ETL Serverless su AWS: Integrazione diretta SNS – Kinesis

23 Dicembre 2022 - 9 min. read

Amazon Kinesis | Amazon QuickSight | AWS Glue | Data and Analytics | Data Lake | ETL

Al giorno d'oggi, più che mai, le aziende capiscono il potenziale e il valore dei dati e dell'importanza di utilizzarli in maniera consona. Per questo motivo soluzioni ETL sono sempre più comuni e dedicate.

In questo articolo parleremo di una pipeline ETL con un'integrazione non comunemente usata in problemi di questo tipo ma necessaria per il business preso in esame.

Extraction, Transformation and Loading (ETL) è ormai uno standard per pipeline dedicate al dato.

Lo step di estrazione raccoglie dalle varie sorgenti il dato grezzo: avere un Data Lake correttamente organizzato è la chiave per semplificare la fase di estrazione e le successive.

Una volta che il dato è stato recuperato si possono applicare tutte le trasformazioni necessarie ad estrarre il valore maggiore possibile dal dato.

Questi step in linea, che si riassumono in ciò che viene chiamato "step di trasformazione", sono specifici in base alle necessità di utilizzo del dato. I risultati saranno passati all'ultimo step di salvataggio: lo step di caricamento (Loading).

I risultati possono essere analizzati e visualizzati per avere una visione chiara e migliore da un punto di vista analitico in modo da aiutare i processi decisionali.

Il vantaggio di questa struttura di processo è la facilità di modifica e/o implementazione di una trasformazione aggiuntiva, il tempo richiesto è drasticamente ridotto garantendo così un vantaggio enorme in termini di tempo nel rispondere a domande chiave dal punto di vista business in modo da guidare, dati alla mano, decisioni a più alto livello.

Ogni step può essere realizzato con diverse modalità, in questo articolo andremo a vedere una di queste possibilità, aggiungendo qualche consiglio riguardo come farlo nel miglior modo.

Prima di iniziare con la descrizione della soluzione tecnica contestualizziamo il flusso dati.

L'idea finale è un servizio a cui le persone si possano sottoscrivere e inviare flussi di dati continui che dovranno essere salvati e processati in, quasi, tempo reale.

Il caricamento in maniera corretta, senza perdita di dati, è critico per il funzionamento del flusso.

Inoltre, i componenti dovranno essere in grado di scalare in maniera efficiente.

Soluzione tecnica: integrazione SNS - Kinesis Firehose e SQS

Ora che abbiamo un quadro generale, possiamo iniziare con la descrizione delle componenti della soluzione tecnica, spiegarne le possibilità e come connetterle tra di esse per farle interagire.

Il servizio AWS SNS può essere configurato per inviare notifiche con diversi protocolli: dalle classiche eMail e messaggi, richieste API HTTP/HTTPS a integrazioni dirette con servizi AWS come SQS, Lambda e Kinesis Data Firehose.

Per questo motivo i topic SNS sono un'ottima soluzione per il disaccoppiamento dell'infrastruttura.

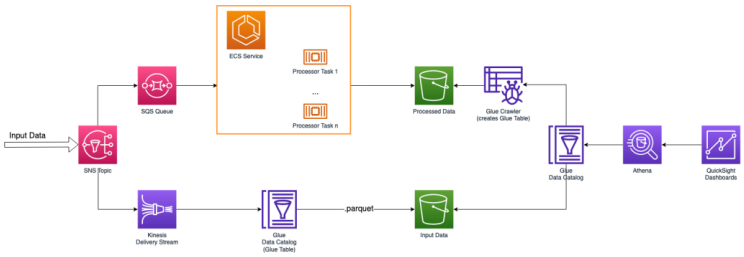
Il topic SNS invierà, nella nostra soluzione, dati a molteplici destinatari: un Kinesis Delivery Stream e una coda SQS.

Nota: possiamo anche usare una sottoscrizione email per ricevere direttamente notifiche su un subset di input grazie alle possibilità fornite da SNS filter.

L'idea è processare i dati in input e, al contempo, salvarli in una sezione differente dell'infrastruttura in modo da poterli visualizzare e, se necessario, ri-processarli successivamente con altri tipi di trasformazioni o con una versione aggiornata del nostro flusso dati.

L'SNS topic si interfaccerà quindi sia con la componente di estrazione che con con quello di trasformazione del nostro flusso ETL: il Delivery Stream salverà i dati che faranno parte del

nostro Data Lake e la coda SQS invierà lo stesso dato al servizio per la trasformazione.



Kinesis Delivery Stream è in grado di aggregare il dato in input e salvarlo come file singolo, questo è esattamente ciò che ci serve per smarcare il primo obiettivo.

Un Delivery Stream aggregherà i dati per periodi di tempo configurabili, o fino al raggiungimento di una quota di MBs e salverà il file generato in maniera compressa su S3 (*.parquet o .orc*).

L'SNS topic indirizzerà i dati sia a Kinesis che al processor; quindi possiamo settare un limite di 3 minuti per il batch dei dati in input.
Salveremo file .parquet su un bucket S3 in una cartella "*data*", partizionando per anno, mese e giorno con l'espressione:

```
data/year=!{timestamp:yyyy}/month=!{timestamp:MM}/day=!{timestamp:dd}/
```

La scelta del formato parquet è guidata dalla facile accessibilità anche tramite S3 Select, Glue e Athena.

I file parquet sono strutturati, dovremo avere la definizione di questa struttura da configurare in Kinesis Data Firehose.

A riguardo, una nota importante: è anche possibile avere una funzione Lambda come transformer per il Delivery Stream.

Lambda può fare qualsiasi tipo di elaborazione sul dato, dalla semplice validazione alla completa ricostruzione del dato in input con l'aggiunta di nuovi campi. La definizione della struttura necessaria a Kinesis Data Firehose per l'aggregazione del dato in file parquet deve, per forza di cose, riflettere quella del dato di output che sarà salvato in S3.

In questo caso specifico, stiamo usando Kinesis Delivery Stream per salvare il nostro input. Non necessitiamo di una Lambda Transformer e quindi la nostra struttura sarà uguale a quella del dato in input.

La comunicazione della struttura dati a Kinesis è fatta tramite i servizi di AWS Glue usando, nello specifico, le feature di Glue Data Catalog. Dobbiamo creare un database Glue con tabelle al suo interno, le tabelle manterranno la struttura dati e saranno usate da Kinesis per la generazione dei file parquet.

Ora che abbiamo il dato di input salvato e aggregato dobbiamo trovare il modo di processarlo.

Processare il dato può essere fatto in diversi modi, come detto in precedenza, i set dei possibili sottoscrittori di SNS è vasto. Da funzioni Lambda, code SQS a integrazioni API per notificare tipi eterogenei di processors.

Nel nostro caso abbiamo necessità di un processing near-real time; abbiamo quindi creato una coda SQS che viene consumata da un servizio ECS.

Il servizio scala il numero di task accesi dipendentemente dal numero di messaggi all'interno della coda.

Abbiamo iniziato il flusso con la validazione dell'input controllando che tutti i campi necessari siano presenti. Questo controllo è necessario per controllare che nessuno abbia manomesso l'applicazione provando a interagirci in modi malevoli.

Inoltre, abbiamo inserito dei controlli sul tipo del dato in modo da garantire che l'input sia in linea con i metodi poi utilizzati per il processing.

Dati validi continueranno all'interno del flusso mentre dati 'sporchi' o non completi saranno categorizzati e, in base al tipo di errore sul dato, salvati in sezioni diverse della soluzione.

Il processing del dato di input può essere diviso in 3 parti principali: appiattimento del dato e pulizia, offuscazione e aggregazione.

Inizialmente appiattiamo il dato portando a livello eventuali dati annidati all'interno di altri filtrando inoltre campi non utilizzati dalla piattaforma.

Di seguito andiamo a offuscare il dato di input, non andremo nel dettaglio di questo step ma è risultato necessario non avere tutti i dati in chiaro per questo specifico flusso di dati. Abbiamo utilizzato tecniche di data masking per nascondere le PII.

Come ultimo step aggreghiamo il dato per poi estrarre metriche e generare informazioni da queste ultime.

Per questa soluzione, siamo interessati nel trovare luoghi fisici in cui persone si ritrovano in gruppi; dobbiamo quindi contare il numero di persone all'interno di un dato raggio con informazioni sul sesso e sull'età dei soggetti.

Infine, lo step di caricamento del flusso ETL: salviamo il dato aggregato all'interno di un bucket S3 per l'utilizzo futuro.

Visualizzazione del dato

Abbiamo ora il nostro dato, sia processato che grezzo salvato all'interno di un Bucket S3, possiamo finalmente utilizzarlo per analisi e raccogliere informazioni.

Vorremo fare più cose col dato tra cui: analizzare con query e visualizzarlo.

Se vogliamo fare query SQL sul nostro dato in S3 possiamo utilizzare Athena.

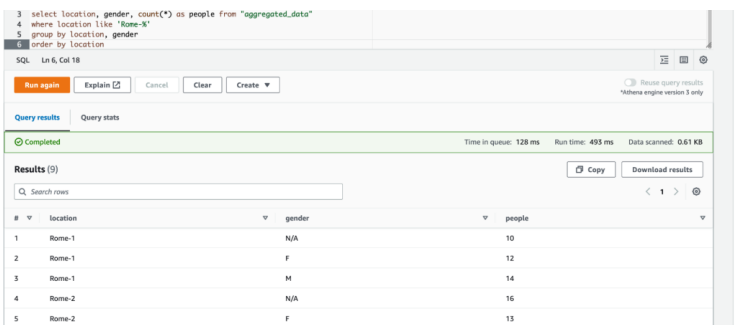
Athena può eseguire query direttamente su S3 o tramite il Glue Data Catalog. Come descritto in precedenza, abbiamo già creato un database Glue e una tabella all'interno del Data Catalog per i nostri dati in input da Firehose.

Possiamo fare lo stesso con i nostri dati aggregati utilizzando un altro servizio dalla suite Glue: il Glue Crawler.

Con poche configurazioni necessarie possiamo puntare il nostro Crawler verso una posizione in un bucket S3 e il crawler estrarrà in autonomia lo schema del dato creando una tabella Glue che lo rifletta.

Suggeriamo di seguire questa metodologia in modo che, se cambiamo il tipo di data processing, il crawler potrà essere rilanciato e modificherà in autonomia lo schema.

Ora possiamo eseguire query sui nostri dati, sia input che aggregati, con Athena recuperando importanti informazioni di business come, ad esempio, il numero di persone in prossimità di un punto d'interesse, raggruppate in base al sesso.

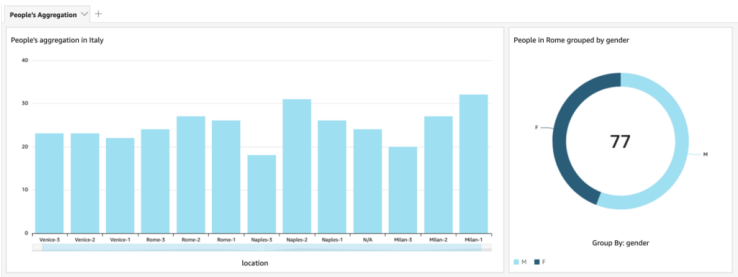


La visualizzazione del dato sarà relativamente semplice da configurare, in questo caso ci viene in aiuto AWS QuickSight per la creazione di dashboard.

Per farlo, andiamo a creare due dataset in QuickSight, uno per il dato grezzo di input e uno per il processato. I dataset possono avere come sorgente diverse entità, nel nostro caso useremo Athena per entrambi.

Una volta che abbiamo i nostri dataset possiamo utilizzare le analisi QuickSight per personalizzare la visualizzazione come preferiamo e, successivamente, quando siamo soddisfatti del risultato ottenuto, salvarla in una dashboard QuickSight da fornire ai fruitori.

Grazie alla dashboard QuickSight abbiamo ora un modo di fruire dei risultati delle nostre analisi in maniera chiara ed immediata. Riusciamo quindi ad avere l'informazione dell'aggregazione di persone presso punti di interesse a colpo d'occhio.



Conclusioni

In questo articolo abbiamo esplorato una parte del mondo ETL, con un assaggio di ogni sua componente: estrazione, trasformazione, caricamento ed eventuali implementazioni su AWS.

Abbiamo imparato come disaccoppiare blocchi infrastrutturali con SNS e come possiamo utilizzare i topic per notificare diversi componenti con lo stesso input.

Grazie a questa possibilità abbiamo notificato sia lo step di Estrazione che di Trasformazione nello stesso momento processando così il dato in quasi tempo reale.

Abbiamo poi descritto le varie possibilità di salvataggio del dato di input in formati compressi, parquet con Kinesis Delivery Stream. Facendo questo, abbiamo esplorato i servizi della suite Glue dal Catalog, con database e tabelle, al Crawler che ci aiutano nell'integrazione di sorgenti esterne con il nostro Data Lake.

Inoltre abbiamo mostrato come è possibile gestire i dati di input con code SQS e task ECS. Abbiamo dato qualche spunto di riflessione sulle tecniche di processing ma, di solito, sono molto specifiche in base al tipo di richiesta business.

Infine, abbiamo delineato come eseguire query con AWS Athena e come creare e condividere dashboard con QuickSight per la visualizzazione del dato.

Un'ultima cosa: i flussi di analisi dati sono spesso specifici in base al contesto, sentitevi liberi di sperimentare in ogni step.

Da sorgenti di dati differenti, come database, a ogni tipo di trasformazione del dato che potrebbe venirvi in mente; potreste provare con diversi servizi di processing come Lambda che, potenzialmente, potrebbe essere la soluzione più adatta per alcuni flussi.

La visualizzazione del dato inoltre è lo step in cui si può sperimentare ancora di più utilizzando al massimo la vostra creatività. Provate a creare visualizzazioni dai diagrammi standard a plot multi-linea con filtri complessi.

L'importante è che le dashboard aiutino ad avere una chiara comprensione dei dati analizzati e dei risultati da essi estratti.

Facci sapere se ti è piaciuto l'articolo e vuoi saperne di più sugli argomenti trattati!

Seguici per non perderti i nostri prossimi articoli su ETL e Data analytics.

About Proud2beCloud

Proud2beCloud è il blog di beSharp, APN Premier Consulting Partner italiano esperto nella progettazione, implementazione e gestione di infrastrutture Cloud complesse e servizi AWS avanzati. Prima di essere scrittori, siamo Solutions Architect che, dal 2007, lavorano quotidianamente con i servizi AWS. Siamo innovatori alla costante ricerca della soluzione più all'avanguardia per noi e per i nostri clienti. Su Proud2beCloud condividiamo regolarmente i nostri migliori spunti con chi come noi, per lavoro o per passione, lavora con il Cloud di AWS. Partecipa alla discussione!

Matteo Goretti

DevOps Engineer @ beSharp. Appassionato di Cloud Computing e Intelligenza Artificiale, in particolare, Machine Learning e Deep Learning. Amo il trekking e la natura in generale. Mi rilasso con la mia chitarra, giocando ai videogames o guardando serie TV.
