Kafka

Kafka

Che cos’è?

Capiamone le basi

- Prima di discutere i dettagli di Apache **Kafka**, è importante comprendere il concetto di messaggistica publisher/subscriber e il motivo per cui è importante.

- Un sistema Publish-Subscribe è essenzialmente costituito da un insieme di client che scambiano eventi tra di loro in modo asincrono comportandosi, secondo il caso, da **produttori** o **consumatori** di informazione.

- Un’entità logica, chiamata event notification service (ENS), agisce come mezzo di interconnessione, garantendo il disaccoppiamento tra i publisher ed i subscriber.

Kafka

Che cos’è?

Capiamone le basi

- Apache Kafka è una piattaforma di streaming di dati open source che consente a un gran numero di applicazioni di elaborare e analizzare i dati in tempo reale.
- La sua architettura è basata su un modello **publisher-subscriber**, dove i produttori pubblicano messaggi su un determinato topic e i consumatori si iscrivono a quel topic per ricevere i messaggi.
- Kafka è composto da diversi componenti principali, tra cui il broker, il topic, il producer e il consumer.

Kafka

Che cos’è?

Capiamone le basi

- In Kafka, l'unità di basè è il messaggio.
- Un messaggio è semplicemente un array di byte per Kafka, il che significa che i dati contenuti al suo interno non hanno un formato o un significato specifico per Kafka.
- Un messaggio può contenere un bit di metadati opzionale, chiamato chiave.
- Le chiavi vengono utilizzate quando i messaggi devono essere scritti in specifiche partizioni.

Kafka

Che cos’è?

Messaggi e batch

- Per efficienza, i messaggi vengono scritti in Kafka in batch.
- Un batch è semplicemente una raccolta di messaggi, tutti i quali vengono prodotti per lo stesso topic e partizione.
- Naturalmente, l’utilizzo dei batch è un compromesso tra latenza e throughput: l'aumento delle dimensioni dei batch consente di gestire più messaggi per unità di tempo, ma aumenta il tempo necessario per la propagazione di un singolo messaggio.

Kafka

Che cos’è?

Schemi

- Ci sono molte opzioni disponibili per lo schema dei messaggi, a seconda delle esigenze specifiche dell'applicazione.
- Sistemi semplici, come JSON (JavaScript Object Notation) e XML (Extensible Markup Language), sono facili da usare e leggibili dall'uomo.
- Tuttavia, mancano di funzionalità come la gestione robusta dei tipi e la compatibilità tra le diverse versioni dello schema.
- Molti sviluppatori di Kafka preferiscono l'uso di Apache Avro, che è un framework di serializzazione originariamente sviluppato per Hadoop.

Json vs Avro

```json
{
  "persona": {
    "nome": "Mario",
    "cognome": "Rossi",
    "età": 30,
    "indirizzo": {
      "via": "Via Roma, 123",
      "città": "Roma",
      "CAP": "00100"
    },
    "contatti": [
      {
        "tipo": "email",
        "valore": "mario.rossi@example.com"
      },
      {
        "tipo": "telefono",
        "valore": "+39 123456789"
      }
    ]
  }
}
```

```json
{
  "type": "record",
  "name": "Persona",
  "fields": [
    {"name": "nome", "type": "string"},
    {"name": "cognome", "type": "string"},
    {"name": "età", "type": "int"},
    {
      "name": "indirizzo",
      "type": {
        "type": "record",
        "name": "Indirizzo",
        "fields": [
          {"name": "via", "type": "string"},
          {"name": "città", "type": "string"},
          {"name": "CAP", "type": "string"}
        ]
      }
    },
    {
      "name": "contatti",
      "type": {
        "type": "array",
        "items": {
          "type": "record",
          "name": "Contatto",
          "fields": [
            {"name": "tipo", "type": "string"},
            {"name": "valore", "type": "string"}
          ]
        }
```

Kafka

Che cos’è?

Schemi

- Un formato dati consistente è importante in Kafka, poiché consente di gestire meglio la trasmissione dei messaggi.
- Ecco alcune ragioni per cui un formato di dati consistente è importante in Kafka:
    - **Interoperabilità.**
    - **Affidabilità del Consumatore.**
    - **Evolutività del Sistema.**
    - **Debugging e Monitoraggio**
    - **Scalabilità.**

Kafka

Che cos’è?

Topic e partizioni

- I messaggi in Kafka sono categorizzati in topic.
- I topic sono suddivisi in un certo numero di partizioni.
- In Kafka, una partizione è una porzione di un topic. I topic sono categorie logiche che raggruppano i messaggi all'interno di Kafka. Ogni partizione di un topic è un registro di eventi ordinati e immutabili.
- Ogni messaggio in una partizione ha un offset univoco, che rappresenta la sua posizione all'interno della partizione.

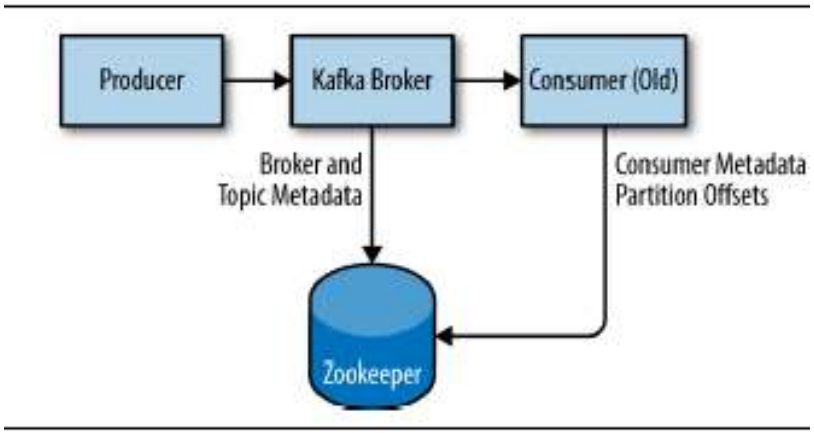
Kafka

Che cos’è?

Topic e partizioni

- I messaggi vengono scritti in esso in modo append-only e vengono letti in ordine dall'inizio alla fine.
- È importante notare che, poiché un topic ha tipicamente più partizioni, non c'è alcuna garanzia sull'ordinamento temporale dei messaggi sull'intero topic, ma solo all'interno di una singola partizione.
- Le partizioni sono anche il modo in cui Kafka fornisce ridondanza e scalabilità. Ciascuna partizione può essere ospitata su un server diverso, il che significa che un singolo topic può essere scalato in modo orizzontale su più server per fornire prestazioni ben oltre le capacità di un singolo server.

Installazione Kafka



Installazione Kafka

Come procedere?

Prerequisiti

- **Scegliere un sistema operativo:** Apache Kafka è un'applicazione Java e può essere eseguita su molti sistemi operativi, tra cui Windows, MacOS, Linux e altri. Linux è il sistema operativo consigliato per distribuire Kafka per un utilizzo generale.

- **Java:** Java è un linguaggio di programmazione ampiamente utilizzato per codificare le applicazioni Web. Per oltre vent'anni è stata una scelta molto diffusa tra gli sviluppatori, tanto che a oggi vengono utilizzate milioni di applicazioni Java. Java è un linguaggio multi-piattaforma, orientato agli oggetti. Si necessita versione di Java >= 8

- **Zookeeper:** Apache Kafka utilizza Zookeeper per memorizzare i metadati sul cluster Kafka, nonché i dettagli dei client consumer, come mostrato nella figura sulla sinistra.

Installazione Kafka

Cos’è ZooKeeper

A cosa serve?

- **ZooKeeper** è utilizzato da diversi progetti open-source per fornire un piano di controllo altamente affidabile per la coordinazione distribuita di applicazioni cluster attraverso uno store gerarchico di chiavi e valori.

- La serie di servizi offerti da ZooKeeper include servizi di configurazione distribuita, servizi di sincronizzazione, servizi di elezione di leadership e un registro di denominazione.

- Un concetto fondamentale di ZooKeeper è lo **znode**.

Installazione Kafka

Cos’è ZooKeeper

zNode

- Uno znode di ZooKeeper è un nodo dati che è tracciato da una struttura di stato che include modifiche dei dati, modifiche delle ACL, un numero di versione e un timestamp.
- Il numero di versione, insieme al timestamp, consente a ZooKeeper di coordinare gli aggiornamenti e memorizzare informazioni nella cache.
- Ogni modifica ai dati di uno znode comporta una modifica della versione, che viene utilizzata per assicurarsi che le modifiche allo znode siano applicate correttamente.

Installazione Kafka

Cos’è ZooKeeper

Come lavorano Kafka e ZooKeeper?

- Kafka e ZooKeeper lavorano in collaborazione per formare un cluster Kafka completo, con ZooKeeper che fornisce i suddetti servizi di clustering distribuito, mentre Kafka gestisce i flussi di dati effettivi e la connettività con i client.

- A un livello dettagliato, ZooKeeper si occupa dell'elezione della leadership dei broker Kafka e gestisce la topologia del cluster, in modo che ogni broker sappia quando i broker entrano o escono dal cluster, quando un broker fallisce e chi è il nodo leader preferito per una coppia di topic/partizione specifica.

Installazione Kafka

Cos’è ZooKeeper

Funzionalità ZooKeeper

- **Elezione del controllore**: Il controller di Kafka è un broker eletto assegnato da ZooKeeper, responsabile della gestione delle partizioni, della leadership delle coppie partizione-log e delle repliche, oltre a svolgere mansioni generali di manutenzione del cluster, come assegnazioni delle partizioni e la gestione di una lista di repliche in sincronia (ISR).

- **ACL**: Apache Kafka è dotato di un autorizzatore integrato che sfrutta ZooKeeper per memorizzare liste di controllo degli accessi o ACL. Le ACL sono un modo conveniente per gestire l'accesso alle risorse di Kafka, come i topic e i gruppi di consumatori.

Installazione Kafka

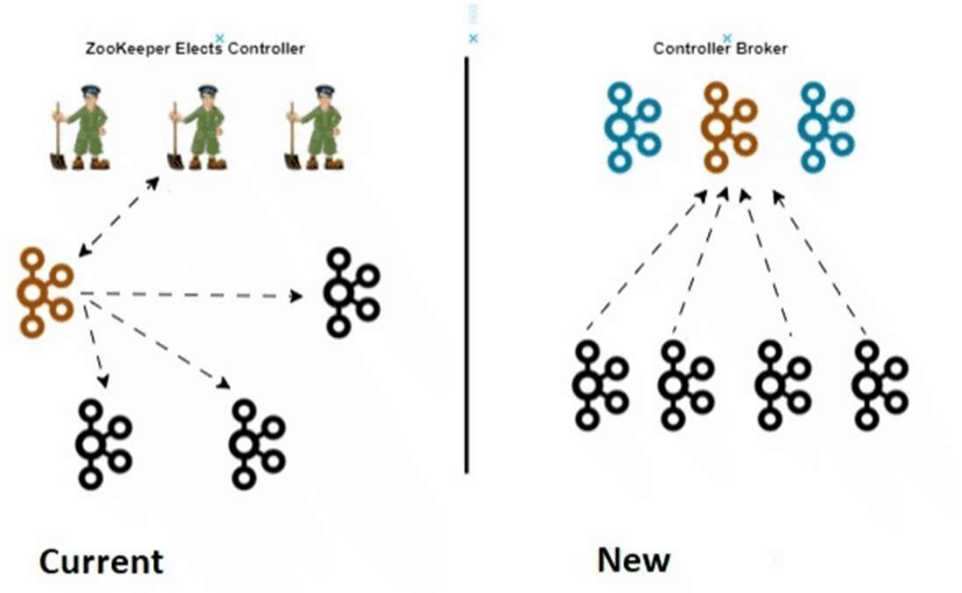
Cos’è ZooKeeper

Funzionalità ZooKeeper

- **Gestione risorse (quote) :** I broker di Kafka hanno la capacità di controllare le risorse del broker utilizzate dai client attraverso le quote.

  Kafka implementa due principali tipi di quote per i client a questo scopo.

  Le quote sulla larghezza di banda di rete sono definite da una soglia di byte-rate e una quota sulla frequenza delle richieste che è definita da una soglia di utilizzo della CPU come percentuale dei thread di I/O di rete.



Installazione Kafka

Perché utilizzare Kafka?

Ma Kafka senza ZooKeeper?

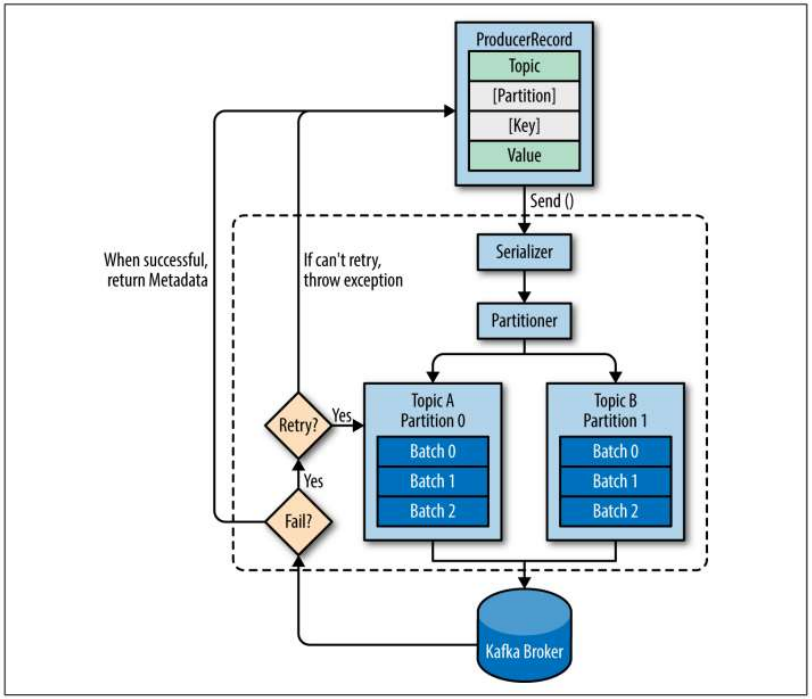
- Con le versioni più recenti di Kafka, è stato introdotto il KIP-500 (o "ZooKeeper Removal") nelle versioni principali.
- Tuttavia, è ancora considerato un elemento in "accesso anticipato" e non è pronto per la produzione a causa di alcuni problemi persistenti. Al momento, non è ancora supportato l'aggiornamento di un cluster basato su ZooKeeper, e manca ancora il supporto per alcune configurazioni dei topic.
- Tenendo presente ciò, ZooKeeper continuerà ad essere rilevante per i cluster di produzione per un po' di tempo.

Produttori (Producer)

Produttori

Kafka producer?

Iniziamo!



- Quando si inizia a produrre messaggi su kafka si utilizza un **ProducerRecord** ,che deve includere il topic al quale si vuole inviare il record e un valore. Opzionalmente, è possibile anche specificare una chiave e/o una partizione.

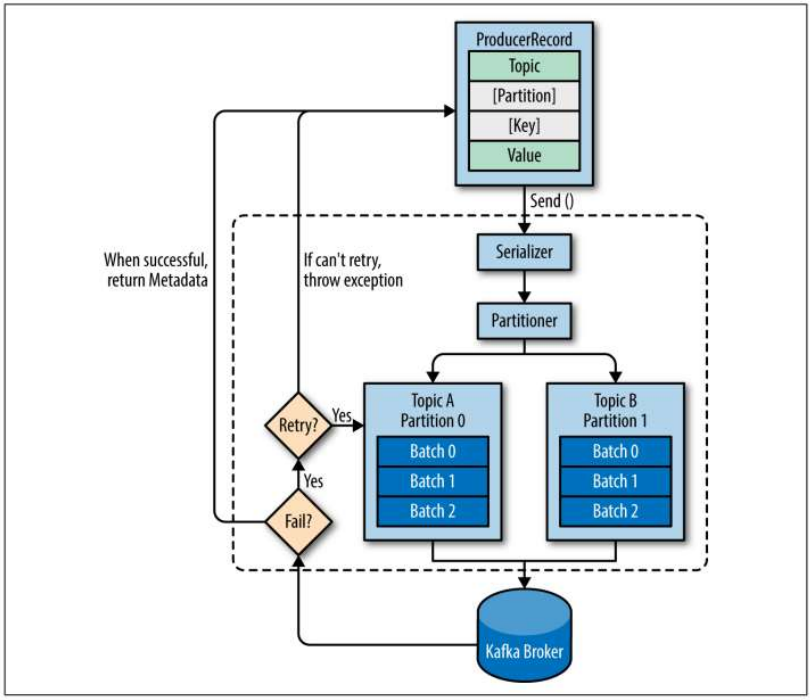
- Successivamente, i dati vengono inviati a un **partizionatore**.

- Se è stato specificato una partizione nel ProducerRecord, il partizionatore non fa nulla e restituisce semplicemente la partizione che specificata. Se non è stato specificato il partizionatore allora, il partizionatore sceglierà una partizione automaticamente, di solito in base alla chiave del ProducerRecord.

Produttori

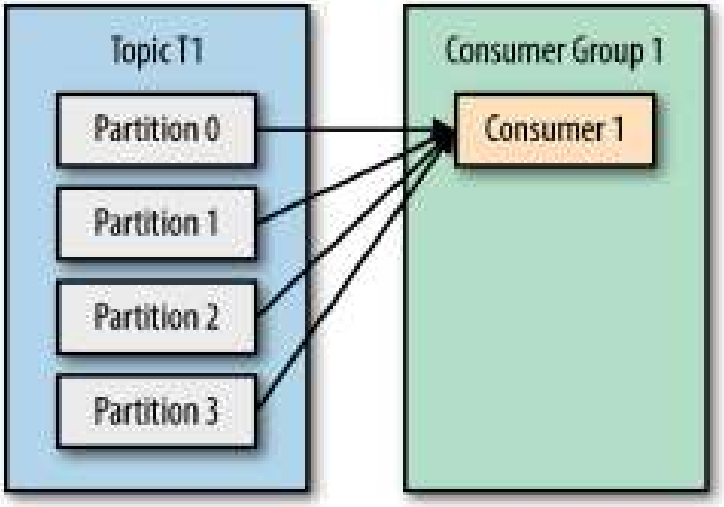
Kafka producer?

Iniziamo!



- Quando il broker riceve i messaggi, invia una risposta. Se i messaggi sono stati scritti con successo su Kafka, restituirà un oggetto RecordMetadata con il topic, la partizione e l'offset del record all'interno della partizione. Se il broker non è riuscito a scrivere i messaggi, restituirà un errore. Quando il produttore riceve un errore, può tentare di inviare il messaggio ancora qualche volta prima di rinunciare e restituire un errore.

Consumatori (Consumer)

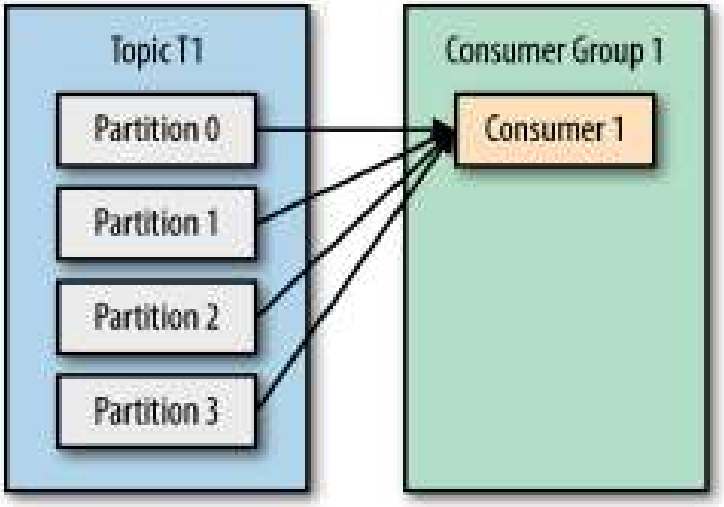


Consumatori

Kafka Consumer?

Iniziamo

- Al cuore dell'API dei consumer c'è un semplice ciclo per richiedere dati aggiuntivi al server. Una volta che il consumatore si iscrive ai topic, il ciclo di poll gestisce tutti i dettagli della coordinazione, dei riequilibri delle partizioni, degli "heartbeats" e dell'acquisizione dei dati, lasciando al programmatore un'API pulita che restituisce semplicemente i dati disponibili dalle partizioni assegnate.



Consumatori

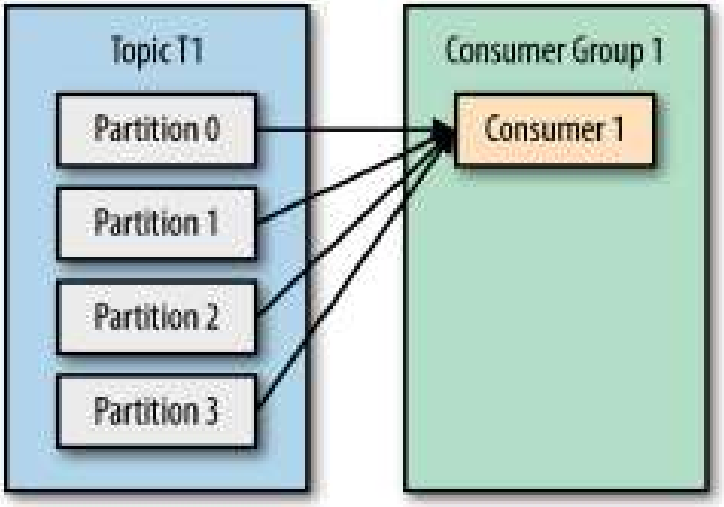
Kafka Consumer?

Iniziamo

- Le applicazioni che hanno bisogno di leggere dati da Kafka utilizzano un KafkaConsumer per iscriversi ai topic di Kafka e ricevere messaggi da questi topic. Leggere dati da Kafka è un po' diverso rispetto alla lettura di dati da altri sistemi di messaggistica, e ci sono pochi concetti e idee uniche coinvolti.

Concetti base

- **Consumer e Gruppi di consumer**:L'applicazione creerà un oggetto consumatore, si iscriverà al topic appropriato e inizierà a ricevere messaggi, convalidandoli e scrivendo i risultati.



Consumatori

Kafka Consumer?

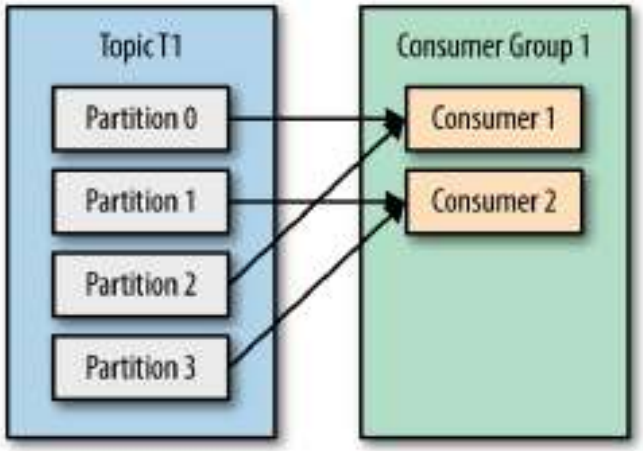
Concetti base

- I consumatori di Kafka sono tipicamente parte di un gruppo di consumatori.
- Quando diversi consumatori sono iscritti a un topic e fanno parte dello stesso gruppo di consumatori, ogni consumatore nel gruppo riceverà messaggi da un sottoinsieme diverso delle partizioni nel topic.
- Si prenda ad esempio il topic T1 con quattro partizioni raffigurato sulla sinistra.
- Si supponga ora di aver creato un nuovo consumatore, C1, che è l'unico consumatore nel gruppo G1, e lo utilizziamo per iscriverci al topic T1.

Consumatori

Kafka Consumer?

Concetti base



- Se si aggiunge un altro consumatore, C2, al gruppo G1, ogni consumatore riceverà solo i messaggi da due partizioni.