public static void main(String[] args) {

Iterable<Record> records = new ArrayList<Record>();

Record r1 = new Record();

RecordId id1 = new RecordId();

id1.setKey("1");

RecordId id2 = new RecordId();

id2.setKey("2");

RecordId id3 = new RecordId();

id3.setKey("3");

RecordId id4 = new RecordId();

id4.setKey("4");

RecordId id5 = new RecordId();

id5.setKey("5");

RecordId id6 = new RecordId();

id6.setKey("6");

r1.setId(id1);

Record r2 = new Record(); r2.setId(id2);

Record r3 = new Record(); r3.setId(id3);

Record r4 = new Record(); r4.setId(id4);

Record r5 = new Record(); r5.setId(id5);

Record r6 = new Record(); r6.setId(id6);

((ArrayList)records).add(r1); ((ArrayList)records).add(r2); ((ArrayList)records).add(r3);

((ArrayList)records).add(r4);((ArrayList)records).add(r5);((ArrayList)records).add(r6);

}

/\* StreamSupport.stream(records.spliterator(),false)

.map(x -> x.getId().getKey())

.collect(Collectors.toList())

\*/

List<String> st = StreamSupport.stream(records.spliterator(),false)

.map( x -> x.getId().getKey()).collect(Collectors.toList());

Lists.partition(st,2).forEach(i -> System.out.println(i));

/\*Seq.of(1,2,3,4,5,6).partition(i -> i.c) .forEach( i -> {

System.out.println(i);

});\*/

/\*Seq.of(StreamSupport.stream(records.spliterator(),false).map( x -> x.getId().getKey())).partition(i -> i % 2 ==0)

.forEach( i -> {

System.out.println(i);

});

\*/

//System.out.println(st);

Map<Object,List<Object>> ls = Seq.seq(records).zipWithIndex().collect(Collectors.groupingBy( tuple-> tuple.v2 /3));

Seq.seq(lazyFileStream) // Seq<String>

.zipWithIndex() // Seq<Tuple2<String, Long>>

.groupBy(tuple -> tuple.v2 / 500) // Map<Long, List<String>>

.forEach((index, batch) -> {

process(batch);

});