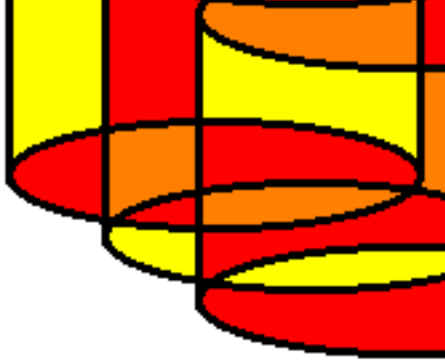
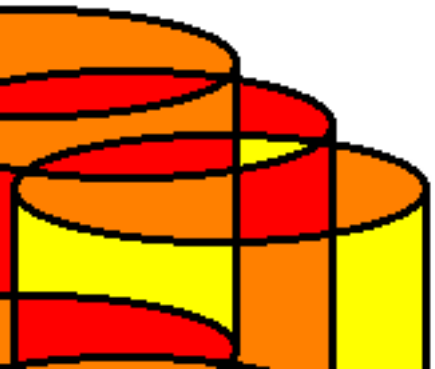
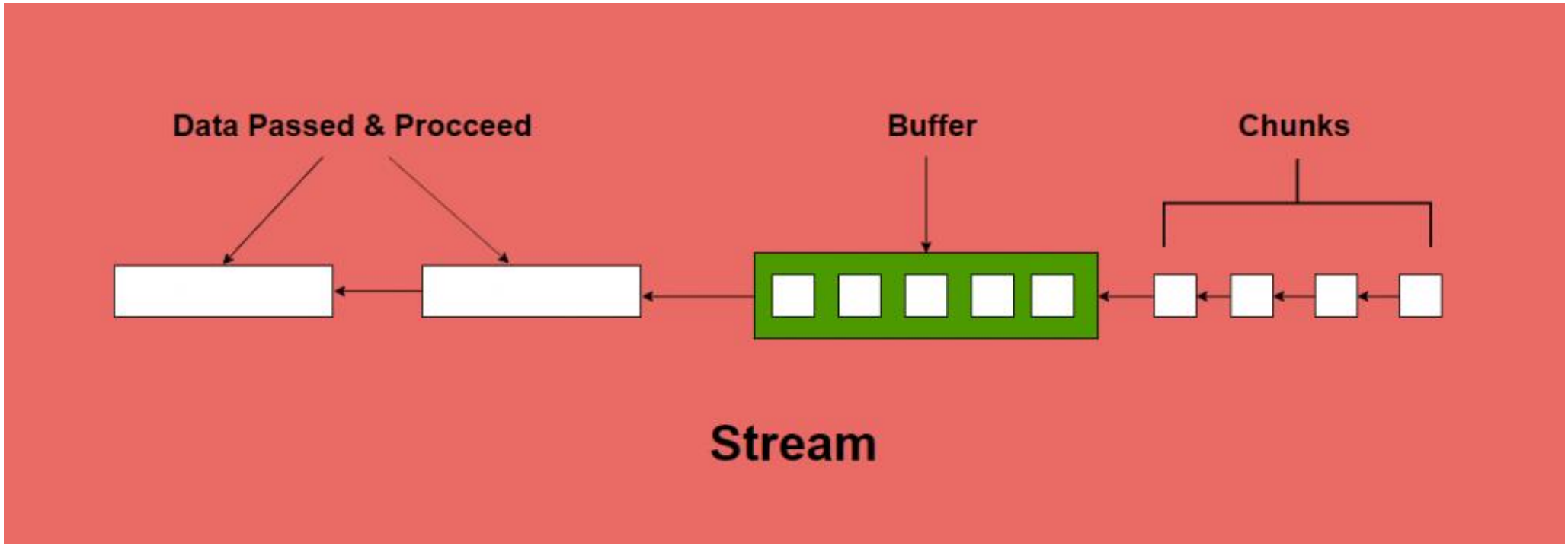
Tokovi podataka

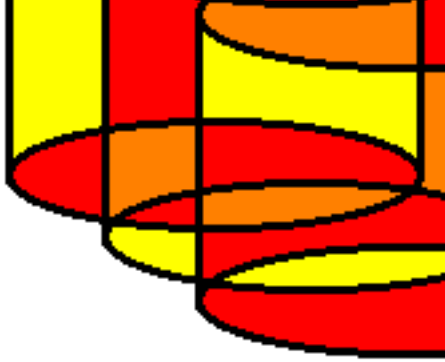


- Tok podataka (engl. *data stream*) označava prijenos i obradu podataka u dijelovima (engl. *chunk*)
- Ako je neki niz podataka prevelik, on se procesuira u manjim dijelovima
- Kada je dio podataka koji se trenutno procesuira u spremniku (engl. *buffer*) obrađen, dohvaća se sljedeći dio



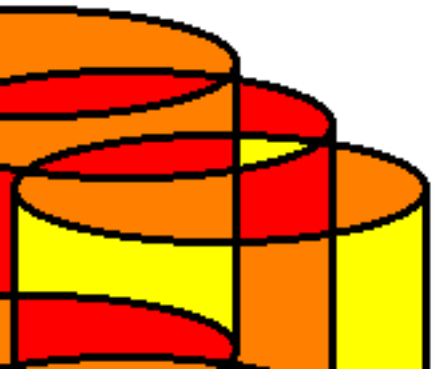
Tokovi podataka



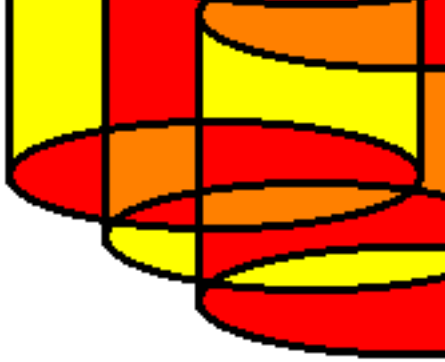


Primjeri

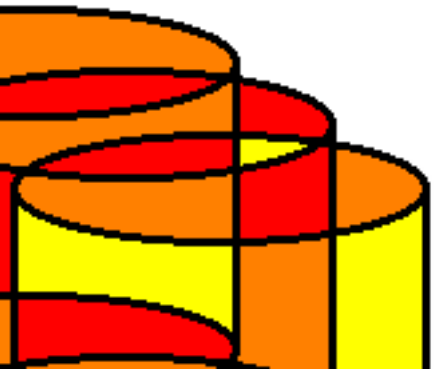
- Preuzimanje velike datoteke putem interneta (konačan skup podataka)
- Gledanje videozapisa na YouTube-u (konačan skup podataka)
- IoT senzori (beskonačan skup podataka)



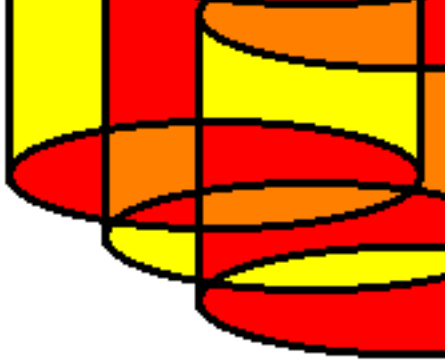
Tipovi podataka



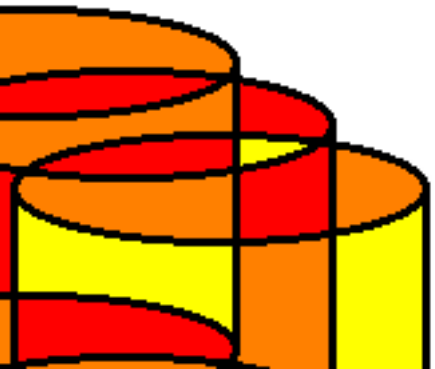
- Podatke možemo klasificirati kao **konačan** te **beskonačan** skup
- Konačan skup podataka su podaci čija veličina je unaprijed poznata
- Beskonačan skup podataka su podaci čija veličina nije unaprijed poznata te u tom slučaju se podaci generiraju i dohvaćaju kontinuirano



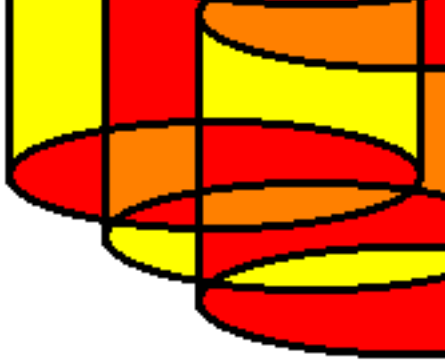
SQL tokovi podataka



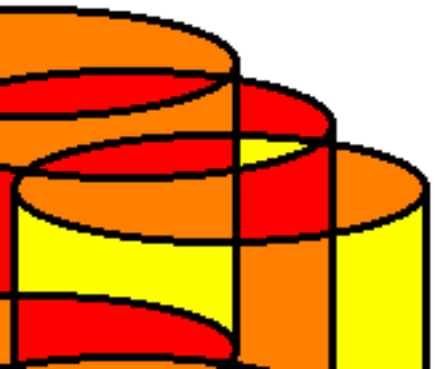
- Prethodnih godina SQL tokovi podataka postaju sve popularniji
- Omogućava zadavanje SQL upita koji kontinuirano dohvaća podatke iz BP



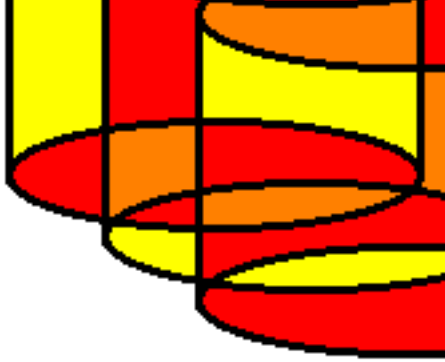
SQL BP s podrškom tokova podataka



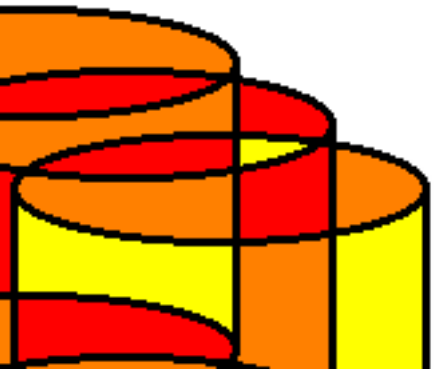
- **ksqlDB** - https://ksqldb.io
- **Samza SQL** - https://samza.apache.org/learn/documentation/latest/api/samza-sql.html
- **Storm SQL** - https://storm.apache.org/releases/2.1.0/storm-sql.html



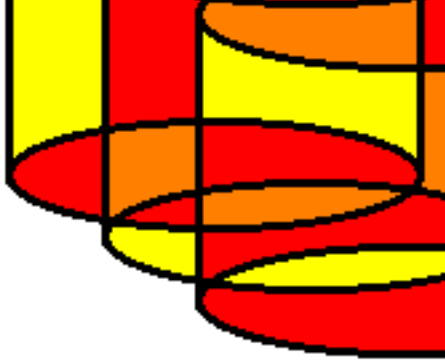
ksqlDB



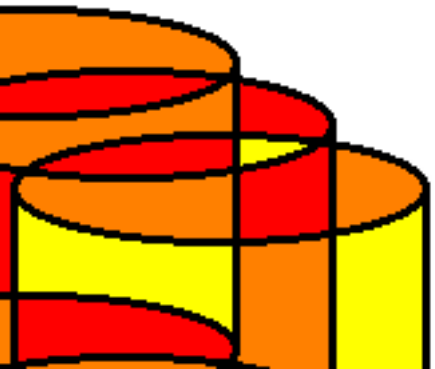
- U sklopu vježbi bavit ćemo se ksqlDB-om
- BP nudi standardne SQL funkcionalnosti i instrukcije uz neke dodatne koje su specifične za tokove podataka
- Bazira se na platformi **Apache Kafka**



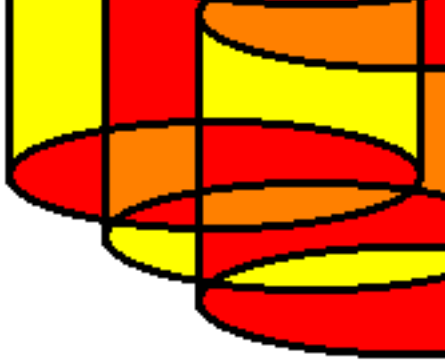
Apache Kafka



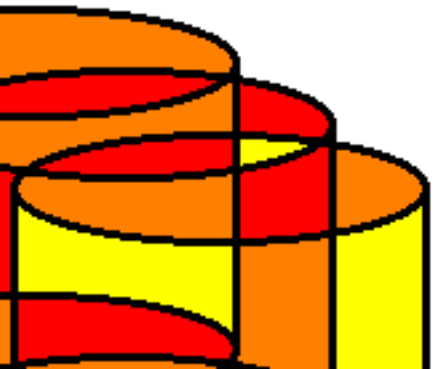
- Platforma koja omogućava procesuiranje tokova podataka
- Pruža bazi podataka API kako bi ona mogla komunicirati u stvarnom vremenu
- Većina BP koja nudi podršku tokovima podataka bazira se na Kafki



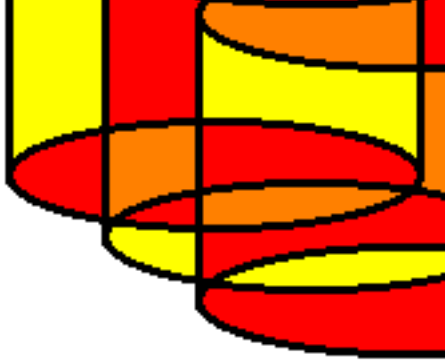
ksqlDB instrukcije



- **CREATE STREAM** – slično kao CREATE TABLE, služi za kreiranja toka podataka (što je zamjena za tablicu)
- Za razliku od relacijskih baza podataka, ova BP nudi spremanje podataka u različitim formatima (npr. JSON, Kafka, Protobuf…) – mi ćemo se fokusirati na **JSON**

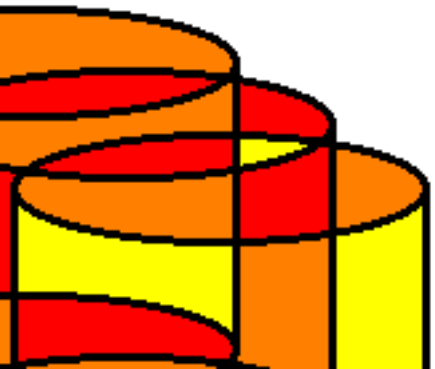


ksqlDB instrukcije

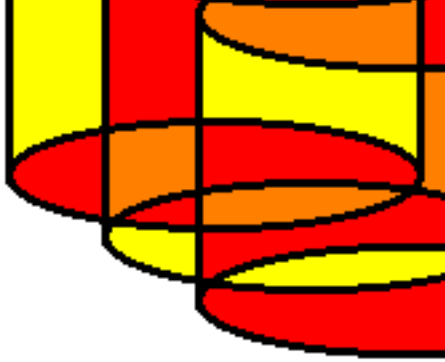


- Format spremanja podataka konfigurira se atributom **value_format**
- **kafka_topic** određuje naziv kanala za komunikaciju (proizvoljno)
- **partitions=1** ako kanal za komunikaciju ne postoji, tada je potrebno dodati partitions atribut kako bi se kanal kreirao

```
CREATE STREAM riderLocations (profileId VARCHAR, latitude DOUBLE, longitude DOUBLE) WITH
(kafka_topic='locations', value_format='json', partitions=1);
```

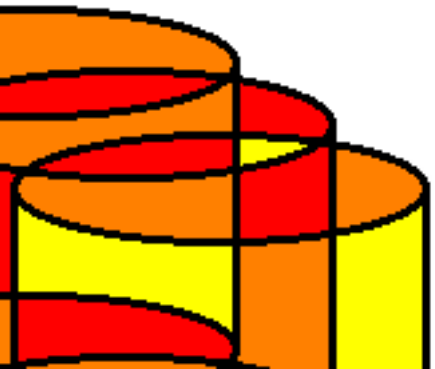


ksqlDB instrukcije

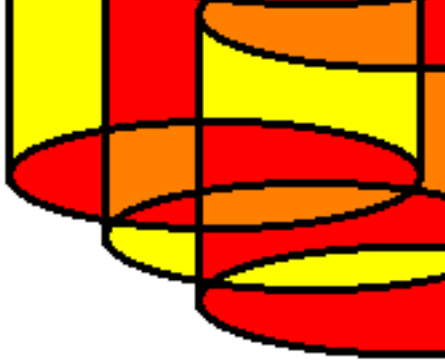


- **SELECT […] WITH CHANGES** – za čitanje iz toka podataka kontinuirano, potrebno je na kraj instrukcije dodati WITH CHANGES
- **WINDOW HOPPING** omogućava grupiranje podataka prema vremenu dospijeća

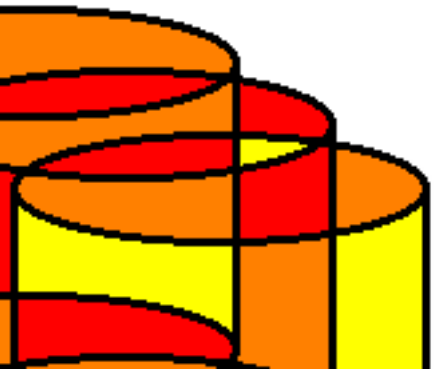
```
SELECT windowstart, windowend, item_id, SUM(quantity) FROM orders WINDOW HOPPING (SIZE
20 SECONDS, ADVANCE BY 5 SECONDS) GROUP BY item_id EMIT CHANGES;
```



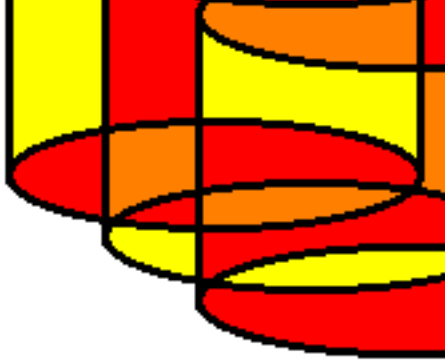
Instalacija



- Za lakšu instalaciju te konfiguraciju ksqlDB-a i Apache Kafka, koristit ćemo **Docker Compose**
- Datoteka **kafka-ksqldb.yml** služi Docker-u kao opisnik koje servise treba pokrenuti
- U priloženom opisniku su servisi potrebni za rad Apache Kafke te ksqlDB-a



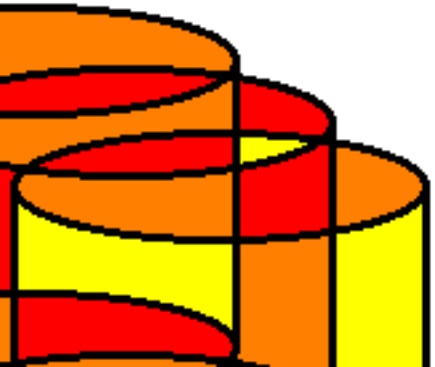
Instalacija



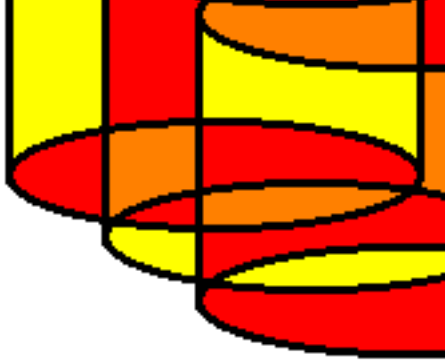
- Instrukcije za instalaciju Docker-a dostupne su na: https://docs.docker.com/compose/install/

```
sudo curl -L
"https://github.com/docker/compose/releases/download/1.27.4/docker-compose-
$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

docker-compose --version // za provjeru uspješne instalacije, ova instrukcija
treba vratiti poruku o verziji Docker Compose-a
```

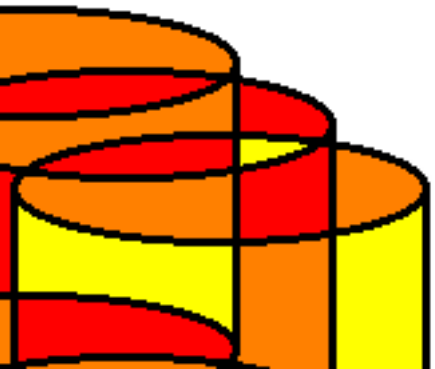
(Instrukcije za Linux)



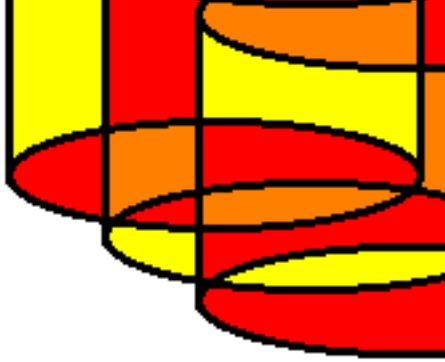
Instalacija



- Nakon uspješne Docker instalacije, na proizvoljnoj lokaciji kreirajte folder te u njega kopirajte **kafka-ksqldb.yml** datoteku (priložena uz prezentaciju)
- Nadalje je potrebno otvoriti terminal te se pozicionirati u novokreirani folder
- U terminal upišite **docker-compose -f kafka-ksqldb.yml up** za pokretanje Kafke i ksqlDB-a

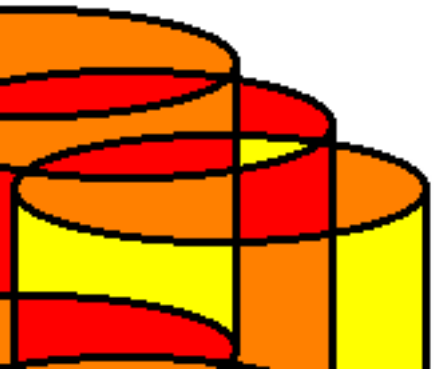


Primjer ksqlDB toka podataka

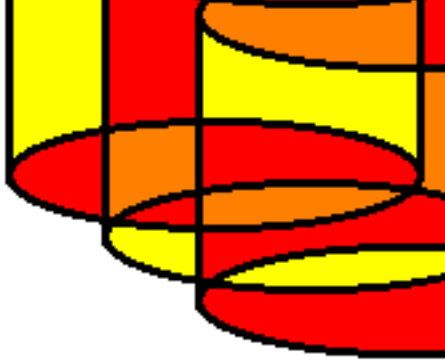


- Sljedeći primjer koji uključuje kreiranje toka podataka i čitanje iz istog dostupan je i na: https://ksqldb.io/quickstart.html#quickstart-content
- Kako bismo simulirali kreiranje retka i čitanje istih kontinuirano, potrebno je otvoriti dva terminala (jedan za kreiranje, jedan za čitanje/dohvaćanje) te u oba upisati instrukciju:

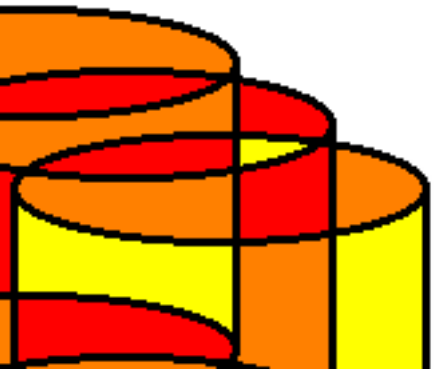
```
docker exec -it ksqldb-cli ksql http://ksqldb-server:8088
```



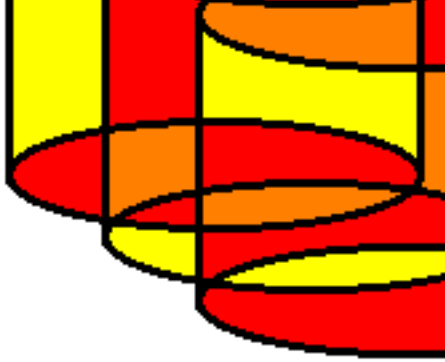
Primjer ksqlDB toka podataka



- Instrukcije priložene u nastavku upisujemo u zasebne terminale
- Najprije treba pokrenuti instrukcije u terminalu #1 (s obzirom da su one vezane za otvaranje i čitanje toka podataka), te nakon toga slijede instrukcije za terminal #2 (instrukcije vezane za upis u tok podataka)



Primjer ksqlDB toka podataka

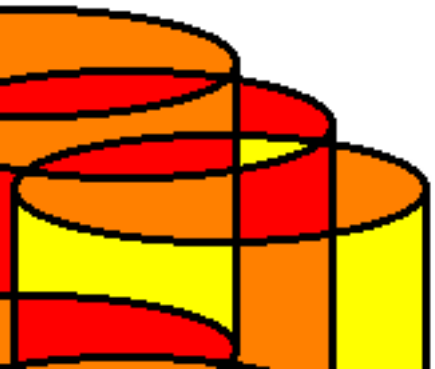


Terminal #1

```
CREATE STREAM riderLocations (profileId VARCHAR, latitude DOUBLE, longitude DOUBLE) WITH (kafka_topic='locations', value_format='json', partitions=1); // kreira tok podataka s pripadajućim atributima

SELECT * FROM riderLocations WHERE GEO_DISTANCE(latitude, longitude, 37.4133, -122.1162) <= 5 EMIT CHANGES; // upit koji pokreće kontinuirano dohvaćanje podataka
```

Terminal #2

```
INSERT INTO riderLocations (profileId, latitude, longitude) VALUES ('c2309eec', 37.7877, -122.4205);

INSERT INTO riderLocations (profileId, latitude, longitude) VALUES ('18f4ea86', 37.3903, -122.0643);

INSERT INTO riderLocations (profileId, latitude, longitude) VALUES ('4ab5cbad', 37.3952, -122.0813);

INSERT INTO riderLocations (profileId, latitude, longitude) VALUES ('8b6eae59', 37.3944, -122.0813);

INSERT INTO riderLocations (profileId, latitude, longitude) VALUES ('4a7c7b41', 37.4049, -122.0822);

INSERT INTO riderLocations (profileId, latitude, longitude) VALUES ('4ddad000', 37.7857, -122.4011);
```



Dokumentacija

- https://docs.ksqldb.io/en/latest/developer-guide/ksqldb-reference/