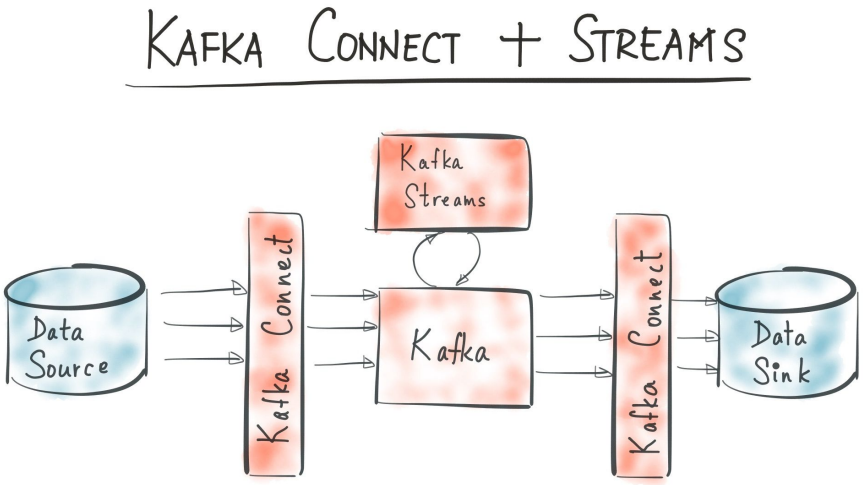
Kafka Streams

Kafka Streams - uvod

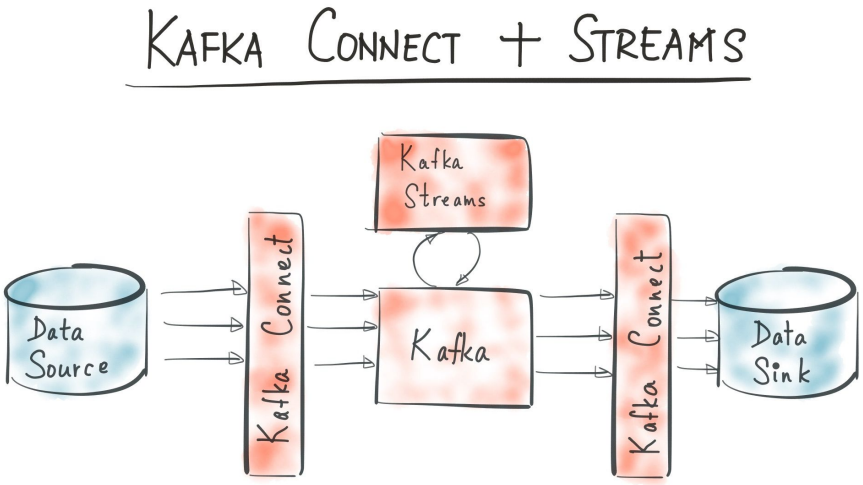
- Jednostavna i “lagana” klijentska biblioteka za kreiranje servisa za obradu tokova podataka.
- Ulazni i izlazni podaci se skladište na *Kafka* klasteru.
- Umesto da se pišu namenski obrađivači toka podataka, koriste se Kafka *producer*-i i *consumer*-i.
  - To znači da će paralelizam, distribuirana koordinacija i otpornost na otkaze biti nativno podržani.



source: https://www.confluent.io/blog/hello-world-kafka-connect-kafka-streams/

Kafka Streams - prednosti

- Nije potreban zaseban klaster za obradu podataka.
- Podržana *record-at-a-time* obrada (ne vrši se micro-batching).
- Sve prednosti kafke nativno podržane:
  - uređivanje, particionisanje, skalabilnost, otpornost na otkaze…
- Lako rukuje zakasnelim i out-of-order podacima.

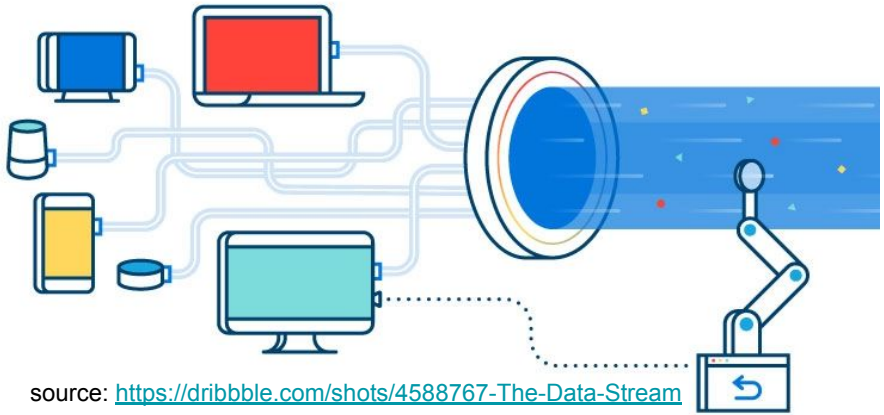


source:
https://www.confluent.io/blog/hello-world-kafka-connect-kafka-streams/

Key Idea: **Outsource hard problems to Kafka**

Kafka Streams - osnovni koncepti

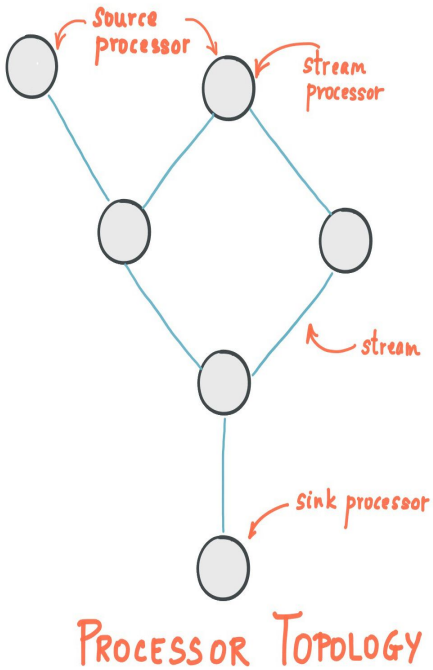
- *Data Stream*
  - neograničeni, kontinualno ažurirani skup podataka;
  - sekvenca nepromenljivih zapisa podataka (*immutable data records*) koja je uređena, *replayable* i otporna na otkaze .
- *Data record* - par ključ-vrednost
- Topologija za obradu toka podataka
  - usmereni aciklični graf izvora i obrađivača, povezanih u cilju analize i obrade podataka



source: https://dribbble.com/shots/4588767-The-Data-Stream

Kafka Streams - topologija za obradu toka podataka

- Svaki program koji koristi Kafka Streams biblioteku zapravo definiše logiku za izvršavanje kroz jednu ili više topologija obrađivača
  - gde je topologija obrađivača usmereni aciklični graf obrađivača toka (čvorova) povezanih tokovima podataka (ivicama).
- Obrađivači toka (*Stream processors*) - čvorovi u topologiji obrađivača; predstavljaju korak obrađivanja kojim se podaci toka transformišu na sledeći način:
  - ulazni podaci se prihvataju *record-at-a-time*;
  - nad njima se vrše specificirane operacije obrađivača;
  - ukoliko ima potrebe, transformisani podaci se šalju nizvodno, na obrađivanje od strane drugih obrađivača.
  - Podvrste - *Source processor* i *Sink processor*



source: https://kafka.apache.org/23/images/streams-architecture-topology.jpg

Kafka Streams - Kafka Streams DSL

- Kako bi se definisala topologija za obradu toka podataka, može se koristiti *Kafka Streams DSL* (*domain specific language*)
  - apstrakcija nad *Stream Processor API*-jem, koja omogućava izražavanje operacija obrađivača na deklarativan način uz korišćenje funkcionalnog programiranja.
  - Pruža mogućnost korišćenja *KStream*, *KTable* i *GlobalKTable* apstrakcija za tokove podataka i tabele.
  - Podržava i *stateless* (npr. mapiranje, filtriranje) i *stateful* (agregacije, spajanja, klizni okviri...) transformacije.

Kafka Streams - primeri

- *Message Broker*: Kafka - kafka klaster sa dva brokera (+ kafka-ui za vizualizaciju saobraćaja)
- *Producer* - Data-Generator aplikacija koja osluškuje mrežni saobraćaj
- *Consumers* - Kafka-Streams - java programi koji koriste kafka streams biblioteku za obradu podataka o mrežnom saobraćaju

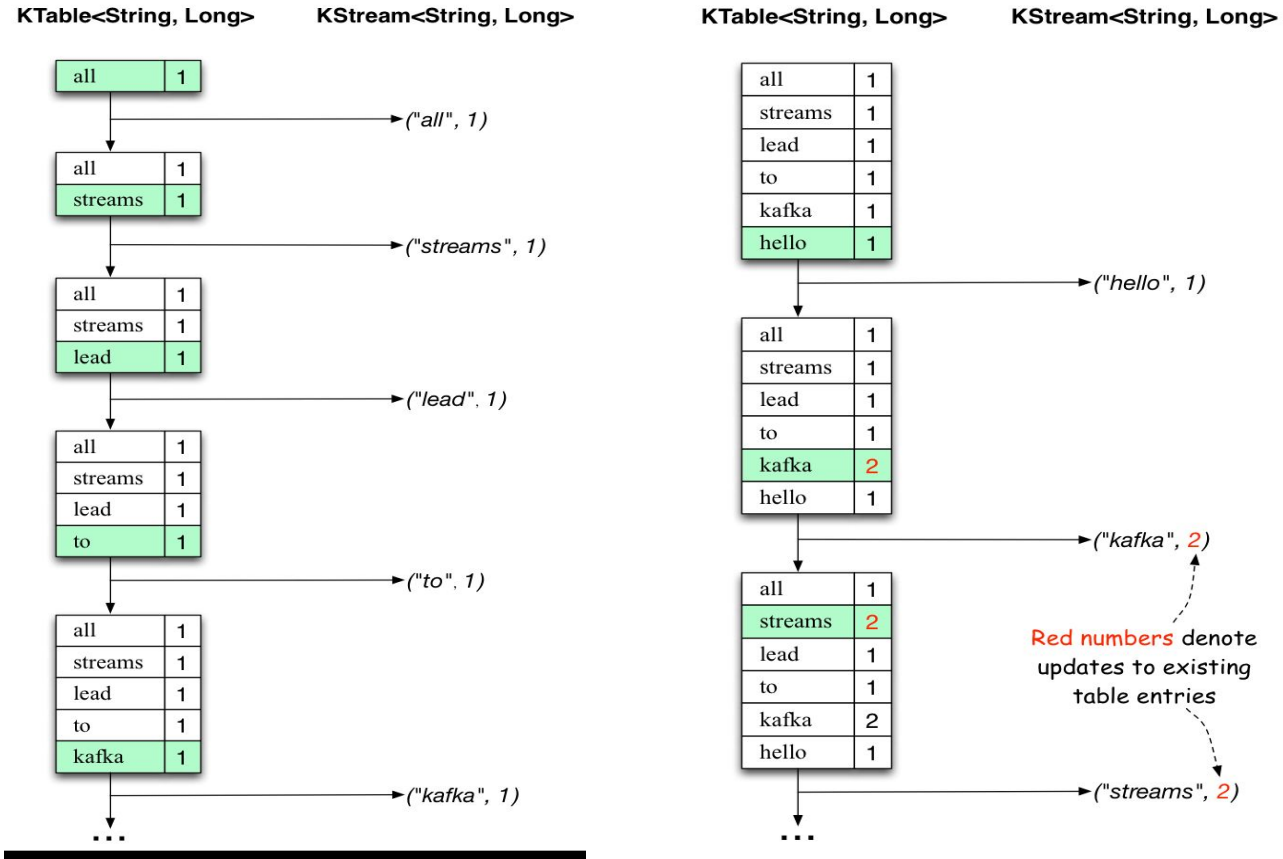
Kafka Streams - *CountByDestination* primer

- *Consumer* prihvata podatke o pojedinačnim paketima na mreži, te vrši grupisanje i prebrajanje po destinaciji paketa.
- Pokretanje primera:
  - Izmeniti docker-compose tako da se pokreće odgovarajući program
  - Pokretanje aplikacije:
    - `docker compose -f ".\Kafka-Streams\docker-compose.yml" up`

Kafka Streams - tokovi podataka i tabele

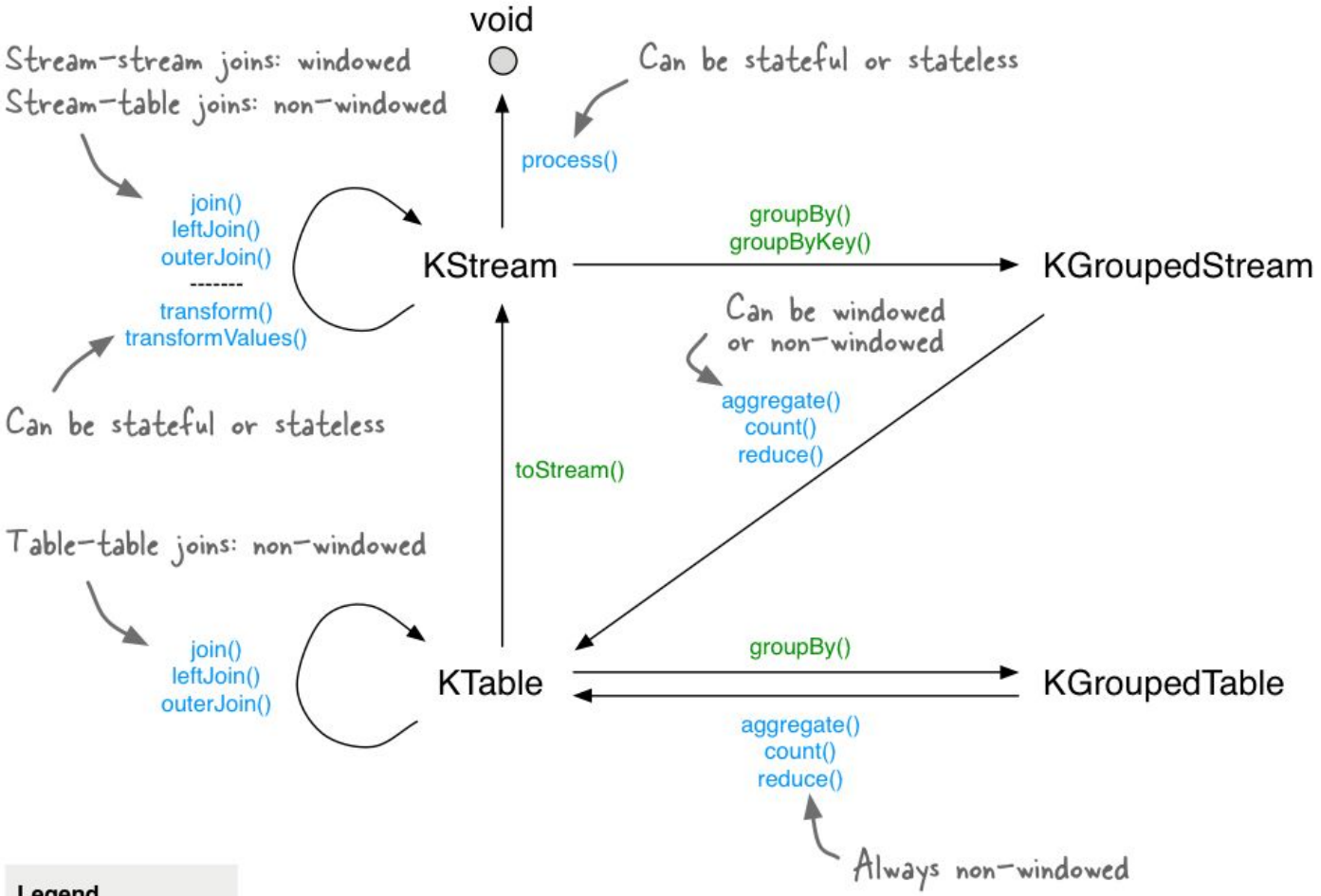
- *KStream* - predstavlja apstrakciju particionisanog toka zapisa (*record stream*), gde svaki zapis predstavlja pojedinačni podatak u neograničenom skupu podataka.
  - Podaci se u *KStream* mogu samo dodavati.
- *KTable* - predstavlja apstrakciju particionisanog toka zapisa promena (*changelog stream*), gde se svaki zapis tumači kao ažuriranje postojeće vrednosti u tabeli sa istom vrednošću ključa, ukoliko postoji (*upsert*).
  - *Null* vrednosti se tumače kao operacija brisanja zapisa iz tabele.
- *GlobalKTable* - kao *KTable*, ali će svaka instanca aplikacije dobijati podatke iz svih particija *topic*-a.

Kafka Streams - tokovi podataka i tabele



Kafka Streams - transformacije

- Izvori tokova podataka
  - input topics → KStream
  - input topics → KTable
  - input topics → GlobalKTable
- Transformacije nad tokovima podataka
  - Stateless
    - Branch
    - Filter
    - Inverse Filter
    - FlatMap
    - FlatMap (values only)
    - Foreach
    - GroupByKey
    - GroupBy
    - Map
    - Map (values only)
    - Merge
    - Peek
    - Print
    - SelectKey
    - Table to Stream
  - Stateful
    - Aggregate
    - Count
    - Reduce
    - Inner Join
    - Left Join
    - Outer Join



source: https://kafka.apache.org/23/documentation/streams/developer-guide/dsl-api.html

Kafka Streams - primeri

- *BranchByProtocol* - deljenje originalnog toka podataka na osnovu vrste poruke.
- *FilterByIP* - primer filtriranja na osnovu zadate IP adrese
- *AnalyzeValues* - primer računanja prosečne vrednosti za zadato polje
- *AnalyzeValuesWindowed* - prethodni primer izmenjen tako da se vrednosti računaju u okviru kliznih prozora
- *AnalyzeValuesWindowedCustomSerDes* - prethodni primer izmenjen tako da se koriste prilagođeni mehanizmi za (de)serijalizaciju

Kafka Streams - zadaci

- Koristeći *network_data topic* iz prethodnih primera, kreirati sledeće aplikacije za obradu toka podataka:
  - aplikacija koja izračunava ukupan broj bajta prenesen za svaki par *src-dst* IP adresa
  - aplikacija koja predstavlja izmenjenu verziju AnalyzeValues gde se vrednosti računaju za broj bajta prenesen za odabranu izvornu IP adresu
  - skalirati broj consumer-a u consumer grupi za primer *FilterByIP* na 2
    - Proveriti kakav uticaj ima na broj obrađivanih poruka, lag…