Kafka Fundamentals - Day 2

Plan

1. Apache Kafka vs. RabbitMQ vs. Apache Pulsar
2. Recap tego co było wczoraj + ew. Pytania
3. Schema Registry (AVRO)
4. Transakcje
5. Spring Framework - szybka wizyta
6. Metryki
7. Kafka Connect
8. Kafka Streams
9. Confluent - co to i co oferuje?

Apache Kafka vs. RabbitMQ vs. Apache Pulsar

Apache Kafka vs. RabbitMQ

Apache Kafka vs. RabbitMQ

Kafka

RabbitMQ

Apache Kafka vs. RabbitMQ

Kafka

- Nowsza koncepcja
- Dużo większa przepustowość
- Trwałość Danych
- Skalowalne

RabbitMQ

- Eventy przeprocesowane są od razu ściągane z topicu. Nie są trwałe
- Skalowalne (możemy mieć klaster Brokerów)
- Wolniejsze
- Możemy usuwać wiadomości

Apache Kafka vs. Apache Pulsar

Apache Kafka vs. Apache Pulsar

- Pulsar jest nowszy. Uczy się na błędach Kafki
- podobne koncepcje

Apache Kafka vs. Apache Pulsar

- Ciekawostka: Lepsza deduplikacja, nawet na “application level”

Apache Kafka vs. Apache Pulsar

- Ciekawostka: Lepsza deduplikacja, nawet na “application level”
- ale więcej flag do ustawiania xD

Recap tego co było wczoraj

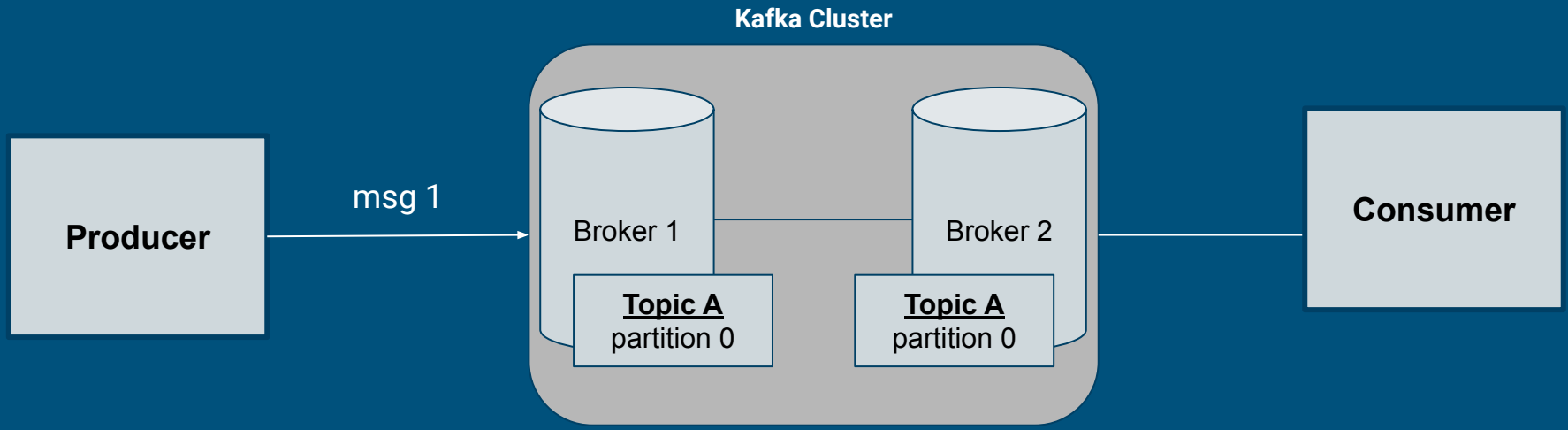
A co było wczoraj? Esencja

- Kafka ma naprawdę dużo flag, warto o nich czytać
- Co to Topic, jego Partycje, jego(ich) Replikacja
- Producer
  - warto rozważyć jakiego ACK potrzebujemy
  - co to ISR i jak może nam się przydać
  - co to Retry i czy chcemy włączyć
  - flaga enable.idempotence jest git
- Consumer
  - Consumer polluje recordy przez pewne okna czasowe
  - domyślnie commituje offsety co… max.poll.interval.ms (default 5 minutes)
  - ale że może to inaczej rozwiązać
- warto pamiętać o CLI i jego możliwościach
- od deduplikacji raczej nie uciekniesz. ale jak dalego w niej zajdziesz?
- At Least One, At Most One, Exactly once (święty Gral), Effectively once (processed)

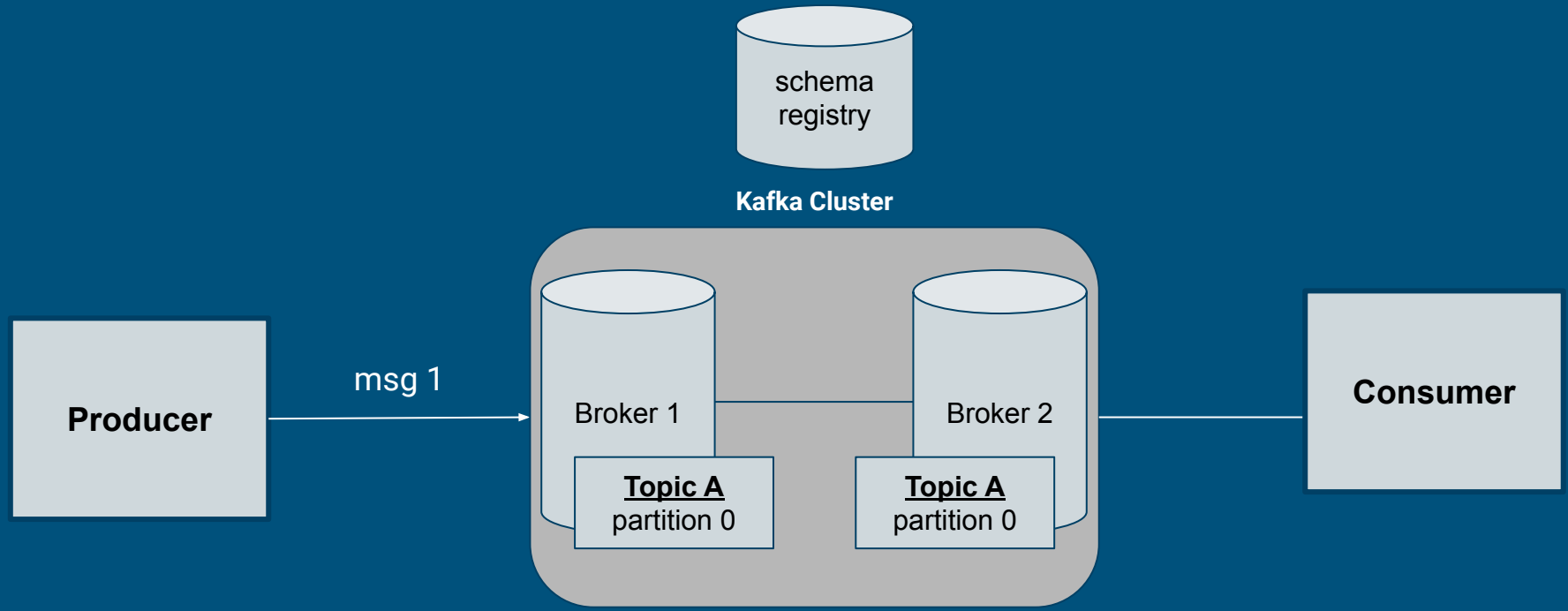
Jakieś pytania po wczoraj?

Schema Registry (AVRO)

Schema Registry (AVRO)

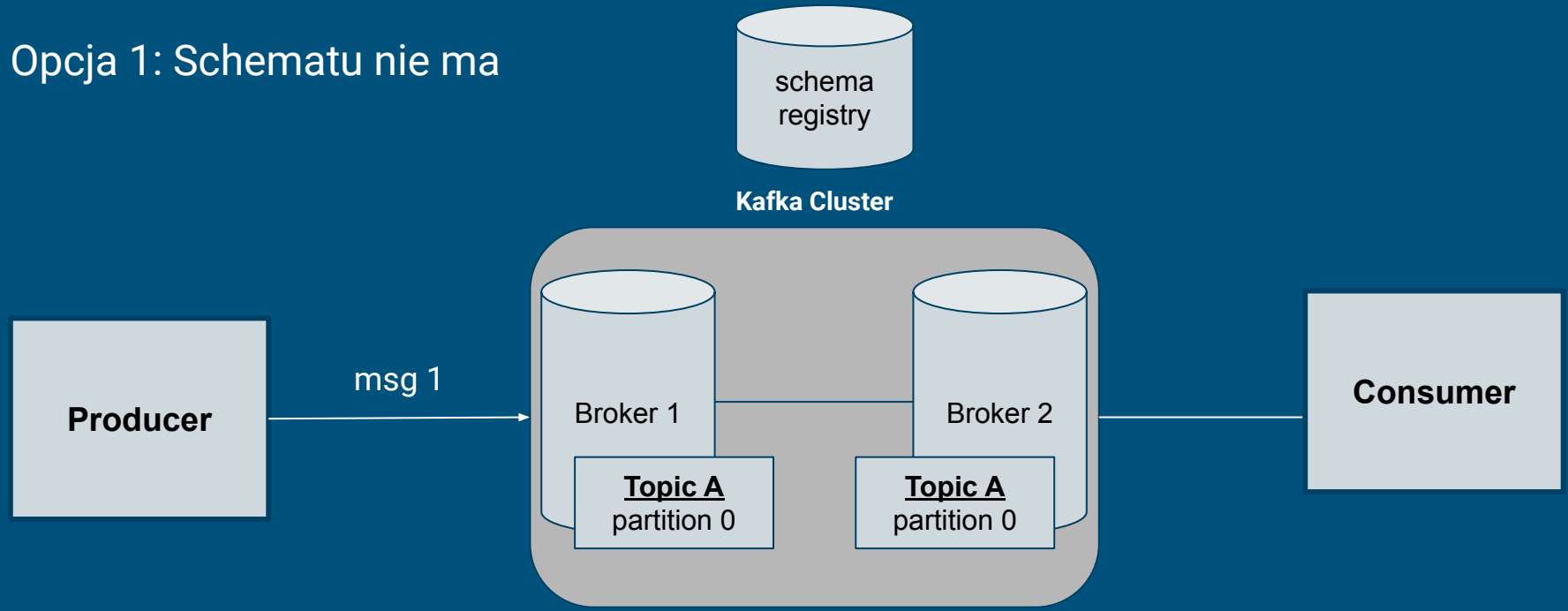


Schema Registry (AVRO)



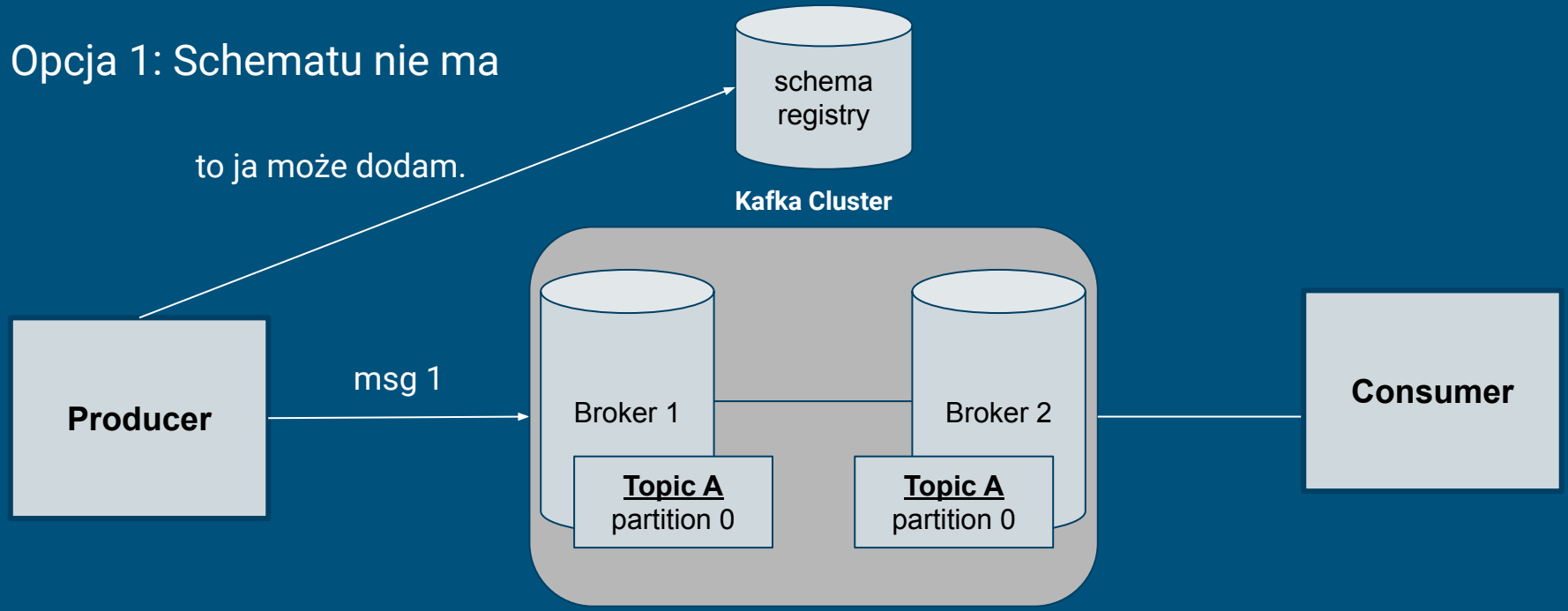
Schema Registry (AVRO)

Opcja 1: Schematu nie ma



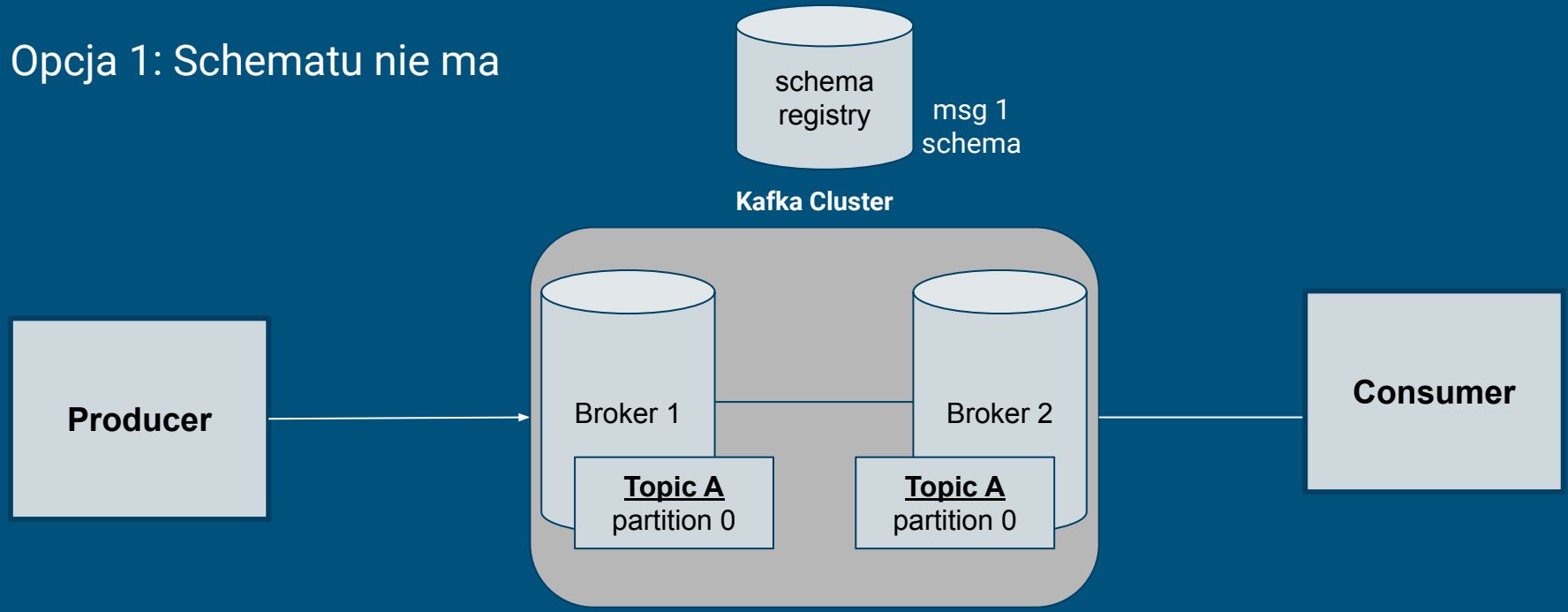
Schema Registry (AVRO)

Opcja 1: Schematu nie ma



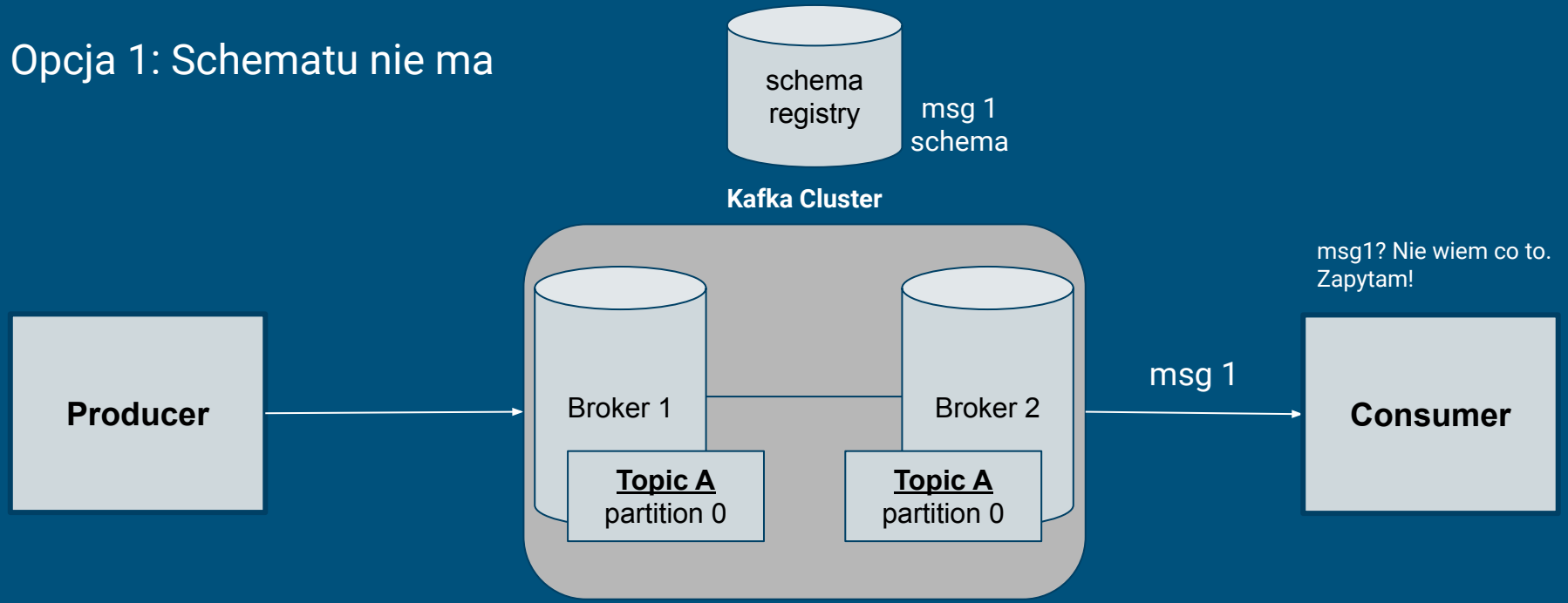
Schema Registry (AVRO)

Opcja 1: Schematu nie ma



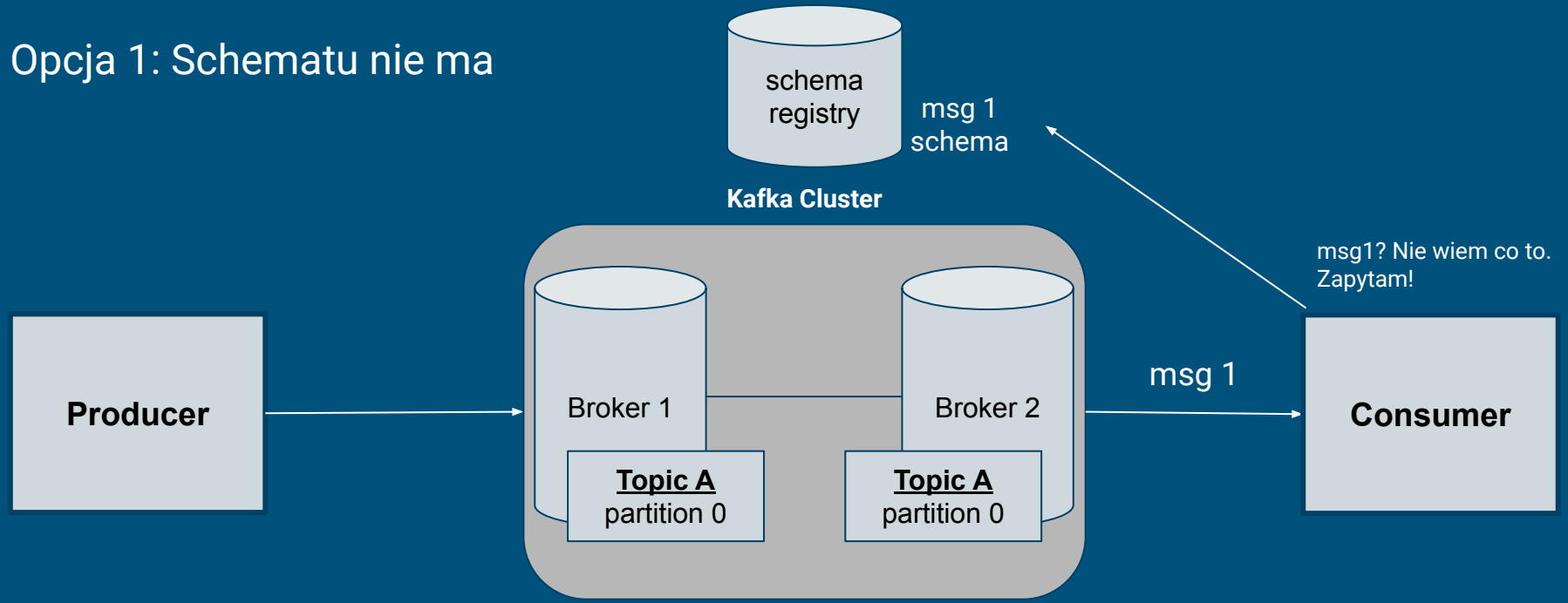
Schema Registry (AVRO)

Opcja 1: Schematu nie ma



Schema Registry (AVRO)

Opcja 1: Schematu nie ma



Schema Registry (AVRO)

- Po co AVRO?
  - większa szansa na forward i backward compatibility
  - zapis do danych binarnych ⇒ lepsze wykorzystanie przestrzeni na dysku
  - bez schematu Consumer sobie nie poradzi z odczytem
- Schema Registry przechowuje schematy na dedykowanym topicu
- SR to nie tylko AVRO - możemy je wymienić na JSON Schema, Protobuf

AVRO - flow stawiania

- Projekt:
  - Konfigurujemy gradle avro plugin
  - Dodajemy schema restry w docker-compose
  - dev-manage ⇒ doszły delete, zobaczcie sobie
- Kodowanie, dodajemy nowe flagi!
  - Producer:
    - properties.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG, StringSerializer.class);
    - properties.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG, KafkaAvroSerializer.class);
    - properties.put(KafkaAvroSerializerConfig.SCHEMA_REGISTRY_URL_CONFIG, "http://localhost:8081");
  - Consumer:
    - properties.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG, StringDeserializer.class);
    - properties.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG, KafkaAvroDeserializer.class);
    - properties.put(KafkaAvroDeserializerConfig.SCHEMA_REGISTRY_URL_CONFIG, "http://localhost:8081");
    - properties.put(KafkaAvroDeserializerConfig.SPECIFIC_AVRO_READER_CONFIG, true); //ensures records are properly converted
- Dodać serilizery:
  - // https://mvnrepository.com/artifact/io.confluent/kafka-avro-serializer
  - implementation group: 'io.confluent', name: 'kafka-avro-serializer', version: '5.3.0'

Schema Registry - trzeba pamiętać że...

- I tak musimy uważać jako deweloperzy na kompatybilność wsteczną. Casus: usunięte pole wymagane przez Consumera.
  - Tutaj oficjalny przewodnik po tych kompatybilności
- trochę zabawy na starcie

inne pomocne linki

- Confluent Developer: o tym jak działa Schema Registry.
- Spring Cloud + Schema Registry + AVRO. Tutorial od podstaw
- Maven + Schema Registry. Tutorial
- Confluent Developer: Przewodnik po typach kompatybilności

Transakcje

Transakcje

- Transakcje w distributed systems NIE są typowe, po prostu my do nich przywykliśmy
- Kafka oferuje feature transakcyjności do zadań specjalnych
- możemy dzięki niemu osiągnąć Efficiently Once Delivery
- dość wolne rozwiązanie - nie przyjmie się raczej w przypadku fast streamingu
- ale do pieniędzy już tak :)
- Tutaj kompletny przewodnik od Confluent

Spring Framework - szybka wizyta

Spring Framework - szybka wizyta

- Spring Kafka. Tutorial od zera
  - Mamy “ProducerFactory<String, String>” oraz “KafkaTemplate<String, String>” i to w nim osadzamy FDD
  - Mamy też ConsumerFactory<String, String>
  - szansę na assignowanie się do partycji
- da się ręcznie commitować offset
- pollowanie o własnym czasie też możliwe, ale bardziej zaszyte
- Dodatkowo do przeczytania
  - Spring Docs: Container Props

Metryki

Metryki

Metryki zapinamy na wielu komponentach:

- ZooKeeper
- Kafka Brokery (JMX exporter)
- Producery
- Consumery

Metryki

Metryki zapinamy na wielu komponentach (przegląd dostępnych metryk):

- ZooKeeper
- Kafka Brokery (JMX exporter) ⇒ HERE
- Producery
- Consumery

Metryki - co nas może intertesować?

- “To zależy” od use case’a, ALE:
- Consumers:
  - Jak bardzo dana Consumer Group jest do tyłu względem topicu (Broker i JMX exporter)
  - Jak bardzo każdy Consumer z danej grupy jest do tyłu względem przypisanej do siebie partycji (Consumer Metrics)
  - Sprawdzać obciążenie pojedynczego Consumera (standardowe metryki JVM). Czasem może dostać za dużo partycji i nie wyrabiać. Czasem może być idle.
- Producers:
  - Jeśli włączyliśmy batching - rozmiar paczek wysyłanych brokera (Producer Metrics). Jeśli jest b. mały to Producer wyrabia i batching może być mało potrzebny. Jeśli za duży to może warto myśleć o skalowaniu Producerów (?)

Metryki - linki

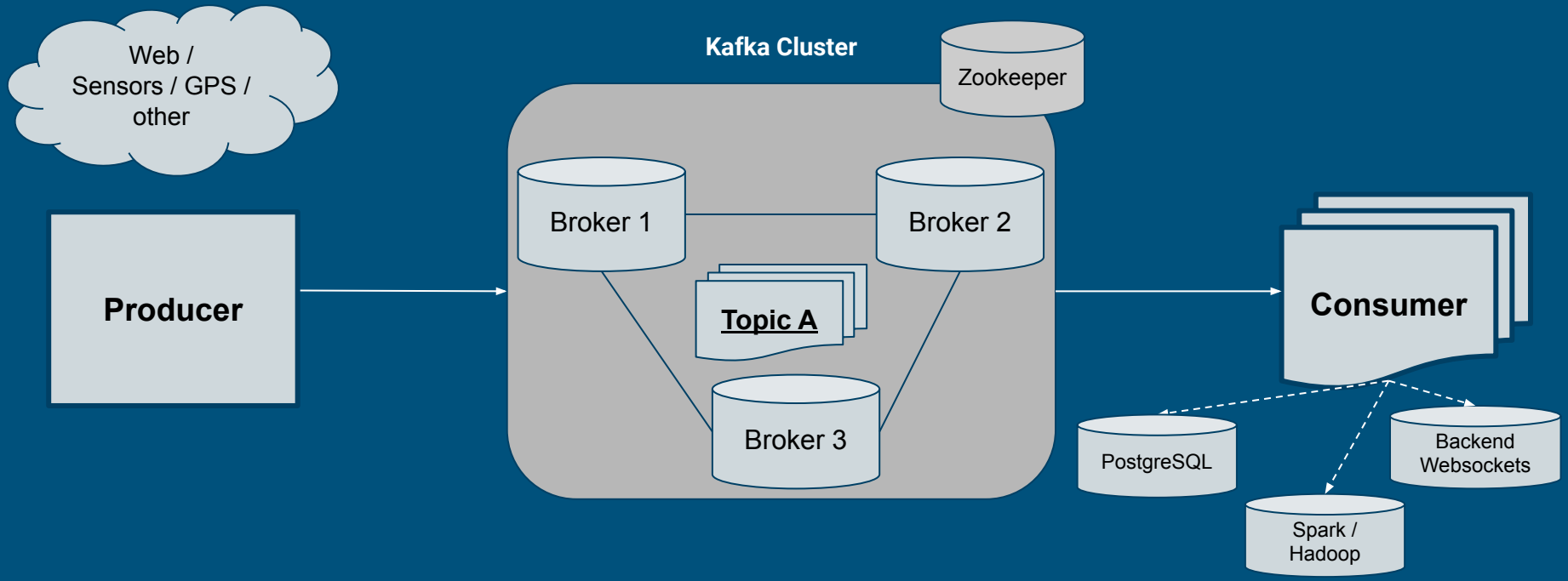
- Dobry Guide o metrykach od Confulenta

Kafka Connect

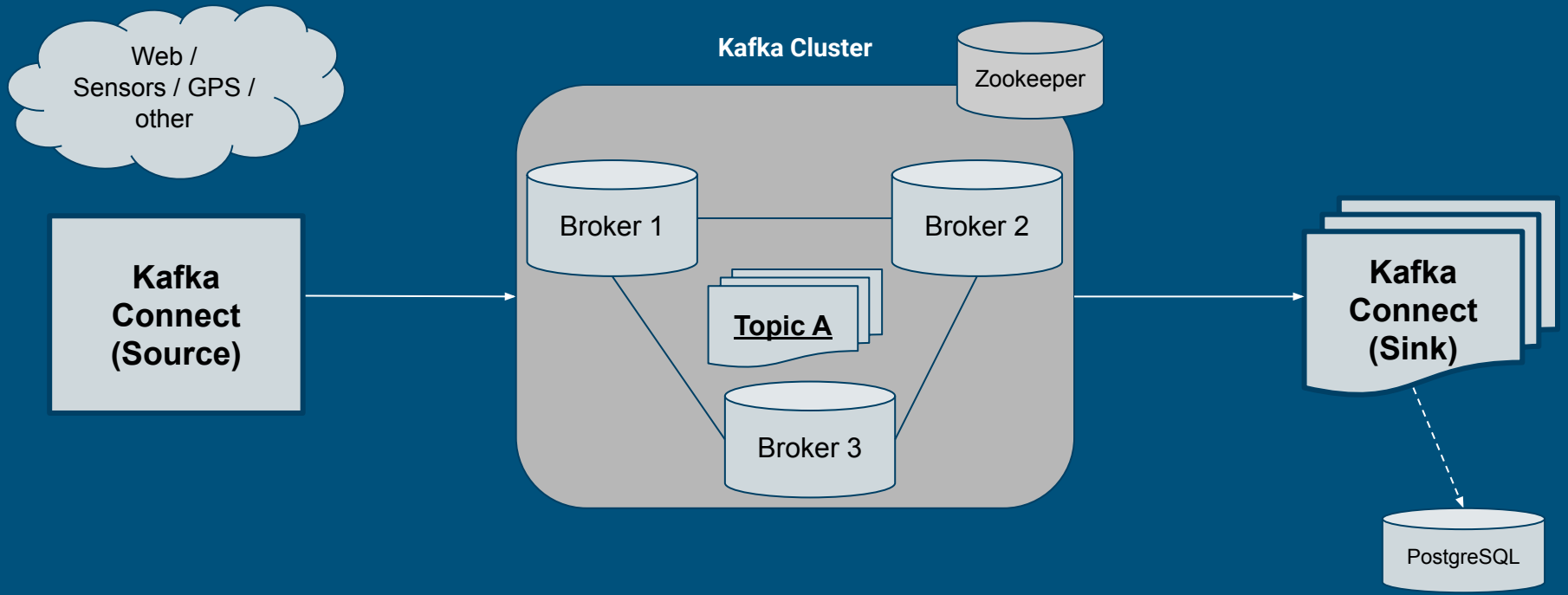
Kafka Connect

- Celem Kafka Connect jest zaoszczędzenie Nam pracy i nie pisanie po raz n-ty tych samych integracji z bazami
- Source Connector - wrzuca z jakiegoś źródła na Kafkę
- Sink Connector - nasłuchuje na topic i publikuje do jakiegoś źródła
- Aby instalować Connectory musimy dołożyć Kafka Connect Cluster.

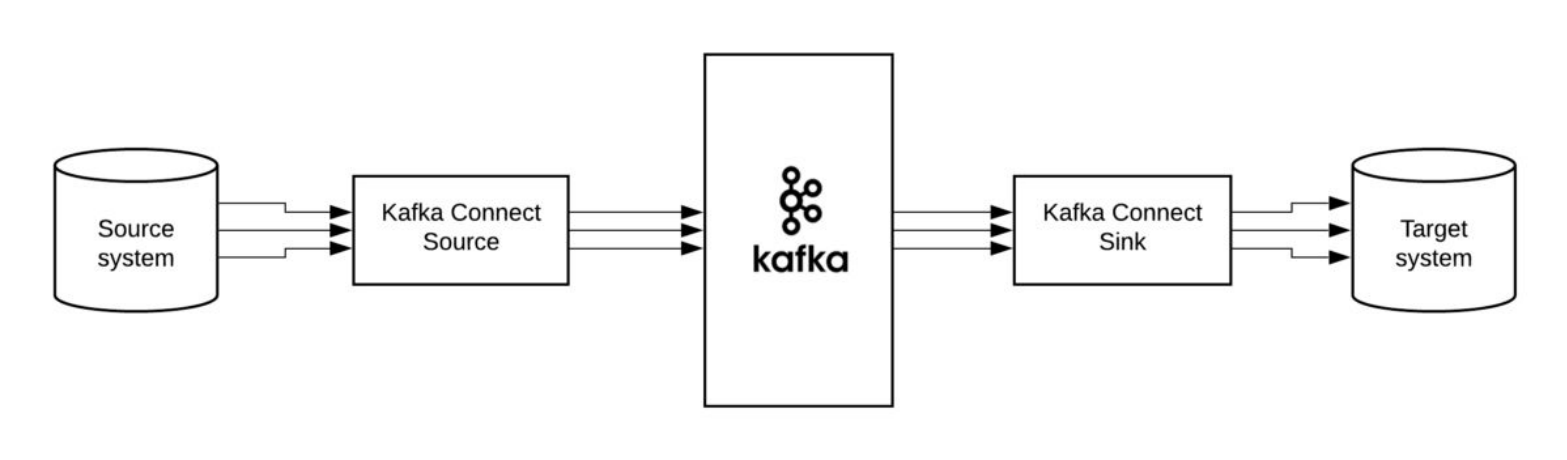
Kafka Connect



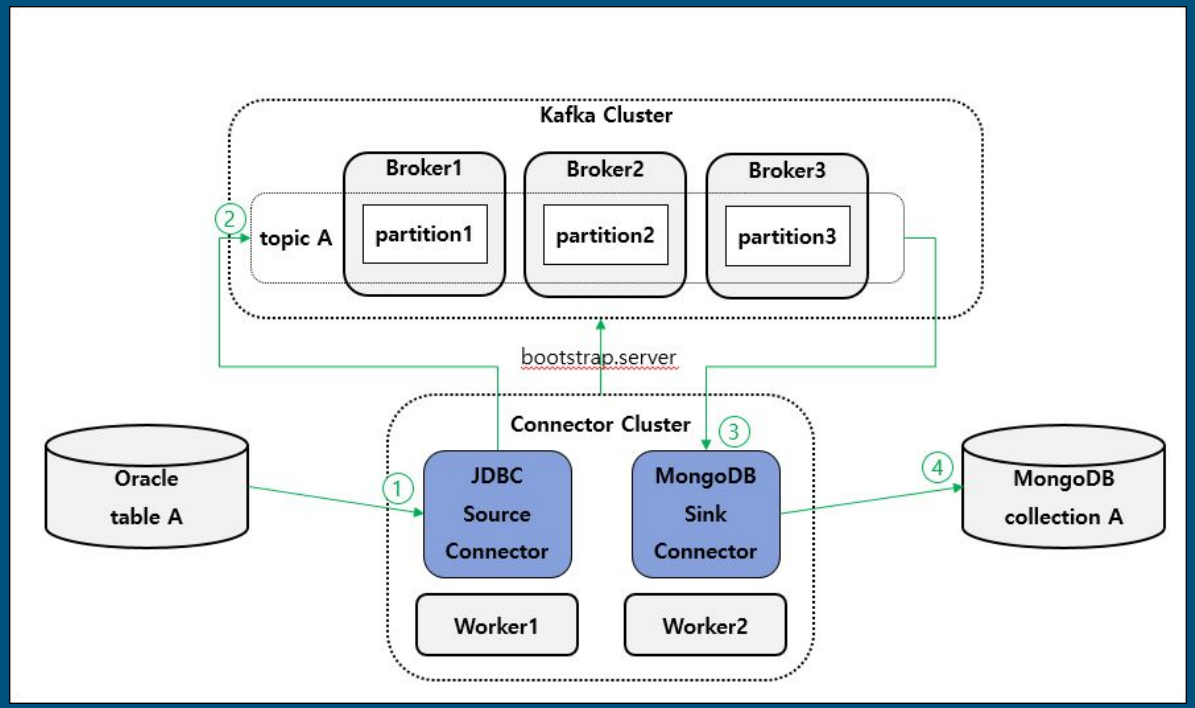
Kafka Connect



Kafka Connect



Kafka Connect



Kafka Connect

- Debezium (PostgreSQL, MySQL (nowsze wersje), Oracle, MsSQL Server, MongoDB)

Kafka Streams

Kafka Streams

- Mniej złożone niż Kafka Connect
- “dostawka do naszej aplikacji”, czyli thick client
- Co Oferuje?
  - ułatwia pracę na pojedynczym Topicu
    - map, filter
  - daje operowanie między wieloma topicami i analitykę
    - join, aggregate, count
  - time windows - analizowanie eventów z pewnego przedziału czasowego
  - Wprowadza dodatkowe opcje takie jak:
    - KStream ⇒ (tak naprawdę topic)
    - KTable
    - GlobalKTable

Kafka Streams

- przykładowy kod

Confluent - co to i co oferuje?

Confluent - co to?

- Kafka powstała w LinkedIn ale po pewnym czasie zaczęła żyć swoim własnym życiem
- Confluent opiekuje się teraz Kafką, rozwija ją i daje różne opcje

Confluent - co oferuje?

- Wiedza
  - Certyfikację z Kafki
    - Confluent Certified Developer for Apache Kafka®
    - Confluent Certified Administrator for Apache Kafka®
  - Materiały szkoleniowe szykujące do egzaminów
- Confluent Platform - komercyjny dodatek
  - Dobry UI do zarządzania, wizualizacji
  - lepsze zarządzanie topicami (zmiana liczby partycji)
  - łatwiejsza praca z Kafka Connect
  - Confluent Replicator - możliwość replikacji Kafki do innego DC

Add-on: Producer Batching

- Parę linków a propo wydajności Kafki i batchingu
- https://medium.com/lydtech-consulting/kafka-producer-message-batching-2f6f0b75a19c
- https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines

Koniec!
Dzięki wszystkim!

Więcej wiedzy!

- SML:
  - Live Wizualizacja tego jak działa Kafka!
  - 7 mistakes when using Apache Kafka
  - Help, Kafka ate my data! - jak zapobiec data loss
  - Does Kafka really guarantee the order of messages?
- Confluent Developer: dużo dobrych tutoriali