OPITZ CONSULTING

Überraschend mehr Möglichkeiten

Die Verwandlung mit Kafka

Coding Kata

Manuel Styrsky, Sebastian Liedl, Eric Nguyen

Agenda

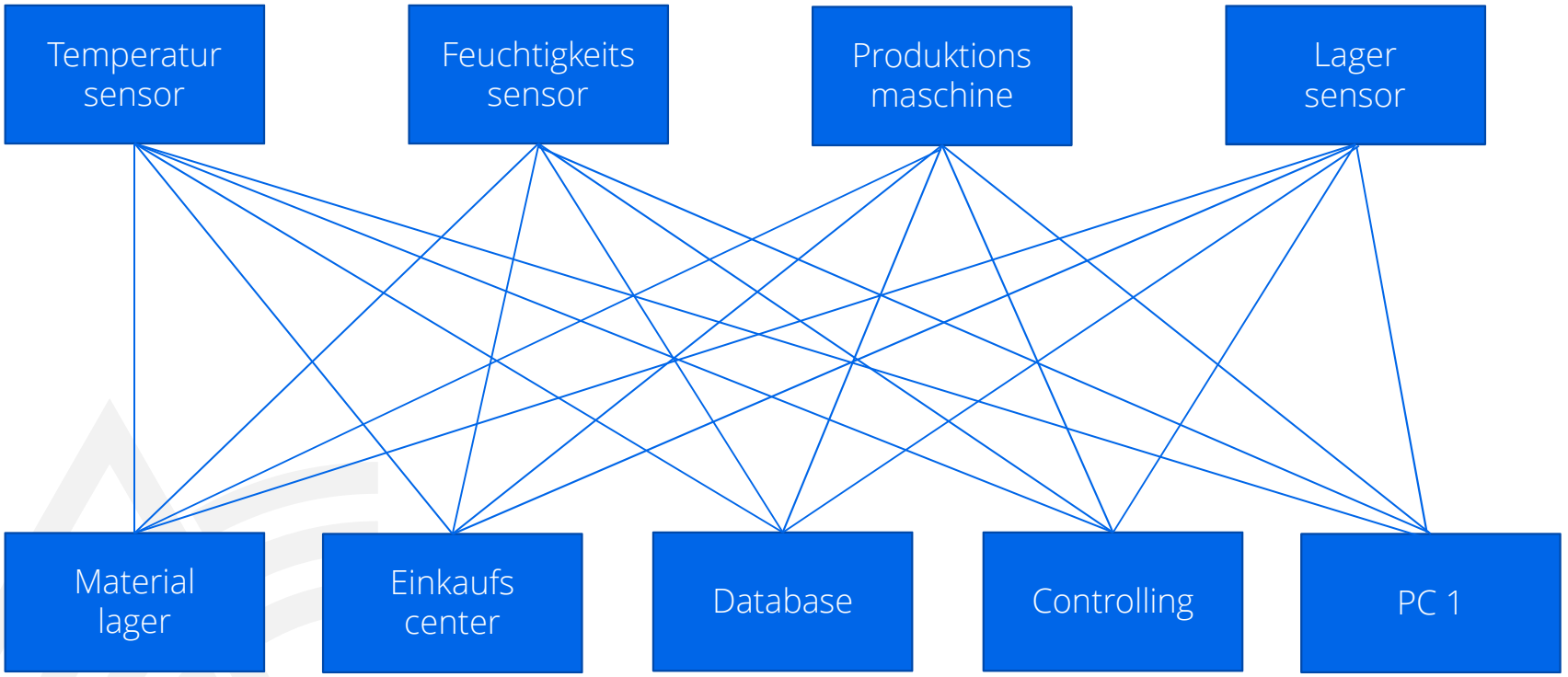
1. Einführung in Kafka
2. Kafka Streaming API
3. Kafka Connect
4. Hands On Hacking

Einführung in Kafka

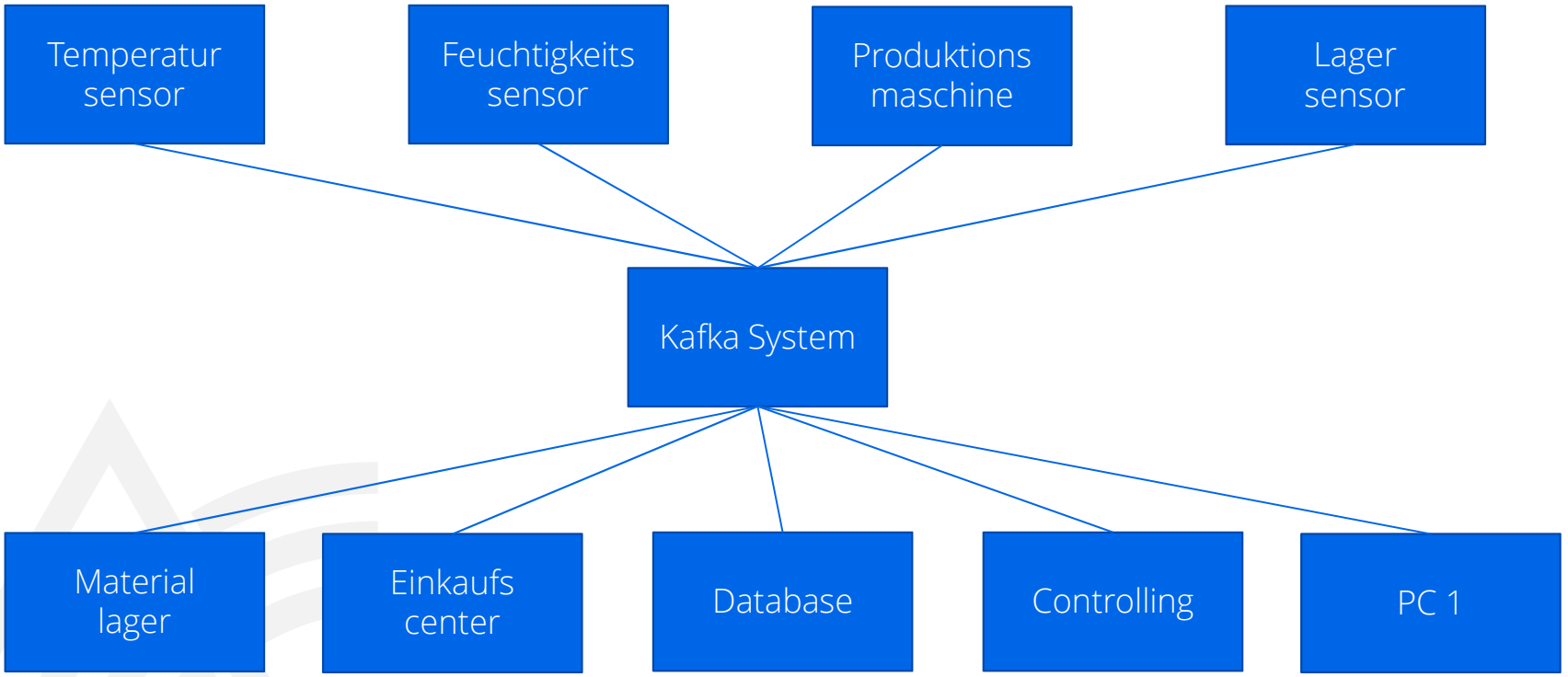
Einführung in Kafka

- Real Time Transport von Daten
- Es wird das sog. Publish/Subscribe Pattern benutzt
- Schnittstellen zum Importieren und Exportieren von Daten
  - Kafka Producer zum importieren
  - Kafka Consumer zum exportieren
  - Kafka Connect (Schon existierende Producer/Consumer)
- Schnittstellen zur Echtzeitverarbeitung von Datenströmen (Kafka Streams)

Einführung in Kafka



Einführung in Kafka

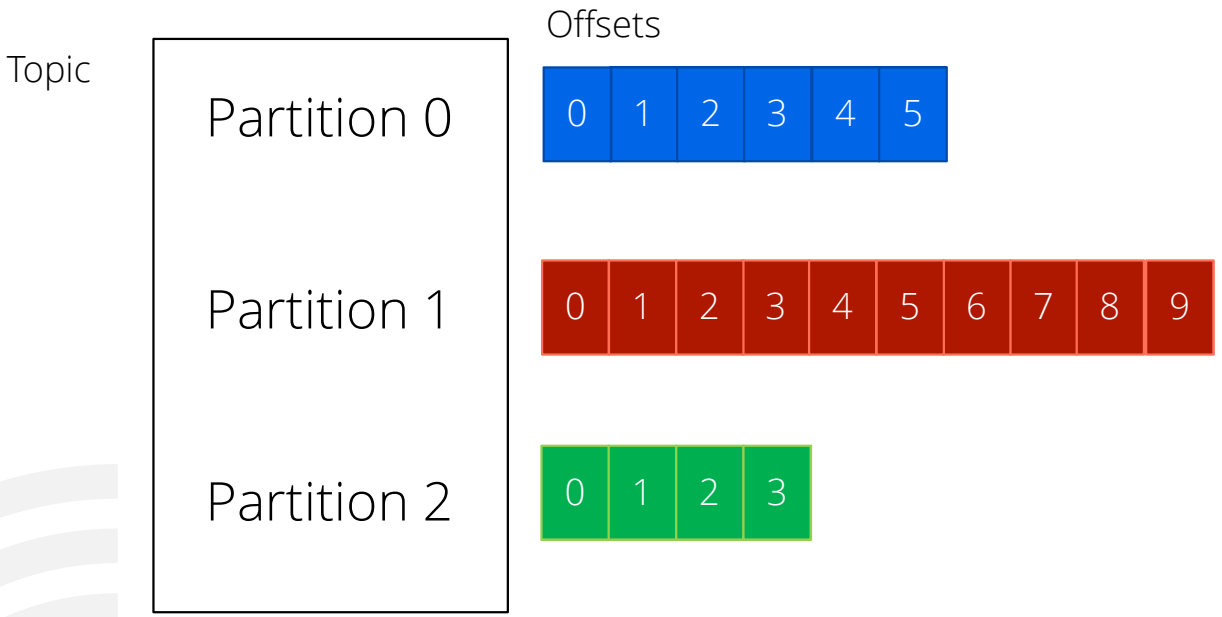


Topics

Topics

- Es können beliebig viele Topics erstellt werden
- Topics werden benötigt, um geordnet Daten zu produzieren und zu konsumieren

Topics


Topic
Partition 0
Partition 1
Partition 2
Offsets
0 1 2 3 4 5
0 1 2 3 4 5 6 7 8 9
0 1 2 3

Partitions

Partitions

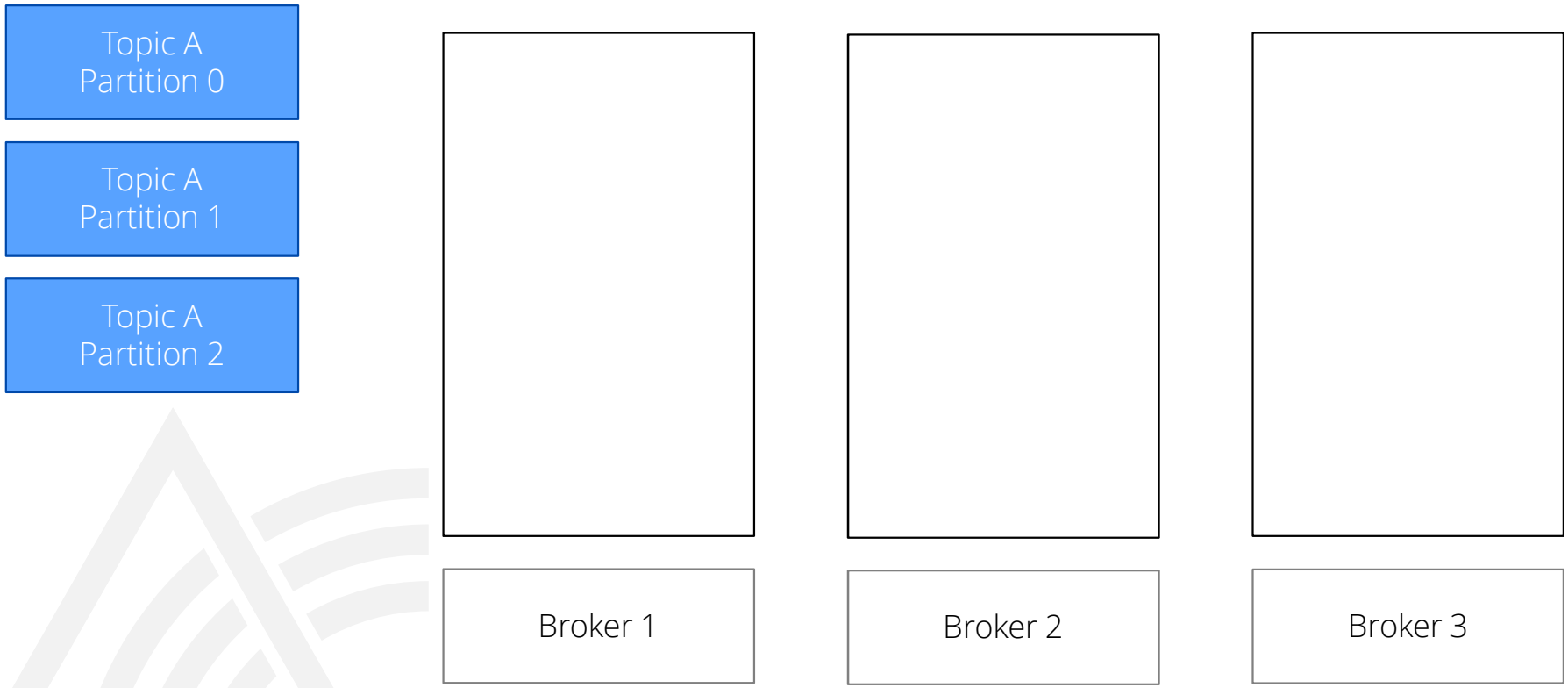
- Innerhalb einer Partition sind Daten geordnet
- Sobald Daten in eine Partition geschrieben sind, können sie nicht mehr verändert werden
- Daten werden nach einer bestimmten Zeit wieder gelöscht
  - Default: Eine Woche

Broker

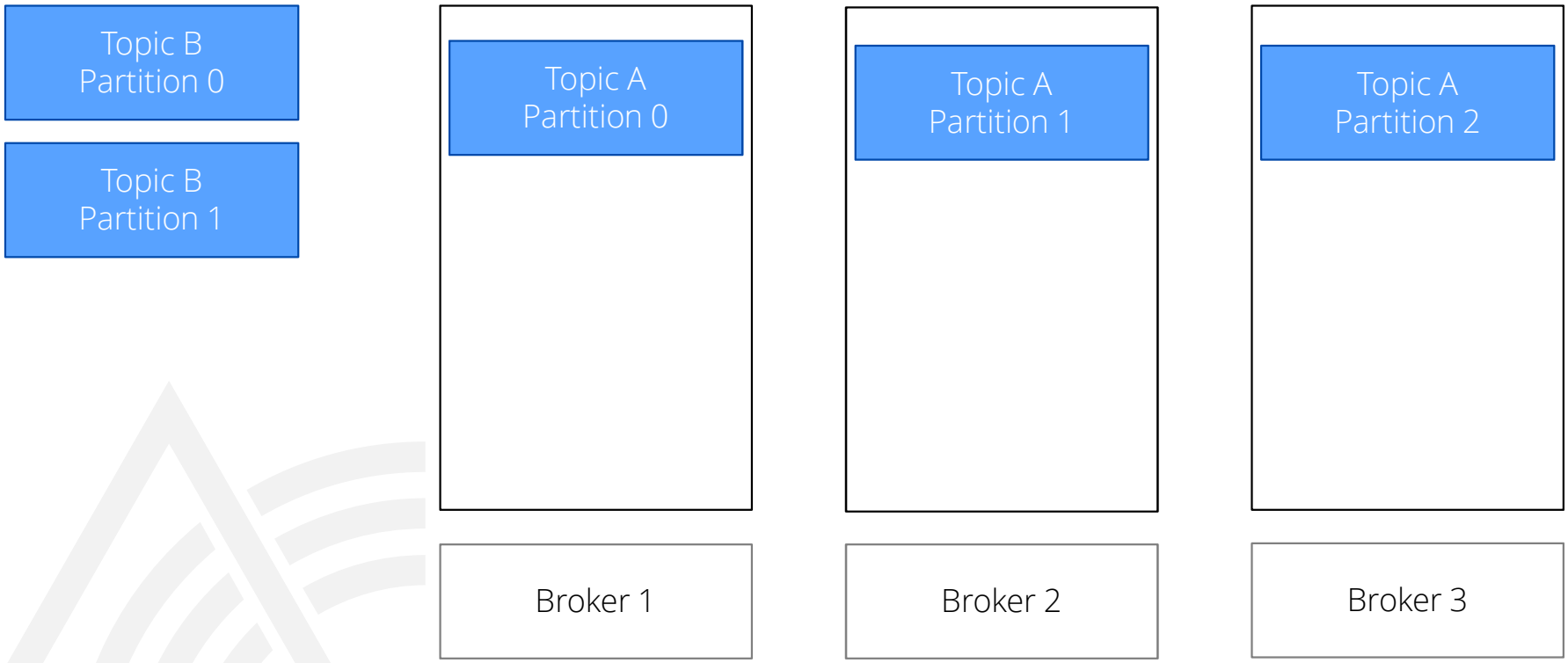
Broker

- Kafka System kann aus mehreren Broker bestehen -> Kafka Cluster
- Jeder Broker ist mit einer eindeutigen ID (Integer) gezeichnet
- Sobald sich ein Client zu einem Broker verbindet, ist er mit dem kompletten Cluster verbunden
- Broker beinhalten Partitions von Topics

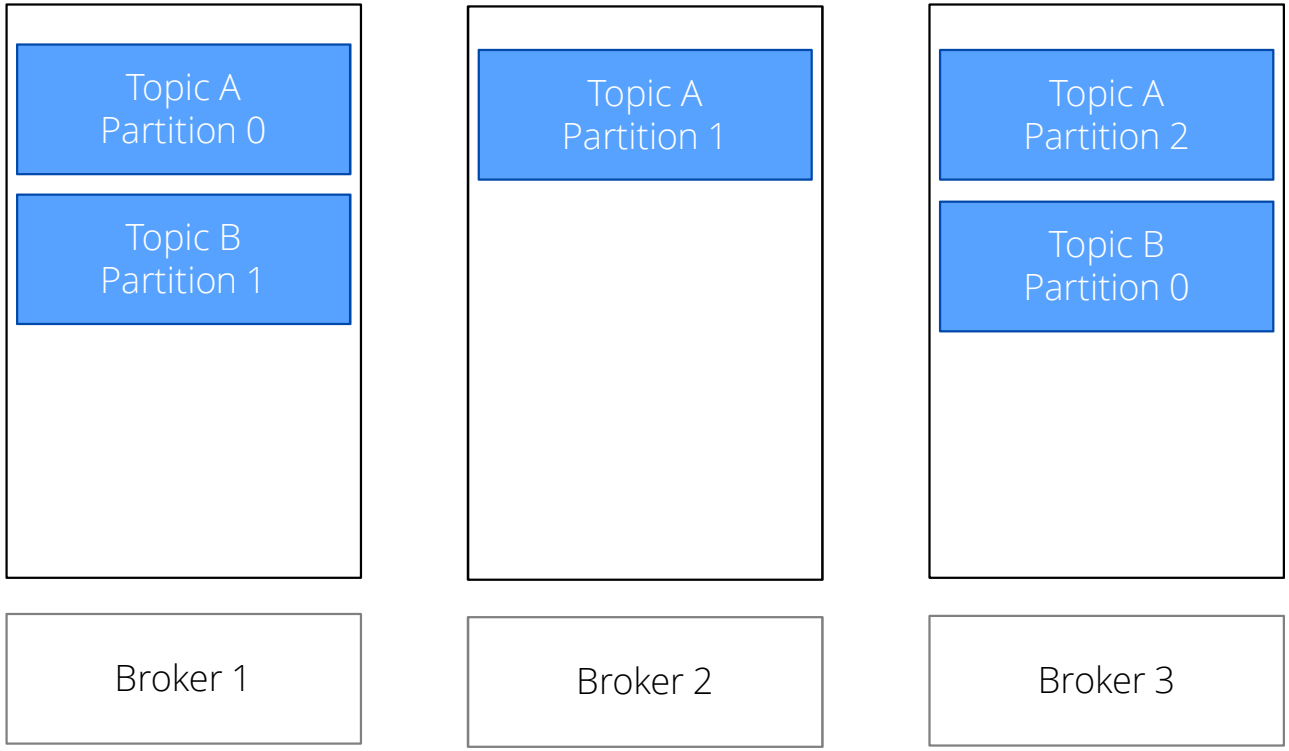
Broker – Verteilung der Partitions



Broker – Verteilung der Partitions



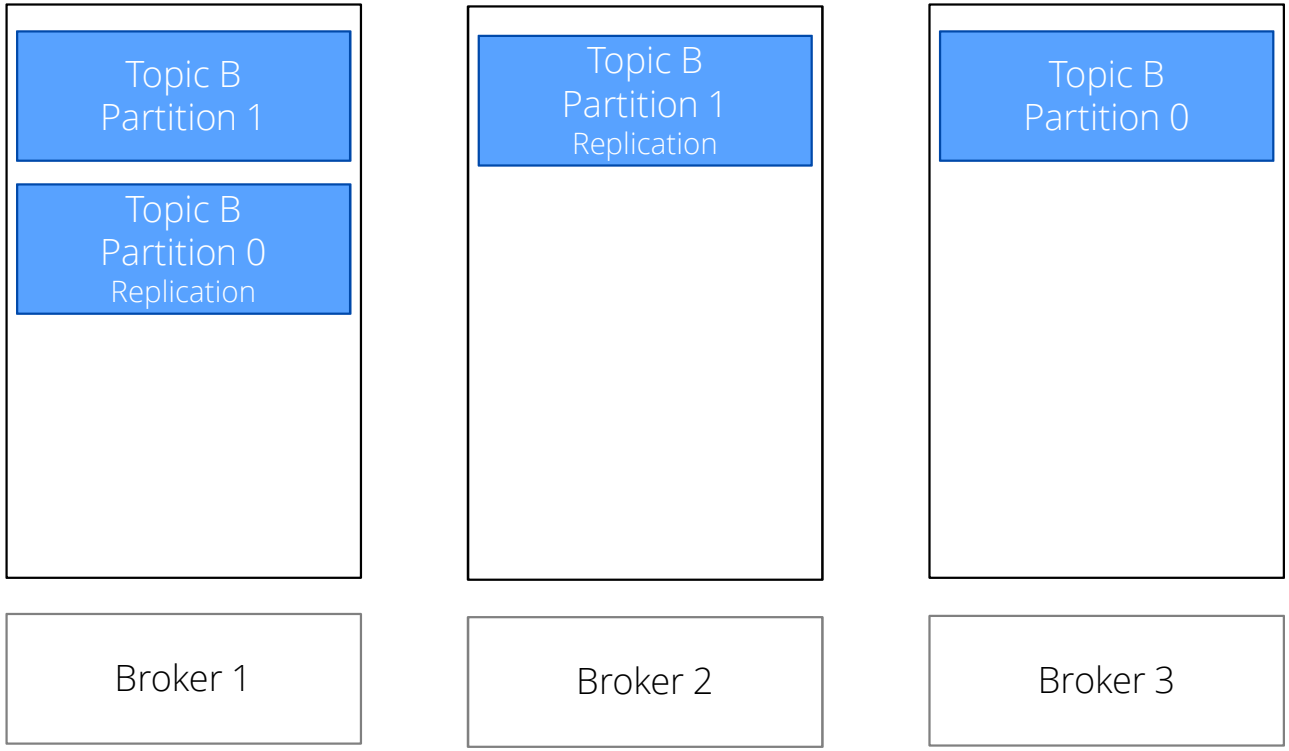
Broker – Verteilung der Partitions



Broker - Ausfallsicherheit

- Pro Topic können sog. Replications konfiguriert werden
- Eine gute Anzahl an Replications sind 3
- Zu beachten ist, dass die Anzahl der Broker größer gleich der Replication-Zahl sein muss

Broker – Ausfallsicherheit



Kafka Streams API

Kafka Streams API

- Ermöglicht die real time Verarbeitung von Daten in einem Topic
- Konsumiert Daten von einem Topic
- Ermöglicht die Bearbeitung der konsumierten Daten
- Schreibt sie in ein anderes Topic

Kafka Connect API

Kafka Connect API

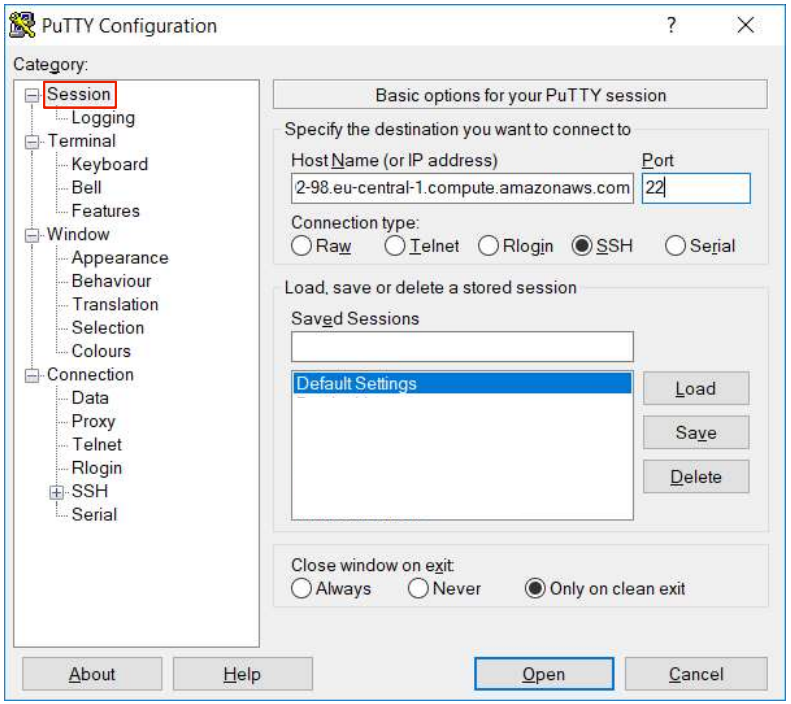
- Bietet eine Schnittstelle, um Daten zu Kafka zu importieren und zu exportieren
- Es gibt bereits fertige Connectoren, welche nur noch an das eigene System angepasst werden müssen
  - Confluent.io/hub (opensource)

Hands-on hacking

- Bitte wartet bis zum Ende der Konfiguration, um das PuTTY Terminal zu öffnen.

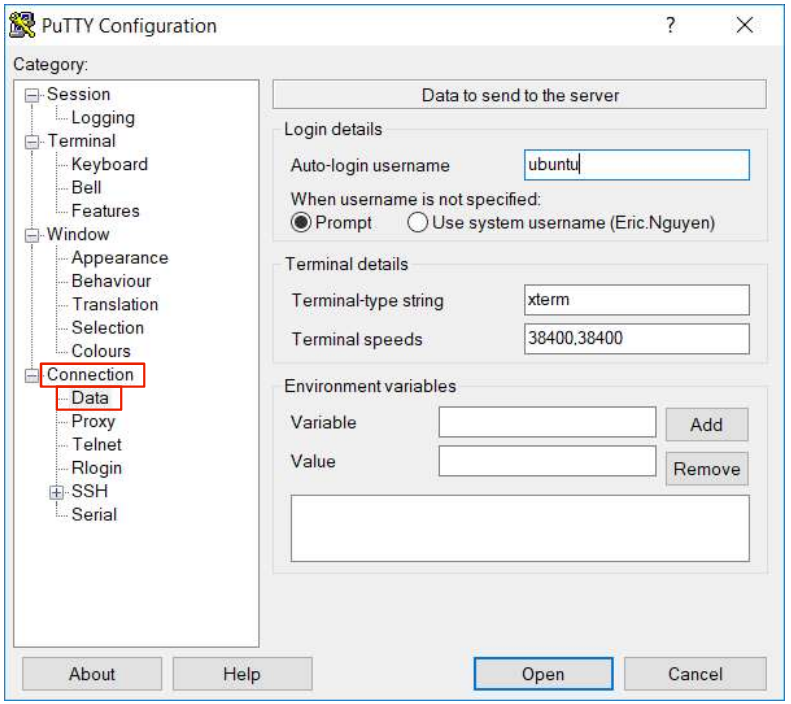
Reiter: Session

- Hostname: IPv4 Adresse auf AWS
- Port: „22“



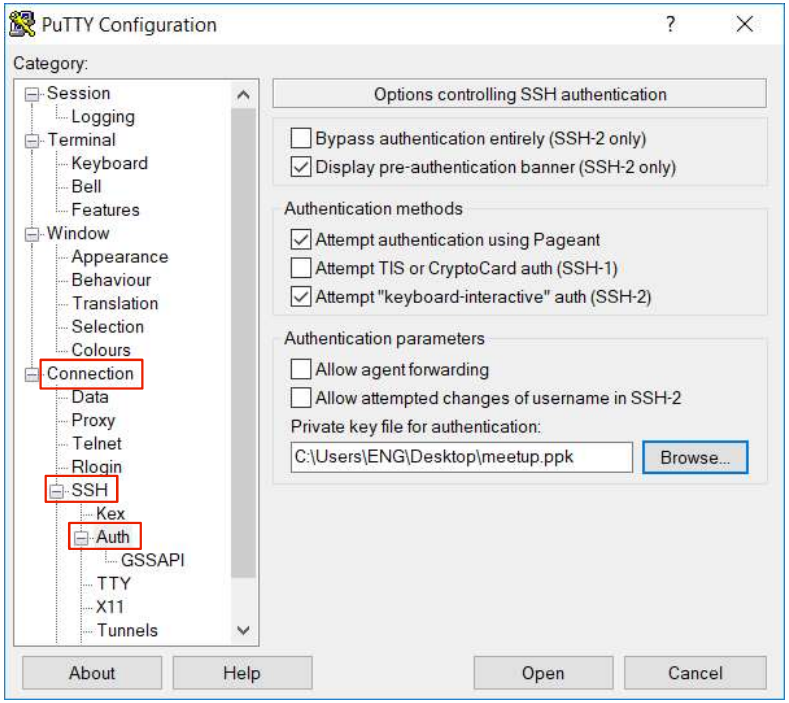
Reiter: Connection → Data

- Auto-login username: „ubuntu“



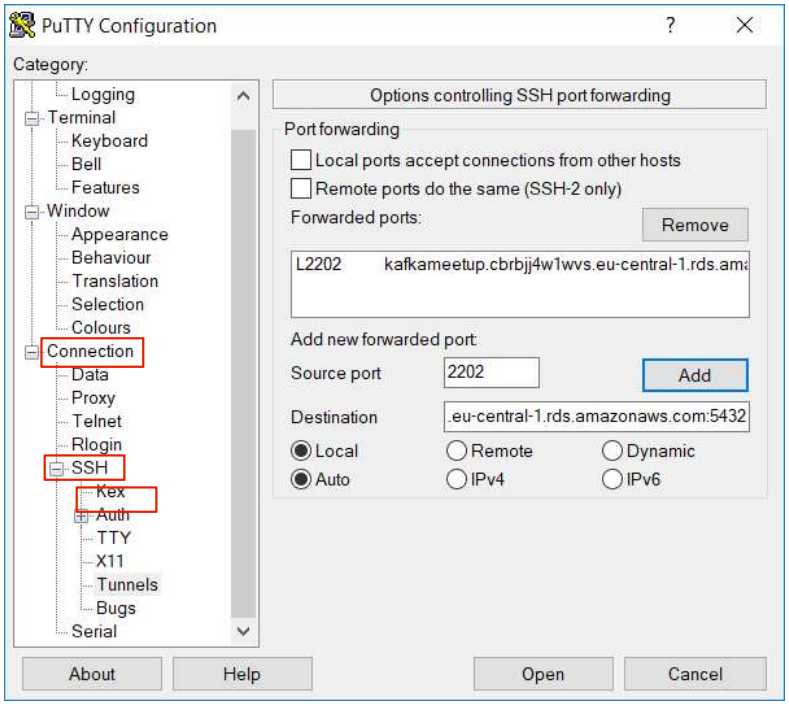
Reiter: Connection → SSH → Auth

- Klickt bitte auf „Browse…“ und wählt „meetup.ppk“ aus



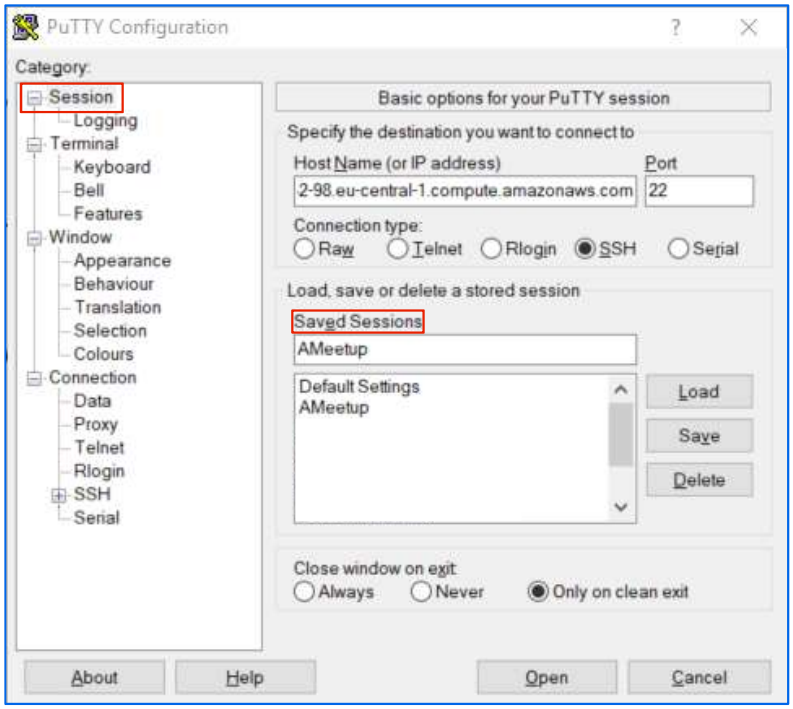
Reiter: Connection → SSH → Tunnels

- Source port: „2202“
- Destination: DB endpoint (cheatsheet)
- Destination port: „5432“



Reiter: Session

- Eintippen eines beliebigen Namens
- Speichert die PuTTY Konfigurationen



Starten des Zookeepers über die Command Line

kafka/bin/zookeeper-server-start.sh kafka/config/zookeeper.properties

Starten des Kafka Servers über die Command Line

kafka/bin/kafka-server-start.sh kafka/config/server.properties

Erstellen eines Topics durch die Command Line

```
kafka/bin/kafka-topics.sh --create --zookeeper localhost:2181
--replication-factor 1 --partitions 1 --topic test
```

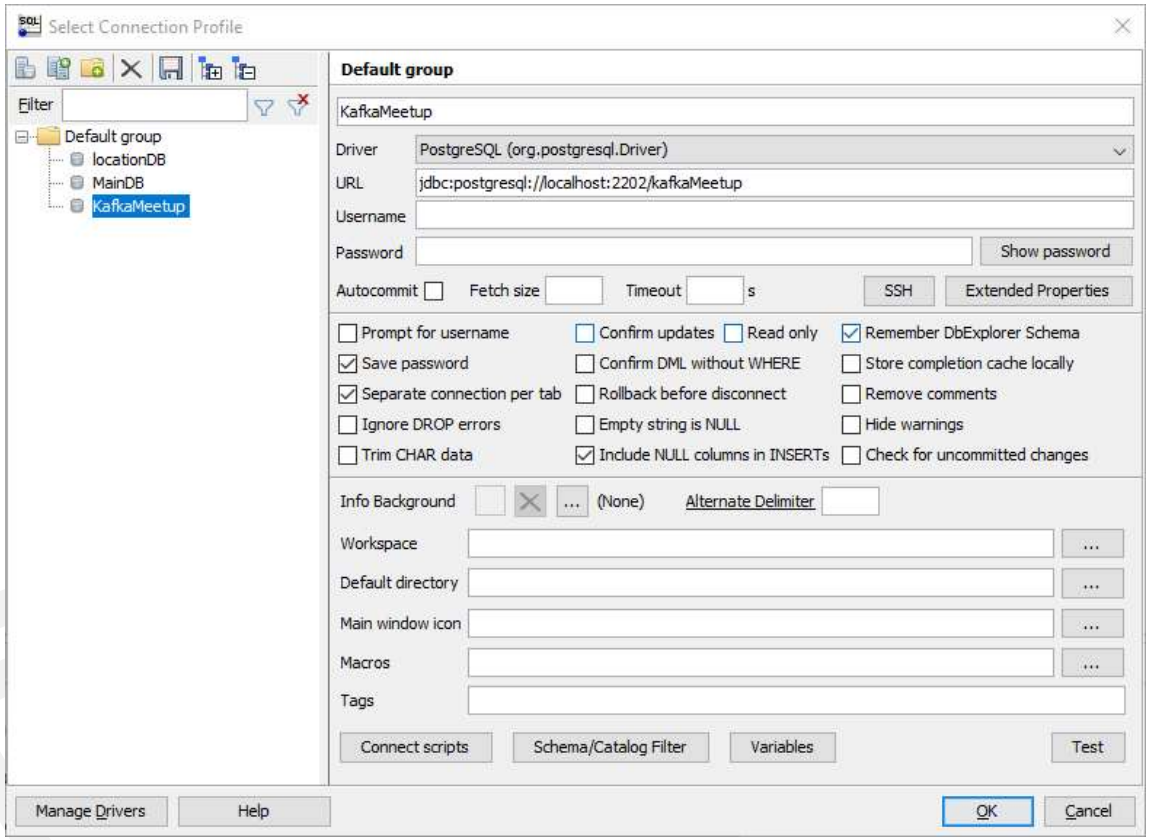
Schicken einer Nachricht über die Command Line

```
kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test
```

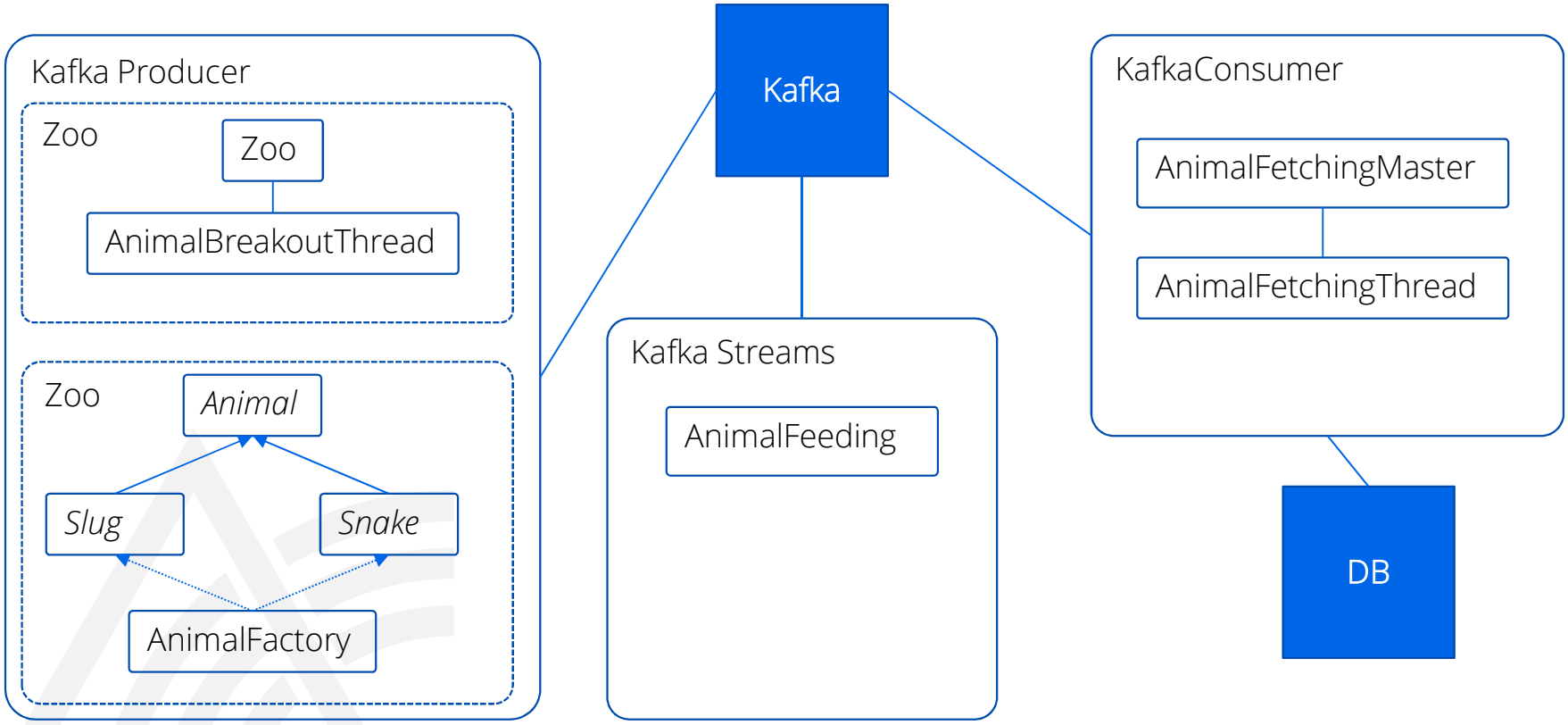
Lesen einer Nachricht über die Command Line

kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092
--topic test --from-beginning

SQL Workbench



Code Gerüst


Kafka Producer
Zoo
Zoo
AnimalBreakoutThread
Zoo
Animal
Slug
Snake
AnimalFactory
Kafka
Kafka Streams
AnimalFeeding
KafkaConsumer
AnimalFetchingMaster
AnimalFetchingThread
DB

Kafka Producer

```
Properties properties = new Properties();
//… set properties
```

```
KafkaProducer<String, String> producer = new KafkaProducer<~,~>(properties);
```

```
ProducerRecord<String, String> record = new ProducerRecord<~,~>(topic, key, value);
```

```
Callback callback = new Callback() {
    public void onCompletion(RecordMetadata recordMetadata, Exception e) {
        //Specify behavior}};
```

```
producer.send(record, callback);
```

```
producer.flush();
producer.close();
```

Kafka Streams

```
Properties properties = new Properties();
//… set properties

//create a topology
StreamsBuilder streamsBuilder = new StreamsBuilder();
KStream<String, String> inputTopic = streamsBuilder.stream(„your topic");
KStream<String, String> filteredStreams = inputTopic.filter(
        (k, text) -> lambdaFunction
         //your lambda function
);
filteredStreams.to(„output topic");

//build a topology
KafkaStreams kafkaStreams = new KafkaStreams(streamsBuilder.build(), properties);

kafkaStreams.start();
```

Kafka Consumer

```
Properties properties = new Properties();
//… set properties
```

```
//create consumer
KafkaConsumer<String, String> consumer = new KafkaConsumer<~,~>(properties);
```

```
//subscribe consumer to our topic(s)
consumer.subscribe(Arrays.asList(topic));
```

```
//poll for new data
while (true){
    ConsumerRecords<String, String> records =consumer.poll(Duration.ofMillis(100));

    for (ConsumerRecord<String, String> record : records){
        //handle your record
    }
}
```

Viel Spaß beim Hacking!

Copy your `id_rsa` file into your ".ssh" folder which is a child of your home folder. This folder is hidden but can be easily revealed via this terminal command:

```
$ open ~/.ssh
```

Now that your `.ssh` folder is open in the finder you can easily drag/drop/copy your pre-existing `id_rsa` file into it.

Next, check to see if the SSH agent is running:

```
$ ps -e | grep ssh-agent
  20207 ?? 0:00.53 /usr/bin/ssh-agent -l // its running
```

If it is not running:

```
$ ssh-agent /bin/bash
```

Check to see what identities have been loaded:

```
$ ssh-add -l
```

Now load your identity with this command:

```
$ ssh-add ~/.ssh/id_rsa
```

Finally, check again for the list of identities – there should be one more than before (or just one if there weren't any previously).

```
$ ssh-add -l
  2048 SHA256:41ZyLeEcsdfwefsdfLegsdftQdm0Ew /Users/my_mac/.ssh/id_rsa (RSA)
```