verificabilità, rielaborando i dati se necessario, senza rileggerli dalla sorgente.

In genere, nel livello Bronze è consigliabile mantenere i dati in formato Delta, in modo che le successive letture dal livello Bronze per i processi ETL siano performanti e si possano aggiornare i dati nel livello Bronze con le modifiche CDC. A volte, quando i dati arrivano nei formati JSON o XML, i clienti li acquisiscono nel formato sorgente originale per poi convertirli in formato Delta per lo staging. Il livello logico Bronze viene così strutturato in zone fisiche di landing e staging.

Archiviare i dati grezzi nel formato sorgente originale in una zona di "atterraggio" aiuta anche a mantenerne la coerenza quando l'acquisizione avviene tramite strumenti che non supportano Delta come destinazione nativa (sink) o quando i sistemi sorgente scaricano i dati direttamente negli archivi di oggetti. Questo modello inoltre si allinea bene con il framework di ingestione automatizzata in cui i sistemi sorgente depositano i dati nella zona di landing per file grezzi prima che Databricks AutoLoader li converta trasferendoli al livello di staging in formato Delta.

Livello Silver — il repository aziendale centrale

Nel livello Silver del lakehouse i dati provenienti dal livello Bronze vengono abbinati, uniti, uniformati e puliti ("quanto basta"), in modo da fornire una "vista aziendale" di tutte le entità, i concetti e le transazioni chiave. Il livello Silver si può paragonare a un archivio dati operazionale (ODS), a un repository centrale o ai domini di dati di un data mesh (ad es. clienti master, prodotti, transazioni non duplicate e tabelle di riferimento incrociato). La vista aziendale unifica i dati provenienti da diverse sorgenti e consente attività di analisi self-service per reportistica mirata, analisi avanzata e ML. Serve anche da fonte per analisti di reparto, data engineer e data scientist, per creare progetti e analisi che rispondano a problemi operativi tramite progetti di gestione dei dati nel singolo reparto in tutta l'impresa nel livello Gold.

Nel paradigma di data engineering del lakehouse si segue in genere la metodologia ELT (Extract, Load, Transform) invece della tradizionale ETL (Extract, Transform, Load). Questo significa che vengono applicate solo le trasformazioni e le regole di pulizia dei dati minime indispensabili, o "quanto basta", mentre si carica il livello Silver. Tutte le regole di "livello aziendale" vengono applicate nel livello Silver, a differenza delle regole di trasformazione specifiche per progetto che vengono invece applicate nel livello Gold. Velocità e agilità nell'acquisizione e nella consegna dei dati al data lake sono prioritarie.

Dal punto di vista della modellazione dei dati, il livello Silver ha più modelli di dati del tipo 3rd-Normal Form. In questo livello si possono utilizzare modelli tipo data vault con prestazioni avanzate di scrittura. Se si usa una metodologia data vault, sia il data vault grezzo sia quello aziendale rientreranno nel livello logico Silver e le viste di presentazione Point-In-Time (PIT) o le viste materializzate verranno presentate nel livello Gold.

Livello Gold — Il livello di presentazione

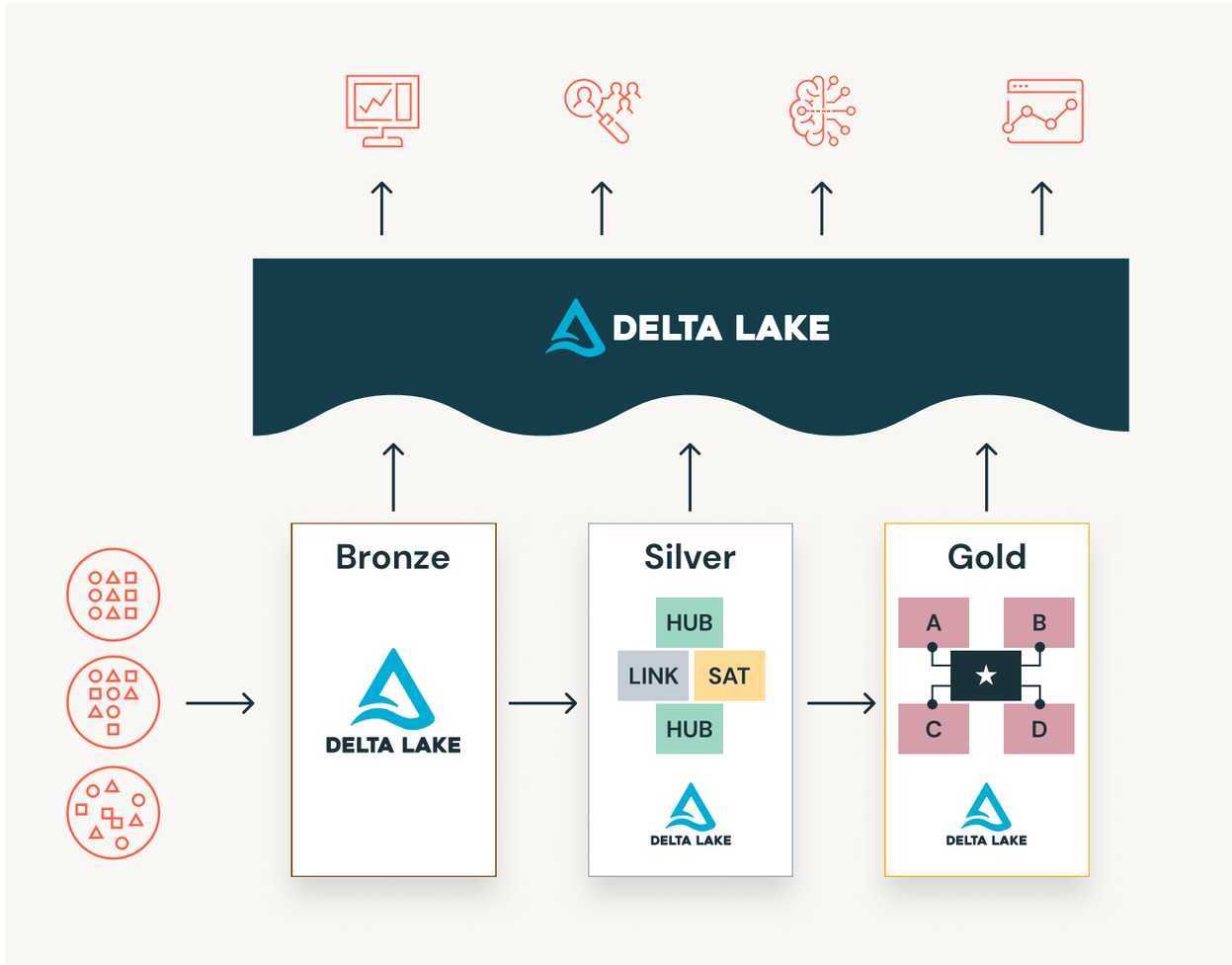
Nel livello Gold si possono costruire più data mart/data warehouse secondo la metodologia di modellazione dimensionale Kimball. Come detto in precedenza, il livello Gold è destinato alla reportistica e usa più modelli di dati denormalizzati e ottimizzati in lettura con meno join rispetto al livello Silver. Le tabelle del livello Gold talvolta possono essere completamente denormalizzate, ad esempio se i data scientist vogliono utilizzarle per alimentare i propri algoritmi per l'ingegneria delle funzionalità.

Le regole ETL e di qualità dei dati specifiche di un progetto vengono applicate nella trasformazione dei dati dal livello Silver al Gold. In questo livello rientrano le presentazioni finali di data warehouse, data mart o prodotti di dati come analisi dei clienti, della qualità dei prodotti e dell'inventario, segmentazione dei clienti, raccomandazioni di prodotti, analisi di marketing/vendite ecc. Nel livello Gold del lakehouse si trovano modelli di dati con schema a stella di tipo Kimball o data mart di tipo Inmon. Sempre nel livello Gold troviamo anche i laboratori di data science e gli ambienti di prova per analitiche di reparto self-service.

Il paradigma di organizzazione dei dati nel lakehouse

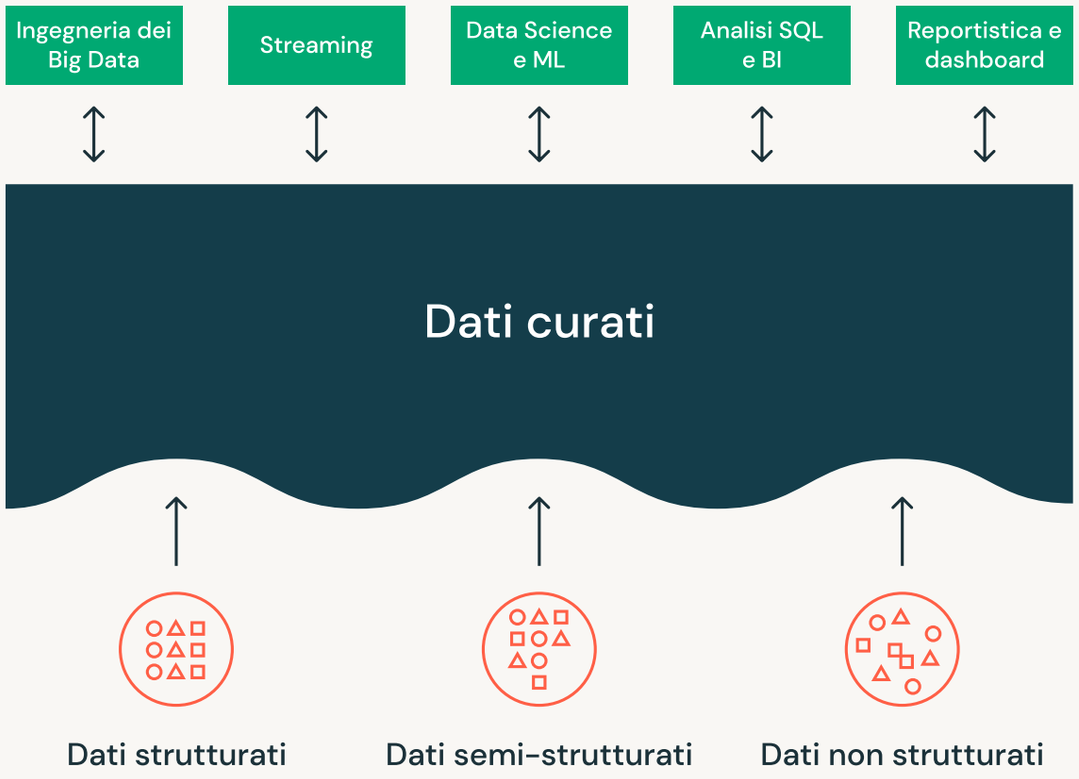
In sintesi, i dati vengono curati man mano che avanzano di livello in un'architettura data lakehouse.

- Il livello Bronze usa i modelli di dati dei sistemi sorgente. Se i dati sono stati acquisiti in formati grezzi, in questo livello vengono convertiti in formato Delta Lake.
- Il livello Silver riunisce per la prima volta i dati provenienti da diverse sorgenti e li uniforma per creare una vista aziendale, in genere usando modelli di dati più normalizzati e ottimizzati per la scrittura, di tipo 3rd-Normal Form o Data Vault.
- Il livello Gold è il livello di presentazione e rispetto al Silver, contiene modelli di dati più denormalizzati o appiattiti, in genere in modelli dimensionali in stile Kimball o schemi a stella. Il livello Gold ospita inoltre ambienti di prova per data science e analitiche di reparto che consentono l'esecuzione di analisi self-service e progetti di data science in tutta l'azienda. Fornire questi ambienti di prova con un cluster di calcolo separati evita che i team commerciali creino copie di dati all'esterno del lakehouse.



Questo approccio all'organizzazione dei dati abbatte i silos, facilita la collaborazione tra i team e consente di eseguire processi ETL, streaming, BI e AI su un'unica piattaforma con una solida governance. I data team centrali dovrebbero farsi promotori dell'innovazione in azienda, accelerando l'onboarding di nuovi utenti self-service e lo sviluppo di molti progetti di dati in parallelo, evitando che il processo di modellazione dei dati diventi un collo di bottiglia. Databricks Unity Catalog offre funzionalità di ricerca, scoperta, governance e derivazione dei dati sul lakehouse per assicurare una gestione costante dei dati.

Costruisci data vault e data warehouse con schema a stella con Databricks SQL. →



I dati vengono curati man mano che avanzano da un livello all'altro di un'architettura data lakehouse.

MAGGIORI INFORMAZIONI

- → Five Simple Steps for Implementing a Star Schema in Databricks With Delta Lake
- → Best practices to implement a Data Vault model in Databricks Lakehouse
- → Dimensional Modeling Best Practice & Implementation on Modern Lakehouse
- → Identity Columns to Generate Surrogate Keys Are Now Available in a Lakehouse Near You!
- → Load an EDW Dimensional Model in Real Time With Databricks Lakehouse

databricks

CAPITOLO 3.2

Le best practice di modellazione dimensionale e la loro implementazione su un'architettura lakehouse moderna

di Leo Mao, Abhishek Dey, Justin Breese e Soham Bhatt

Molti dei nostri clienti stanno migrando i loro data warehouse legacy su Databricks Lakehouse perché ciò consente loro non solo di modernizzare i sistemi di gestione dei dati, ma anche di accedere istantaneamente a una robusta piattaforma per streaming e analisi avanzate. Il lakehouse risponde a tutte le esigenze di streaming, ETL, BI e AI dell'azienda e permette a team commerciali e data team di collaborare facilmente su un'unica piattaforma.

Aiutando i nostri clienti sul campo, abbiamo notato che molti sono interessati a conoscere le best practice per la modellazione e l'implementazione di modelli fisici di dati in Databricks.

In questa sezione analizzeremo in dettaglio le best practice della modellazione dimensionale sulla Databricks Data Intelligence Platform e forniremo un esempio concreto di implementazione di un modello fisico usando le nostre best practice per creazione di tabelle e DDL.

Ecco gli argomenti che tratteremo in questa sezione:

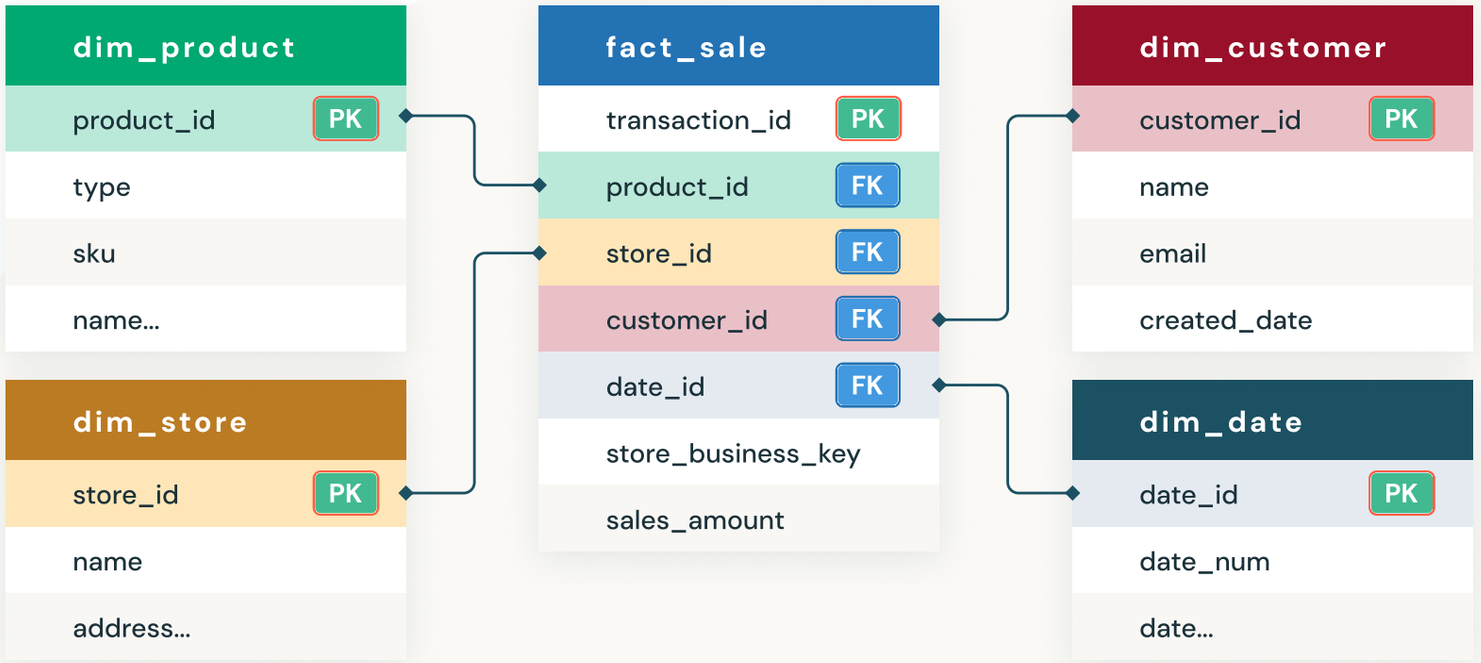
- L'importanza della modellazione dei dati
- Tecniche comune di modellazione dei dati
- Implementazione del DDL nella modellazione del data warehouse
- Best practice e raccomandazioni per la modellazione dei dati nel lakehouse

L'importanza della modellazione dei dati per il data warehouse

I modelli di dati hanno un ruolo di primo piano nella costruzione di un data warehouse. In genere, il processo inizia definendo il Semantic Business Information Model, poi un Logical Data Model e infine un Physical Data Model (PDM). Tutto inizia con una fase di analisi e progettazione dei sistemi in cui prima si creano un modello di Business Information e i flussi di processo, poi vengono acquisiti le entità chiave, gli attributi e le loro interazioni in base ai processi aziendali all'interno dell'organizzazione. Viene quindi creato il Logical Data Model, che illustra i collegamenti tra le entità ed è neutro rispetto alla tecnologia. Infine, viene creato un PDM basato sulla piattaforma tecnologica sottostante per consentire un'esecuzione più efficiente delle operazioni di scrittura e lettura. Come sappiamo, per il data warehousing si prediligono generalmente stili di modellazione ottimizzati per l'analisi, come lo schema a stella e il data vault.

Best practice per la creazione di un modello di dati fisico in Databricks

Sulla base del problema aziendale definito, lo scopo del progetto del modello di dati è quello di rappresentare i dati in modo riutilizzabile, flessibile e scalabile. Questo tipico schema a stella mostra una tabella dei fatti relativa alle vendite che contiene ogni transazione e varie tabelle delle dimensioni come clienti, prodotti, negozi, data ecc. che possono essere usate per sezionare i dati. Le dimensioni possono essere unite alla tabella dei fatti per rispondere a specifiche domande, come ad esempio quali sono stati i prodotti più richiesti in un certo mese o quali negozi hanno ottenuto risultati migliori nel trimestre. Vediamo come implementarlo in Databricks.



Modello dimensionale sull'architettura lakehouse.

Si noti che ciascuna tabella delle dimensioni include anche le colonne **__START_AT** e **__END_AT** per supportare SCD (Slowly Changing Dimension) di Tipo 2, non visualizzate qui per mancanza di spazio.

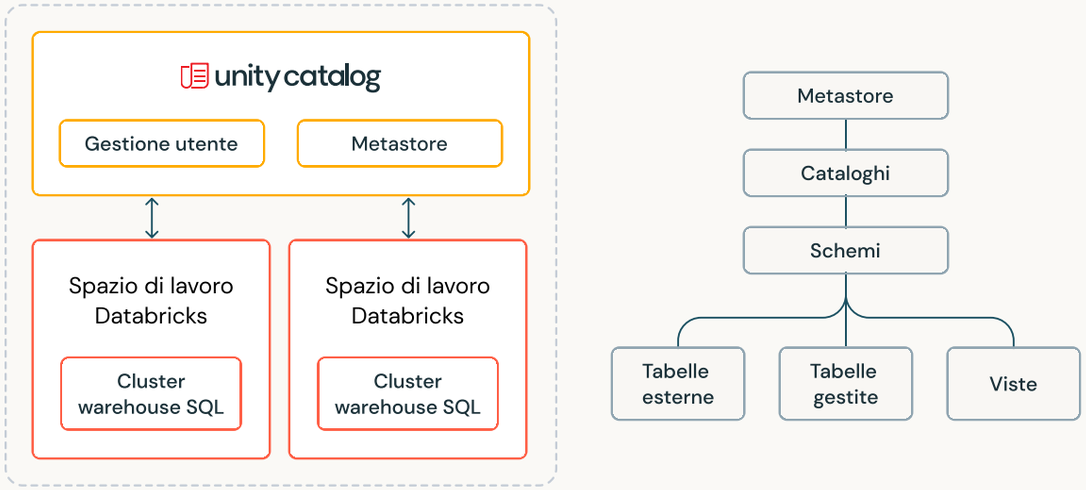
Implementazione del DDL nella modellazione del data warehouse su Databricks

Nelle prossime sezioni, mostreremo come realizzare quanto segue usando i nostri esempi.

- Creazione di catalogo, database e tabella a 3 livelli
- Definizioni di chiave primaria e chiave esterna
- Colonne identità per chiavi surrogate
- Vincoli di colonna per la qualità dei dati
- Indicizzare, ottimizzare e analizzare
- Tecniche avanzate

1. Unity Catalog - namespace a 3 livelli

Unity Catalog è un livello di Databricks Governance che permette ad amministratori e data steward di gestire gli utenti e il loro accesso ai dati in tutti gli spazi di lavoro in modo centralizzato da un account Databricks usando un unico Metastore. Gli utenti di diversi spazi di lavoro possono condividere l'accesso agli stessi dati, in base ai privilegi accordati centralmente in Unity Catalog. Unity Catalog ha un namespace a 3 livelli (catalog.schema(database). table) per l'organizzazione dei dati.
Per maggiori informazioni su Unity Catalog, vedi qui.



Ecco come impostare il catalogo e lo schema prima di creare tabelle nel database. Per il nostro esempio, creiamo un catalogo **US_Stores** e uno schema (database) **Sales_DW** come si vede qui sotto e li usiamo per la parte successiva della sezione.

```
1  CREATE CATALOG IF NOT EXISTS US_Stores;
2  USE CATALOG US_Stores;
3  CREATE SCHEMA IF NOT EXISTS Sales_DW;
4  USE SCHEMA Sales_DW;
```

Impostare il catalogo e il database

Ecco un esempio di interrogazione della tabella **fact_sales** con un namespace a 3 livelli.

```
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.

Esempio di interrogazione della tabella con catalog.database.tablename

2. Definizioni di chiave primaria e chiave esterna

Le definizioni di chiave primaria (Primary Key, PK) e chiave esterna (Foreing Key, FK) sono molto importanti quando si crea un modello di dati. La possibilità di supportarle rende facilissimo definire il modello di dati in Databricks. Aiuta inoltre gli analisti a individuare rapidamente le relazioni di join in Databricks SQL Warehouse, permettendo di scrivere query efficaci. Come nella maggior parte dei casi di elaborazione a elevata parallelizzazione (MPP), EDW e data warehouse in cloud, i vincoli di PK/FK sono puramente informativi. Databricks non supporta l'applicazione della relazione PK/FK, ma dà la possibilità di definirla per semplificare la progettazione del modello di dati semantico.

Questo esempio mostra la creazione della tabella **dim_store** con **store_id** sia come colonna identità sia come chiave primaria.

```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
);
```

Implementazione del DDL per creare la dimensione Store con definizioni di chiave primaria.

Possiamo vedere come, una volta creata la tabella, la chiave primaria (store_id) venga usata come vincolo nella definizione della tabella qui sotto.

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.

La chiave primaria store_id appare come vincolo di tabella.

Questo esempio mostra la creazione della tabella **fact_sales** con **transaction_id** come chiave primaria e di chiavi esterne che fanno riferimento alle tabelle delle dimensioni.

```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
);
```

Implementazione del DDL per creare la tabella di fatti Sales con definizioni di chiavi esterne.

Possiamo vedere come, una volta creata la tabella, la chiave primaria **(transaction_id)** e le chiavi esterne vengano usate come vincoli nella definizione della tabella qui sotto.

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.

Definizione della tabella di fatti con chiave primaria e chiavi esterne che fanno riferimento alle dimensioni.

3. Colonne identità per chiavi surrogate

Una colonna identità è una colonna di un database che genera automaticamente un ID unico per ogni nuova riga di dati. Queste colonne vengono comunemente usate nei data warehouse per creare chiavi surrogate. Le chiavi surrogate sono chiavi generate dal sistema e prive di qualsiasi significato reale che ci evitano di dover fare affidamento su varie chiavi primarie naturali e concatenazioni su più campi per identificare l'unicità della riga. Generalmente, queste chiavi surrogate vengono usate come chiavi primarie ed esterne nei data warehouse. Per informazioni più dettagliate sulle colonne identità vedi questo articolo. L'esempio qui sotto mostra la creazione della colonna identità customer_id, con valori assegnati automaticamente a partire da 1 e con incrementi di 1.

```
-- Customer dimension
CREATE OR REPLACE TABLE dim_customer(
  customer_id BIGINT GENERATED ALWAYS AS IDENTITY (START WITH 1 INCREMENT BY 1)
PRIMARY KEY,
  name STRING,
  email STRING,
  address STRING,
  created_date TIMESTAMP,
  updated_date TIMESTAMP,
  start_at TIMESTAMP,
  end_at TIMESTAMP
);
```

Implementazione del DDL per creare la dimensione Customer con colonna identità.

4. Vincoli di colonna per la qualità dei dati

Oltre ai vincoli informativi di chiave primaria ed esterna, Databricks supporta anche vincoli di controllo della qualità dei dati a livello di colonna che vengono applicati per assicurare la qualità e l'integrità dei dati aggiunti in una tabella. La verifica dei vincoli avviene automaticamente. Rientrano in questa tipologia i vincoli NOT NULL e i vincoli sui valori nelle colonne. A differenza di altri data warehouse in cloud, Databricks fornisce vincoli di controllo sui valori nelle colonne, utilissimi per assicurare la qualità dei dati di una determinata colonna. Come si vede qui sotto, il vincolo di controllo **valid_sales_amount** verificherà che tutte le righe esistenti soddisfino il vincolo (cioè **sales amount** > 0) prima di aggiungere un valore alla tabella. Per maggiori informazioni, vedi qui.

Questi esempi mostrano l'aggiunta di vincoli per dim_store e fact_sales, rispettivamente, per assicurare che store_id e sales_amount abbiano valori validi.

```
-- 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);
```

Aggiunta di vincoli su colonne alle tabelle esistenti per assicurare la qualità dei dati.

5. Indicizzare, ottimizzare e analizzare

Mentre i database tradizionali hanno indici b-tree e bitmap, Databricks ha un'indicizzazione clusterizzata Z-order multidimensionale molto più avanzata e supporta inoltre gli indici di filtro Bloom. Prima di tutto, Delta usa il formato Parquet, un formato di file colonnare compresso già di per sé molto efficiente nella "potatura" delle colonne; aggiungere un indice Z-order dà la possibilità di vagliare dati nell'ordine dei petabyte nel giro di qualche secondo. Tanto gli indici Z-order che quelli di filtro Bloom riducono drasticamente la quantità di dati da scansionare per rispondere a query altamente selettive su ampie tabelle Delta, il che si traduce in un miglioramento del runtime e una riduzione notevole dei costi. Si deve utilizzare Z-order sulle chiavi primarie ed esterne che vengono usate per i join più frequenti. E gli indici di filtro Bloom aggiuntivi secondo necessità.

```
-- 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);
```

Ottimizzazione di fact_sales su customer_id e product_id per prestazioni migliori.

```
-- 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)
```

Creazione di un indice di filtro Bloom per consentire lo skipping dei dati di una determinata colonna.

Come in qualsiasi altro data warehouse, è possibile usare ANALYZE TABLE per aggiornare le statistiche e assicurarsi così che l'ottimizzatore delle query disponga delle migliori statistiche per creare il miglior piano di query.

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

Raccolta di statistiche di tutte le colonne per un miglior piano di esecuzione delle query.

6. Tecniche avanzate

Sebbene Databricks supporti tecniche avanzate come il partizionamento delle tabelle, suggeriamo di limitarne l'uso ai soli casi in cui si hanno troppi terabyte di dati compressi. Questo perché, nella maggior parte dei casi, i nostri indici OPTIMIZE e Z-ORDER consentiranno il miglior pruning di file e dati tanto da rendere addirittura sconsigliato il ricorso al partizionamento di una tabella. Tuttavia, è buona norma accertarsi che i DDL della tabella siano configurati per ottimizzazione e compattazione automatiche. In questo modo, i dati che vengono scritti di frequente in piccoli file vengono compattati nei formati colonnari compressi di Delta.

Vuoi sfruttare uno strumento di modellazione dei dati visuale? Il nostro partner erwin Data Modeler di Quest può eseguire rapidamente e facilmente il reverse engineering, la creazione e l'implementazione in Databricks di schemi a stella, data vault e ogni altro modello di dati.

Esempio di notebook Databricks

La Databricks Platform semplifica la progettazione e l'implementazione di vari modelli di dati. Per vedere tutti gli esempi discussi finora riuniti in un flusso di lavoro completo, vedi questo esempio.

Consulta anche l'articolo sul nostro blog
Five Simple Steps for Implementing a Star Schema in Databricks With Delta Lake →

CAPITOLO 3.3

Caricare un modello di dati data warehouse in tempo reale con la Databricks Data Intelligence Platform

Implementazione di un modello dimensionale sul lakehouse moderno usando Delta Live Tables

di Leo Mao, Soham Bhatt e Abhishek Dey

La modellazione dimensionale è una delle tecniche più popolari per costruire un data warehouse moderno. Permette agli utenti di sviluppare rapidamente tabelle di fatti e dimensioni in base alle specifiche esigenze dell'azienda. Aiutando i nostri clienti sul campo, abbiamo riscontrato che molti si rivolgono a Databricks per conoscere le best pratice e l'architettura di riferimento per l'implementazione.

In questa sezione, analizzeremo nel dettaglio le best practice della modellazione dimensionale sull'architettura lakehouse e forniremo un esempio concreto di come caricare un modello dimensionale EDW in tempo reale usando Delta Live Tables.

Ecco gli argomenti che tratteremo in questa sezione:

- Definire un problema aziendale
- Progettare un modello dimensionale
- Best practice e raccomandazioni per la modellazione dimensionale
- Implementare un modello dimensionale su un'architettura lakehouse
- Conclusioni

Definire un problema aziendale

La modellazione dimensionale è business-oriented e inizia sempre con un problema aziendale. Prima di costruire un modello dimensionale, dobbiamo capire il problema aziendale da risolvere: ciò determina il modo in cui le risorse di dati verranno presentate e consumate dagli utenti finali. Dobbiamo progettare il modello di dati per supportare query più accessibili e veloci.

La matrice aziendale è un concetto fondamentale nella modellazione dimensionale. Nell'esempio di matrice riportato di seguito, le colonne sono dimensioni condivise mentre le righe rappresentano processi aziendali. Il problema aziendale definito determina la granularità dei dati dei fatti e le dimensioni richieste. L'aspetto importante è che potremmo costruire in modo incrementale ulteriori risorse di dati sulla base della matrice aziendale e delle sue dimensioni condivise o conformi.

Dimensioni condivise →

Processi aziendali ↓	Data	Cliente	Prodotto	Fornitore	Promozione	Rivenditore	Area di vendita	Dipendente	Account	Organizzazione
Vendite online	✓	✓	✓	✓	✓		✓			
Vendite da rivenditori	✓		✓		✓	✓	✓	✓		
Registro generale	✓								✓	✓
Piano di vendite	✓		✓				✓			
Inventario	✓		✓	✓					✓	
Sondaggi clienti	✓	✓								
Chiamate servizio clienti	✓	✓	✓					✓		

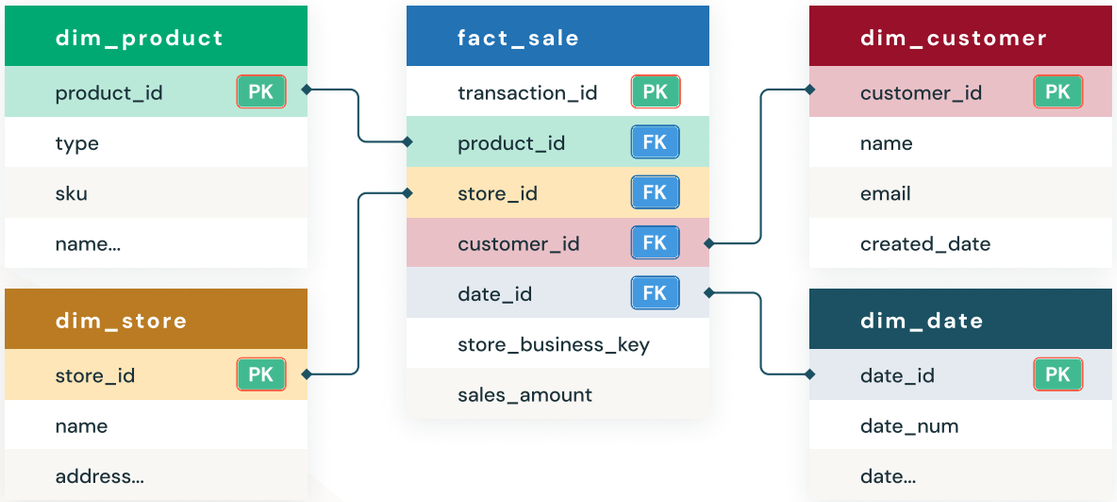
Una matrice aziendale con dimensioni condivise e processi aziendali.

Ipotizziamo che uno sponsor abbia richiesto un rapporto con informazioni dettagliate su:

- Quali sono i prodotti più venduti, in modo da valutare la richiesta dei vari articoli.
- Quali sono i negozi che hanno ottenuto prestazioni migliori, in modo da individuare le buone prassi di gestione di un punto vendita.

Progettare un modello dimensionale

Sulla base del problema aziendale definito, lo scopo del progetto è rappresentare i dati in modo efficiente in termini di riusabilità, flessibilità e scalabilità. Il seguente modello di dati di alto livello può rispondere alle domande sopracitate.



Modello dimensionale sull'architettura lakehouse.

Si noti che ciascuna tabella delle dimensioni include anche le colonne **__START_AT** e **__END_AT** per supportare SCD (Slowly Changing Dimension) di Tipo 2, non visualizzate qui per mancanza di spazio.

Il progetto dovrebbe essere semplice da comprendere e funzionare in modo efficiente con diverse tipologie di query. In base al modello, abbiamo progettato la tabella dei fatti Sales per rispondere alle nostre domande. Come si può vedere, a parte le chiavi esterne (FK) sulle dimensioni, contiene solo i parametri numerici usati per misurare le vendite, ovvero sales_amount.

Abbiamo anche progettato tabelle delle dimensioni come prodotto, negozio, cliente, data che forniscono informazioni contestuali sui dati dei fatti. Le tabelle delle dimensioni vengono generalmente unite alle tabelle dei fatti per rispondere a specifiche domande, ad esempio quali sono stati i prodotti più richiesti in un dato mese o quali negozi hanno avuto le prestazioni migliori nel trimestre.

Best practice e raccomandazioni per la modellazione dimensionale

Con la Databricks Data Intelligence Platform è possibile progettare e implementare facilmente modelli dimensionali e costruire tabelle di fatti e dimensioni per l'area tematica che ci interessa.

Ecco alcune best practice per l'implementazione di un modello dimensionale:

- Denormalizzare le tabelle delle dimensioni. Diversamente da tipi di modelli quali 3rd-Normal Form o lo schema a fiocco di neve, le tabelle delle dimensioni sono in genere altamente denormalizzate, con relazioni appiattite molti-a-uno all'interno di una singola tabella delle dimensioni.
- Usare tabelle delle dimensioni uniformate quando più attributi in diverse tabelle delle dimensioni hanno gli stessi nomi di colonna e contenuti di dominio. Il vantaggio è che i dati di diverse tabelle dei fatti possono essere combinati in un singolo report usando attributi di dimensioni uniformati con ciascuna tabella dei fatti.
- Una tendenza comune nelle tabelle delle dimensioni è quella di tenere traccia delle modifiche alle dimensioni nel tempo per supportare la generazione di rapporti "as-is" o "as-was". È possibile applicare facilmente le seguenti tecniche per gestire le dimensioni in base alle diverse esigenze.
  - La tecnica Tipo 1 sovrascrive il valore iniziale dell'attributo della dimensione.
  - La tecnica Tipo 2, la più comune tecnica SCD, viene usata per tenere traccia in modo accurato delle modifiche nel tempo.

  Tutto questo può essere facilmente ottenuto in versione pronta all'uso con l'implementazione di Delta Live Tables.
  - È possibile eseguire facilmente SCD di tipo 1 o 2 con Delta Live Tables usando APPLY CHANGES INTO.
- I vincoli **chiave primaria + chiave esterna** consentono agli utenti finali di comprendere le relazioni tra le tabelle.
- **L'uso delle colonne IDENTITÀ** genera automaticamente valori interi unici quando vengono aggiunte nuove righe. Le colonne identità sono una forma di chiavi surrogate. Per maggiori dettagli, vedi questo articolo.
- **Vincoli di CONTROLLO applicati** per non doversi mai preoccupare di problemi imprevisti relativi a qualità o correttezza dei dati.

Implementare un modello dimensionale su un'architettura lakehouse

Vediamo ora un esempio di implementazione di un modello dimensionale basata su Delta Live Tables.

Il codice mostrato qui sotto illustra come creare una tabella delle dimensioni (dim_store) usando SCD di Tipo 2, dove i dati modificati vengono acquisiti dal sistema sorgente.

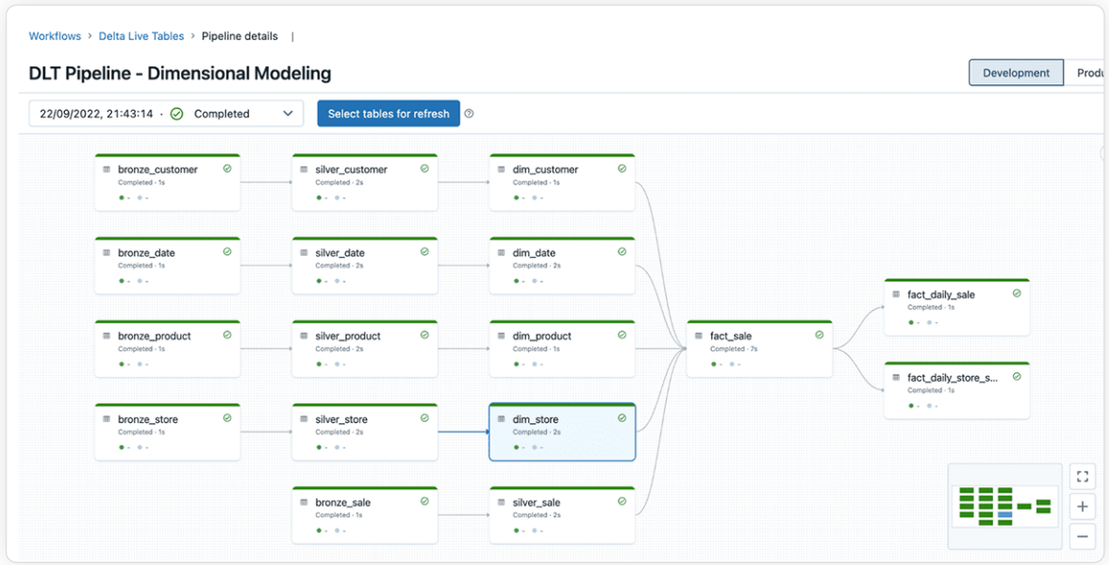
```
-- create the gold table
CREATE INCREMENTAL LIVE TABLE dim_store
TBLPROPERTIES (“quality” = “gold”)
COMMENT “Slowly Changing Dimension Type 2 for store dimension in the gold layer”;

-- 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;
```

Il codice di esempio mostrato qui sotto illustra come creare una tabella dei fatti (fact_sale) e accertarsi, con il vincolo valid_product_id, che tutti i record dei fatti caricati siano associati a un prodotto valido.

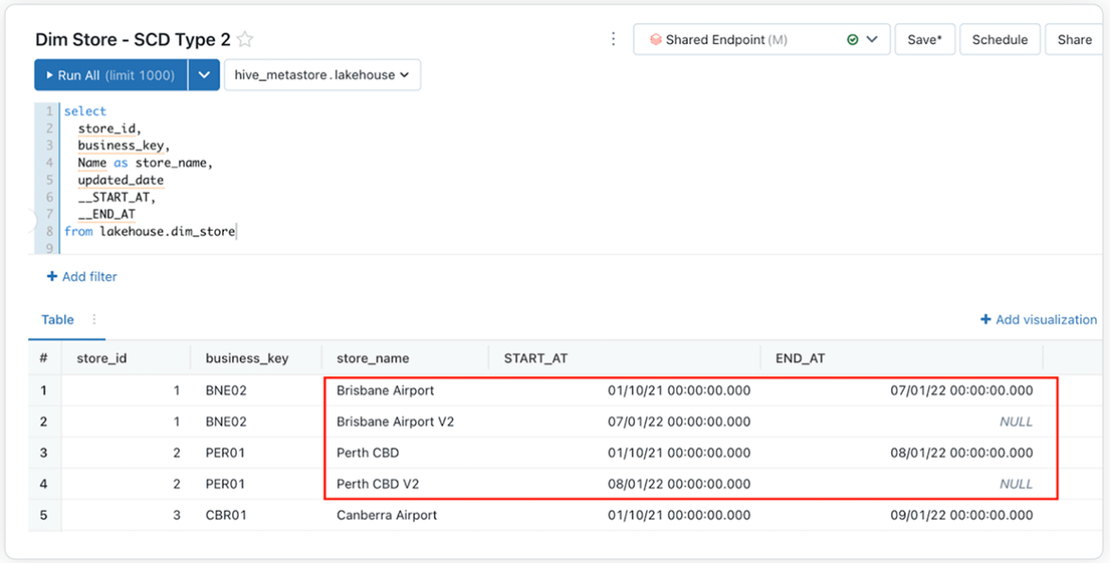
```
-- create the fact table for sales in gold layer
CREATE STREAMING LIVE TABLE fact_sale (
  CONSTRAINT valid_store_business_key EXPECT (store_business_key IS NOT NULL) ON
VIOLATION DROP ROW,
  CONSTRAINT valid_product_id EXPECT (product_id IS NOT NULL) ON VIOLATION DROP
ROW
)
TBLPROPERTIES (“quality” = “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
```

Puoi trovare un esempio di pipeline Delta Live Tables qui. Consulta la guida rapida a Delta Live Tables per informazioni su come creare una pipeline Delta Live Tables. Come si vede qui sotto, DLT offre una visibilità completa sulla pipeline ETL e sulle dipendenze tra i diversi oggetti nei livelli Bronze, Silver e Gold seguendo l'architettura medallion del lakehouse.



Pipeline DLT completa.

Ecco un esempio di come viene aggiornata la tabella delle dimensioni dim_store in base alle modifiche in ingresso. Nell'esempio qui sotto, il punto vendita Brisbane Airport è stato aggiornato a Brisbane Airport V2 e, con il supporto pronto all'uso per SCD di Tipo 2, il 7 gennaio 2022 il record originale è terminato ed è stato creato un nuovo record che inizia lo stesso giorno e ha una data di fine aperta (NULL), a significare che è il record più aggiornato per l'aeroporto di 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 di Tipo 2 per la dimensione Store

Per maggiori dettagli sull'implementazione e per l'esempio di notebook completo, vedi qui.

Conclusioni

In questa sezione abbiamo appreso in dettaglio alcuni concetti relativi alla modellazione dimensionale, quali sono le best practice e come implementarle usando Delta Live Tables.

Per maggiori informazioni sulla modellazione dimensionale, vedi Kimball Technology.

CAPITOLO 3.4

Quali novità introduce Databricks SQL?

Ottimizzazioni basate sull'AI, federazione di lakehouse e altro per BI di livello aziendale

di Alex Lichen, Miranda Luna, Can Efeoglu e Cyrielle Simeone

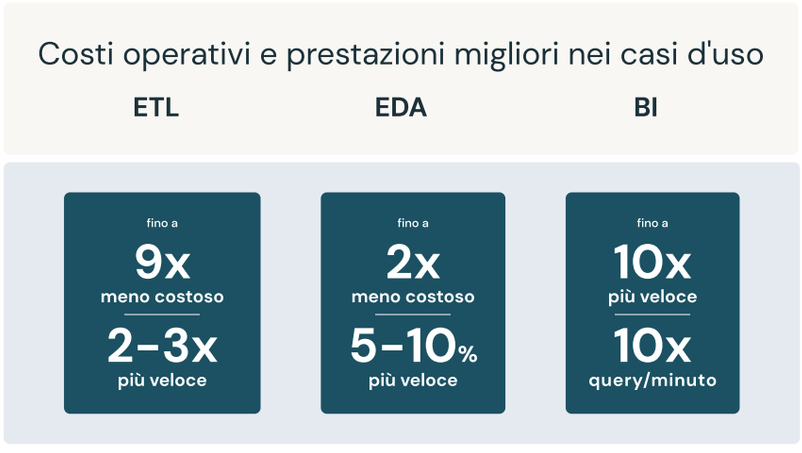
AI Data + AI Summit di quest'anno, Databricks SQL ha ulteriormente alzato l'asticella del data warehousing, utilizzando l'AI a tutti i livelli di prodotto per estendere la sua leadership nelle prestazioni e nell'efficienza, semplificando l'esperienza e offrendo nuove opportunità ai clienti. Parallelamente, continuiamo a migliorare le nostre funzionalità di data warehousing base per aiutarti a unificare i tuoi stack di dati in un'architettura lakehouse sulla Databricks Data Intelligence Platform.

In questa sezione siamo entusiasti di condividere con te le novità presenti e future che ti aspettano su Databricks SQL.

- Ottimizzazioni delle prestazioni basate sull'AI, come Predictive I/O, che assicurano performance e risparmi ai vertici del settore senza necessità di messa a punto manuale.
- Nuove esperienze utente con funzioni AI, warehouse SQL in notebook, nuove dashboard e funzioni Python definite dall'utente.
- Robusto supporto a sorgenti di dati esterne con Lakehouse Federation.
- Nuovi modi per accedere ai tuoi dati con l'API per l'esecuzione delle istruzioni SQL.
- Elaborazione dei dati semplice ed efficiente con tabelle streaming, viste materializzate e integrazione dei flussi di lavoro.
- Assistenza intelligente basata su DatabricksIQ, il nostro knowledge engine.
- Strumenti amministrativi migliorati con Databricks SQL System Tables e Live Query Profile.
- Funzionalità aggiuntive per le nostre integrazioni partner con Fivetran, dbt labs e PowerBI.

Il data warehouse ottimizzato dall'AI: pronto per tutti i carichi di lavoro, senza necessità di messa a punto

Siamo convinti che il miglior data warehouse sia un lakehouse; di conseguenza, continuiamo a estendere la nostra leadership nei carichi di lavoro ETL e a sfruttare la potenza dell'AI. Databricks SQL ti offre prestazioni leader di settore per i tuoi carichi di lavoro EDA e BI e un maggiore risparmio sui costi. Il tutto senza bisogno di ottimizzazione manuale.



Dimentica l'indicizzazione manuale. Con Predictive I/O per le operazioni di lettura (in commercio) e di aggiornamento (anteprima pubblica), Databricks SQL analizza i pattern storici di lettura e scrittura per costruire indici e ottimizzare i carichi di lavoro in maniera intelligente. I clienti che l'hanno già provato hanno riscontrato un miglioramento nell'efficienza delle ricerche point-in-time di ben 35 volte, di 2-6 volte per le operazioni di MERGE e di 2-10 volte per le operazioni di DELETE.

Con le ottimizzazioni predittive (anteprima pubblica), Databricks ottimizzerà clustering e dimensioni dei file eseguendo automaticamente i comandi OPTIMIZE, VACUUM, ANALYZE e CLUSTERING. Questa funzionalità ha consentito ad Anker Innovations di aumentare di 2,2 volte le prestazioni delle query e di conseguire un risparmio del 50% sui costi di archiviazione.

> **Le ottimizzazioni predittive di Databricks hanno migliorato in maniera intelligente il nostro archivio Unity Catalog, con un risparmio del 50% sui costi di archivio annuali e l'esecuzione delle query è più che raddoppiata. La funzionalità ha imparato a dare la priorità alle nostre tabelle più grandi e più utilizzate. Il tutto automaticamente, con risparmio di tempo per il team.**
>
> — ANKER INNOVATIONS

Non vuoi più gestire svariati warehouse per carichi di lavoro di dimensioni diverse o regolare i parametri di scalabilità? Intelligent Workload Management è una suite di funzionalità che assicura velocità nell'esecuzione delle query a costi contenuti. Analizzando i pattern in tempo reale, Intelligent Workload Management garantisce che i carichi di lavoro abbiano la quantità ottimale di potenza di calcolo per eseguire le istruzioni SQL in ingresso senza compromettere le query già in esecuzione.

Con ottimizzazioni basate sull'AI, Databricks SQL offre TCO e prestazioni leader di settore per qualsiasi tipo di carico di lavoro, eliminando ogni necessità di regolazione manuale. Per maggiori informazioni sulle anteprime di ottimizzazione disponibili, guarda l'intervento di Reynold Xin al Data+AI Summit: Databricks SQL Serverless Under the Hood: How We Use ML to Get the Best Price/Performance.

Sblocca i dati isolati con Lakehouse Federation

Le organizzazioni si trovano oggi a dover individuare, governare e interrogare sorgenti di dati isolate distribuite su sistemi frammentati. Con Lakehouse Federation, i team di gestione dei dati possono usare Databricks SQL per scoprire, interrogare e gestire i dati presenti su piattaforme esterne, incluse MySQL, PostgreSQL, Amazon Redshift, Snowflake, Azure SQL Database, Azure Synapse, Google's BigQuery (a breve) e altre.

Inoltre, Lakehouse Federation si integra con le funzionalità avanzate di Unity Catalog quando si accede a sorgenti di dati esterne dall'interno di Databricks. Applica sicurezza a livello di riga e colonna per limitare l'accesso a informazioni sensibili. Utilizza la provenienza dei dati per tracciare l'origine dei dati e assicurarne qualità e conformità. Assegna un tag alle risorse del catalogo federato per semplificare l'individuazione dei dati e l'organizzazione e la gestione delle risorse.

Infine, per accelerare trasformazioni complesse o operazioni di cross-join (prodotto cartesiano) su risorse federate, Lakehouse Federation supporta viste materializzate per ridurre la latenza di query.

Per maggiori informazioni, guarda la nostra sessione dedicata Lakehouse Federation: Access and Governance of External Data Sources from Unity Catalog dal Data+AI Summit.

Sviluppare sull'architettura lakehouse con l'API per l'esecuzione delle istruzioni SQL

L'API per l'esecuzione delle istruzioni SQL consente di accedere al warehouse Databricks SQL tramite un'interfaccia API REST per eseguire query e recuperarne i risultati. Con framework HTTP disponibili per quasi tutti i linguaggi di programmazione, puoi connettere facilmente una varietà di applicazioni e piattaforme direttamente a un warehouse Databricks SQL.

L'API per l'esecuzione delle istruzioni SQL di Databricks è disponibile con i livelli Premium e Enterprise di Databricks. Per maggiori informazioni, guarda la nostra sessione, segui il tutorial (AWS | Azure), leggi la documentazione (AWS | Azure) o consulta il nostro repository di esempi di codice.

Snellisci l'elaborazione dei dati con Streaming Tables, Materialized Views e DB SQL in Workflows

Con Streaming Tables, Materialized Views e DB SQL in Workflows, qualsiasi utente SQL può applicare le migliori practice del data engineering all'elaborazione dei dati. È possibile acquisire, trasformare, orchestrare e analizzare dati in modo efficiente con poche righe di SQL.

Le Streaming Tables sono il modo ideale per immettere i dati nelle tabelle del livello Bronze. Con una singola istruzione SQL, si possono acquisire in maniera scalabile dati da varie sorgenti come sistemi di archiviazione in cloud (S3, ADLS, GCS), message bus (EventHub, Kafka, Kinesis) e altro ancora. L'acquisizione avviene per incrementi, il che permette di avere pipeline a bassa latenza e basso costo senza bisogno di gestire un'infrastruttura complessa.

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

Le Materialized Views riducono i costi e la latenza di query pre-elaborando le query lente e i calcoli più frequentemente e vengono aggiornate per incrementi per ridurre la latenza complessiva. In un contesto di data engineering, vengono usate per trasformare i dati. Sono però preziose anche per i team di analisi in un contesto di data warehousing, perché possono essere usate per (1) accelerare l'esecuzione delle query e le dashboard BI degli utenti finali e (2) condividere i dati in modo sicuro. Con appena quattro righe di codice, qualsiasi utente può creare una vista materializzata per elaborare i dati in maniera efficiente.

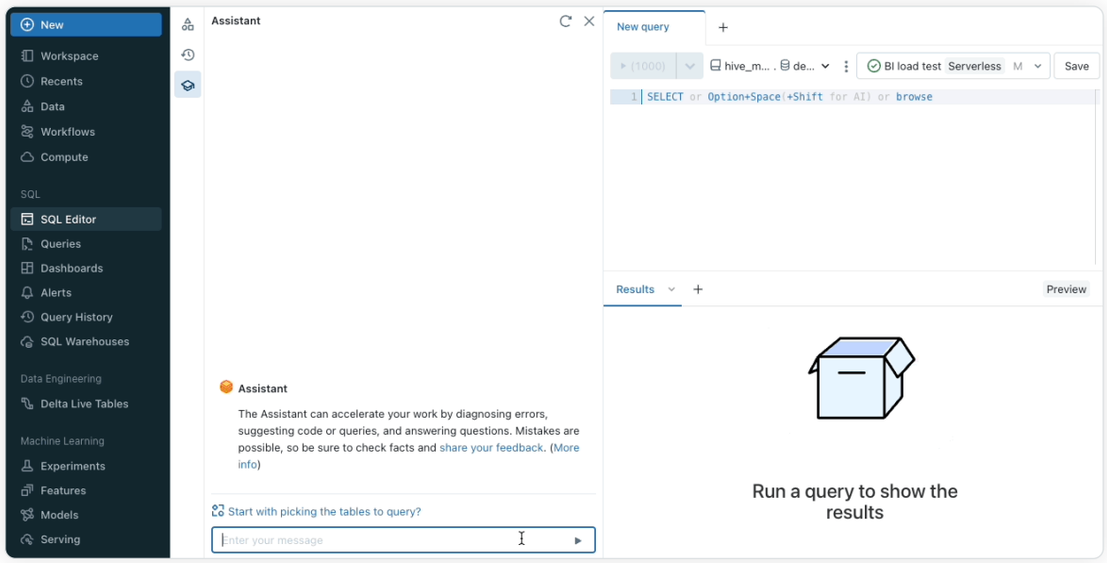
```
CREATE MATERIALIZED VIEW customer_orders
AS
SELECT
  customers.name,
  sum(orders.amount),
  orders.orderdate
FROM orders
  LEFT JOIN customers ON
    orders.custkey = customers.c_custkey
GROUP BY
  name,
  orderdate;
```

Ti serve l'orchestrazione con DB SQL? Workflows consente di pianificare query, dashboard e avvisi SQL. Gestisci con facilità dipendenze complesse tra attività e monitora le esecuzioni di job pregressi usando l'intuitiva interfaccia di Workflows o un'API.

Streaming Tables e Materialized Views sono ora in anteprima pubblica. Per maggiori informazioni, vedi il nostro articolo dedicato sul blog. Per iscriverti all'anteprima pubblica di entrambi, compila questo modulo. Workflows in DB SQL è disponibile a livello generale. Per maggiori informazioni, leggi la documentazione (AWS | Azure).

Databricks Assistant: scrivi in SQL meglio e più velocemente usando il linguaggio naturale

Databricks Assistant è un assistente AI sensibile al contesto integrato in Databricks Notebooks e SQL Editor. Databricks Assistant può prendere una domanda in linguaggio naturale e suggerire una query SQL per rispondere a quella domanda. Quando si trovano di fronte a una query complessa, gli utenti possono chiedere a Databricks Assistant di spiegarla usando il linguaggio naturale. In questo modo, chiunque può capire la logica alla base dei risultati delle query.



Databricks Assistant usa diversi segnali per fornire risultati più accurati e pertinenti. Usa il contesto estratto da celle di codice, librerie, tabelle comuni, schemi di Unity Catalog e tag per mappare domande in linguaggio naturale nelle query e nel codice.

In futuro, aggiungeremo l'integrazione con DatabricksIQ per fornire ancora più contesto alle richieste.

Gestisci il tuo data warehouse in tutta sicurezza

Amministratori e team IT hanno bisogno di strumenti che li aiutino a comprendere come viene utilizzato il data warehouse. Con System Tables, Live Query Profile e Statement Timeouts, gli amministratori possono monitorare e risolvere i problemi nel momento in cui si verificano, contribuendo a mantenere efficiente il data warehouse.

Ottieni più insights e maggiore visibilità nell'ambiente SQL con System Tables. Le System Tables sono tabelle fornite da Databricks che contengono informazioni storiche su esecuzioni di istruzioni, costi, derivazione dei dati e altro. Esplora metadati e metriche di utilizzo per rispondere a domande come: "Quali istruzioni sono state eseguite e da chi?"; "Come e quando è scalato il warehouse?" e "Quali costi mi sono stati addebitati?". Dal momento che le System Tables sono integrate in Databricks, avrai accesso a funzionalità native come avvisi e dashboard SQL per automatizzare il processo di monitoraggio e notifica.

Ad oggi, ci sono tre System Tables in anteprima pubblica: Audit Logs, Billable Usage System Table e Lineage System Table (AWS | Azure). Prossimamente renderemo disponibili ulteriori tabelle di sistema per eventi nel warehouse e cronologia delle istruzioni.

Sarà così possibile, ad esempio, interrogare le Billable Usage System Tables per calcolare le DBU mensili usate per SKU.

```
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
```

Live Query Profile dà agli utenti informazioni in tempo reale sulle prestazioni delle query per aiutarli a ottimizzare i carichi di lavoro nell'immediato. Visualizza i piani di esecuzione delle query e valuta in tempo reale esecuzioni di attività di query per correggere errori SQL comuni, come join "esplosivi" o scansioni di tabelle complete. Con Live Query Profile, puoi accertarti che le query in esecuzione sul data warehouse siano ottimizzate e gestite in maniera efficiente. Per maggiori informazioni, leggi la documentazione (AWS | Azure).

Vuoi dei controlli automatizzati? Gli Statement Timeouts ti consentono di impostare un timeout personalizzato a livello di spazi di lavoro o query. Se il tempo di esecuzione di una query eccede la soglia del timeout, questa verrà automaticamente interrotta. Per maggiori informazioni, leggi la documentazione (AWS | Azure).

Nuove funzionalità coinvolgenti in DBSQL

Nell'ultimo anno, abbiamo lavorato con grande impegno per aggiungere funzionalità nuove e all'avanguardia a Databricks SQL. Siamo entusiasti di annunciare nuove funzionalità che mettono la potenza dell'AI nelle mani degli utenti SQL (ad esempio rendendo possibile la costruzione di warehouse SQL sull'intera piattaforma Databricks), introducono una nuova generazione di dashboard SQL e portano la potenza di Python in Databricks SQL.

databricks

Democratizza l'analisi di dati non strutturati con AI Functions

Con AI Functions, DB SQL porta la potenza dell'AI nel warehouse SQL. Sfrutta facilmente il potenziale dei dati non strutturati eseguendo attività quali analisi del sentiment o classificazione, riassunto e traduzione di testi. Gli analisti possono applicare modelli AI in modalità self-service, mentre i data engineer possono costruire autonomamente pipeline basate sull'AI.

Usare AI Functions è semplicissimo. Consideriamo ad esempio uno scenario in cui un utente voglia classificare il sentiment di alcuni articoli in base a una di queste quattro categorie: Frustrated, Happy, Neutral o Satisfied.

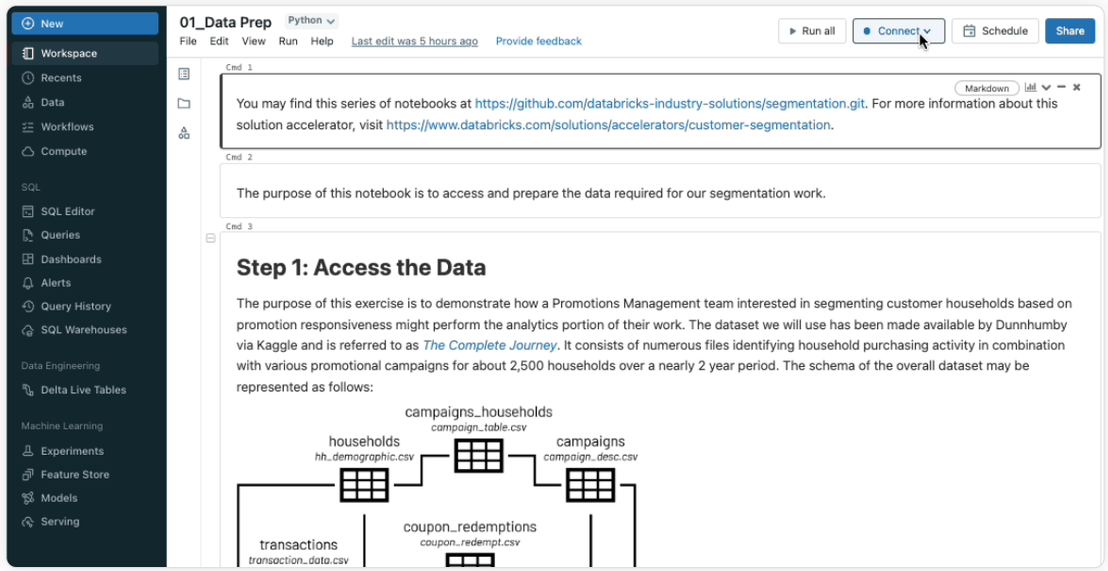
```
-- create a udf for sentiment classification
CREATE FUNCTION classify_sentiment(text STRING)
  RETURNS STRING
  RETURN ai_query(
    'Dolly', -- the name of the model serving endpoint
    named_struct(
      'prompt',
      CONCAT('Classify the following text into one of four categories [Frustrated,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

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

AI Functions are now in Public Preview. Per iscriverti all'anteprima, compila il modulo qui. Per maggiori informazioni, vedi il nostro dettagliato articolo sul blog o la documentazione (AWS | Azure).

Portare la potenza dei warehouse SQL nei notebook

I warehouse Databricks SQL sono in anteprima pubblica nei notebook. Potrai così combinare la flessibilità dei notebook con le prestazioni e il TCO dei warehouse SQL Serverless e Pro di Databricks. Per abilitare i warehouse SQL nei notebook, seleziona un warehouse SQL dal menu a discesa "Compute" dei notebook.



Connessione di warehouse SQL serverless dai notebook Databricks.

Trovare e condividere informazioni con una nuova generazione di dashboard

Scopri un'esperienza di dashboard rinnovata direttamente sull'architettura lakehouse. Agli utenti basterà selezionare un set di dati per costruire visualizzazioni di alto livello senza dover necessariamente usare SQL. Dimentica il fastidio di dover gestire query e oggetti dashboard separati: il modello di contenuto tutto in uno semplifica i processi di autorizzazione e gestione.

Pubblica una dashboard in tutta l'organizzazione in modo che qualsiasi utente autenticato nel tuo fornitore di identità possa accedervi tramite un link web sicuro, anche se non ha accesso a Databricks.

Le nuove dashboard di Databricks SQL sono al momento in anteprima privata. Contatta il tuo account team per saperne di più.

Sfruttare la flessibilità di Python in SQL

Porta la flessibilità di Python in Databricks SQL con le funzioni Python definite dall'utente (UDF). Integra modelli di machine learning o applica una logica di redazione personalizzata all'elaborazione e all'analisi dei dati chiamando funzioni Python personalizzate direttamente dalla tua query SQL. Le UDF sono funzioni riutilizzabili che consentono di applicare un'elaborazione uniforme a pipeline e analisi di dati.

Per cancellare indirizzi e-mail e numeri di telefono da un file, ad esempio, si potrebbe usare l'istruzione CREATE FUNCTION, come qui sotto:

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

Scopri come iscriverti all'anteprima privata qui.

databricks

Integrazioni con il tuo ecosistema di dati

Al Data+AI Summit, Databricks SQL ha annunciato nuove integrazioni per utilizzare una serie di strumenti in modo fluido.

Databricks + Fivetran

Siamo entusiasti di annunciare che Fivetran sarà accessibile in Partner Connect a tutti gli utenti, inclusi i non amministratori con sufficienti autorizzazioni in un catalogo. Questa innovazione rende 10 volte più semplice per tutti gli utenti acquisire dati in Databricks usando Fivetran. Si tratta di una grande vittoria per tutti i clienti Databricks, che potranno ora portare nell'architettura lakehouse i dati provenienti dalle centinaia di connettori che Fivetran offre, come Salesforce e PostgreSQL. Inoltre, Fivetran ora supporta pienamente anche i warehouse serverless.

Per maggiori informazioni, vedi l'articolo del blog qui.

Databricks + dbt Labs

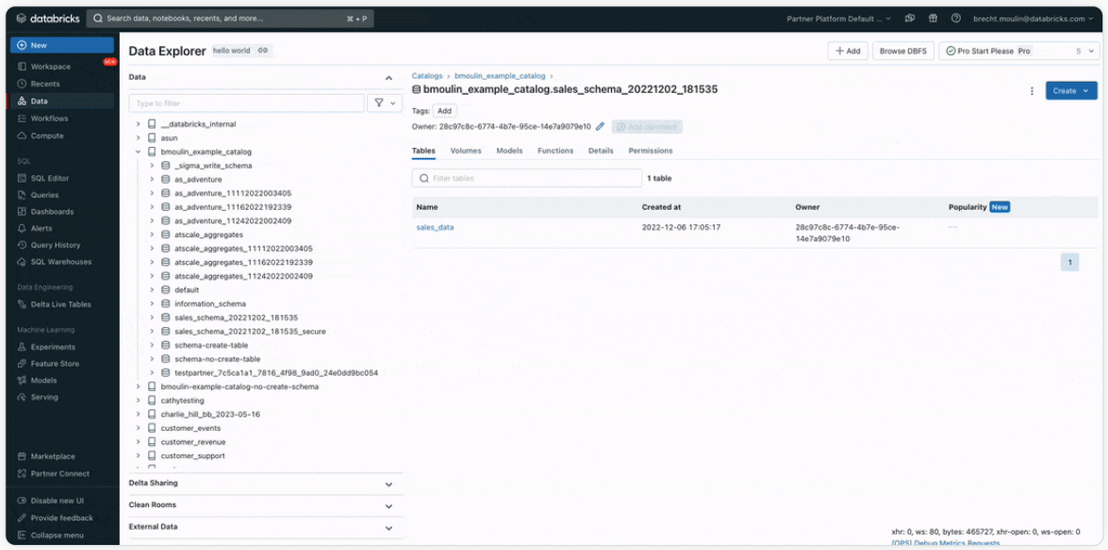
Semplifica l'ingegneria analitica in tempo reale sull'architettura lakehouse con Databricks e dbt Labs. La combinazione tra il popolarissimo framework per l'ingegneria analitica di dbt e la Databricks Platform offre potenti funzionalità:

- dbt + Streaming Tables. L'acquisizione in streaming da qualsiasi sorgente è ora integrata nei progetti dbt. Usando SQL, gli analisti possono definire e acquisire dati in cloud/streaming direttamente nelle loro pipeline dbt.
- dbt + Materialized Views. Costruire pipeline efficienti diventa più semplice con dbt, grazie alle potenti funzionalità di aggiornamento incrementale di Databricks. Gli utenti possono usare dbt per costruire ed eseguire pipeline, riducendo i costi di infrastruttura con un'efficiente elaborazione incrementale.

Per maggiori informazioni, vedi questo articolo dettagliato.

Databricks + PowerBI: pubblicare sugli spazi di lavoro PowerBI

Pubblica set di dati provenienti dal tuo spazio di lavoro Databricks sugli spazi di lavoro PowerBI online con pochi clic. Non sarà più necessario gestire connessioni odbc/jdbc: basterà selezionare il set di dati da pubblicare. Scegli i set di dati o gli schemi da pubblicare e poi seleziona il tuo spazio di lavoro PBI. In questo modo sarà più semplice per amministratori e creatori di report BI supportare spazi di lavoro PowerBI senza dover necessariamente usare Power BI Desktop.



Primi passi con Databricks SQL

Segui la guida (AWS | Azure | GCP) sulla costruzione di un warehouse SQL per cominciare a usare Databricks SQL. Databricks SQL Serverless è ora disponibile con uno sconto promozionale di oltre il 20%. Consulta il nostro listino prezzi per maggiori informazioni.

Puoi anche guardare **Databricks SQL: Why the Best Serverless Data Warehouse is a Lakehouse** and **What's New in Databricks SQL — With Live Demos** per una panoramica completa.

CAPITOLO 3.5

Governance dei dati distribuita e ambienti isolati con Unity Catalog

di Max Nienu, Zeashan Pappa, Paul Roome e Sachin Thakur

Una governance efficace è essenziale per qualsiasi organizzazione che si affidi ai dati, all'analisi e all'AI per le sue attività operative. Molte organizzazioni sono ora consapevoli del valore aggiunto di una governance dei dati centralizzata. Tuttavia, anche con le migliori intenzioni, implementare una tale governance centralizzata può essere difficoltoso in assenza di processi e risorse organizzative adeguati. Il ruolo del Chief Data Officer (CDO) è ancora relativamente nuovo, lasciando aperta la questione su chi debba definire ed eseguire le policy di governance dei dati.

Di conseguenza, la responsabilità di definire ed eseguire tali policy spesso non è centralizzata, con conseguenti variazioni nelle policy o negli organismi di governance dei vari settori, sub-unità e reparti di un'organizzazione. Per semplicità, possiamo chiamare questo modello governance distribuita: un contesto in cui vi è consenso generale sulle distinzioni tra questi organismi, ma non necessariamente una funzione di governance centralizzata.

In questa sezione vedremo come implementare un modello di governance distribuita usando Databricks Unity Catalog, che offre una soluzione unificata per dati, analisi e AI nel lakehouse.

Evoluzione della governance dei dati in Databricks

Prima dell'introduzione di Unity Catalog, il workspace era considerato una realtà monolitica: ciascuno spazio di lavoro aveva un suo metastore, una gestione degli utenti e uno store di tabelle ACL. Questo portava a un'intrinseca compartimentalizzazione di dati e governance tra spazi di lavoro e a un raddoppio degli sforzi per risolvere i problemi di coerenza tra di essi.

Per gestire la situazione, alcuni clienti hanno deciso di eseguire pipeline o codice per sincronizzare i loro metastore e ACL, mentre altri hanno predisposto dei metastore gestiti autonomamente da usare nei vari spazi di lavoro. Queste soluzioni, tuttavia, comportavano un aumento delle spese operative e dei costi di manutenzione, costringendo a decidere in anticipo l'architettura da adottare per ripartire i dati nell'organizzazione e portando alla creazione di silos.

Governance dei dati con Unity Catalog

Per superare questi limiti Databricks ha sviluppato Unity Catalog che facilita l'implementazione di una governance dei dati massimizzando al tempo stesso la capacità di collaborazione e condivisione. Il primo passo è stato quello di implementare un namespace comune che permetta di accedere a tutti i dati dell'organizzazione.

Questo approccio può sembrare una sfida al modello di governance distribuito, ma Unity Catalog offre nuovi meccanismi di isolamento all'interno del namespace per riprodurre ciò che si otteneva utilizzando più metastore Hive. I meccanismi di isolamento consentono ai gruppi di operare in modo indipendente, con interazioni scarse o nulle, ed essere applicati ad altri scenari, ad esempio per separare gli ambienti di produzione e sviluppo.

Hive Metastore vs. Unity Catalog in Databricks

Con Hive, i metastore avevano un limite di servizio: diversi metastore implicavano infatti diversi servizi Hive ospitati e diversi database sottostanti. Unity Catalog è un servizio di piattaforma integrato nella Databricks Data Intelligence Platform e, di conseguenza, senza limiti di servizio.

Unity Catalog offre un namespace comune che consente di eseguire governance e audit dei dati in un unico luogo.

Con Hive, era comune usare più metastore, ciascuno con un proprio namespace, per isolare l'ambiente di sviluppo da quello di produzione, o per separare i dati tra le diverse unità operative.

Unity Catalog risponde a queste esigenze attraverso meccanismi di isolamento dinamici sui namespace che non compromettono la capacità di condividere i dati e di collaborare su di essi e non richiedono difficili e irreversibili decisioni iniziali sull'architettura da adottare.

Lavorare con team e ambienti diversi

Quando si usa una piattaforma di dati, c'è spesso la pressante esigenza di isolare ambienti diversi, come sviluppo e produzione, oppure gruppi, team o unità operative dell'organizzazione.

Cominciamo definendo i termini di isolamento in una piattaforma di dati come Databricks:

- L'accesso degli utenti ai dati deve avvenire solo in base a regole concordate.
- I dati possono essere gestiti da utenti o team di utenti designati.
- I dati devono essere fisicamente separati in archivi.
- L'accesso ai dati deve essere possibile solo in ambienti designati.

L'accesso degli utenti ai dati deve avvenire solo in base a regole concordate

Le organizzazioni hanno in genere regole stringenti sull'accesso ai dati, basate su requisiti organizzativi o normativi fondamentali per garantire la sicurezza dei dati stessi. Esempi tipici sono le informazioni relative ai salari dei dipendenti o ai pagamenti con carte di credito.

L'accesso a queste informazioni è in genere strettamente controllato e sottoposto a periodici audit. Unity Catalog offre alle organizzazioni un controllo granulare sulle risorse di dati all'interno del catalogo per soddisfare questi standard di settore. Grazie ai controlli, Unity Catalog assicura che gli utenti possano vedere e interrogare solo i dati ai quali sono autorizzati ad accedere.

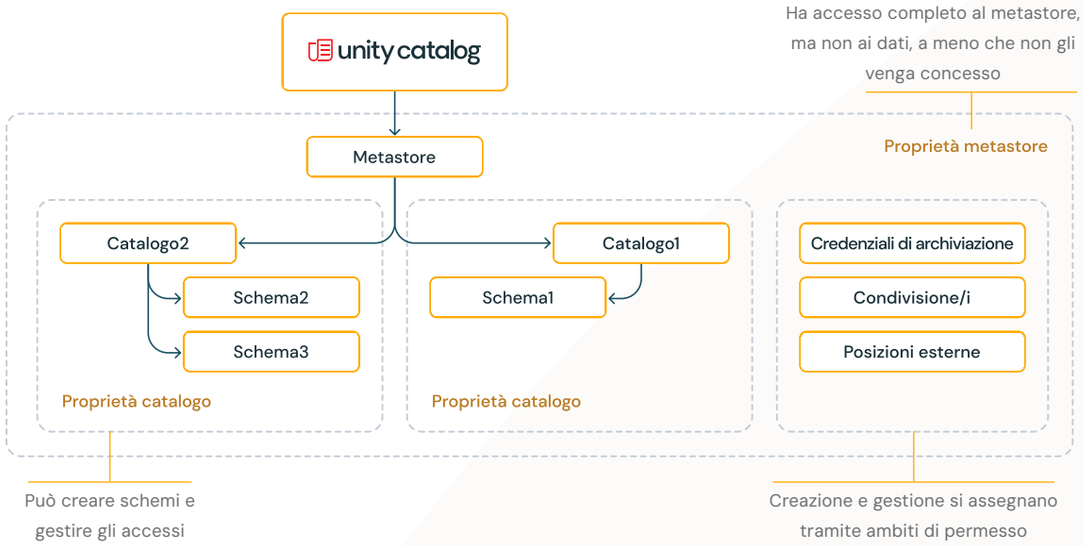
I dati possono essere gestiti da utenti o team di utenti designati

Unity Catalog dà la possibilità di scegliere tra un modello di governance centralizzato e uno distribuito.

Nel modello di governance centralizzato, gli amministratori della governance sono proprietari del metastore e possono assumere la proprietà di qualsiasi oggetto e stabilire ACL e policy.

In un modello di governance distribuito, un catalogo o un insieme di cataloghi vengono considerati un dominio di dati. Il proprietario di quel catalogo può creare tutte le risorse, assumerne la proprietà e gestire la governance in quel dominio. Di conseguenza, i proprietari dei vari domini possono operare indipendentemente l'uno dall'altro.

In entrambi i casi, consigliamo vivamente di designare un gruppo come proprietario o service principal se la gestione viene fatta tramite altri strumenti.



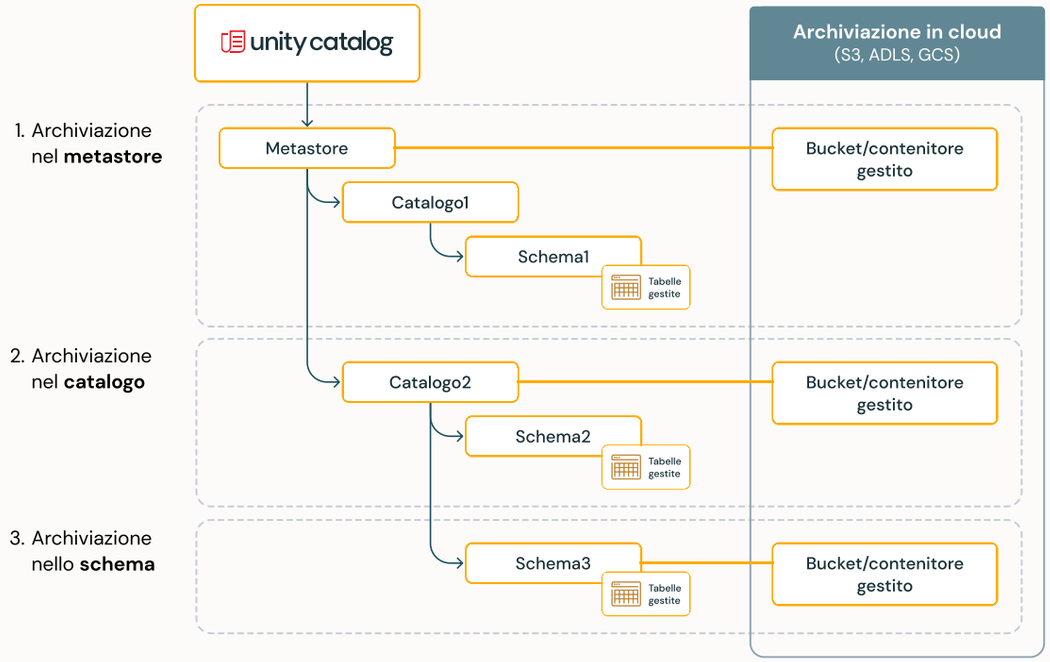
I dati devono essere fisicamente separati in archivi

Di default, quando si crea un metastore Unity Catalog, il Databricks Account Admin fornisce una singola posizione di archiviazione e una credenziale nel cloud come posizione predefinita per le tabelle gestite.

Le organizzazioni che necessitano dell'isolamento fisico dei dati, per ragioni normative o ad esempio in ambiti SDLC tra unità aziendali, o anche per finalità di allocazione dei costi, dovrebbero valutare l'utilizzo di funzionalità per sorgenti di dati gestite a livello di catalogo e di schema.

Unity Catalog permette di scegliere in modo predefinito come separare i dati nell'archivio. Di default, tutti i dati vengono memorizzati nel metastore. Con il supporto alle funzionalità per sorgenti di dati gestite su cataloghi e schemi è possibile isolare fisicamente archiviazione e accesso ai dati, permettendo all'organizzazione di soddisfare i requisiti di governance e gestione.

Quindi, quando si creano tabelle gestite, i dati verranno archiviati usando la posizione dello schema (se presente), seguita dalla posizione del catalogo (se presente) e si userà la posizione del metastore solo se le prime due posizioni non sono state definite.

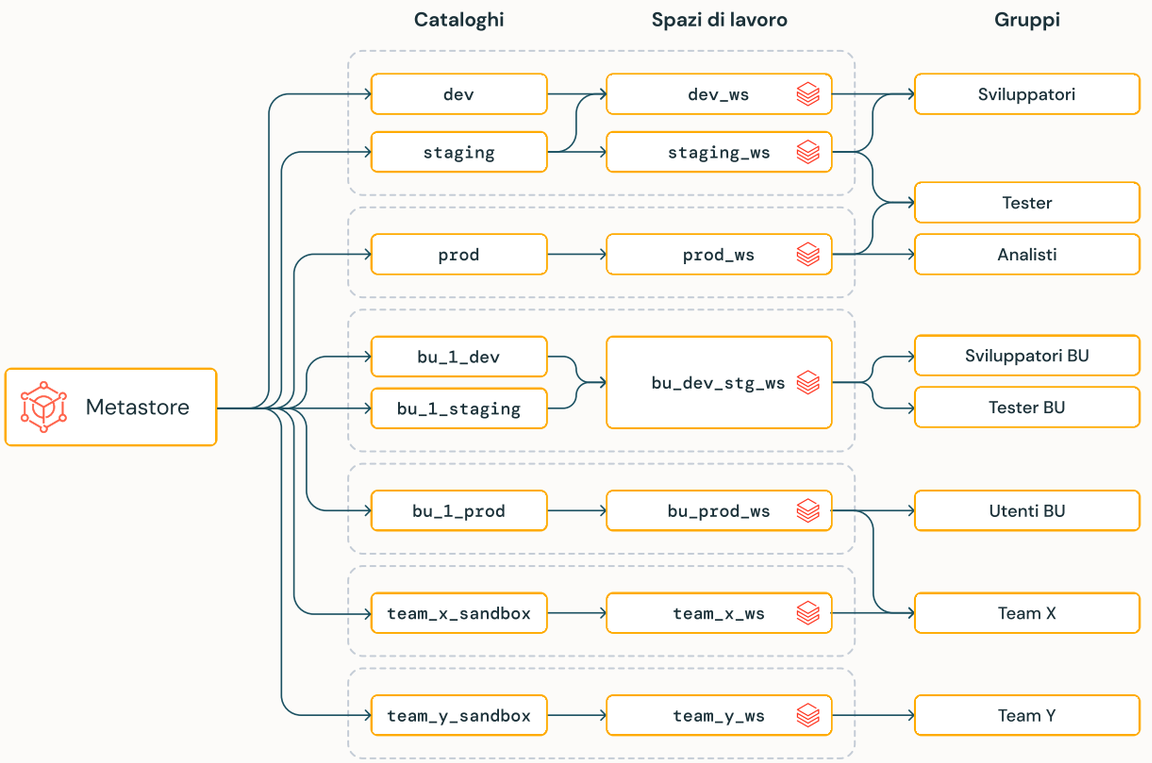


databricks

L'accesso ai dati deve essere possibile solo in ambienti designati, in base al loro scopo

Spesso i requisiti organizzativi e normativi impongono di rendere accessibili determinati dati solo in specifici ambienti. È il caso ad esempio degli ambienti di sviluppo e produzione, o di ambienti HIPAA o PII che contengono dati PII per l'analisi e hanno speciali regole d'accesso che determinano chi può accedere e da quali ambienti. A volte, i requisiti impongono che certi set o domini di dati non possano essere incrociati o combinati.

In Databricks, consideriamo ogni spazio di lavoro come un ambiente. Unity Catalog ha una funzionalità che consente di "associare" cataloghi a spazi di lavoro. Questi ACL basati sull'ambiente garantiscono che solo certi cataloghi siano disponibili in un determinato spazio di lavoro, indipendentemente dagli ACL individuali di un utente. Ciò significa che l'amministratore del metastore o il proprietario del catalogo possono definire gli spazi di lavoro dai quali è possibile accedere a un catalogo di dati. Questo processo può essere controllato tramite la nostra UI o con API/terraform per integrazioni semplici. Recentemente abbiamo anche pubblicato un articolo su come controllare Unity Catalog tramite terraform in modo da adattarlo a uno specifico modello di governance.



L'accesso ai dati e la loro disponibilità si possono isolare a livello di spazi di lavoro e gruppi. Gli utenti possono accedere solo a certi cataloghi, in certi ambienti.

Conclusioni

Con Unity Catalog al centro della tua architettura lakehouse, puoi implementare un modello di governance flessibile e scalabile senza sacrificare la capacità di gestire e condividere i dati in modo efficace. Unity Catalog ti permette di superare i limiti e vincoli del tuo attuale metastore Hive per isolare più facilmente i dati e renderli disponibili per la collaborazione in base alle specifiche esigenze aziendali. Consulta le guide su Unity Catalog (AWS, Azure) per iniziare. Scarica questo eBook gratuito su governance di dati, analisi e AI per apprendere le best practice di una strategia di governance efficace per il tuo data lakehouse.

CAPITOLO 3.6

Guida pratica al privilege model e al controllo degli accessi in Unity Catalog

Estrapola i concetti del modello di privilegi di Unity Catalog in modo semplice e assimilabile per supportare differenti esigenze e modelli di accesso

di Som Natarajan e Vuong Nguyen

Con l'aumentare di volume, velocità e varietà dei dati, le organizzazioni si trovano a fare sempre più affidamento su una robusta governance per conseguire i risultati di business attesi. Unity Catalog è una soluzione di governance granulare per dati e AI integrata nella Databricks Data Intelligence Platform. Semplifica la messa in sicurezza e la governance del patrimonio di dati dell'azienda fornendo un meccanismo centralizzato per amministrare e controllare gli accessi.

In passato, prima che Unity Catalog unificasse il modello dei permessi per file e tabelle e aggiungesse il supporto per tutti i linguaggi, i clienti implementavano soluzioni di controllo granulare degli accessi su Databricks usando la tabella ACL (TACL) legacy a livello di spazio di lavoro, che era essenzialmente limitata a certe configurazioni di cluster e funzionava solo per Python e SQL. Sia Unity Catalog che TACL consentono di controllare l'accesso a oggetti da proteggere come cataloghi, schemi (database), tabelle e viste, ma ci sono alcune differenze nel funzionamento dei due modelli.

Un'adeguata comprensione del modello di accesso agli oggetti è essenziale per implementare una governance dei dati su larga scala usando Unity Catalog. Soprattutto se si è già implementato il modello Table ACL e si vuole aggiornarlo a Unity Catalog per approfittare di tutte le funzionalità più recenti, come supporto multi-linguaggio, controllo degli accessi centralizzato e derivazione dei dati.

Gli assiomi del modello di accesso di Unity Catalog

- In Unity Catalog i privilegi sono definiti a livello di metastore. I permessi in Unity Catalog si riferiscono sempre a identità a livello di account, mentre i permessi TACL definiti nel catalogo hive_metastore si riferiscono sempre alle identità locali nello spazio di lavoro.
- Ereditarietà dei privilegi. Gli oggetti in Unity Catalog sono gerarchici e i privilegi vengono ereditati a cascata. L'oggetto di livello più alto da cui vengono ereditati i privilegi è il catalogo.
- La proprietà degli oggetti è importante. I privilegi possono essere concessi solo da un amministratore di metastore, dal proprietario di un oggetto o dal proprietario del catalogo o schema che contiene l'oggetto. Solo il proprietario di un oggetto, o il proprietario del catalogo e dello schema che lo contengono, possono eliminare l'oggetto.
- Privilegi USE come limiti. Il privilegio USE CATALOG/SCHEMA è richiesto per interagire con gli oggetti contenuti in un catalogo/schema. Tuttavia, il privilegio USE non consente di esplorare i metadati degli oggetti presenti all'interno del catalogo/schema.
- I permessi sugli oggetti derivati sono semplificati. Unity Catalog richiede solo che il proprietario di una vista abbia il privilegio SELECT, insieme a USE SCHEMA sullo schema genitore delle viste e USE CATALOG sul catalogo genitore. Con TACL, invece, il proprietario di una vista deve essere il proprietario di tutte le tabelle e viste a cui si fa riferimento.

Altri assiomi più complessi

- Sicurezza di default. Solo i cluster con modalità d'accesso (condivise o per utente singolo) specifiche per Unity Catalog possono accedere ai dati di Unity Catalog. Con TACL tutti gli utenti hanno accesso a tutti i dati su cluster non condivisi.
- Restrizione sui cluster a utente singolo. I cluster a utente singolo non supportano viste dinamiche. Gli utenti devono avere il privilegio SELECT su tutte le tabelle e le viste a cui si fa riferimento per poter leggere da una vista.
- Nessun supporto per privilegi ANY FILE o ANONYMOUS FUNCTION: Unity Catalog non supporta questi permessi, in quanto potrebbero essere usati per aggirare i controlli e consentire a un utente privo di privilegi di eseguire codice privilegiato.

Alcuni pattern interessanti

Il modello di accesso di Unity Catalog può essere utilizzato per implementare diversi pattern di governance.

Esempio 1. Permessi estesi a più spazi di lavoro

L'Assioma 1 consente ai team di prodotto di definire permessi per i prodotti di dati all'interno del loro spazio di lavoro e di vederli riflessi e applicati su tutti gli altri spazi di lavoro, indipendentemente dalla provenienza dei consumatori.

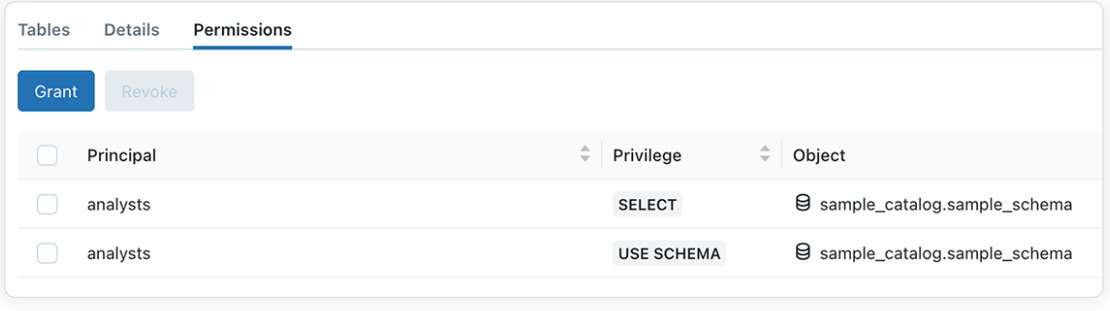
Esempio 2. Stabilire dei limiti per il data sharing

L'Assioma 2 consente ai proprietari di cataloghi/schemi di impostare regole di accesso predefinite ai loro dati. Il seguente comando, ad esempio, permette ai membri del team di machine learning di creare tabelle all'interno di uno schema e di leggere l'uno le tabelle dell'altro:

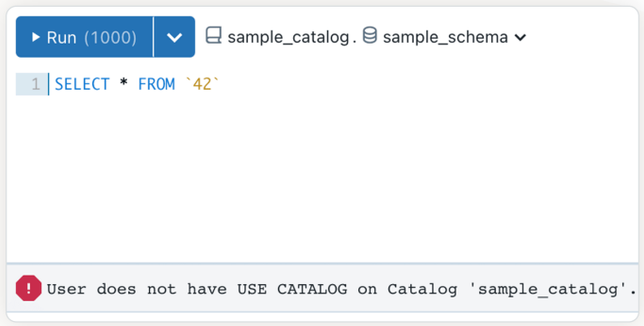
```
CREATE CATALOG ml;
CREATE SCHEMA ml.sandbox;
GRANT USE_CATALOG ON CATALOG ml TO ml_users;
GRANT USE_SCHEMA ON SCHEMA ml.sandbox TO ml_users;
GRANT CREATE TABLE ON SCHEMA ml.sandbox TO ml_users;
GRANT SELECT ON SCHEMA ml.sandbox TO ml_users;
```

Un aspetto ancora più interessante è che l'Assioma 4 permette ai proprietari di cataloghi/schemi di decidere quanto proprietari di singoli schemi e tabelle possono condividere i dati che producono. Il proprietario di una tabella che concede il privilegio SELECT a un altro utente non permette a quell'utente di accedere alla tabella in lettura, a meno che non abbia anche privilegi USE CATALOG sul suo catalogo genitore e privilegi USE SCHEMA sul suo schema genitore.

Nell'esempio qui sotto, sample_catalog è di proprietà dell'utente A; l'utente B ha creato lo schema sample_schema e la tabella 42. Anche se al team di analisi si concedono privilegi USE SCHEMA e SELECT, i suoi membri non potranno comunque interrogare la tabella, dato il limite sui permessi impostato dall'utente A.



Pagina dei permessi che mostra un gruppo di analisti con permessi SELECT e USE SCHEMA su sample_catalog.sample_schema

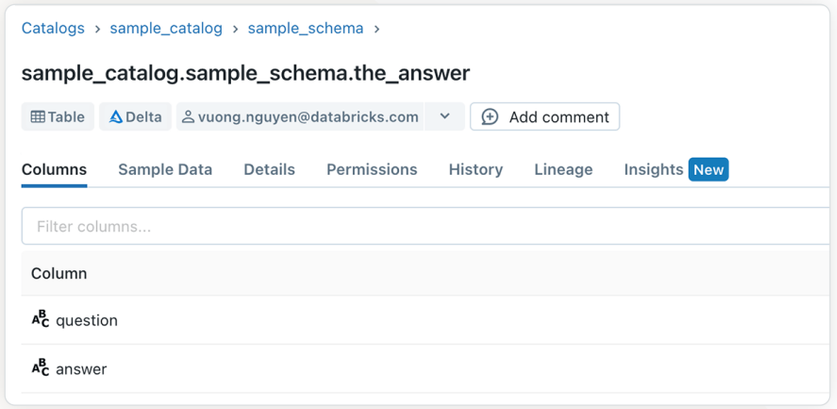


Pagina della query con un messaggio di errore.

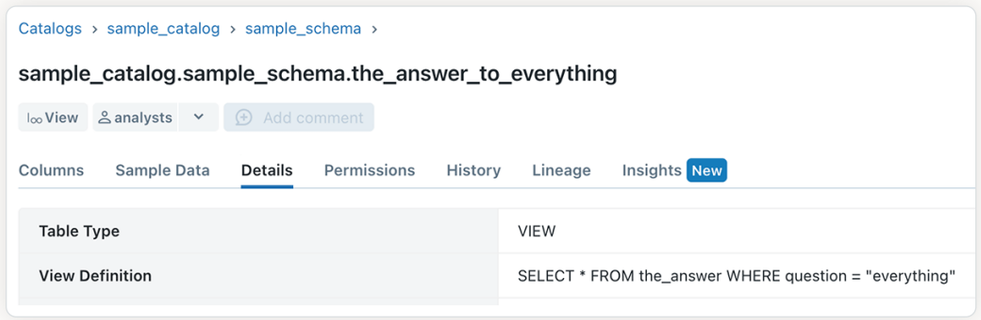
Esempio 3. Condivisione semplificata della logica operativa

I fruitori di dati hanno l'esigenza di condividere la propria logica operativa e di trasformazione; per farlo in un modo riutilizzabile si possono creare viste e condividerle con altri fruitori.

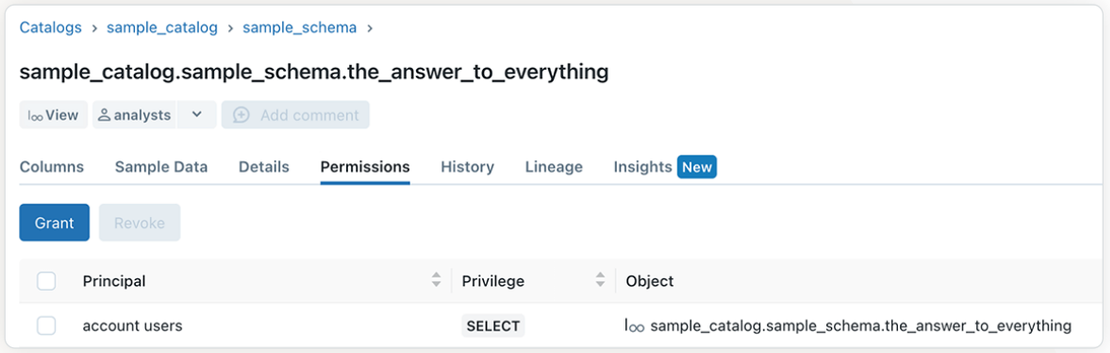
L'Assioma 5 consente ai fruitori di dati di eseguire questa operazione facilmente, senza dover passare continuamente e in modo manuale dai proprietari delle tabelle.



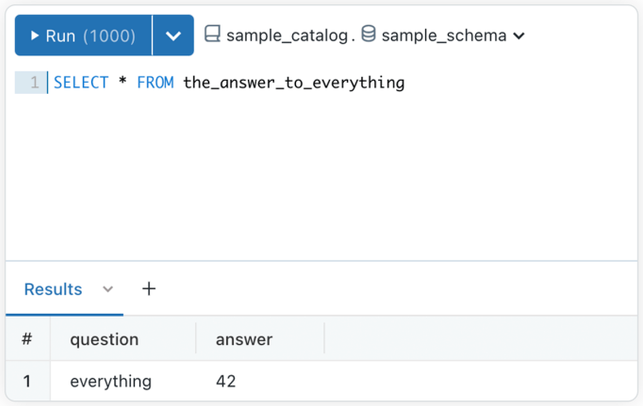
Proprietà della tabella.



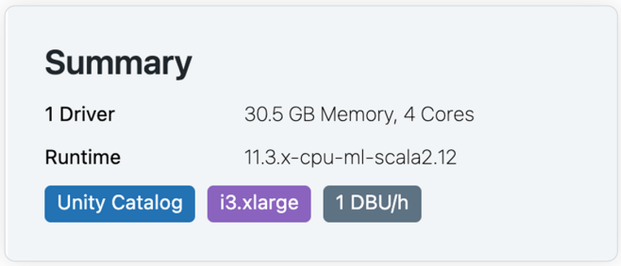
Definizione della vista.



Pagina dei permessi con una vista di proprietà di un gruppo di analisti e un gruppo di utenti di account con privilegio SELECT



Pagina delle query che mostra il risultato per “the answer to everything”.



Riepilogo del cluster che mostra il supporto a Unity Catalog.

Esempio 4. Niente più fughe di dati

Grazie all'Assioma 6, i proprietari di dati possono assicurarsi che non vi siano accessi non autorizzati ai loro dati in caso di errata configurazione dei cluster. Qualsiasi cluster non configurato con la corretta modalità di accesso non potrà accedere ai dati nello Unity Catalog.

Gli utenti possono verificare che i loro cluster possano accedere ai dati nello Unity Catalog grazie a questo pratico suggerimento sulla pagina Create Clusters.

Ora che i proprietari dei dati sono in grado di comprendere il modello privilegi e il controllo degli accessi, possono usare Unity Catalog per semplificare la gestione della policy di accesso su larga scala.

Prossimamente, saranno disponibili ulteriori funzionalità che permetteranno ad amministratori e proprietari di dati di creare policy di accesso ancora più complesse:

- **Filtri di righe e maschere di colonne.** Usa funzioni SQL standard per definire filtri di righe e maschere di colonne per controllare in modo granulare l'accesso a righe e colonne.
- **Controllo degli accessi basato sugli attributi.** Definisce policy di accesso basate su tag (attributi) delle risorse di dati.

CAPITOLO

04

Casi d'uso di analisi su Databricks

CAPITOLO 4.1

Come costruire una soluzione per analisi di marketing usando Fivetran e dbt su Databricks

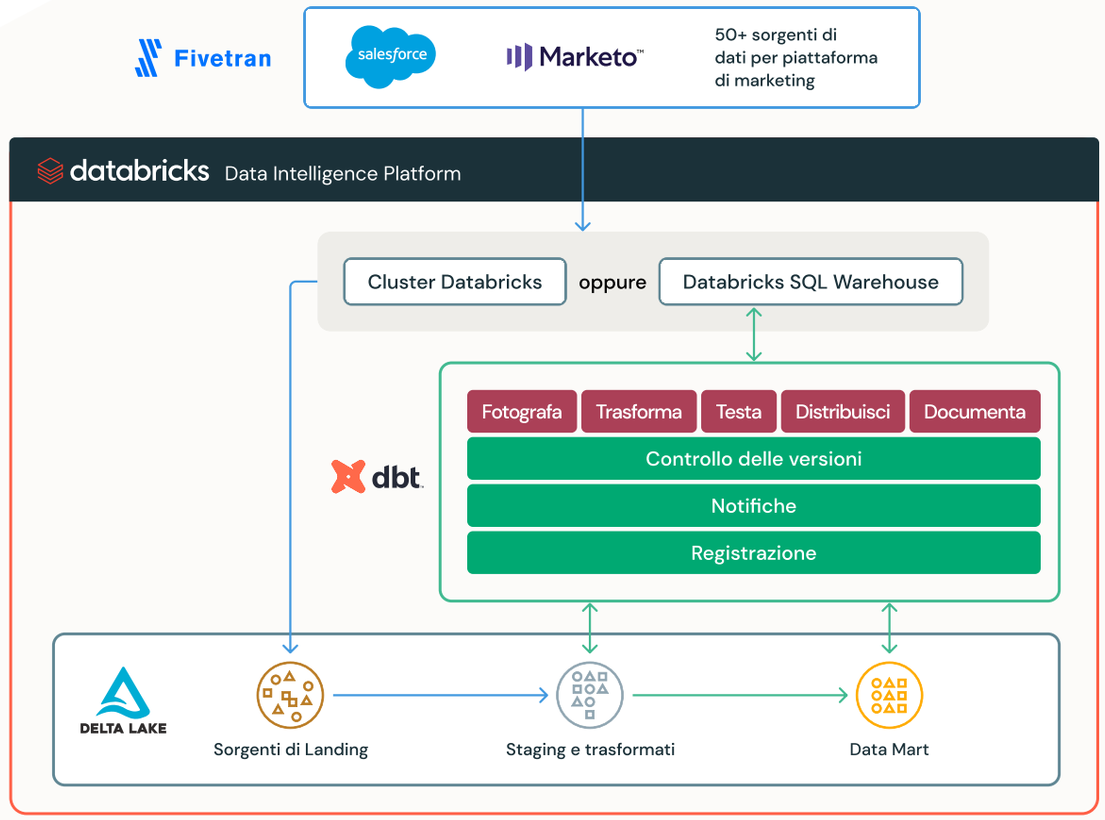
di Tahir Fayyaz, Bilal Aslam e Robert Saxby

I team di marketing usano diverse piattaforme per condurre campagne di marketing e di vendita, che possono generare un volume significativo di dati preziosi ma slegati. Unificare tutti questi dati può tradursi in un considerevole ritorno sull'investimento, come dimostra l'esperienza di Publicis Groupe, che ha aumentato di ben il 50% la redditività delle sue campagne di marketing.

Databricks, che unifica casi d'uso di data warehousing e AI su una singola piattaforma, è l'ideale per costruire una soluzione per analisi di marketing: manteniamo una singola fonte di verità e supportiamo i casi d'uso con AI/ML. Utilizziamo anche due soluzioni di partner, Fivetran e dbt, per supportare un'ampia gamma di casi d'uso di analisi di marketing, incluse quelle sul tasso di abbandono e sul valore totale dei clienti, sulla segmentazione della clientela e sull'efficacia degli annunci.

Fivetran consente di acquisire facilmente dati da oltre 50 piattaforme di marketing in Delta Lake senza dover costruire e mantenere complesse pipeline. Se un'API di una qualsiasi piattaforma di marketing cambia o cessa di funzionare, Fivetran aggiornerà e riparerà le integrazioni in modo che il flusso dei dati non si interrompa.

dbt è un popolare framework open source che permette agli utenti di lakehouse di costruire pipeline di dati usando semplici istruzioni in SQL. Tutto è organizzato all'interno di directory in formato testo, semplificando così controllo di versione, distribuzione e testabilità. Una volta che i dati sono stati acquisiti in Delta Lake, usiamo dbt per trasformarli, testarli e documentarli. Il data mart per analisi di mercato costruito sulla base dei dati acquisiti è quindi pronto per supportare la gestione di nuove campagne e iniziative di marketing.



Fivetran e dbt possono leggere e scrivere su Delta Lake usando un cluster Databricks o un warehouse Databricks SQL

Fivetran e dbt appartengono entrambi a Databricks Partner Connect, un portale omnicomprensivo per indagare e integrare in sicurezza dati, analisi e strumenti di AI direttamente all'interno della Databricks Platform. In pochi clic è possibile configurare e connettere questi strumenti e molti altri direttamente dal proprio spazio di lavoro Databricks.

Come costruire una soluzione per analisi di marketing

In questa demo pratica, ti mostreremo come acquisire dati da Marketo e Salesforce in Databricks tramite Fivetran e usare poi dbt per trasformare, testare e documentare il modello di dati per analisi di mercato.

Tutto il codice per la demo è disponibile su Github nel repository degli esempi di flussi di lavoro.

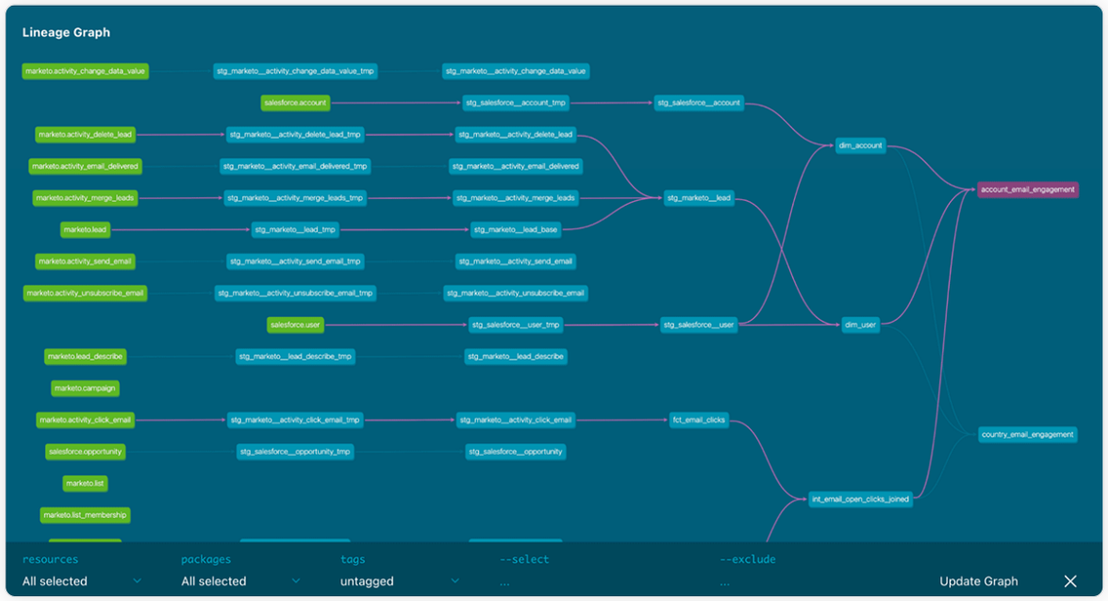
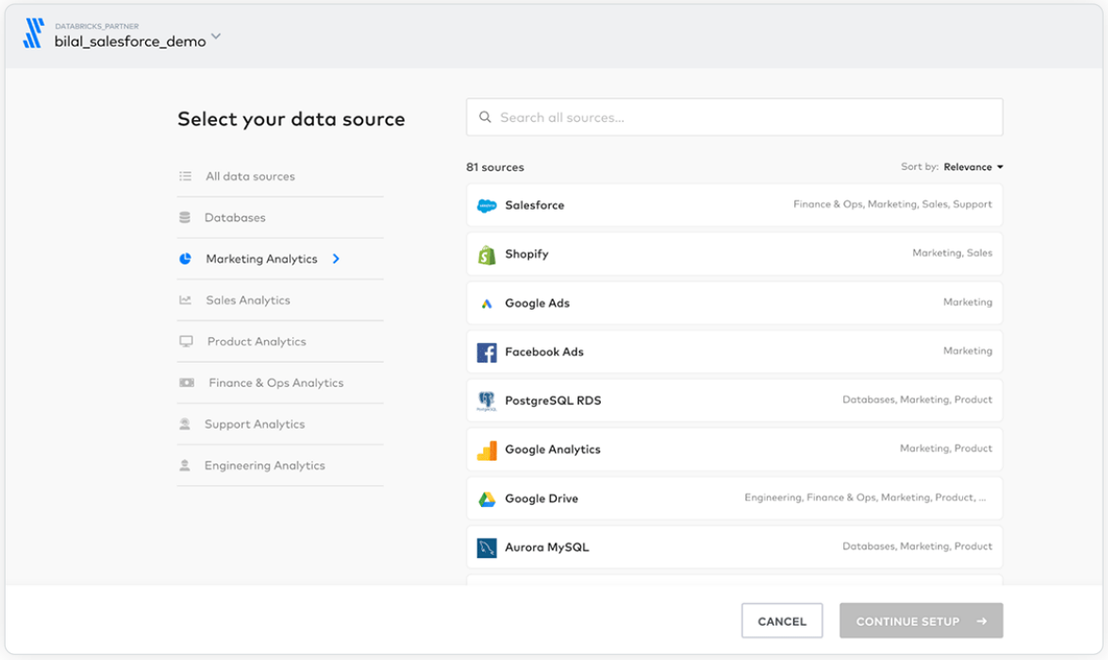


Grafico di derivazione dbt che mostra sorgenti e modelli di dati.

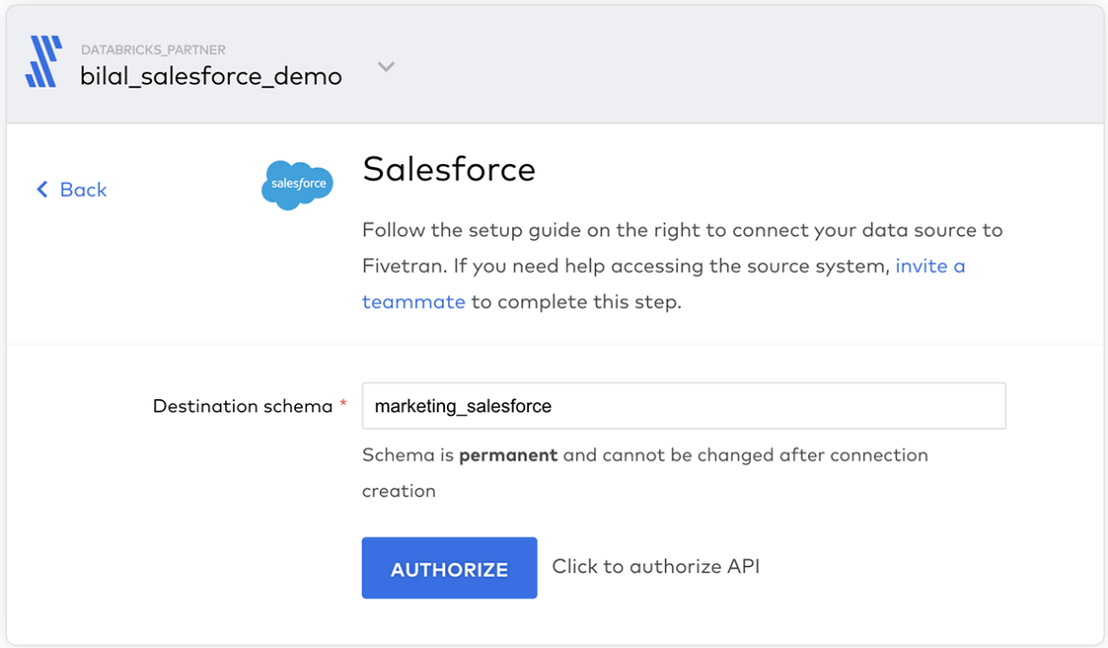
Il grafico di derivazione dbt del modello finale avrà questo aspetto. Le tabelle sorgente di Fivetran sono a sinistra, in verde, mentre i modelli finali per analisi di mercato sono a destra. Selezionando un modello, puoi vedere le corrispondenti dipendenze con i vari modelli evidenziate in porpora.

Acquisizione di dati con Fivetran



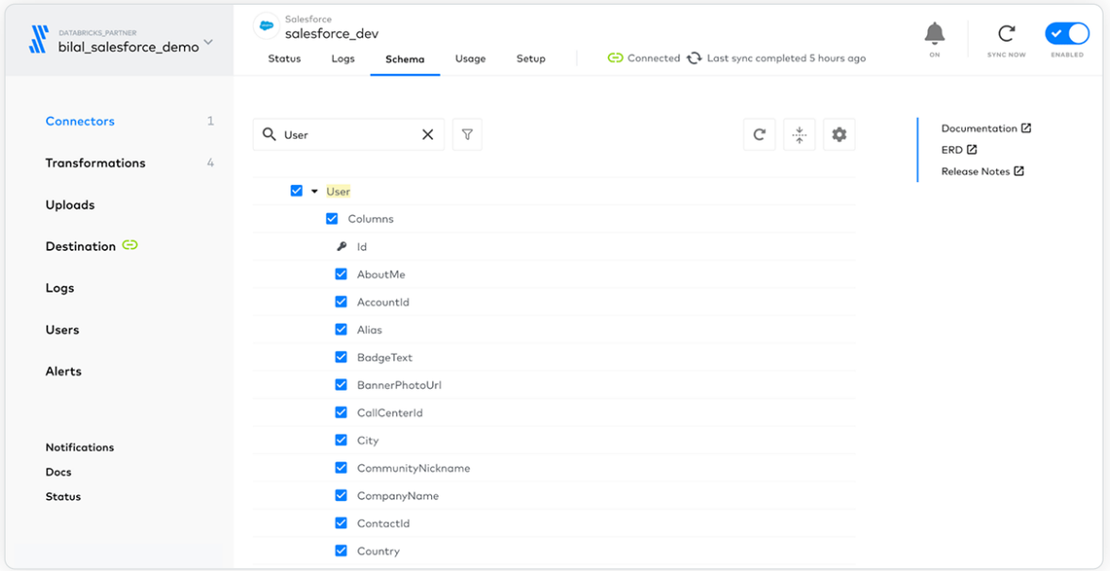
Fivetran ha molti connettori per sorgenti di dati destinati all'analisi di marketing.

Imposta nuove connessioni a Salesforce e Marketo in Fivetran per iniziare ad acquisire i dati di marketing in Delta Lake. Quando imposta le connessioni, Fivetran automaticamente crea e gestisce anche uno schema per ciascuna sorgente di dati in Delta Lake. In seguito, puoi usare dbt per trasformare, pulire e aggregare questi dati.



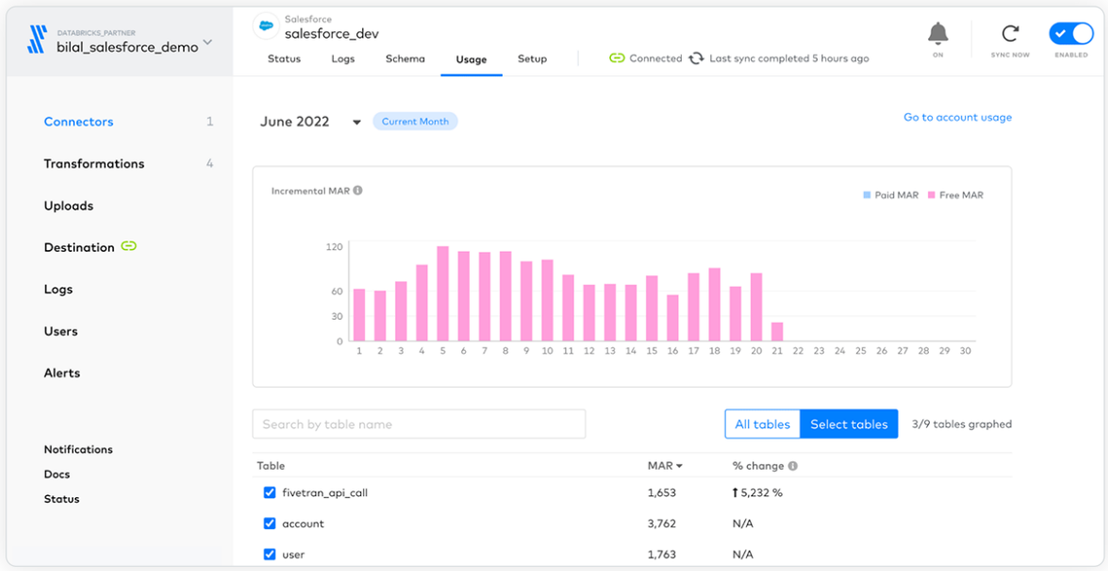
Definizione di uno schema di destinazione in Delta Lake per la sorgente di dati Salesforce.

Per la demo, assegna agli schemi che saranno creati in Delta Lake i nomi marketing_salesforce e marketing_marketo. Se gli schemi non esistono, Fivetran li creerà come parte del carico di acquisizione iniziale.



Selezione degli oggetti della sorgente di dati da sincronizzare come tabelle Delta Lake.

A questo punto, puoi scegliere quali oggetti sincronizzare in Delta Lake, dove ciascun oggetto verrà salvato come tabella individuale. Con Fivetran è semplice gestire e visualizzare quali colonne vengono sincronizzate per ciascuna tabella:



Dashboard di Fivetran per il monitoraggio delle righe attive mensilmente sincronizzate.

Inoltre, Fivetran offre una dashboard di monitoraggio per analizzare quante righe attive mensilmente vengono sincronizzate ogni giorno e ogni mese per ciascuna tabella, insieme ad altri log e statistiche utili.

Modellazione dei dati con dbt

Ora che tutti i dati di marketing sono stati acquisiti in Delta Lake, puoi usare dbt per creare i modelli di dati seguendo questi passaggi.

Configurazione di un progetto dbt locale e connessione a Databricks SQL

Configura l'ambiente di sviluppo dbt locale in un ambiente IDE a scelta seguendo le istruzioni di configurazione per dbt Core e dbt-databricks.

Esegui lo scaffolding di un nuovo progetto dbt e connettilo a un Databricks SQL Warehouse usando dbt init, che chiederà le seguenti informazioni:

```
$ 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]:
```

Una volta configurato il profilo, puoi testare la connessione usando:

```
$ dbt debug
```

Installare i pacchetti di modelli dbt di Fivetran per lo staging

Per usare i dati Marketo e Salesforce, per prima cosa dovremo creare le tabelle come fonti del nostro modello. Fivetran velocizza questa operazione grazie ai suoi pacchetti di modelli dbt Fivetran preconfigurati. In questa demo useremo i pacchetti `marketo_source` e `salesforce_source`.

Per installare i pacchetti aggiungi un file **`packages.yml`** alla root del progetto dbt e quindi aggiungi i pacchetti `marketo-source`, `salesforce-source` e `fivetran-utils`:

```
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
```

Dovresti ora vedere i pacchetti Fivetran installati nella cartella relativa.

Aggiornare dbt_project.yml per i modelli dbt di Fivetran

Ci sono alcuni parametri di configurazione nel file `dbt_project.yml` da modificare per far funzionare correttamente i pacchetti Fivetran in Databricks.

Il file `dbt_project.yml` si trova nella cartella principale del progetto dbt.

spark_utils sostituisce le macro in dbt_utils

I modelli dbt di Fivetran usano le macro contenute nel pacchetto dbt_utils ma, per poter funzionare con Databricks, alcune di esse devono essere modificate. A tal fine, useremo il pacchetto spark_utils .

Funziona fornendo degli shim per certe macro di `dbt_utils` che puoi impostare usando dispatch config nel file `dbt_project.yml` in modo che dbt cerchi le macro prima nel pacchetto `spark_utils` quando risolve macro dal namespace `dbt_utils`.

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

Variabili per gli schemi marketo_source e salesforce_source

I pacchetti Fivetran richiedono che vengano definiti catalogo (chiamato database in dbt) e schema della posizione di landing dei dati acquisiti da Fivetran.

Aggiungi queste variabili al file `dbt_project.yml` con i nomi corretti di catalogo e schema. Il catalogo predefinito è `hive_metastore`, che verrà usato se `_database` è lasciato in bianco. I nomi degli schemi saranno quelli che hai definito al momento della creazione delle connessioni in Fivetran.

```
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
```

Schema di destinazione per i modelli di staging di Fivetran

Per evitare che tutte le tabelle di staging create dai modelli d'origine di Fivetran vengano ricreate nello scherma di destinazione predefinito, può essere utile definire uno schema di staging separato.

Nel file `dbt_project.yml` scrivi il nome dello schema di staging, che verrà aggiunto come suffisso al nome dello schema predefinito.

```
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
```

In base a quanto scritto sopra, se lo schema di destinazione definito in `profiles.yml` è **`mkt_analytics`**, lo schema usato per le tabelle `marketo_source` e `salesforce_source` sarà **`mkt_analytics_your_staging_name`**.

Disabilitare le tabelle mancanti

A questo punto, puoi eseguire i pacchetti di modelli Fivetran per verificare che funzionino correttamente.

```
dbt run -select marketo_source
```

```
dbt run -select marketo_source
```

Se uno dei modelli fallisce a causa di tabelle mancanti, perché hai scelto di non sincronizzare quelle tabelle in Fivetran, puoi disabilitare quei modelli nello schema d'origine aggiornando il file **`dbt_project.yml`**.

Ad esempio, se le tabelle "email bounced" e "email template" mancano dallo schema d'origine di Marketo, puoi disabilitare i relativi modelli aggiungendo nella configurazione dei modelli quanto segue:

```
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
```

Sviluppare i modelli per le analisi di marketing

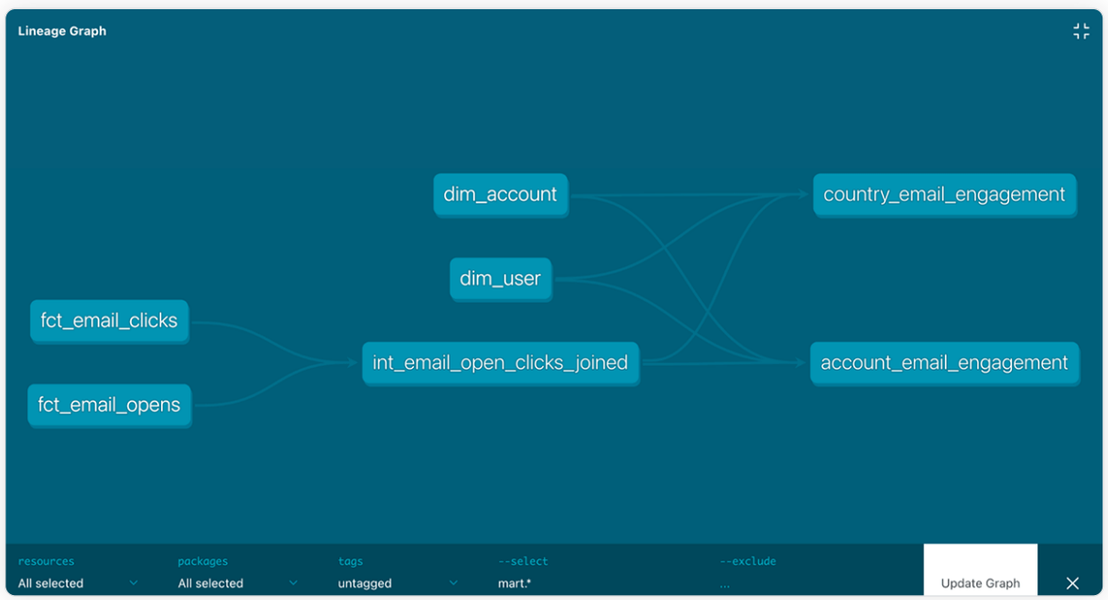


Grafico di derivazione dbt che mostra lo schema a stella e il modello di dati con tabelle aggregate.

Ora che i pacchetti Fivetran hanno creato e testato i modelli di staging, puoi cominciare a sviluppare i modelli di dati per i tuoi casi d'uso di analisi di marketing. Tali modelli saranno composti da uno schema a stella con tabelle aggregate materializzate.

Per la prima dashboard potresti decidere di valutare, ad esempio, l'engagement di determinate aziende e aree di vendita in base al numero di campagne di e-mail aperte e cliccate.

A tal fine, puoi unire le tabelle Salesforce e Marketo usando e-mail utente e `account_id` di Salesforce e `lead_id` di Marketo.

I modelli saranno strutturati nella cartella mart nel seguente modo:

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

Puoi visualizzare il codice per tutti i modelli su Github, nella directory /models/mart, dove troverai anche descrizioni del contenuto di ogni cartella insieme a un esempio.

Modelli fondamentali

I modelli fondamentali sono le tabelle di fatti e dimensioni che verranno usate come base da tutti i modelli a valle.

Questo è il codice SQL di dbt per il modello `dim_user`:

```
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
```

Puoi anche aggiungere documentazione e test per i modelli usando un file yaml nella cartella.

Qui abbiamo aggiunto 2 semplici test nel file `core.yml` :

```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
```

Modelli intermedi

Alcuni dei modelli finali a valle potrebbero fare affidamento sulle stesse metriche calcolate. Di conseguenza, per evitare di ripetere comandi SQL, puoi creare dei modelli intermedi riutilizzabili.

Questo è il codice SQL di dbt per il modello `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
```

Modelli per analisi di marketing

Sono i modelli per le analisi di marketing finali su cui si baseranno le dashboard e gli strumenti di reportistica usati dai team di marketing e vendite.

Questo è il codice SQL di dbt per il modello `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
```

Esecuzione e test dei modelli dbt

Ora che il modello è pronto, puoi eseguire tutti i modelli usando:

```
dbt run
```

Esegui quindi i test usando:

```
dbt test
```

Visualizzare la documentazione dbt e il grafico di derivazione

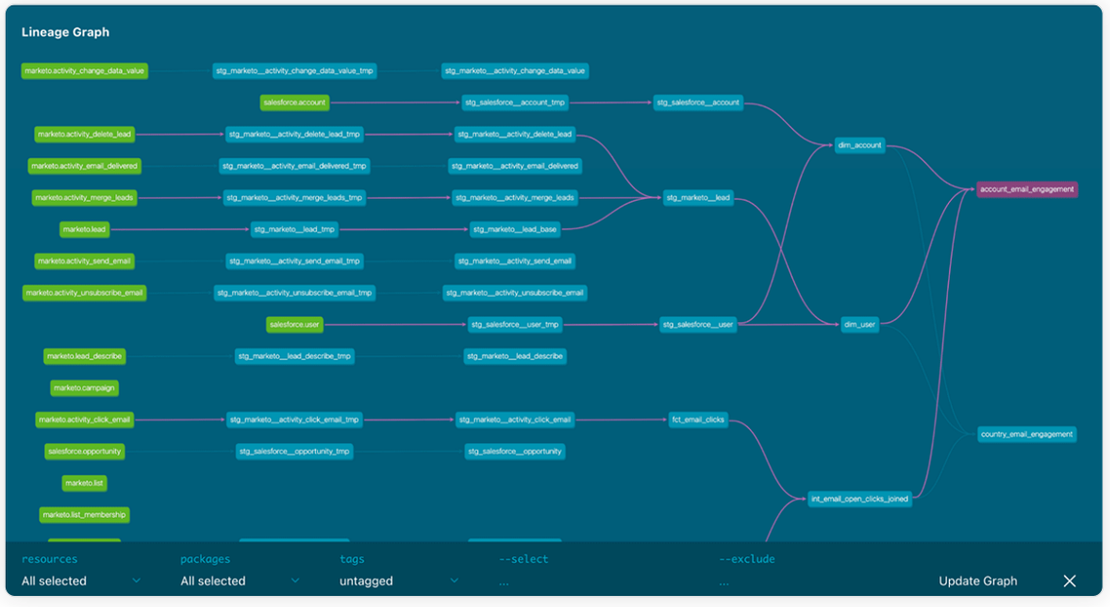


Grafico di derivazione dbt dei modelli per analisi di marketing.

Una volta eseguiti i modelli, puoi generare la documentazione e il grafico di derivazione usando:

```
$ dbt docs generate
```

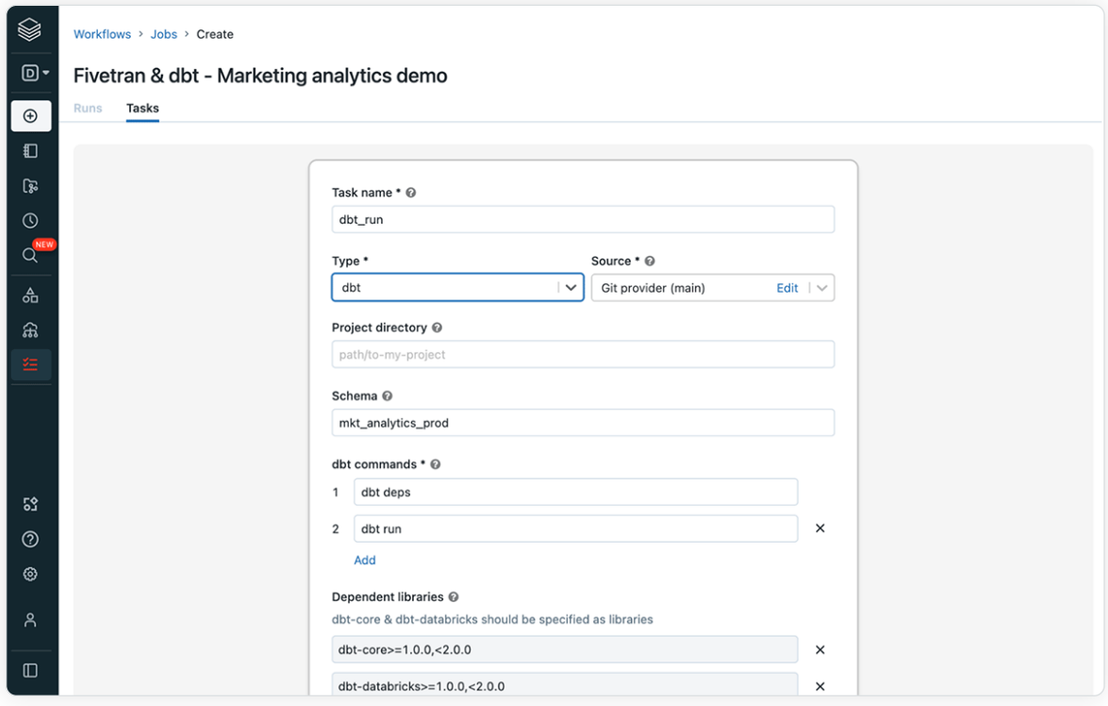
Quindi, per visualizzarli localmente, esegui:

```
$ dbt docs serve
```

Implementare i modelli dbt in produzione

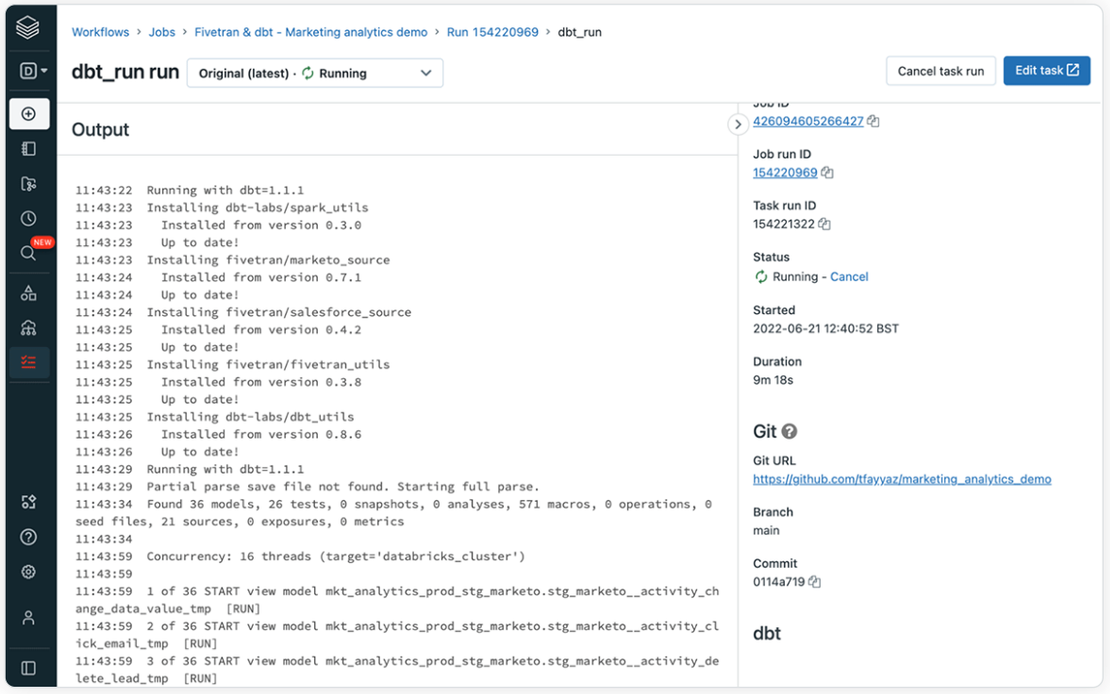
Una volta sviluppato e testato il modello dbt a livello locale ci sono diverse opzioni per implementarlo in produzione. Una di queste è il nuovo tipo di attività dbt in Databricks Workflows (in anteprima privata).

La gestione del progetto dbt e il controllo della versione dovrebbero avvenire in un repository Git. Puoi creare nel job Databricks Workflows un'attività dbt che punti al repository Git.



Utilizzo di un tipo di attività dbt in Databricks Workflows per orchestrare dbt.

Quando usi dei pacchetti nel progetto dbt, il primo comando dovrebbe essere dbt deps, seguito da dbt run per la prima attività e dbt test per la successiva.



Visualizzazione dei registri dbt per ogni esecuzione di dbt.

A ogni esecuzione, puoi vedere i registri di ciascun comando dbt eseguito per procedere al debugging e risolvere eventuali problemi.

Analisi di marketing basate su Fivetran e dbt

Come abbiamo visto, usare Fivetran e dbt unitamente alla Databricks Data Intelligence Platform permette di costruire una potente soluzione per analisi di marketing facile da configurare e da gestire e così flessibile da adattarsi a qualsiasi requisito di modellazione dei dati.

Per iniziare a costruire una soluzione, consulta la documentazione per l'integrazione di Fivetran e dbt con Databricks e riutilizza l'esempio di progetto marketing_analytics_demo per partire subito con il piede giusto.

Il tipo di attività dbt in Databricks Workflows è in anteprima privata. Se desideri provarlo, contatta il tuo referente locale Databricks.

CAPITOLO 4.2

Automatizzazione dei claim su Databricks

Smart Claims aumenta l'efficienza automatizzando tutti gli aspetti dell'elaborazione dei claim: dall'acquisizione, all'analisi fino ai processi decisionali.

di Anindita Mahapatra e Marzi Rasooli

Secondo gli ultimi report della società di consulenza globale EY, il settore assicurativo del futuro sarà sempre più **guidato dai dati** e **basato sull'analisi**. La recente diffusione del cloud ha migliorato l'accesso a un'infrastruttura tecnologica avanzata, ma la maggior parte delle organizzazioni ha ancora bisogno di aiuto per implementarne e sfruttarne le funzionalità. Per creare valore, è il momento di spostare l'attenzione sull'operazionalizzazione dei servizi.

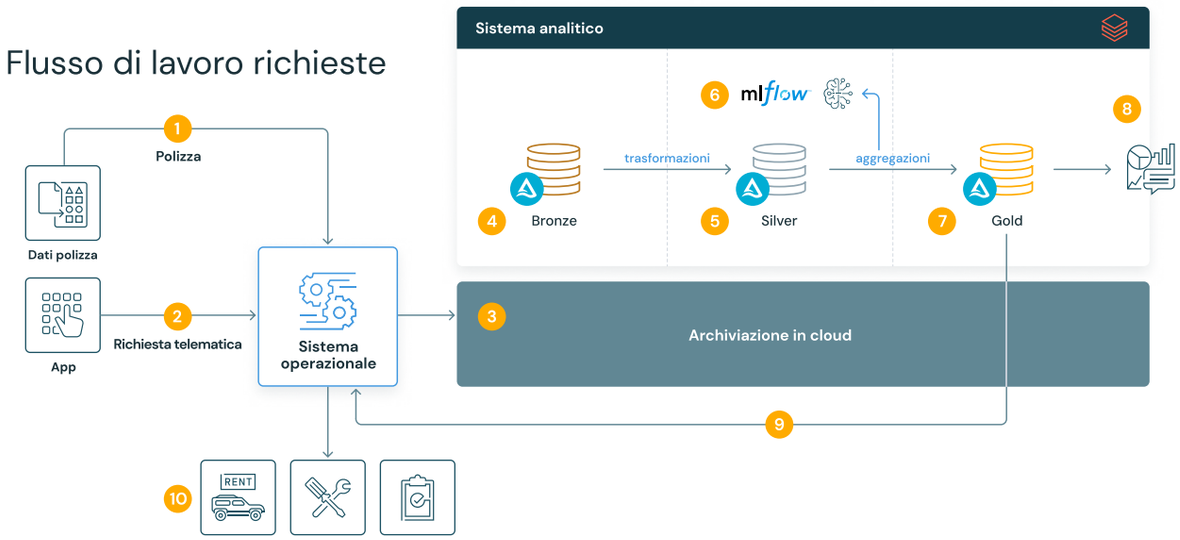
Nello scenario economico attuale, le sfide che le compagnie assicurative devono affrontare sono in continuo aumento e gli assicuratori devono a sfruttare a proprio vantaggio i dati e accelerare **l'innovazione**. Per gli assicuratori che operano nel ramo danni e infortuni, questo significa un'attenzione maggiore alla **personalizzazione** dei servizi e alla fidelizzazione dei clienti. La fedeltà a un marchio è ai minimi storici; i clienti sono costantemente alla ricerca di tariffe più vantaggiose ed esperienze complessivamente migliori, il che aumenta il tasso di abbandono. L'incremento nel numero di richieste di indennizzo fraudolente erode ulteriormente i margini di profitto, per cui gli assicuratori devono trovare nuovi modi per ridurre i costi e migliorare la gestione dei rischi.

Automatizzare e ottimizzare il processo di gestione dei claim può ridurre i costi in maniera significativa facendo risparmiare tempo e diminuendo la dipendenza dal capitale umano. Inoltre, utilizzare in modo efficace le informazioni ricavate da dati e analisi avanzate riduce considerevolmente l'esposizione al rischio complessiva.

La motivazione alla base dell'acceleratore di soluzioni Smart Claims è semplice: utilizzare il lakehouse per migliorare il processo di gestione delle richieste di risarcimento così da renderne più veloce la liquidazione, diminuire i costi di elaborazione e rendere subito disponibili le informazioni relative a richieste potenzialmente fraudolente. Implementare il paradigma lakehouse semplifica l'architettura attuale e prepara il campo a future espansioni dell'organizzazione. Le risorse correlate sono consultabili qui.

Architettura di riferimento e flusso di lavoro

Il flusso di lavoro tipico di una richiesta di risarcimento prevede un certo livello di orchestrazione tra sistemi operazionali come Guidewire e sistemi analitici come Databricks. Il diagramma sottostante mostra un esempio di flusso di lavoro per un assicuratore nel ramo auto.



Smart Claims: architettura di riferimento e flusso di lavoro.

Automatizzare e ottimizzare il processo di gestione delle richieste di risarcimento richiede una profonda comprensione dell'interazione dei clienti con i sistemi operazionali e delle varie fonti di informazioni disponibili per l'analisi.

In questo esempio, partiamo dal presupposto che i clienti interagiscano prevalentemente tramite un'app mobile che usano per inviare le richieste di risarcimento e monitorare lo stato di pratiche esistenti. Questo punto di contatto offre informazioni essenziali sul loro comportamento. Un'altra importante fonte sono i dispositivi IoT installati sui veicoli dei clienti. I dati telematici sono inviati in streaming ai sistemi operazionali e analitici, fornendo dati preziosi sul comportamento e i modelli di guida dei clienti. Altre sorgenti esterne possono includere dati relativi al meteo e alle condizioni della strada che integrano le tradizionali categorie di dati quali caratteristiche del veicolo (marca, modello, anno), caratteristiche del conducente ed esposizione/copertura (limiti, franchigie).

L'accesso a sorgenti di dati aggiuntive diventa sempre più importante, soprattutto in assenza di dati provenienti da fonti tradizionali come le agenzie di credito. L'affidabilità creditizia è in genere alla base della modellazione del rischio e viene usata per stimare l'esposizione del conducente, che influisce sulla quantificazione del suo premio assicurativo. I dati provenienti da app mobili e dispositivi IoT, d'altro canto, forniscono una visione più personalizzata del comportamento dei clienti, che si può utilizzare per creare un indicatore di rischio più accurato. Questo approccio alternativo alla modellazione e quantificazione del rischio **basato sui comportamenti** è essenziale per offrire un'esperienza cliente iper personalizzata.

L'architettura lakehouse su Databricks Data Intelligence Platform è l'unica piattaforma che combina tutte le funzionalità e i servizi necessari per un processo di gestione delle richieste di risarcimento a prova di futuro. Dallo streaming al machine learning e al reporting, Databricks offre la migliore piattaforma per costruire una soluzione completa per il settore assicurativo di domani.

Il processo si articola nelle seguenti fasi:

- I dati della polizza vengono acquisiti.
- I dati telematici vengono continuamente acquisiti dai sensori IoT. I dati della richiesta di risarcimento vengono inviati tramite un'app mobile.
- Tutti i dati operazionali vengono acquisiti nello storage in cloud.
- Questi vengono caricati incrementalmente come "dati grezzi" nelle tabelle Delta di livello Bronze.
- I dati grezzi vengono sottoposti a wrangling e affinati tramite varie trasformazioni.
- Ai dati viene assegnato un punteggio usando il modello addestrato.
- Le previsioni vengono caricate in una tabella di livello Gold.
- La Claims Dashboard viene aggiornata per la visualizzazione
- Le informazioni risultanti vengono immesse nuovamente nel sistema operazionale. Si attiva così un circuito di retroazione che consiste nell'estrarre dati da Guidewire e nel rimandarli poi a Guidewire stesso per stabilire in tempo reale la "migliore azione successiva" e sapere a quali richieste dare la priorità.
- I flussi di lavoro decisionali (Claims Decisioning) usano le informazioni generate per instradare correttamente la pratica. (Ad esempio, approvare le spese di riparazione, rimborsare il noleggio o allertare le autorità.)

Come il paradigma lakehouse aiuta Smart Claims

L'architettura lakehouse consente a tutti coloro che lavorano con i dati (data engineer, data scientists, analytics engineer e BI analysts) di collaborare su una singola piattaforma. Il supporto di tutti i carichi di lavoro e paradigmi big data (elaborazione in batch, streaming, DataOps, MLOps e BI) su una singola piattaforma collaborativa semplifica enormemente l'architettura complessiva, aumenta la stabilità e abbatte i costi.

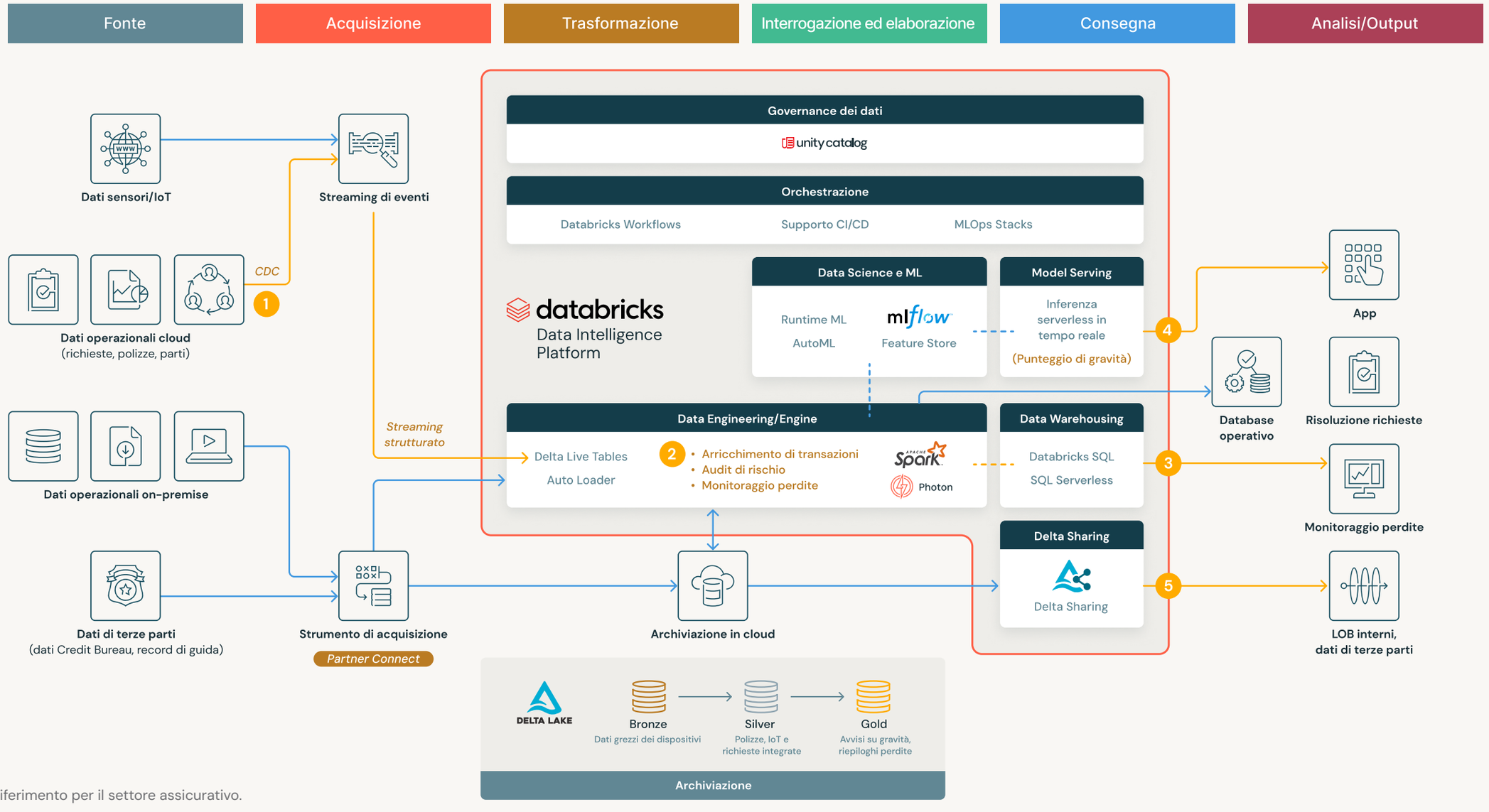
Le pipeline Databricks Delta Live Tables (DLT) offrono un semplice framework dichiarativo per sviluppare e implementare i carichi di lavoro in modo rapido. Forniscono inoltre supporto nativo per la gestione della data quality con vincoli granulari per garantire l'integrità degli output.

I carichi di lavoro di ML e AI possono essere facilmente creati e gestiti con MLflow, garantendo riproducibilità e verificabilità. MLFlow semplifica l'intero ciclo di vita del modello: sperimentazione, implementazione, distribuzione e archiviazione. Il ML può essere eseguito su tutti i tipi di dati, inclusi quelli non strutturati in formato diverso dal testo (immagini, audio, video, ecc.) In questo esempio useremo delle funzionalità di visione artificiale per stimare i danni al veicolo.

Infine, Databricks SQL offre un motore rapido ed efficiente per interrogare dati curati e aggregati. Queste informazioni possono quindi essere accorpate e visualizzate tramite dashboard interattive nel giro di qualche minuto.

Unity Catalog offre una soluzione di governance multi-cloud centralizzata per tutte le risorse di dati e AI, come tabelle, modelli di machine learning e dashboard, e include funzioni integrate di ricerca, scoperta e lineage automatizzato dei carichi di lavoro.

Il seguente diagramma mostra un'architettura di riferimento per il lakehouse nel contesto dei tipici casi d'uso nel settore assicurativo:



Architettura di riferimento per il settore assicurativo.

Acquisizione di dati con DLT e Multitask Workflows

L'automatizzazione del processo di gestione delle richieste di risarcimento inizia con l'ottimizzazione del flusso di lavoro di acquisizione e data engineering. La figura qui sotto mostra un riepilogo delle tipiche sorgenti di dati, strutturate, semi-strutturate e non strutturate, che si incontrano in questo ambito. Alcune sorgenti cambiano lentamente, altre si aggiornano più rapidamente. Alcune sorgenti, inoltre, potrebbero essere additive e richiedere l'aggiunta di dati (appending), mentre altre offrono aggiornamenti incrementali e devono essere trattate come SCD (Slowly Changing Dimension).

Volume	Velocità	Varietà
Alto • Dati telematici	**In tempo quasi reale** • Dati telematici	**Strutturati** • Polizza, intestatario polizza, telematici
Medio • Dati richiesta	**In streaming** • Dati richiesta, dati incidente	**Semi-strutturati** • Dati richiesta
Basso • Polizza, dati intestatario polizza • Dati dell'incidente	**In batch** • Polizza, dati intestatario polizza	**Non strutturati** • Dati incidente • Immagini, video e testi

Esempi di set di dati usati nell'elaborazione delle richieste di risarcimento.

DLT può semplificare e operazionalizzare la pipeline di elaborazione dei dati. Il framework offre supporto per Auto Loader per facilitare l'acquisizione dei dati dalle sorgenti in streaming, autoscalabilità efficiente per gestire repentini cambiamenti nei volumi di dati e resilienza grazie al riavvio delle attività non andate a buon fine.

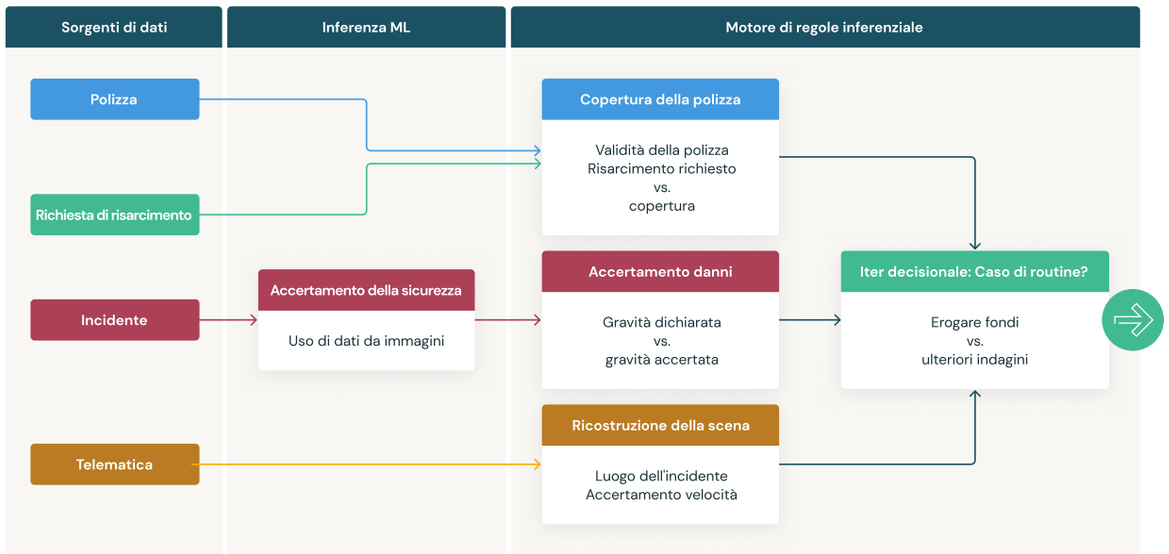
Databricks Workflows può gestire molteplici attività e carichi di lavoro (come notebook, DLT, ML, SQL). Workflows supporta funzionalità "ripara ed esegui" e di condivisione delle risorse di calcolo tra più attività per carichi di lavoro robusti, scalabili ed economici. Inoltre, può essere facilmente automatizzato tramite pianificazioni o chiamate programmatiche via API REST.

Generare informazioni usando ML e un motore di regole dinamiche

L'uso del ML è essenziale per scoprire pattern precedentemente non rilevati, portare alla luce nuove informazioni e segnalare attività sospette. Tuttavia, è ancora più utile combinare ML e approcci tradizionali basati su regole.

Nell'ambito del processo di gestione delle richieste di risarcimento, il ML può essere utilizzato per diversi casi d'uso. Un esempio è l'impiego della visione artificiale e del ML per esaminare e valutare le immagini inviate insieme alle richieste di risarcimento per sinistri stradali. I modelli possono essere addestrati per concentrarsi sulla validità e la gravità dei danni. Qui, MLFlow gioca un ruolo cruciale nel semplificare l'addestramento e il serving del modello con le sue funzionalità MLOps complete. MLFlow offre un serving del modello serverless tramite API REST. I modelli addestrati possono essere operazionalizzati e messi in produzione con un semplice clic.

I motori di regole, dal canto loro, offrono una soluzione flessibile per definire categorie operazionali e controlli statistici noti, che possono essere automatizzati e applicati senza intervento umano. Ogni volta che i dati non sono conformi alle aspettative, l'anomalia viene segnalata e i dati vengono sottoposti a revisione e indagine umana. Incorporare un simile approccio in flussi di lavoro basati su ML offre una misura aggiuntiva di supervisione e riduce in maniera significativa il tempo necessario ai periti assicurativi per analizzare e rivedere i casi segnalati.



ML e motore di regole inferenziale.

Visualizzazione delle informazioni sulle dashboard

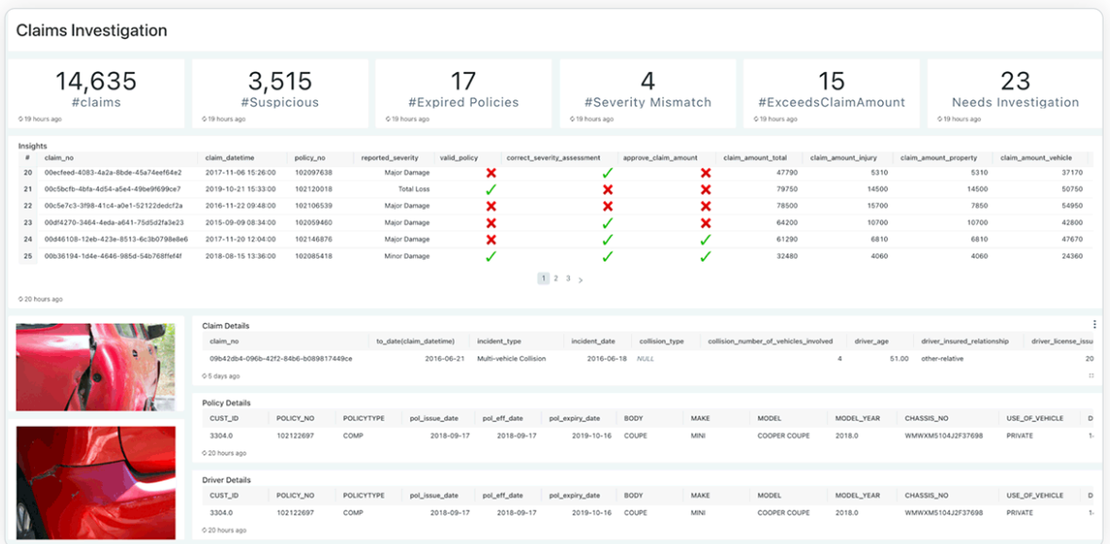
In questo esempio, abbiamo creato due dashboard per ottenere informazioni strategiche per l'azienda. Le dashboard includono:

- Una dashboard **Loss Summary** (Riepilogo delle perdite) per una vista ad alto livello delle attività operative aziendali nel loro complesso.
- Una dashboard **Claims Investigation** (Indagini sulle richieste di risarcimento), con una vista granulare dei dettagli delle richieste per conoscere i particolari di un caso specifico.

L'analisi delle tendenze recenti può dare un contributo nella revisione di casi simili, come:

- Il tasso di perdita è calcolato dividendo per la somma dei premi il numero di richieste di risarcimento pagate (comprensive di spese di perizia). Ad esempio, il tasso di perdita tipico (combinando tutte le coperture, lesioni personali e danni fisici) di un'automobile privata dovrebbe attestarsi intorno al 65%.
- Il riepilogo mostra il numero di tipi di incidente per gravità dei danni.
- Linee di tendenza su varie caratteristiche/dimensioni.
- Distribuzione geografica delle polizze.

La dashboard di Claims Investigation facilita e velocizza l'indagine fornendo tutte le informazioni relative a una richiesta di risarcimento. Il perito può esaminare in dettaglio una specifica richiesta e vedere le immagini del veicolo danneggiato, i dettagli della richiesta, della polizza o del conducente. I dati telematici tracciano il percorso del veicolo e i dati dichiarati vengono confrontati con le informazioni ricavate dalla valutazione.



Dashboard Claim Investigation

Fornisce inoltre richieste di risarcimento recenti che vengono autovalutate nella pipeline usando inferenze ML e un motore di regole:

- Un segno di spunta verde indica che l'autovalutazione corrisponde alla descrizione del sinistro.
- Una x rossa indica che c'è un'incongruenza che necessita di ulteriori indagini manuali.

Conclusioni

Innovazione e personalizzazione sono indispensabili per consentire alle compagnie assicurative di distinguersi dalla concorrenza. Databricks offre agli assicuratori una piattaforma di Data Intelligence per favorire e accelerare l'innovazione con un'architettura aperta, sicura e scalabile che si integra facilmente con strumenti e servizi di terze parti. Questo acceleratore di soluzioni mostra come il paradigma possa essere applicato alla gestione delle richieste di risarcimento. Inoltre, l'ecosistema Databricks fornisce un'ampia gamma di funzionalità per consentire ai data team e agli stakeholder aziendali di collaborare, generare e condividere informazioni a supporto dei processi decisionali per generare un valore tangibile in termini di redditività.

Le risorse tecniche, incluse configurazioni di pipeline, modelli e dati campione usati in questo esempio sono disponibili qui o direttamente su Git.

CAPITOLO 4.3

Schemi di progettazione per l'elaborazione in batch nel settore dei servizi finanziari

Gettare le basi per flussi di lavoro automatizzati

di Eon Retief

Introduzione

Tra volatilità dei mercati, instabilità politica e cambiamenti nella legislazione e nelle normative, gli istituti di servizi finanziari (FIS) di tutto il mondo si trovano ad affrontare sfide senza precedenti. Le aziende sono costrette ad accelerare i loro programmi di trasformazione digitale, automatizzando i processi fondamentali per ridurre i costi operativi e migliorare i tempi di risposta. Tuttavia, con dati generalmente dispersi su più sistemi, accedere alle informazioni necessarie per attuare queste iniziative è più facile a dirsi che a farsi.

Progettare un ecosistema di servizi in grado di supportare la grande varietà di casi d'uso basati sui dati in un contesto aziendale improntato al digitale può sembrare un'impresa impossibile. Questa sezione si concentrerà su un aspetto cruciale del data stack moderno: l'elaborazione in batch. Benché apparentemente obsoleto, questo paradigma resta un componente vitale ed estremamente efficiente dell'architettura di dati. Vedremo inoltre come Databricks possa aiutare le istituzioni finanziarie a superare alcune delle difficoltà legate alla costruzione di un'infrastruttura per supportare questi flussi di lavoro programmati o periodici.

L'importanza dell'acquisizione in batch

Negli ultimi due decenni, la transizione globale verso una società istantanea ha costretto le organizzazioni a ripensare i propri modelli operativi e di engagement. Una strategia orientata al digitale non è un'opzione, bensì una scelta essenziale per la sopravvivenza. Le esigenze e le richieste dei clienti cambiano ed evolvono più rapidamente che mai. Questo desiderio di gratificazione istantanea ha fatto aumentare l'interesse per funzionalità che supportino elaborazione e decisioni in tempo reale. Ci si potrebbe chiedere se l'elaborazione in batch sia ancora rilevante in questo nuovo mondo così dinamico.

Se i sistemi in tempo reale e i servizi di streaming possono aiutare le istituzioni finanziarie ad affrontare agilmente la volatilità dei mercati, di solito non soddisfano i requisiti delle funzioni di back-office. La maggior parte delle decisioni aziendali non è reattiva, ma richiede anzi un ragionamento ponderato e strategico. Per definizione, questo approccio necessita di una revisione sistematica dei dati aggregati raccolti nel corso del tempo. In questo contesto, l'elaborazione in batch rappresenta ancora il metodo più efficiente ed economico per elaborare grandi volumi di dati aggregati. Inoltre, l'elaborazione in batch può essere eseguita offline, riducendo i costi operativi e offrendo un maggiore controllo sull'intero processo.

Il mondo della finanza sta cambiando, ma tanto gli attori consolidati quanto le startup continuano a fare ampio affidamento sull'elaborazione in batch per alimentare le principali attività dell'azienda. Che si tratti di generazione di report e gestione del rischio o di rilevazione di anomalie e sorveglianza, le istituzioni finanziarie hanno bisogno dell'elaborazione in batch per ridurre gli errori umani, aumentare la velocità di consegna e ridurre i costi operativi.

Per cominciare

Se guardiamo al quadro generale, la maggior parte delle istituzioni finanziarie avrà una moltitudine di sorgenti di dati disseminate tra sistemi on-premise, servizi in cloud e persino applicazioni di terze parti. La creazione di un framework di acquisizione in batch che soddisfi tutte queste connessioni richiede un'ingegnerizzazione complessa e può diventare rapidamente un onere per i team di manutenzione. Le cose si complicano ulteriormente se consideriamo aspetti come l'acquisizione dei dati modificati (CDC), la pianificazione e l'evoluzione degli schemi. In questa sezione, dimostreremo come l'architettura lakehouse per i servizi finanziari e il suo ecosistema di partner possano essere utilizzati per affrontare queste sfide chiave e semplificare notevolmente l'architettura complessiva.

L'architettura lakehouse è progettata per fornire una piattaforma unificata che supporta tutti i carichi di lavoro analitici e scientifici sui dati. La Figura 1 mostra l'architettura di riferimento per un design svincolato e facile da integrare con altre piattaforme che supportano il moderno ecosistema di dati. Il lakehouse facilita la costruzione di livelli di acquisizione e di serving che operano indipendentemente dalla sorgente, dal volume, dalla velocità e dalla destinazione dei dati.

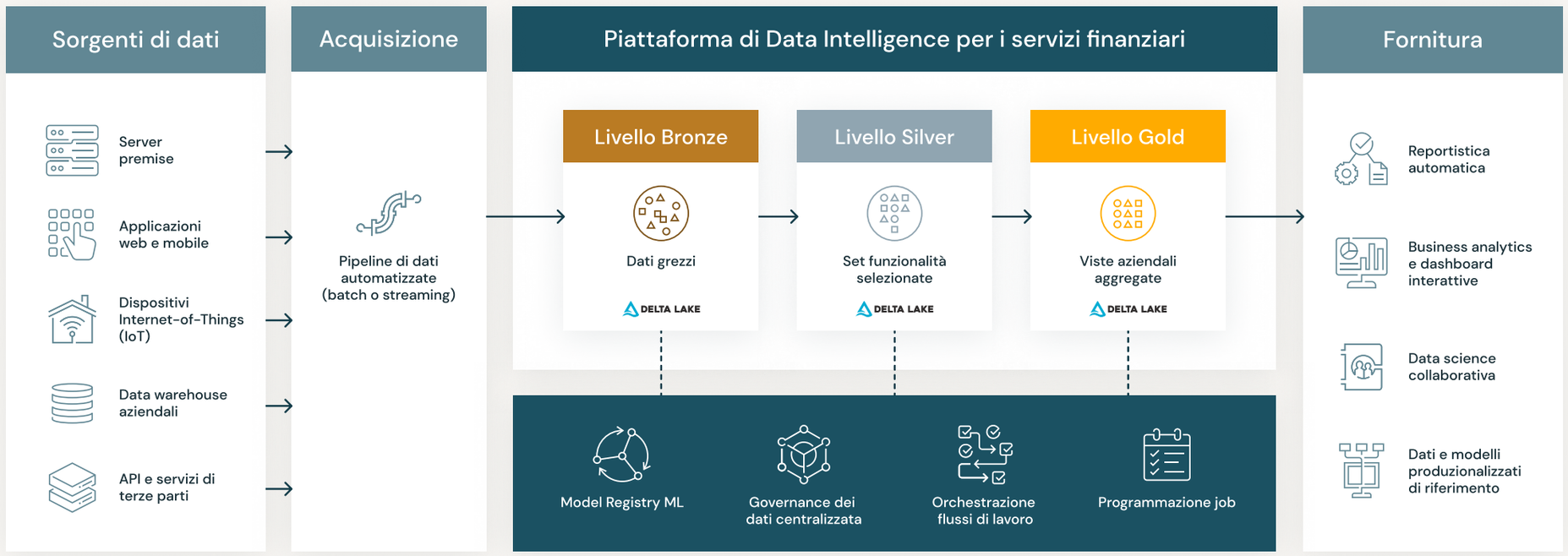


Figura 1. Architettura di riferimento del lakehouse per i servizi finanziari.

Per dimostrare la potenza e l'efficienza del lakehouse per i servizi finanziari (LFS), ci rivolgiamo al settore assicurativo. Consideriamo i requisiti di base per la reportistica in un tipico flusso di lavoro di richieste di risarcimento. In questo scenario, l'organizzazione potrebbe essere interessata alle metriche chiave guidate dai processi di gestione delle richieste di risarcimento. Ad esempio:

- Numero di polizze attive
- Numero di richieste di risarcimento
- Valore delle richieste di risarcimento
- Esposizione totale
- Tasso di perdita

Inoltre, l'azienda potrebbe desiderare una panoramica delle richieste di risarcimento potenzialmente sospette e una suddivisione delle richieste per tipo di incidente e gravità. Tutte queste metriche sono facilmente calcolabili sulla base di due sorgenti di dati chiave: 1) il registro delle polizze e 2) le richieste di risarcimento presentate dai clienti. I registri delle polizze e delle richieste di risarcimento sono generalmente archiviati in una combinazione di data warehouse aziendali (EDW) e database operativi. La sfida principale diventa connettersi a queste sorgenti e acquisire i dati nel lakehouse, dove si può sfruttare la potenza di Databricks per ottenere gli output desiderati.

Fortunatamente, il design flessibile del LFS rende facile sfruttare i prodotti di punta di un'ampia gamma di tecnologie e strumenti SaaS per gestire compiti specifici. Una possibile soluzione per il nostro caso d'uso di analisi delle richieste di risarcimento potrebbe essere utilizzare Fivetran per il livello di acquisizione in batch. Fivetran fornisce una piattaforma semplice e sicura per connettersi a numerose sorgenti di dati e consegnare i dati direttamente alla Databricks Data Intelligence Platform. Inoltre, offre supporto nativo per CDC, evoluzione degli schemi e pianificazione dei carichi di lavoro. La Figura 2 mostra l'architettura tecnica di una soluzione pratica per questo caso d'uso.

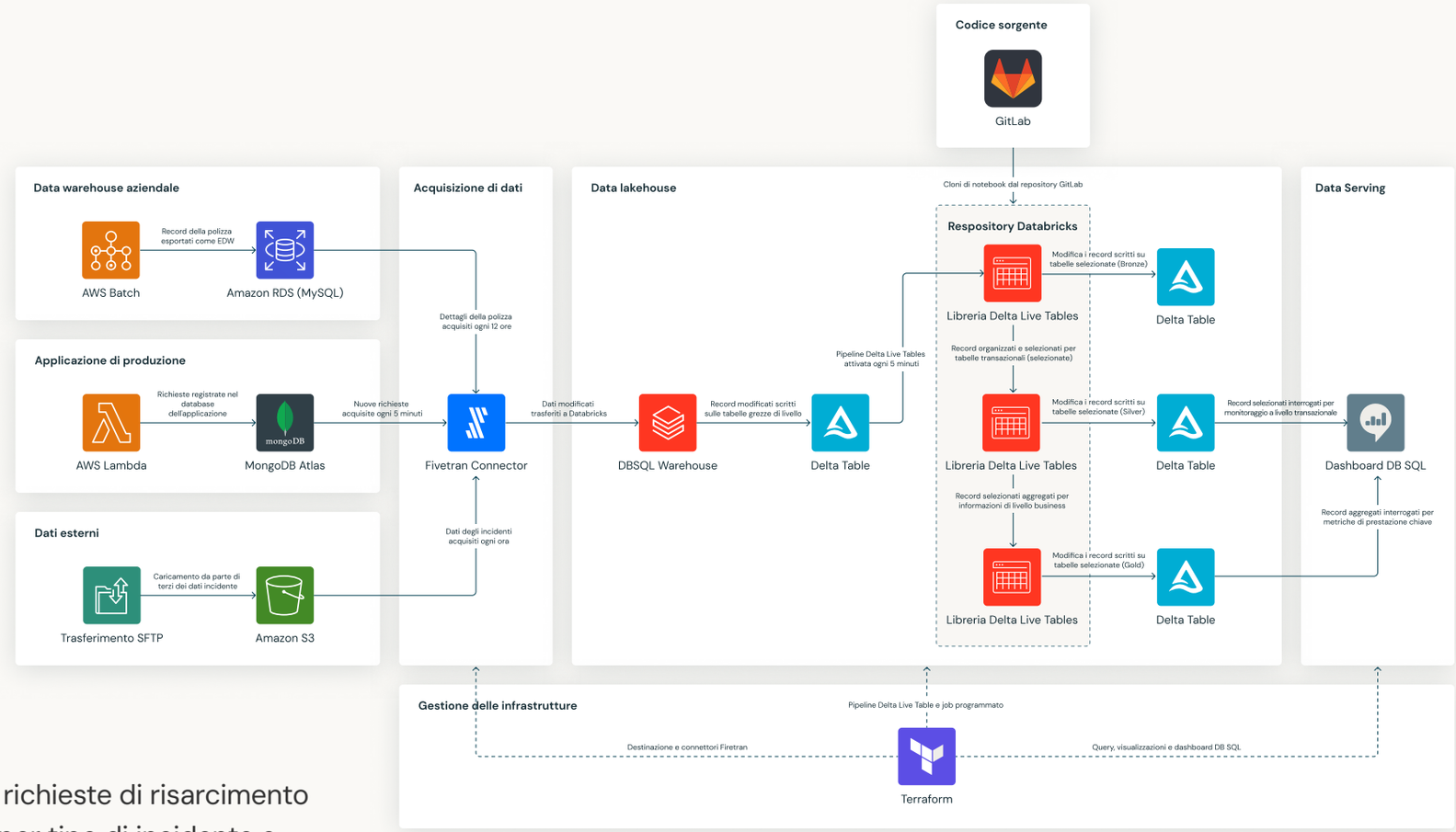


Figura 2. Architettura tecnica per un semplice flusso di lavoro delle richieste di risarcimento assicurativo.

Una volta che i dati sono acquisiti e consegnati al LFS, possiamo utilizzare Delta Live Tables (DLT) per l'intero flusso di lavoro di engineering. DLT fornisce un framework dichiarativo semplice e scalabile per automatizzare flussi di lavoro complessi ed effettuare controlli sulla qualità dei dati. Gli output dal flusso di lavoro DLT, le nostre risorse di dati curati e aggregati, possono essere interrogati utilizzando Databricks SQL (DB SQL). DB SQL porta il data warehousing sul LFS per alimentare carichi di lavoro analitici di importanza cruciale per l'azienda. I risultati delle query DB SQL possono essere confezionati in dashboard facili da consultare e messe a disposizione agli utenti aziendali.

Fase 1. Creare il livello di acquisizione

La creazione di un livello di acquisizione in Fivetran si articola in due passaggi. Il primo è la configurazione della cosiddetta destinazione in cui i dati verranno consegnati, il secondo è la creazione di una o più connessioni con i sistemi sorgente. Per il primo passaggio, utilizziamo la procedura guidata sull'interfaccia di Partner Connect per collegare Fivetran a un warehouse Databricks. Fivetran utilizzerà il warehouse per convertire i dati grezzi provenienti dalla sorgente in tabelle Delta e archiviare i risultati sulla Databricks Data Intelligence Platform. Le Figure 3 e 4 mostrano i passaggi da eseguire sulle interfacce di Partner Connect e Fivetran per configurare una nuova destinazione.

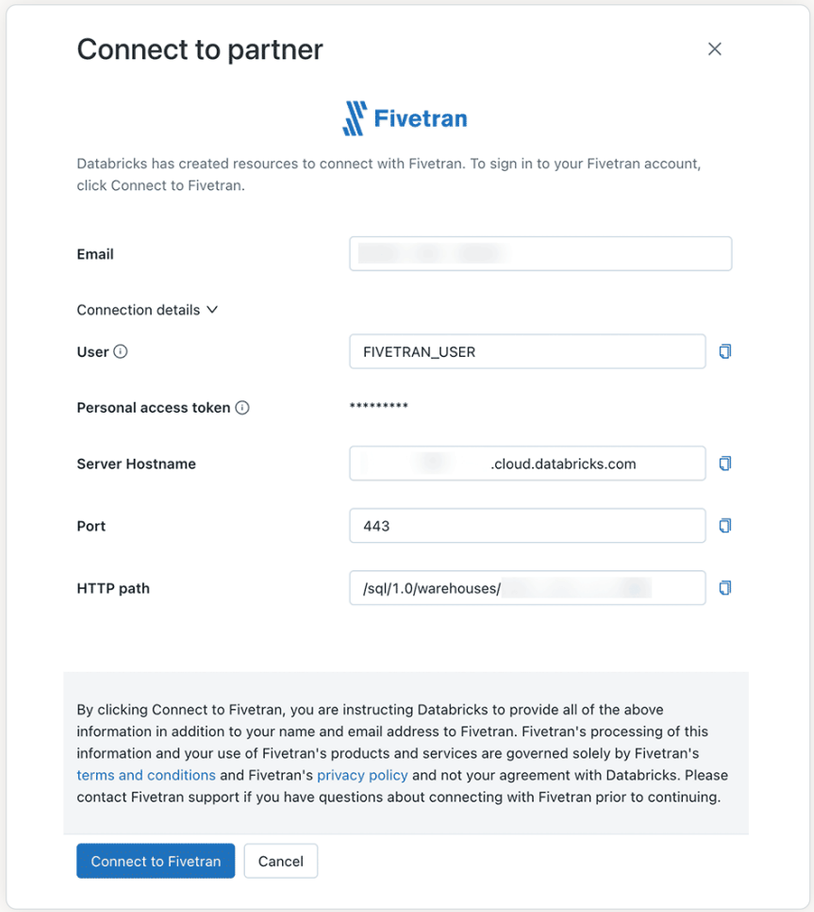


Figura 3. Interfaccia di Databricks Partner Connect per creare una nuova connessione.

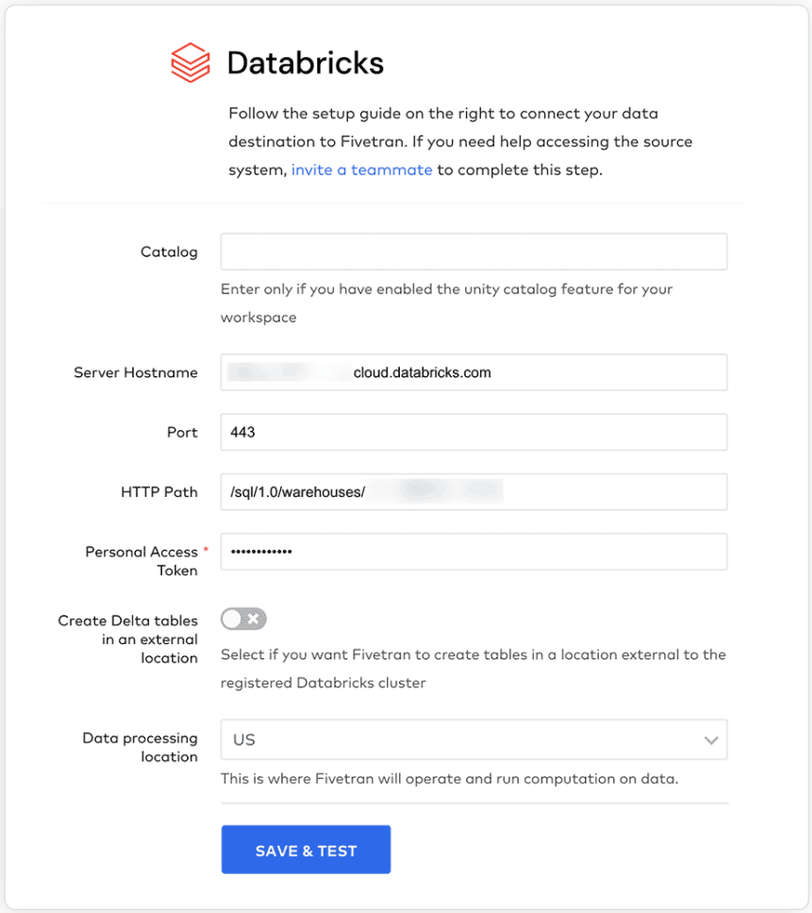


Figura 4. Interfaccia di Fivetran per confermare una nuova destinazione.

Per il prossimo passaggio, ci spostiamo sull'interfaccia di Fivetran. Da qui è possibile creare e configurare facilmente connessioni a diversi sistemi sorgente (consulta la documentazione ufficiale per un elenco completo di tutte le connessioni supportate). Nel nostro esempio, consideriamo tre sorgenti di dati: 1) I registri delle polizze archiviati in un archivio dati operazionale (ODS) o in un data warehouse aziendale (EDW), 2) i registri delle richieste di risarcimento archiviati in un database operativo e 3) i dati esterni consegnati a uno storage blob. Di conseguenza, avremo bisogno di configurare tre diverse connessioni in Fivetran. Per ognuna di queste, possiamo seguire il semplice processo guidato di Fivetran per configurare una connessione con il sistema sorgente. Le Figure 5 e 6 mostrano come configurare nuove connessioni a sorgenti di dati.

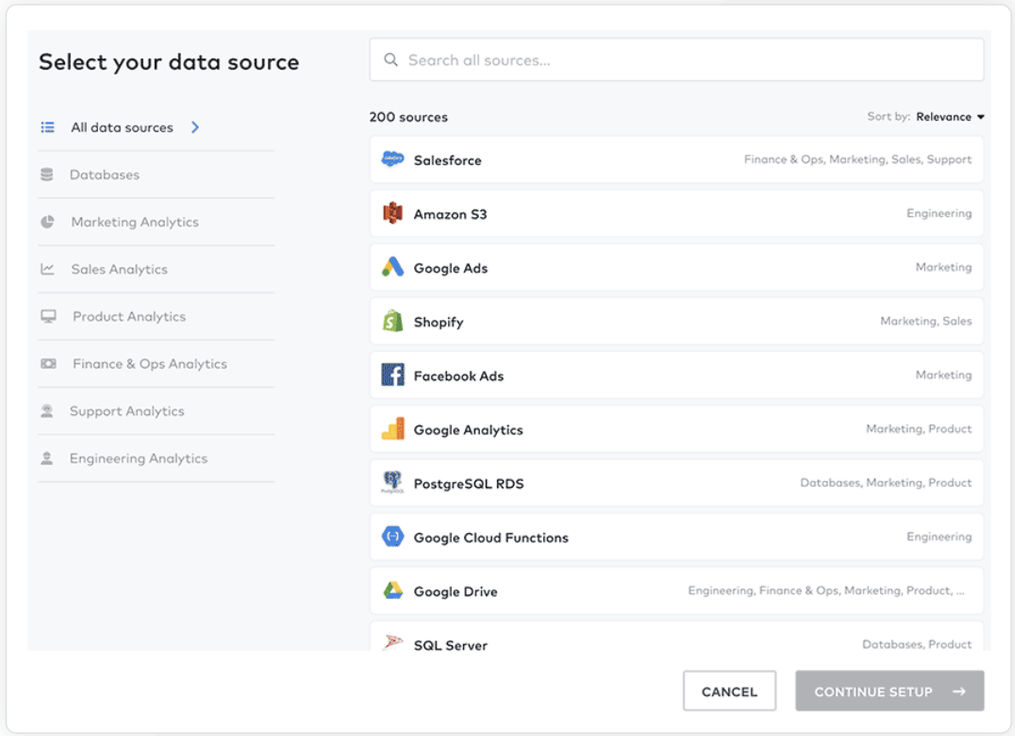


Figura 5. Interfaccia di Fivetran per la selezione di un tipo di sorgente di dati.

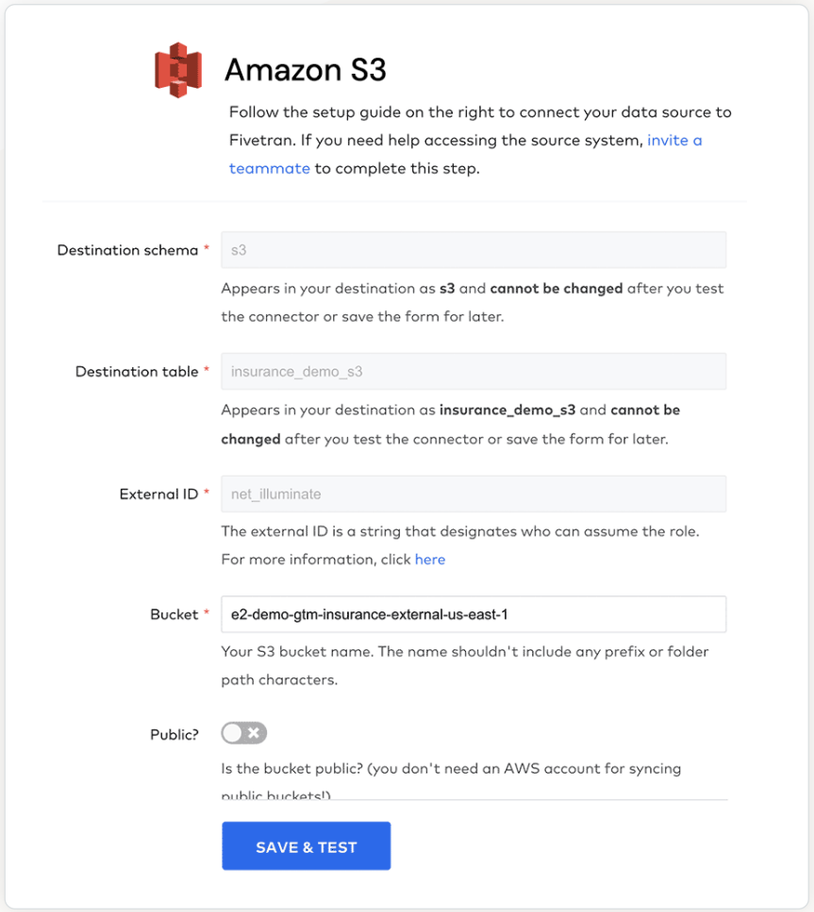


Figura 6. Interfaccia di Fivetran per la configurazione di una connessione a una sorgente di dati.

Le connessioni possono essere ulteriormente configurate una volta che sono state validate. Un'opzione importante da impostare è la frequenza con cui Fivetran interrogherà il sistema sorgente per i nuovi dati. Nella Figura 7 possiamo vedere quanto sia facile impostare la frequenza di sincronizzazione in Fivetran, con intervalli che vanno da 5 minuti a 24 ore.

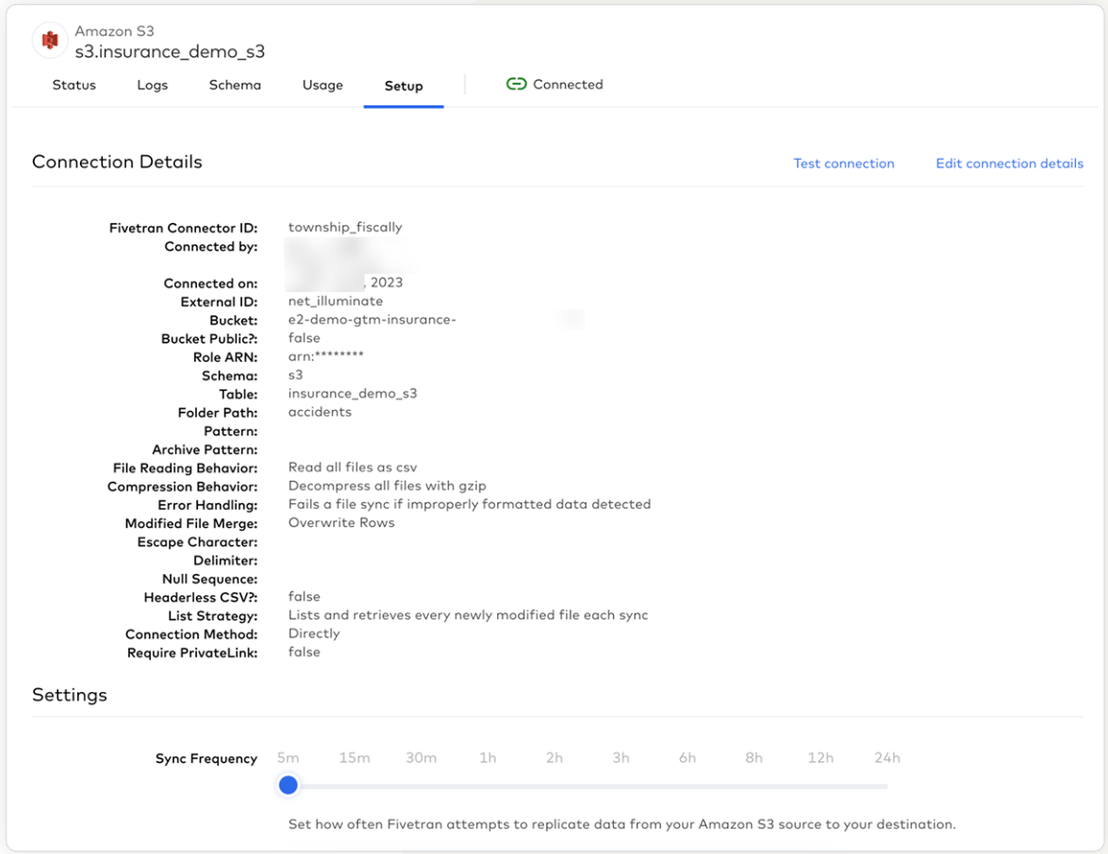


Figura 7. Panoramica della configurazione di un connettore Fivetran.

Una volta convalidata una connessione, Fivetran interrogherà immediatamente i sistemi sorgente e ne acquisirà i dati. I dati vengono archiviati come tabelle Delta e possono essere visualizzati in Databricks tramite il Catalog Explorer. Per impostazione predefinita, Fivetran archivierà tutti i dati nel metastore Hive. Per ogni nuova connessione viene creato un nuovo schema, che conterrà almeno due tabelle: una con i dati e un'altra con i registri di ogni ciclo di acquisizione tentato (vedi Figura 8).

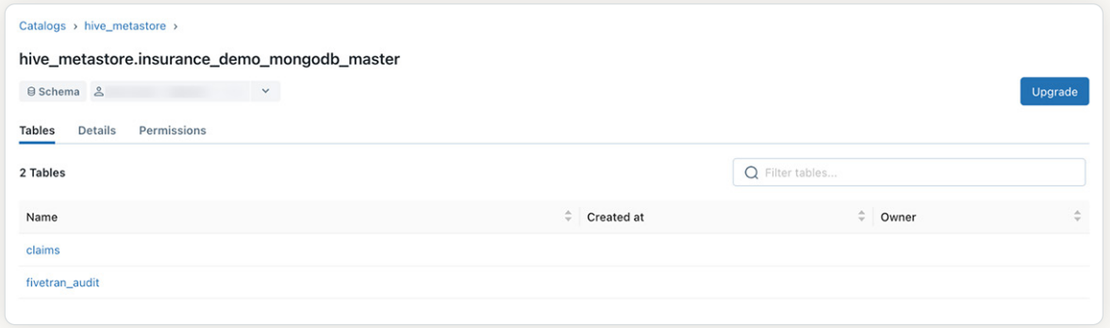


Figura 8. Riepilogo delle tabelle create da Fivetran nel Databricks Warehouse per una connessione di esempio.

Avere i dati archiviati in tabelle Delta rappresenta un significativo vantaggio. Delta Lake supporta nativamente il controllo granulare delle versioni dei dati, consentendo di viaggiare nel tempo attraverso ciascun ciclo di acquisizione (vedi Figura 9). Possiamo utilizzare DB SQL per interrogare versioni specifiche dei dati e analizzare come si sono evoluti i record di origine.

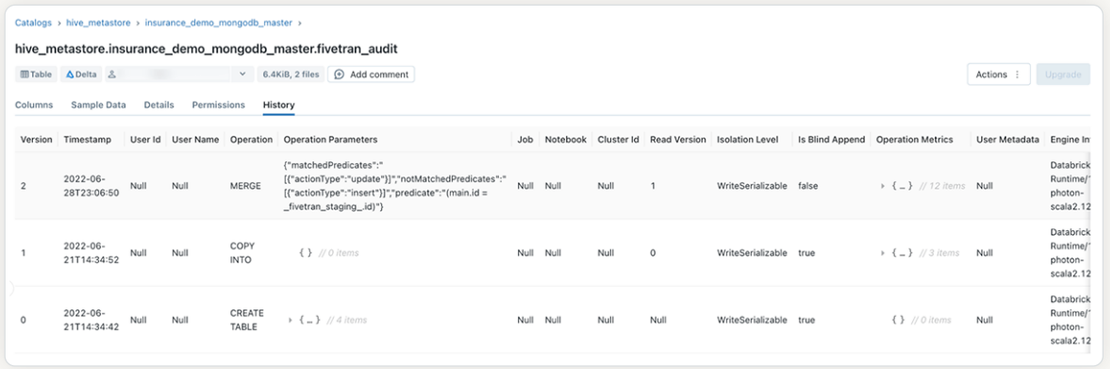


Figura 9. Vista della cronologia che mostra le modifiche apportate alla tabella di audit di Fivetran.

È importante notare che, se i dati di origine contengono valori semi-strutturati o non strutturati, quegli attributi verranno appiattiti durante il processo di conversione. Ciò significa che i risultati saranno memorizzati in colonne di tipo testo raggruppate e queste entità dovranno essere dissezionate e scompattate con DLT nel processo di cura per creare attributi separati.

Fase 2. Automatizzare il flusso di lavoro

Con i dati nel lakehouse, possiamo utilizzare Delta Live Tables (DLT) per costruire un flusso di lavoro di data engineering semplice e automatizzato. DLT fornisce un framework dichiarativo per specificare dettagliati passaggi di ingegnerizzazione delle feature. Attualmente, DLT supporta API sia per Python sia per SQL. In questo esempio, useremo API per Python per costruire il nostro flusso di lavoro.

Il costrutto fondamentale in DLT è la definizione di una tabella. DLT interroga tutte le definizioni delle tabelle per creare un flusso di lavoro completo su come i dati dovrebbero essere elaborati. In Python, ad esempio, le tabelle sono create usando definizioni di funzioni e il decorator "dlt.table" (vedi un esempio del codice Python qui sotto). Il decorator è usato per specificare il nome della tabella risultante, un commento descrittivo che spiega lo scopo della tabella e una raccolta di proprietà della tabella.

```python
@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)

   ...
```

Le istruzioni per l'ingegnerizzazione delle feature sono definite all'interno del corpo delle funzioni utilizzando API PySpark standard e comandi Python nativi. L'esempio seguente mostra come PySpark unisca i record delle richieste di risarcimento con i dati della tabella delle polizze per creare una singola vista curata delle richieste.

```
    ...

    # 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
```

Un vantaggio significativo di DLT è la possibilità di specificare e applicare standard di qualità ai dati. Possiamo impostare le aspettative per ogni tabella DLT con dettagliati vincoli di qualità dei dati da applicare ai contenuti della tabella. Attualmente, DLT supporta aspettative per tre diversi scenari:

Decorator	Descrizione
`expect`	Conserva i record che violano le aspettative
`expect_or_drop`	Elimina i record che violano le aspettative
`expect_or_fail`	Ferma l'esecuzione se uno o più record viola i vincoli

Le aspettative possono essere definite con uno o più vincoli di qualità dei dati. Ogni vincolo richiede una descrizione e un'espressione Python o SQL da valutare. È possibile definire più vincoli usando i decorator `expect_all`, `expect_all_or_drop` e `expect_all_or_fail`. Ogni decorator prevede un dizionario Python nel quale le chiavi sono le descrizioni dei vincoli e i valori sono le rispettive espressioni. L'esempio riportato di seguito mostra diversi vincoli di qualità dei dati per gli scenari di conservazione/eliminazione sopra descritti.

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

Possiamo utilizzare più di un Databricks Notebook per dichiarare le nostre tabelle DLT. Se adottiamo l'architettura medallion possiamo ad esempio utilizzare diversi notebook per definire le tabelle che compongono i livelli Bronze, Silver e Gold. Il framework DLT può assimilare istruzioni definite su più notebook per creare un singolo flusso di lavoro; tutte le dipendenze e le relazioni tra le tabelle vengono elaborate e considerate automaticamente. La Figura 10 mostra il flusso di lavoro completo per il nostro esempio sulle richieste di risarcimento. Partendo da tre tabelle sorgente, DLT costruisce una pipeline completa che fornisce tredici tabelle a uso dell'azienda.

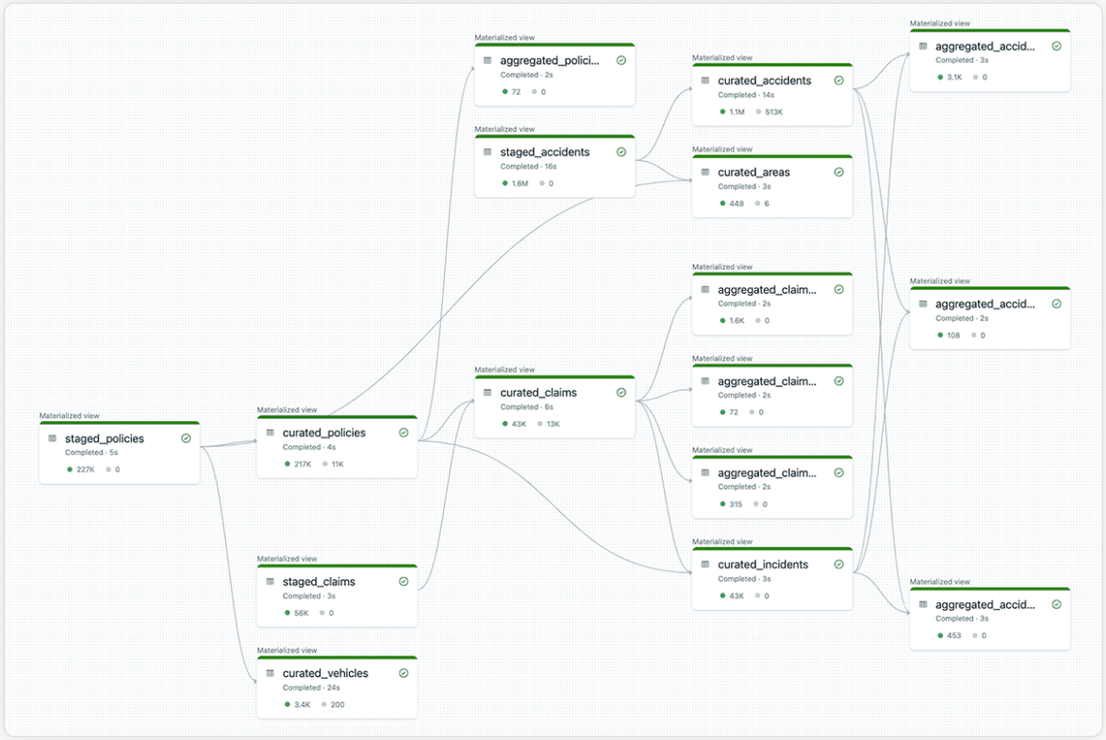


Figura 10. Panoramica di un flusso di lavoro completo di Delta Live Tables (DLT).

I risultati di ciascuna tabella possono essere analizzati selezionando l'entità desiderata. La Figura 11 fornisce un esempio dei risultati della tabella delle richieste di risarcimento curata. DLT fornisce una panoramica generale dei risultati dei controlli di qualità sui dati:

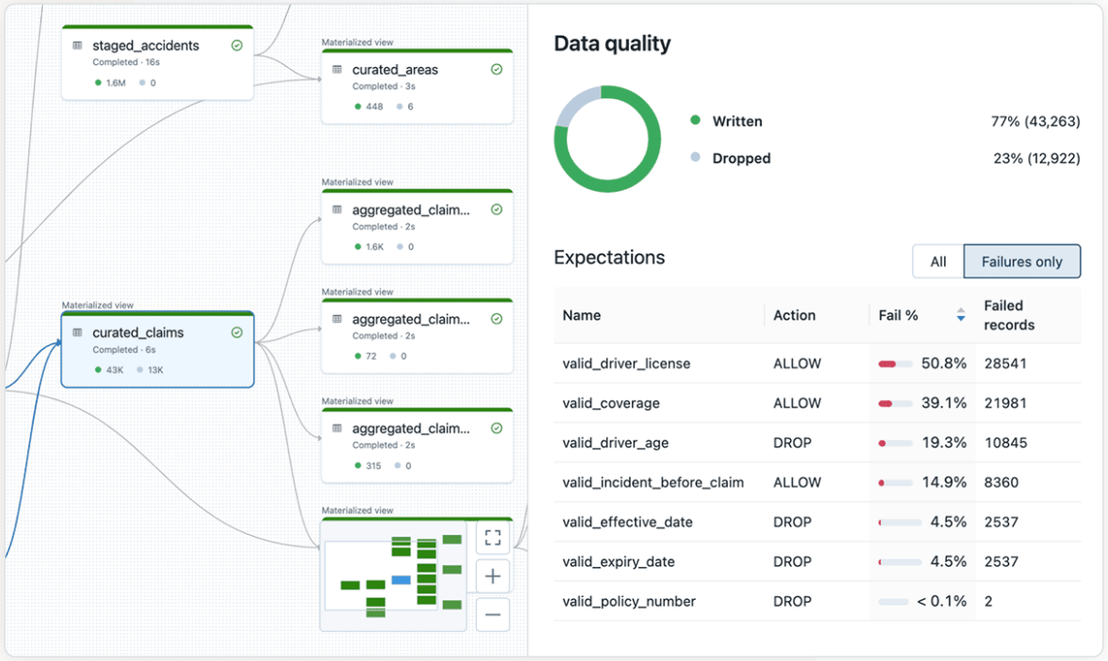
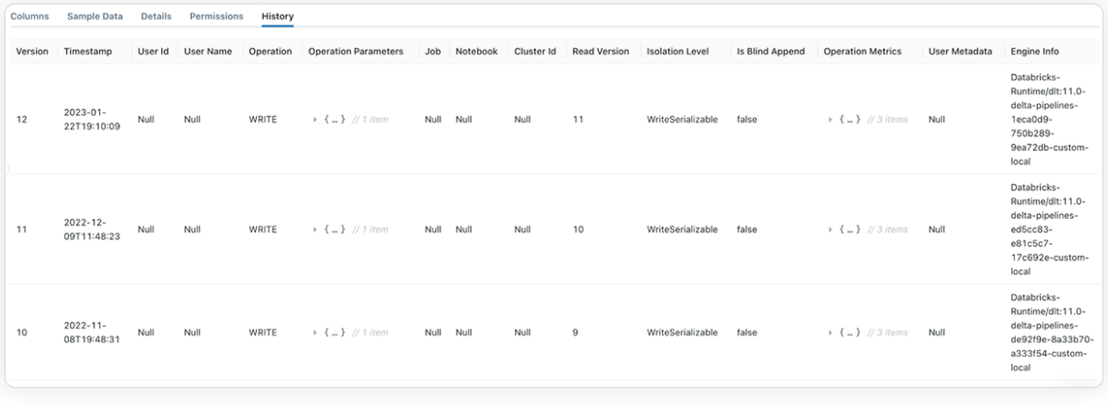


Figura 11. Esempio di vista dettagliata per un'entità di tabella di Delta Live Tables (DLT) con il relativo rapporto sulla qualità dei dati.

I risultati dalle aspettative di qualità dei dati possono essere analizzati ulteriormente interrogando il registro eventi, che contiene metriche dettagliate su tutte le aspettative definite per il flusso di lavoro della pipeline. La seguente query fornisce un esempio per visualizzare le metriche chiave dall'ultimo aggiornamento della pipeline, incluso il numero di record che hanno soddisfatto o fallito le aspettative:

```
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;
```

Anche in questo caso, possiamo visualizzare l'intera cronologia delle modifiche apportate a ciascuna tabella DLT guardando i registri cronologici Delta (vedi Figura 12). Ciò consente di capire come le tabelle si evolvono nel tempo e di indagare thread completi di aggiornamenti in caso di fallimento di una pipeline.


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

Figura 12. Visualizzazione della cronologia delle modifiche apportate a un'entità di tabella Delta Live Tables (DLT) risultante.

Possiamo inoltre utilizzare l'acquisizione dei dati modificati (CDC) per aggiornare le tabelle in base alle modifiche intervenute nei set di dati sorgente. Il CDC di DLT supporta l'aggiornamento di tabelle con slow-changing dimensions (SCD) di tipo 1 e 2.

Per avviare la pipeline DLB, scegliamo una di due opzioni per il nostro processo in batch. Possiamo utilizzare Databricks Auto Loader per elaborare incrementalmente i nuovi dati man mano che arrivano nelle tabelle sorgente, oppure creare job pianificati che si attivano in determinati momenti o intervalli. In questo esempio, abbiamo optato per quest'ultima opzione con un job pianificato che esegue la pipeline DLT ogni cinque minuti.

Operazionalizzare gli output

La capacità di elaborare i dati in modo incrementale ed efficiente è solo metà dell'equazione. I risultati del flusso di lavoro di DLT devono essere resi operativi e consegnati agli utenti aziendali. Nel nostro esempio, possiamo usare gli output della pipeline DLT tramite analisi ad hoc o informazioni preconfezionate rese disponibili tramite una dashboard interattiva.

Analisi ad hoc

Databricks SQL (o DB SQL) fornisce un data warehouse efficiente e conveniente basato sull'architettura lakehouse. Ci consente di eseguire i carichi di lavoro SQL direttamente sui dati sorgente con un rapporto prezzo/prestazioni fino a 12 volte migliore rispetto alle alternative.

Possiamo utilizzare DB SQL per eseguire query ad hoc specifiche sulle tabelle curate e aggregate. Potremmo, ad esempio, eseguire una query sulla tabella curata delle polizze che calcola l'esposizione totale. L'editor di query di DB SQL fornisce un'interfaccia semplice e facile da usare per costruire ed eseguire tali query (vedi esempio sotto).

```
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
      ...
```

Possiamo anche utilizzare l'editor di query di DB SQL per eseguire query su diverse versioni delle nostre tabelle Delta. Ad esempio, possiamo interrogare una vista dei record di richieste di risarcimento aggregati per una data e un'ora specifiche (vedi esempio sotto). Possiamo utilizzare DB SQL anche per confrontare i risultati di diverse versioni e analizzare solo i record modificati tra uno stato e l'altro.

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

DB SQL consente di utilizzare un motore di calcolo serverless, eliminando la necessità di configurare, gestire o scalare l'infrastruttura cloud e contenendo il più possibile i costi. Si integra anche con workbench SQL alternativi (ad esempio DataGrip), permettendo agli analisti di utilizzare i loro strumenti preferiti per consultare i dati e generare informazioni.

Informazioni aziendali

Infine, possiamo utilizzare le query di DB SQL per creare visualizzazioni dettagliate con i risultati delle nostre interrogazioni. Queste visualizzazioni possono quindi essere accorpate e rese disponibili agli utenti finali tramite dashboard interattive (vedi Figura 13).

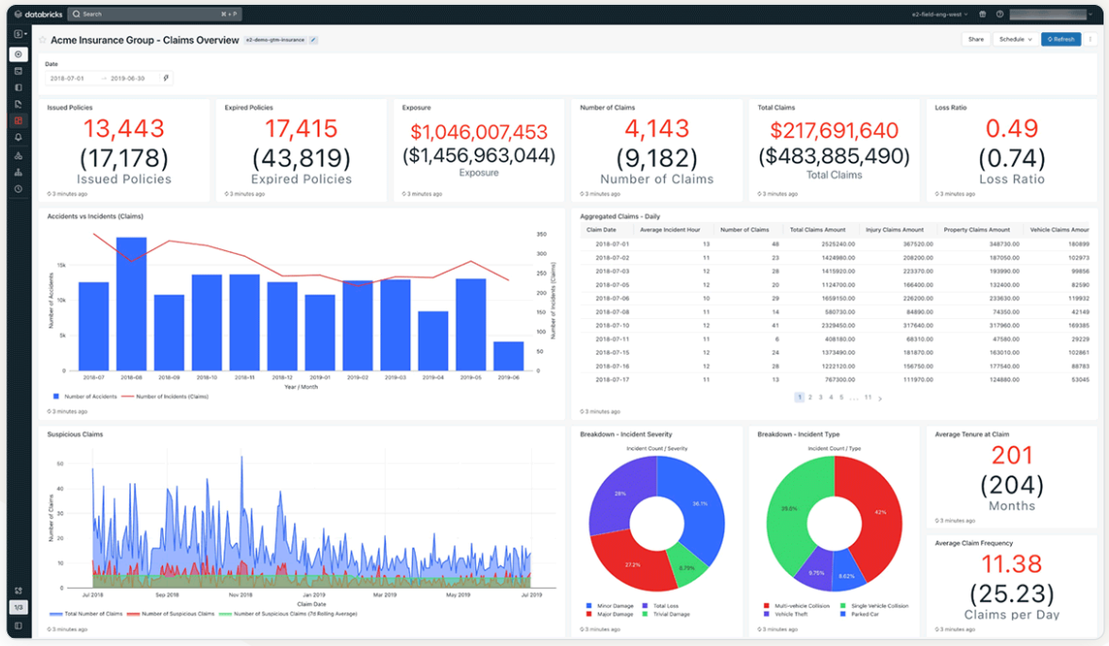


Figura 13. Esempio di dashboard operativa costruita su un insieme di entità di tabelle Delta Live Tables (DLT) risultanti.

Per il nostro caso d'uso, abbiamo creato una dashboard con una raccolta di metriche chiave, calcoli mobili, disaggregazioni di alto livello e visualizzazioni aggregate. La dashboard fornisce un riepilogo completo del nostro processo di gestione delle richieste di risarcimento a colpo d'occhio. Abbiamo anche aggiunto un'opzione per specificare intervalli di date specifici. DB SQL supporta una serie di parametri di query che possono sostituire i valori in una query durante l'esecuzione. Questi parametri di query possono essere definiti a livello di dashboard per garantire che tutte le query correlate vengano aggiornate di conseguenza.

DB SQL si integra con numerosi strumenti di analisi e BI di terze parti come Power BI, Tableau e Looker. Come abbiamo fatto per Fivetran, possiamo utilizzare Partner Connect per collegare la nostra piattaforma esterna con DB SQL. Ciò consente di creare e distribuire dashboard nelle piattaforme preferite dall'azienda senza sacrificare le prestazioni di DB SQL e della Databricks Data Intelligence Platform.

Conclusioni

Nel mondo veloce e volatile della finanza, l'elaborazione in batch rimane una parte essenziale dello stack di dati moderno, in grado di competere con le funzionalità e i vantaggi dei servizi in streaming e in tempo reale. Abbiamo visto come sia possibile utilizzare l'architettura lakehouse per i servizi finanziari e il suo ecosistema di partner per progettare un framework semplice, scalabile ed estensibile che supporta complessi carichi di lavoro di elaborazione in batch, con un esempio pratico incentrato sull'elaborazione delle richieste di risarcimento assicurativo. Con Delta Live Tables (DLT) e Databricks SQL (DB SQL), possiamo costruire una piattaforma dati con un'architettura scalabile all'infinito, facile da estendere per rispondere a esigenze mutevoli e in grado di resistere alla prova del tempo.

Per maggiori informazioni sulla pipeline di esempio descritta, inclusa la configurazione e l'allestimento dell'infrastruttura utilizzata, vedi a questo repository GitHub o guarda questo video dimostrativo.

CAPITOLO

05

Referenze: Risultati reali su Databricks

CAPITOLO 5.1

InMobi: creare connessioni significative tra clienti e brand

Quando si tratta di pubblicità, i consumatori desiderano vedere contenuti pertinenti e personalizzati, specialmente se appaiono sui loro dispositivi mobili, dove il tempo è una risorsa preziosa. Pertanto, è importante catturare immediatamente l'attenzione e stimolare l'engagement. Per ottenere questo risultato, InMobi utilizza i dati dei clienti in tempo reale per fornire pubblicità mirate e contenuti sulla schermata di blocco. Tuttavia, quando i requisiti di elaborazione dei dati sono aumentati fino a superare i 20 terabyte all'ora, il costo di gestione del data warehouse multicloud è schizzato alle stelle e la sua natura proprietaria ha creato silos che ostacolavano la collaborazione e la condivisione dei dati.

InMobi ha eseguito la migrazione dal suo data warehouse multicloud a Databricks per unificare i vari carichi di lavoro (data warehousing, intelligenza artificiale e analisi), ottimizzare le operazioni e occupare gli ingegneri su compiti più importanti, raggiungendo maggiore efficienza e agilità operativa. Passando al lakehouse, l'azienda non solo ha ridotto significativamente il costo totale di proprietà (TCO) rispetto all'utilizzo di un data warehouse multicloud, ma ha anche migliorato la produttività in tutta l'organizzazione, accelerando il time-to-market dei nuovi prodotti.

Databricks ha pienamente soddisfatto il nostro obiettivo di ottimizzare il rapporto prezzo/prestazioni. Il lakehouse ci ha aiutato a ridurre i costi senza sacrificare le prestazioni su carichi di lavoro misti, consentendoci di ottimizzare le operazioni su dati e AI oggi e in prospettiva futura.

— MOHIT SAXENA
Co-founder e Group CTO,
InMobi

32%
Riduzione del TCO rispetto al precedente data warehouse multicloud

15%
Aumento della velocità delle query rispetto al precedente data warehouse multicloud

20%
Miglioramento delle prestazioni della reportistica dei fornitori

databricks

Un'infrastruttura legacy complessa e un data warehouse multicloud difficile da gestire

L'attività di InMobi si concentra sulla pubblicità mirata per aiutare i brand a raggiungere e coinvolgere i consumatori in modo significativo ed economicamente vantaggioso. Ma per consegnare annunci pertinenti in modo accurato, servono dati. Moltissimi dati. Nel corso del tempo, InMobi ha esteso il suo sistema Hadoop on-premise aggiungendo diversi data warehouse cloud per far fronte a vari problemi. Tuttavia, mentre la quantità di dati che l'azienda doveva elaborare aumentava esponenzialmente (fino a 20 TB l'ora), InMobi ha continuato a costruire sul proprio sistema legacy, finendo per ritrovarsi con un data warehouse multicloud che presentava numerose criticità. Era infatti eccessivamente complesso, soggetto a guasti e interruzioni ed estremamente costoso da scalare. Inoltre, isolava i dati in silos, limitando la collaborazione e la condivisione. Il team di InMobi si è reso conto che, continuando con questo sistema, avrebbe rallentato la propria capacità di innovare e sperperato preziose risorse ingegneristiche per la manutenzione.

> **"L'infrastruttura dati che avevamo costruito funzionava, ma a prezzo di una complessità e di un costo di gestione significativi che ci distoglievano dal focus sui nostri prodotti principali", ha dichiarato Madhan Sundaram, Senior Director of Platform Engineering di InMobi. "I nostri ingegneri dovevano impiegare il proprio talento per generare più valore per i nostri clienti, e per questo ci serviva un sistema più semplice e unificato".**

InMobi voleva un singolo sistema in grado di risolvere diversi problemi. Per raggiungere questo obiettivo, l'azienda doveva riunire in un'unica piattaforma i suoi diversi sistemi, così da permettere agli ingegneri di concentrarsi su attività di maggior valore, come lo sviluppo di modelli di ML e LLM. Il team si è affidato a Databricks Data Intelligence Platform per unificare i carichi di lavoro di data warehousing e AI su una singola piattaforma.

La migrazione al lakehouse permette di unificare dati, analisi e intelligenza artificiale

Gli esperti ingegneri di InMobi hanno imparato una lezione preziosa sulla produttività e l'agilità operativa. "Abbiamo scoperto che tutto il tempo dedicato alla manutenzione del nostro ambiente stava danneggiando la nostra capacità di supportare e collaborare agli obiettivi aziendali", ha spiegato Sundaram. "Volevamo essere più strategici e individuare modi per operazionalizzare i dati in modo più efficiente". Dopo un'attenta valutazione del data warehouse multicloud esistente e delle prospettive di costruzione in-house, InMobi ha ritenuto che la Databricks Data Intelligence Platform fosse quella che si allineava meglio ai suoi obiettivi: migliorare la produttività degli sviluppatori attraverso la riduzione della complessità dell'infrastruttura, ottenendo nel contempo la migliore resa possibile in termini di prezzo/prestazioni.

Una volta presa la decisione, il team ha collaborato con Databricks per pianificare la migrazione. Con oltre un decennio di personalizzazioni sviluppate sui sistemi esistenti e più di un 1 petabyte di dati, InMobi sapeva che la migrazione sarebbe stata complessa. Solo sul lato ETL c'erano 150 pipeline e otto team coinvolti nella migrazione da Apache Spark™ open source. La migrazione riguardava anche circa 300 dashboard di reportistica e doveva avvenire in modo da evitare interruzioni nel flusso di informazioni verso fornitori e clienti. Per la gestione di questo processo, Databricks ha collaborato strettamente con InMobi (che ha destinato due ingegneri interni per team al supporto) e il suo partner di implementazione Celebal Technologies per ottenere le ottimizzazioni necessarie per una migrazione senza intoppi.

Databricks ha permesso a InMobi di migrare con facilità dal suo data warehouse multicloud al lakehouse. "Databricks ci consente di utilizzare funzionalità all'avanguardia nel settore che ci aiutano a fare la differenza in ambito tecnico", ha dichiarato Sundaram. "La sua competenza è emersa durante la migrazione, quando ci ha aiutati a ottimizzare le risorse di calcolo in termini di costo e prestazioni. "La responsabilità che Databricks si è assunta nel guidare le ottimizzazioni e la trasparenza e il valore formativo del processo sono stati encomiabili".

databricks

Portare annunci più personalizzati sui dispositivi mobili di una clientela globale

Ora, con un'architettura lakehouse unificata e più snella, InMobi può sfruttare appieno i suoi robusti dati sui clienti per fornire pubblicità più intelligenti e personalizzate sui dispositivi mobili. Diversi team usano i notebook Databricks per analisi ad hoc, Power BI per visualizzazioni su Databricks SQL (il data warehouse serverless sul lakehouse) e MLflow per costruire una piattaforma AI di prossima generazione. Anche l'adozione di Delta Live Tables per il rilevamento delle anomalie si è rivelata un grande successo, con un miglioramento del 50% negli SLA e una riduzione dell'80% dei costi. I silos e la visibilità dei dati non sono più un problema, grazie a Unity Catalog, con cui InMobi può ora gestire gli accessi a livello di tabella e colonna, acquisendo al contempo l'intera derivazione dei dati per assicurarsi di avere visibilità sulla loro provenienza e sull'eventuale obsolescenza. Con una piattaforma progettata per soddisfare le sue esigenze di analisi e intelligenza artificiale, il team di InMobi sta sperimentando nuove tecnologie, compresi i modelli linguistici di grandi dimensioni (LLM), per fornire le informazioni ai clienti in modo più efficiente. "Stiamo considerando la possibilità di implementare LLM per rendere più facile ai nostri utenti finali fare una domanda e trovare le informazioni di cui hanno bisogno. L'architettura lakehouse faciliterà questo sforzo, perché i job verranno eseguiti automaticamente dietro le quinte. Questo consentirà ai nostri team di porre semplicemente una domanda e di avere risposte contestuali a portata di mano, senza che questo richieda alcuna particolare competenza da parte loro", ha spiegato Sundaram.

In termini di impatto sull'azienda, la migrazione ha apportato numerosi miglioramenti misurabili su tutta la linea. Non solo i costi dell'infrastruttura sono inferiori del 34% rispetto al passato, ma anche la velocità delle query è aumentata del 15% e i job falliti sono diminuiti del 20% rispetto all'ambiente dati precedente, il che ha contribuito a migliorare del 20% le prestazioni nella reportistica e nella consegna di informazioni agli utenti finali. Il TCO è inferiore del 32% rispetto a quando utilizzavano un data warehouse multicloud e si è registrata una riduzione dei costi del 24% anche nell'esecuzione delle pipeline ETL. Dal punto di vista qualitativo, il team ha riscontrato un miglioramento generale dell'affidabilità, con sistemi più stabili e un aumento della reputazione presso i clienti.

"Il nostro tasso di sperimentazione è migliorato enormemente", ha dichiarato Mohit Saxena, Co-founder Group CTO di InMobi. "Databricks ha semplificato la collaborazione e l'approccio unificato offerto dal lakehouse ci permette di essere più efficienti, produttivi e coerenti quando forniamo nuove funzionalità e prodotti."

Con il passaggio a una piattaforma unificata sulla Databricks Data Intelligence Platform, InMobi può ora concentrarsi sull'innovazione nello spazio pubblicitario mobile per offrire una personalizzazione in tempo reale che genera valore sia per i clienti di InMobi sia per i suoi utenti finali interni.

Leggi altre referenze

CAPITOLO 5.2

Akamai: eseguire analisi in tempo reale su larga scala con Delta Lake

Akamai gestisce una rete di distribuzione dei contenuti (CDN) capillare e altamente distribuita. Il suo CDN utilizza circa 345.000 server in più di 135 Paesi e oltre 1.300 reti in tutto il mondo per instradare il traffico Internet di alcune delle più grandi aziende nei settori dei media, del commercio, della finanza, della vendita al dettaglio e altri. Circa il 30% del traffico Internet passa attraverso i server di Akamai, che fornisce anche soluzioni di sicurezza per il cloud.

Nel 2018, l'azienda ha lanciato uno strumento di analisi della sicurezza web che offre ai clienti un'interfaccia unificata per valutare un'ampia gamma di eventi di sicurezza in streaming ed eseguirne l'analisi. Lo strumento di analisi web aiuta i clienti Akamai a intraprendere azioni informate in relazione agli eventi di sicurezza in tempo reale. Akamai è in grado di trasmettere enormi quantità di dati e di rispettare i rigorosi SLA che fornisce ai clienti sfruttando Delta Lake e la Databricks Data Intelligence Platform per il suo strumento di analisi web.

<1
minuto per l'acquisizione dei dati, invece di 15 min

>85%
delle query hanno un tempo di risposta di 7 secondi o inferiore

Delta Lake ci permette non solo di interrogare meglio i dati, ma anche di aumentarne il volume. Nell'ultimo anno abbiamo registrato un aumento dell'80% del traffico e dei dati, per cui la capacità di scalare rapidamente è per noi fondamentale.

— TOMER PATEL
Engineering Manager,
Akamai

Acquisizione e streaming di enormi quantità di dati

Lo strumento di analisi della sicurezza web di Akamai acquisisce circa 10 GB di dati relativi agli eventi al secondo. Il volume può aumentare in modo significativo quando i clienti del settore retail effettuano un gran numero di vendite o in occasione di giornate come il Black Friday o il Cyber Monday. Lo strumento di analisi della sicurezza web archivia svariati petabyte di dati a scopo di analisi. Queste analisi vengono eseguite per proteggere i clienti di Akamai e fornire loro la possibilità di analizzare e interrogare autonomamente gli eventi di sicurezza.

Lo strumento di analisi si basava inizialmente su un'architettura on-premise con Apache Spark™ su Hadoop. Akamai offre ai suoi clienti rigorosi accordi sui livelli di servizio (SLA) che prevedono un intervallo di 5-7 minuti dal momento in cui si verifica un attacco alla sua visualizzazione nello strumento. Per rispettare tali SLA, l'azienda desiderava migliorare la velocità di acquisizione e di interrogazione dei dati. "I dati devono essere il più possibile in tempo reale, in modo che i clienti possano vedere cosa li sta attaccando", afferma Tomer Patel, Engineering Manager presso Akamai. "Fornire dati interrogabili ai clienti in modo veloce è fondamentale. Volevamo abbandonare l'on-premise per migliorare prestazioni e SLA in modo che la latenza fosse di pochi secondi invece che di minuti".

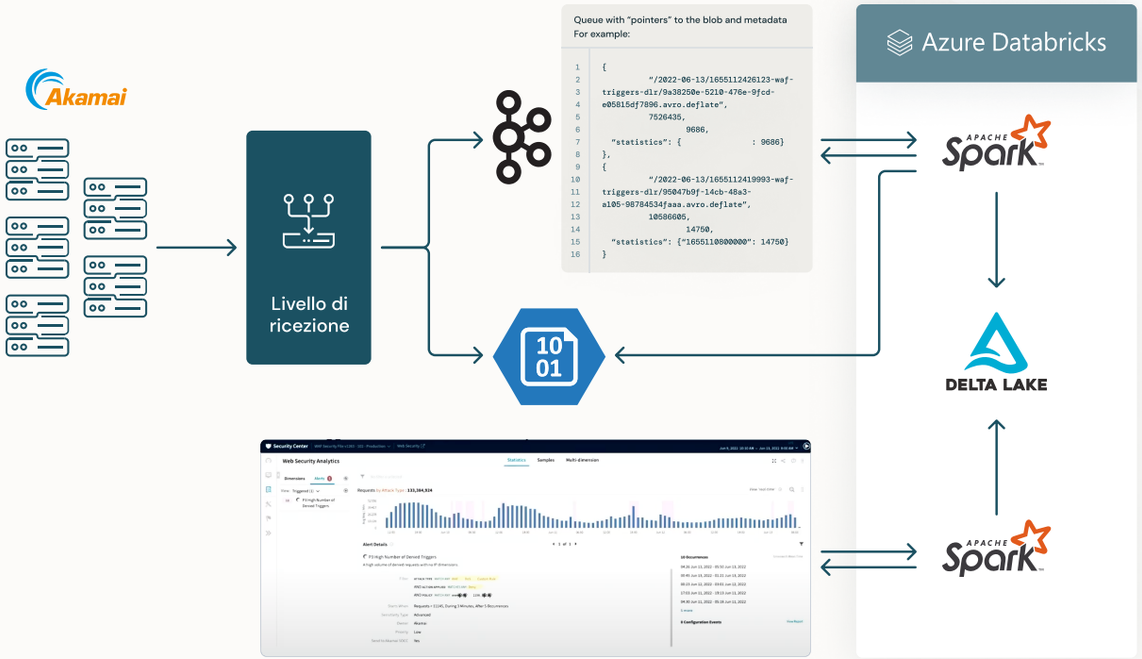
Dopo aver condotto prove di concetto con diverse aziende, Akamai ha scelto di basare la sua architettura di analisi in streaming su Spark e sulla Databricks Data Intelligence Platform. "Alla luce delle nostre dimensioni e dei requisiti dei nostri SLA, abbiamo stabilito che Databricks fosse la soluzione giusta per noi", afferma Patel. "Non avremmo potuto raggiungere lo stesso livello di prestazioni nell'ottimizzazione dell'archivio e nel caching se avessimo scelto un'altra soluzione".

Aumentare la velocità e ridurre i costi

Oggi, lo strumento di analisi della sicurezza web acquisisce e trasforma i dati, li archivia in uno storage cloud e invia la posizione del file tramite Kafka. Poi utilizza un job Databricks come applicazione di acquisizione. Delta Lake, il formato di archiviazione open source alla base della Databricks Data Intelligence Platform, supporta query in tempo reale sui dati di analisi della sicurezza web. Delta Lake consente inoltre ad Akamai di scalare rapidamente. "Delta Lake ci permette non solo di interrogare meglio i dati, ma anche di aumentarne il volume", afferma Patel. "Abbiamo osservato un aumento dell'80% nel traffico e nei dati nell'ultimo anno, quindi essere in grado di scalare rapidamente per noi è essenziale."

Akamai utilizza anche Databricks SQL (DBSQL) e Photon, che offrono prestazioni di interrogazione estremamente veloci. Patel ha dichiarato che Photon ha consentito un notevole miglioramento nelle prestazioni delle query. Nel complesso, l'architettura di streaming di Databricks, combinata con DBSQL e Photon, consente ad Akamai di eseguire analisi in tempo reale, con immediati vantaggi commerciali.

Patel apprezza che Delta Lake sia open source e sottolinea come la sua azienda abbia tratto vantaggio dall'esistenza di una comunità di utenti che lavora per migliorare il prodotto. "Il fatto che Delta Lake sia open source e abbia una grande comunità alle spalle significa che non dobbiamo implementare tutto da soli," spiega. "Beneficiamo delle correzioni ai bug e delle ottimizzazioni che altri apportano al progetto." Akamai ha collaborato strettamente con Databricks per garantire che Delta Lake potesse soddisfare le sue esigenze di scalabilità e prestazioni. Tutti questi miglioramenti sono stati introdotti nel progetto (e molti resi disponibili come parte di Delta Lake 2.0) di conseguenza, qualsiasi utente che esegue Delta Lake può trarre vantaggio dalla tecnologia testata su scala così ampia in uno scenario di produzione reale.



Soddisfare requisiti stringenti di scalabilità, affidabilità e prestazioni

L'uso di Spark Structured Streaming sulla Databricks Data Intelligence Platform consente allo strumento di analisi della sicurezza web di eseguire lo streaming di vasti volumi di dati e di fornire Analytics-as-a-Service (AaaS) in tempo reale ai clienti di Akamai. Akamai è così in grado di rendere disponibili ai clienti i dati relativi agli eventi di sicurezza entro l'intervallo di 5-7 minuti dal momento in cui si verifica un attacco, come previsto dai suoi SLA. "Per noi, le prestazioni sono tutto", dice Patel. "Prestazioni e scalabilità della piattaforma sono alla base del nostro lavoro", afferma.

Ora, con la Databricks Data Intelligence Platform, basta meno di 1 minuto per acquisire i dati degli eventi di sicurezza. "Ridurre i tempi per l'acquisizione da 15 minuti a meno di 1 minuto è un enorme progresso", dice Patel. "È un vantaggio per i nostri clienti, perché possono visualizzare i dati degli eventi di sicurezza più rapidamente e avere una visione precisa di ciò che sta accadendo, anche attraverso filtri".

Per Akamai, la massima priorità è fornire ai clienti un'esperienza positiva e tempi di risposta rapidi. Ad oggi, Akamai ha trasferito circa il 70% dei dati degli eventi di sicurezza dalla propria architettura on-premise a Databricks, migliorando significativamente i livelli di servizio (SLA) per le query dei clienti e i tempi di risposta. "Ora, con il passaggio a Databricks, i nostri clienti ottengono un tempo di risposta molto inferiore, con oltre l'85% delle query completate in meno di 7 secondi." Fornendo questo tipo di dati in tempo reale, Akamai aiuta i suoi clienti a rimanere vigili e a mantenere una configurazione di sicurezza ottimale.

Leggi altre referenze

CAPITOLO 5.3

Quartile: diventare la più grande piattaforma pubblicitaria per l'e-commerce

Quartile è la piattaforma pubblicitaria cross-channel per l'e-commerce più grande del mondo e serve oltre 5.000 account pubblicitari collegati da più di 10 canali. Costruita su sei tecnologie di machine learning brevettate, la piattaforma automatizza e ottimizza la pubblicità per i siti di e-commerce su Google, Facebook, Amazon, Instacart, Walmart e altri canali. Quartile si basa su una tecnologia all'avanguardia e su esperti di marketing che creano strategie su misura degli obiettivi aziendali dei clienti. Migliaia di venditori in tutto il mondo si affidano all'ottimizzazione del funnel di conversione di Quartile per sbloccare il potenziale di vendite e pubblicità su più canali, completamente integrati e su larga scala.

80%
Riduzione dello spazio di archiviazione rispetto ai database tradizionali

10x più veloce
45 minuti per ottimizzare le offerte, invece delle 7,5 ore precedenti

La Databricks Data Intelligence Platform ha consentito ai nostri team tecnologici di migliorare il time-to-market delle nuove soluzioni e di rendere felici i nostri clienti con set di dati di alta qualità. Diventare la più grande piattaforma pubblicitaria cross-channel per l'e-commerce sarebbe stato molto più difficile senza Databricks e l'architettura lakehouse.

— DANIEL KNIJNIK
CEO,
Quartile

databricks

Dimensione dei dati e prestazioni necessarie per scalare

Quartile doveva far fronte a diverse criticità nell'archiviazione e nell'elaborazione dei dati per molteplici canali pubblicitari a causa di stringenti esigenze nella generazione di report sui dati di vendita e nel loro consolidamento, compresa un'attribuzione di oltre 60 giorni. L'architettura precedente non riusciva a tenere il passo con la mole di dati che Quartile doveva processare. Il team elaborava in batch oltre 10 TB di dati in un singolo job che applicava tutte le trasformazioni necessarie per la generazione dei report, causando l'indisponibilità del server e ritardi nella consegna dei data point. Questo processo, che serve per migliorare le prestazioni degli annunci, presentava gravi problemi a livello di prestazioni, con singoli job che richiedevano fino a 7,5 ore ogni giorno.

Evoluzione tecnologica, dall'architettura legacy al data stack moderno

Nel quadro dell'evoluzione dell'azienda, l'architettura dati di Quartile ha raggiunto nuovi livelli di maturità, passando da tradizionali database SQL eseguiti su cloud Azure a una nuova soluzione basata sulla Databricks Data Intelligence Platform. Questa modernizzazione ha avuto un impatto diretto su diverse aree dell'azienda e ha apportato vantaggi evidenti per i clienti di Quartile: Delta Lake ha aumentato l'affidabilità dei dati, velocizzato le prestazioni e ridotto i costi. Grazie alla migrazione dei dati da un tradizionale database SQL a Databricks, Quartile ha riscontrato una considerevole riduzione del volume dei dati, principalmente grazie alle ottimizzazioni di Delta con il controllo delle versioni e la compattazione di Parquet, con una riduzione dello spazio di archiviazione da 90 TB a circa 18 TB.

L'architettura lakehouse ha permesso allo stack di dati di Quartile di evolvere. Questo risultato è stato possibile attraverso diversi processi: Databricks Auto Loader per l'elaborazione incrementale ed efficiente dei nuovi file di dati man mano che arrivano nello storage cloud; Delta Lake per il suo livello di archiviazione in formato aperto, che offre affidabilità, sicurezza e prestazioni nel data lake; Databricks SQL per avvicinare ingegneri e analisti dei dati con query e dashboard facili da creare; e Databricks Workflows che collega tutti i pezzi in modo stabile e scalabile.

Per offrire la migliore esperienza ai clienti, Quartile deve essere in grado di ottenere dati accurati. Questa sfida importante è più facile da affrontare con le funzioni definite dall'utente di Spark, perché l'azienda sfrutta la potenza del parallelismo per suddividere l'elaborazione in tutte le parti necessarie. Per rendere scalabile la soluzione, Quartile utilizza Terraform per implementare tutti gli spazi di lavoro, creando facilmente nuovi cluster e job e garantendo che in tutta l'azienda siano seguiti gli standard.

Aiutare i partner a gestire meglio la pubblicità

È fondamentale che i clienti di Quartile dispongano di una soluzione centralizzata in cui analizzare vendite, costi e altre metriche relative alle loro campagne. Quartile approfitta del solido lavoro di ingegneria dei dati e integra Databricks con Power BI per incorporare le dashboard direttamente nel suo portale. In questo modo, i clienti hanno un unico luogo in cui configurare campagne di marketing multicanale e seguire le variazioni delle prestazioni, mantenendo comunque una minore impronta di archiviazione dei dati, con un costo inferiore dell'80% rispetto allo storage nei data warehouse tradizionali, sfruttando il formato di file di Delta Lake per l'archiviazione degli oggetti. Molti clienti dell'azienda hanno sfruttato la possibilità di combinare dati provenienti da tutti i canali; SmartyPants, ad esempio, è cresciuta oltre il 100% da quando ha stretto una partnership con Quartile.

Ma non è tutto. Quartile ha anche brevettato degli algoritmi per migliorare le prestazioni degli annunci, implementati utilizzando il machine learning di Databricks. Avere un'architettura lakehouse centralizzata per costruire l'intero stack di dati ha semplificato notevolmente il lavoro degli sviluppatori di Quartile, consentendo loro di concentrarsi sullo sviluppo di soluzioni innovative e di ottenere risultati sempre migliori per i clienti. Un ulteriore esempio è fornito da OfficeSupply, che ha ottenuto eccellenti risultati durante il primo anno di collaborazione con Quartile, con un aumento del 67% dei ricavi da Google Ads e un aumento del 103% dei clic su Google Shopping per i termini di marchio migliorando le prestazioni dei singoli job: in passato richiedevano 7,5 ore, ora vengono eseguiti in 45 minuti sull'architettura lakehouse.

Per il futuro, Quartile continuerà la sua partnership con Databricks per costruire un moderno stack di dati, integrando e testando le nuove soluzioni. Tra queste, Delta Live Tables per controlli di qualità più accurati, Delta Sharing per inviare i dati ai clienti e Data Marketplace per consentire ai clienti di muovere rapidamente i primi passi. Quartile ha l'ambizioso obiettivo di sviluppare il primo apprendimento cross-channel per un algoritmo di ottimizzazione degli annunci del settore, e Databricks sarà al centro di queste innovazioni.

Informazioni su Databricks

Databricks è un'azienda di dati e AI. Più di 10.000 organizzazioni in tutto il mondo (fra cui Comcast, Condé Nast, Grammarly e oltre il 50% delle aziende Fortune 500) si affidano alla piattaforma Databricks Data Intelligence per unificare dati, analisi e AI. Databricks ha la sede principale a San Francisco e uffici in tutto il mondo ed è stata fondata dai creatori di Lakehouse, Apache Spark™, Delta Lake e MLflow.

Per maggiori informazioni, segui Databricks su Twitter, LinkedIn e Facebook.

COMINCIA LA PROVA GRATUITA

Contattaci per una demo personalizzata
databricks.com/company/contact

databricks