Над текстом нажмите кнопку "**Создать стенд**". До 10 минут создается стенд. Каждый стенд создан под определенный модуль курса, поэтому при выполнении нижеследующих инструкций, нужно запустить отсюда стенд.

Где получить логин/пароль? В настройках профиля или над текстом по кнопке "**Доступы**".

Практический стенд этого урока состоит из одной ноды. Как зайти? Тут.

В данном практическом занятии мы будем реплицировать данные из **source** кластера Kafka, слушающего на порту 9092 в **destination** кластер Kafka, который работает на порту 9093.

Для этого сперва выполним команды ниже и поднимем два кластера.

1. Копируем архив с кафкой в домашний каталог:

```
cp /opt/kafka_2.13-2.7.0.tgz ~/
```

В случае, если что-то пойдет не так, вы сможете удалить каталог полностью и взять чистый архив для того, чтобы начать практику с начала.

Если вы его случайно удалили, можно скачать архив по ссылке:

```
wget https://archive.apache.org/dist/kafka/2.7.0/kafka_2.13-2.7.0.tgz
```

Распакуем архив и перейдем в каталог с приложением:

```
tar -xzf kafka_2.13-2.7.0.tgz
```

```
cd kafka_2.13-2.7.0
```

2. Первым делом мы запускаем Zookeeper.

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

3. Запускаем два кластера Apache Kafka. Обратите внимание как мы используем разный корень в ZooKeeper в конфигурационном файле брокера (`localhost:2181/9092` и `localhost:2181/9093`). Это позволяет нам использовать один кластер ZooKeeper сразу для двух кластеров Kafka. Но использовать такой подход в Production окружении не рекомендуется.

```
cat << 'EOF' > kafka-two-clusters-start.sh
#!/usr/bin/env bash
for i in {9092..9093}
do
  cp ./config/server.properties ./config/server$i.properties
  echo "listeners=PLAINTEXT://:$i" >> ./config/server$i.properties
  rm -rf /tmp/kafka-logs$i
  mkdir /tmp/kafka-logs$i
  sed -i "s/log.dirs=\/tmp\/kafka-logs/log.dirs=\/tmp\/kafka-logs$i/g"
./config/server$i.properties
  sed -i
"s/zookeeper.connect=localhost:2181/zookeeper.connect=localhost:2181\/$i/g"
./config/server$i.properties
  (./bin/kafka-server-start.sh ./config/server$i.properties &)
done
EOF
chmod +x kafka-two-clusters-start.sh
./kafka-two-clusters-start.sh
```

Далее создадим топик в source кластере на виртуальном сервере node-1 и начнем записывать в него данные.

4. Создаем топик в source кластере

```
./bin/kafka-topics.sh \
--create \
--topic registrations \
--partitions 1 \
--bootstrap-server localhost:9092 \
--replication-factor 1
```

5. Проверяем, что топик создан

```
./bin/kafka-topics.sh \
--describe \
--topic registrations \
```

```
--bootstrap-server localhost:9092 | head -n1
```

6. Запускаем продюсера в source кластере

```
./bin/kafka-run-class.sh org.apache.kafka.tools.ProducerPerformance  \
--topic=registrations  \
--num-records=100000000 \
--throughput=100 \
--record-size=100 \
--producer-props \
  bootstrap.servers=localhost:9092 \
  batch.size=64000 \
--print-metrics
```

7. Теперь запустим кластер Kafka Connect, использующий destination кластер.

```
cat << EOF > worker.properties
bootstrap.servers=localhost:9093

group.id=kafka-connect-group

producer.acks=all

offset.flush.interval.ms=30000
offset.flush.timeout.ms=2000

security.protocol=PLAINTEXT

plugin.path=/usr/share/java

offset.storage.topic=__connect-offset-storage
offset.storage.replication.factor=1
config.storage.topic=__connect-config-storage
config.storage.replication.factor=1
status.storage.topic=__connect-status-storage
status.storage.replication.factor=1

key.converter.schemas.enable=false
key.converter=org.apache.kafka.connect.converters.ByteArrayConverter
value.converter.schemas.enable=false
value.converter=org.apache.kafka.connect.converters.ByteArrayConverter
EOF
./bin/connect-distributed.sh worker.properties
```

8. И наконец запустим сам коннектор MM 2.0

```
cat << EOF > mm2.json
```

```
{
  "name": "mm2-connector",
  "config": {
    "connector.class": "org.apache.kafka.connect.mirror.MirrorSourceConnector",
    "source.cluster.alias": "source",
    "source.cluster.bootstrap.servers": "localhost:9092",
    "target.cluster.alias": "destination",
    "target.cluster.bootstrap.servers": "localhost:9093",
    "topics": "registrations",
    "consumer.group.id": "kafka-mm2-group",
    "producer.enable.idempotence":"true"
  }
}
EOF
curl -X POST -H "Content-Type: application/json" \
  localhost:8083/connectors \
  --data @mm2.json | jq
```

9. Убедимся, что коннектор успешно запустился

```
curl localhost:8083/connectors/mm2-connector/status | jq
```

10. Проверим, что в destination кластере появились новые топики (включая топик `registrations` из source кластера).

```
./bin/kafka-topics.sh \
--list \
--bootstrap-server localhost:9093
```

11. Попробуем прочитать данные из реплицируемого топика `source.registrations`

```
./bin/kafka-console-consumer.sh \
--topic source.registrations \
--bootstrap-server localhost:9093
```