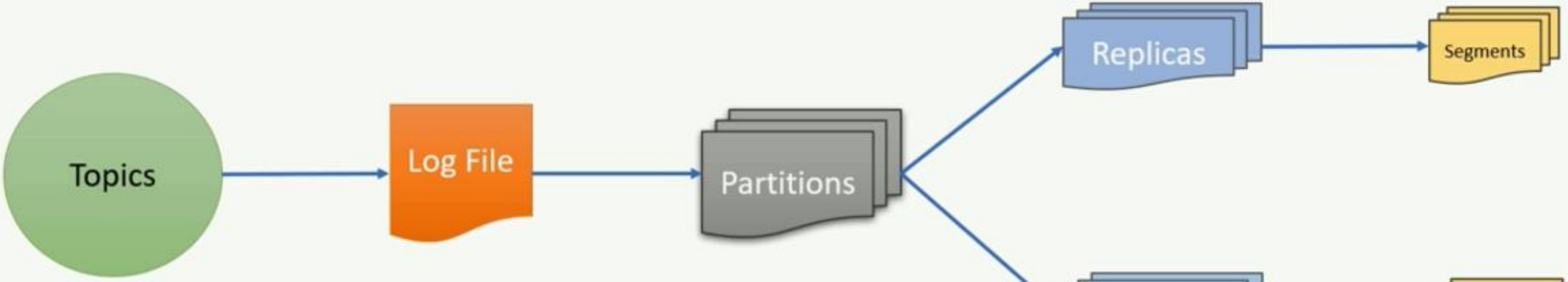
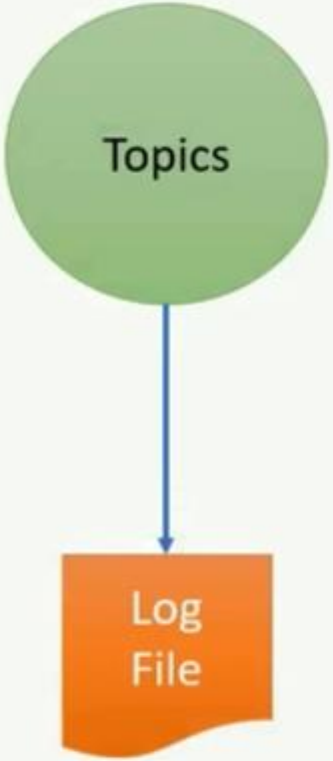
Kafka Storage System


Topics
Log File
Partitions
Replicas
Segments
Replicas
Segments


Topics
Log File

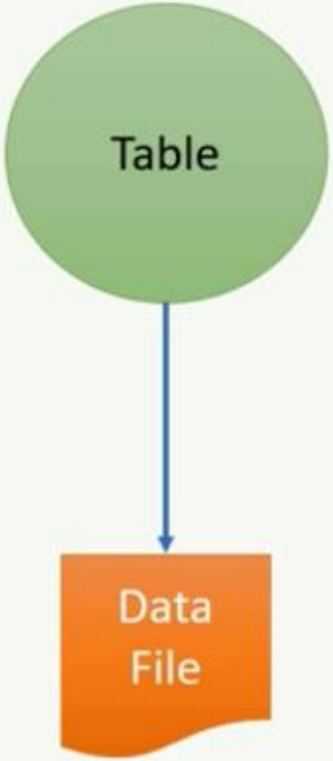
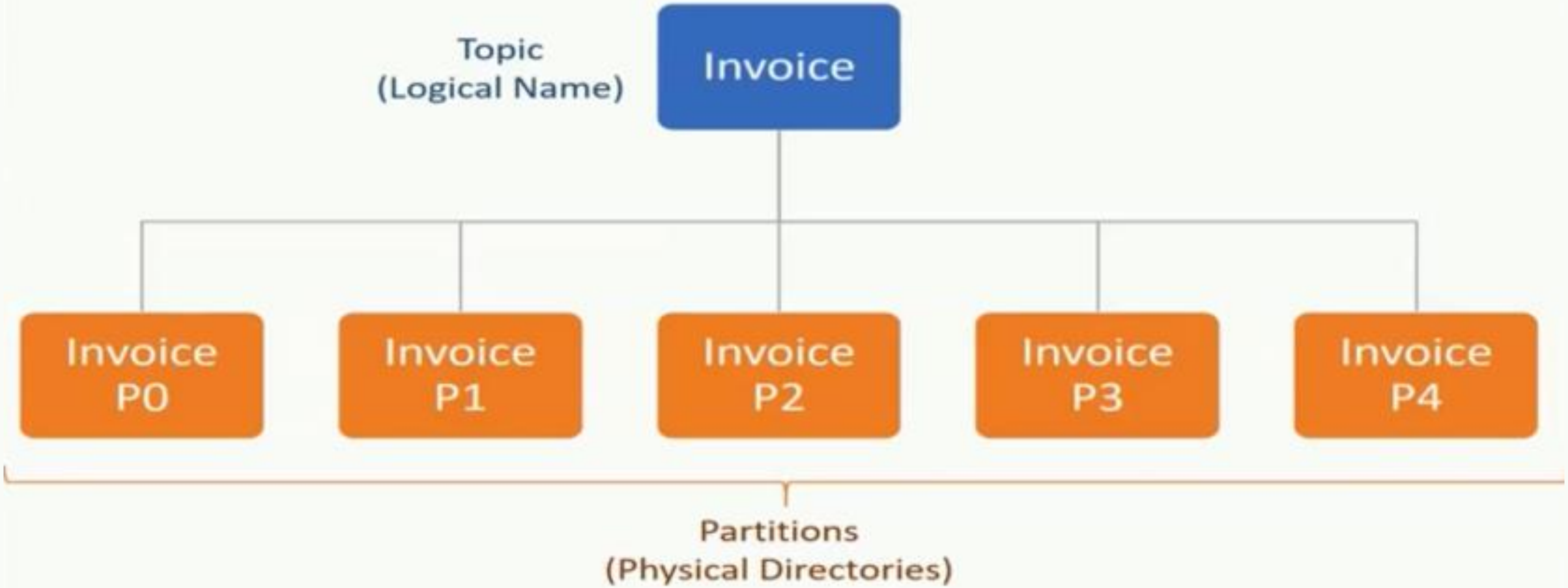


Table
Data File



Topic
(Logical Name)

Invoice

Invoice
P0

Invoice
P1

Invoice
P2

Invoice
P3

Invoice
P4

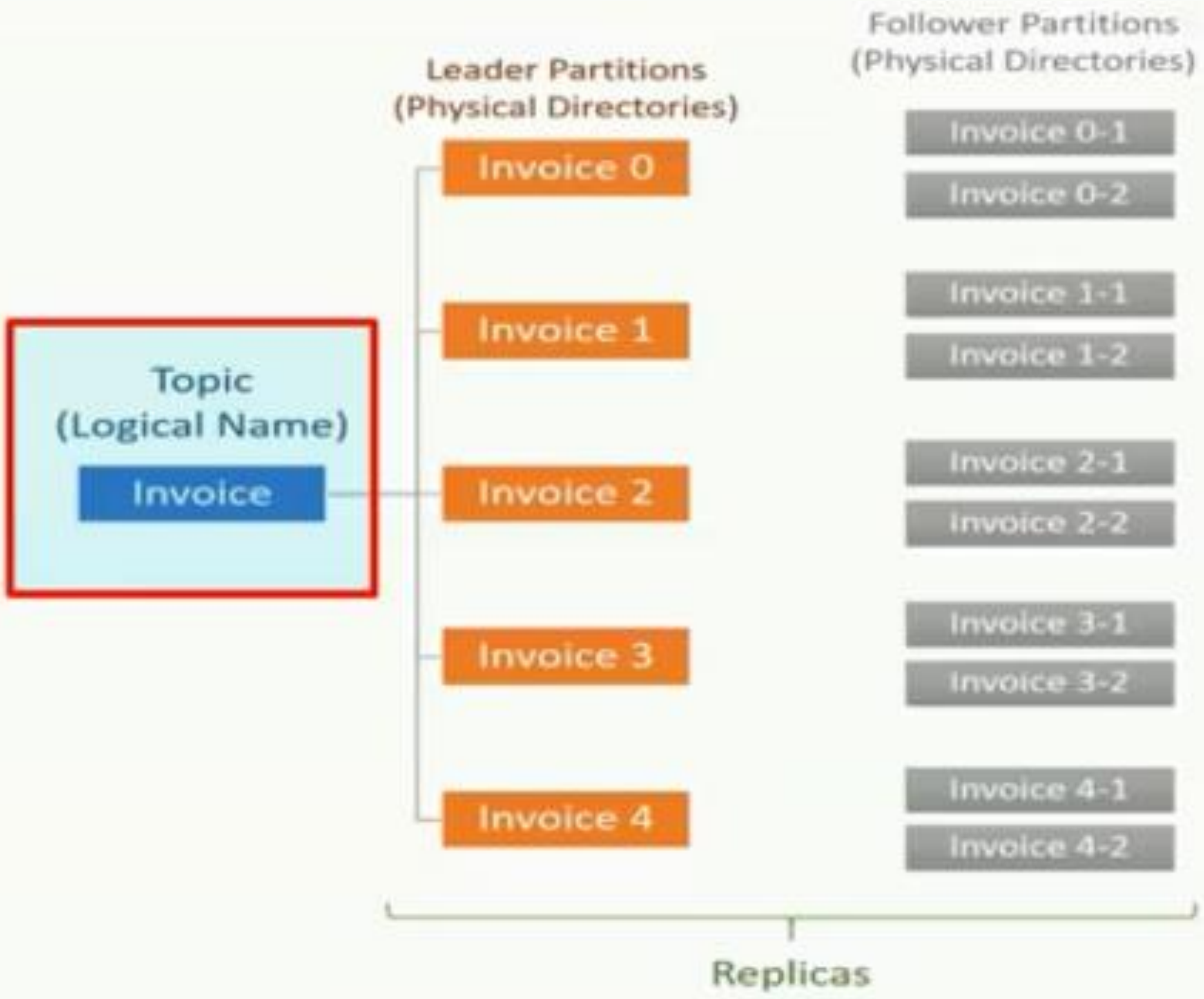
Partitions
(Physical Directories)

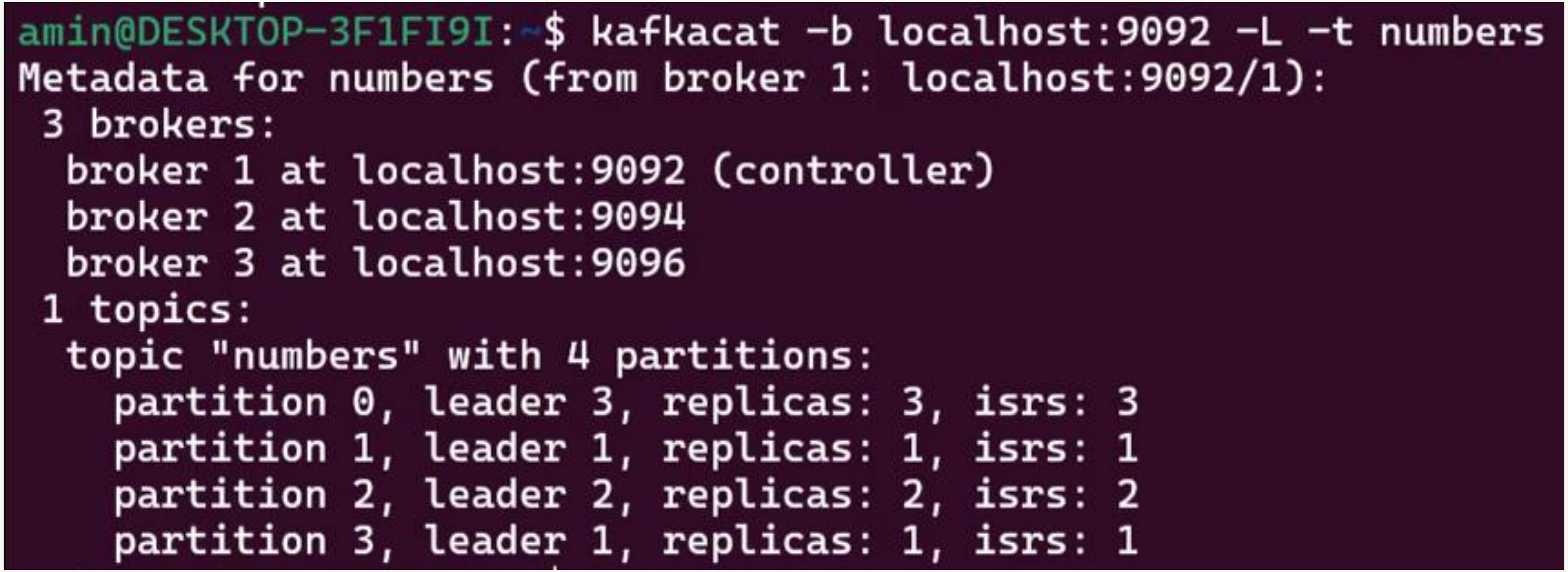
What is
Replication Factor?

Number of Copies for each Partition

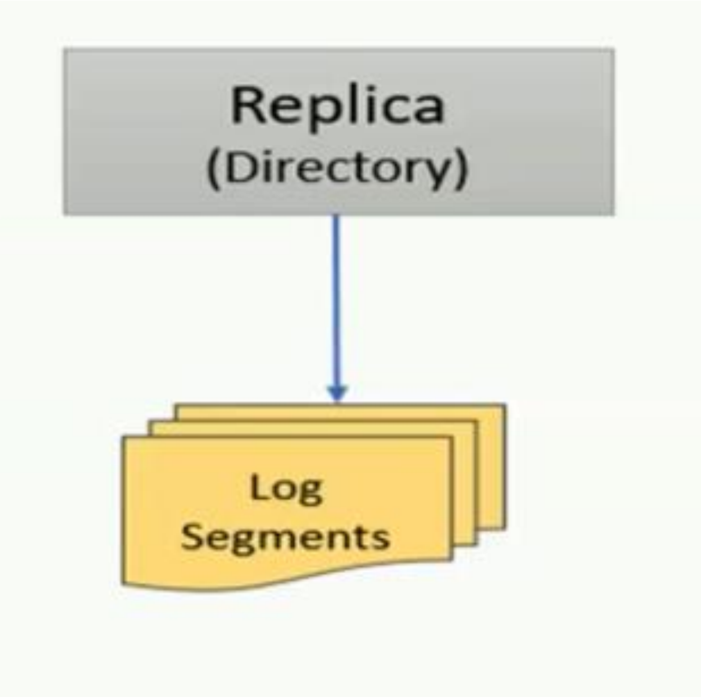
Number of Replicas (15) = Partitions (5) X Replication (3)

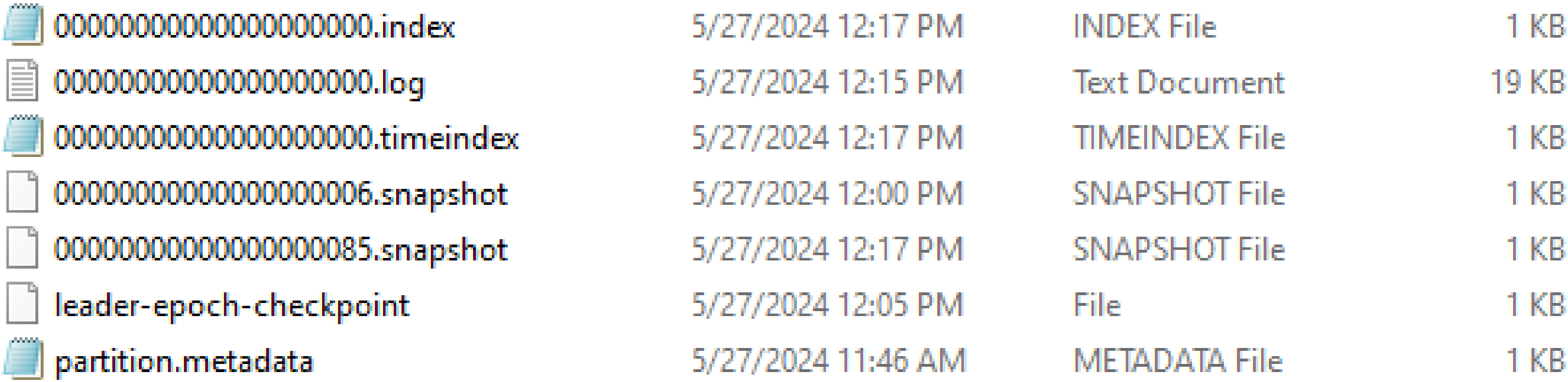
1. Leader Partitions
2. Follower Partitions



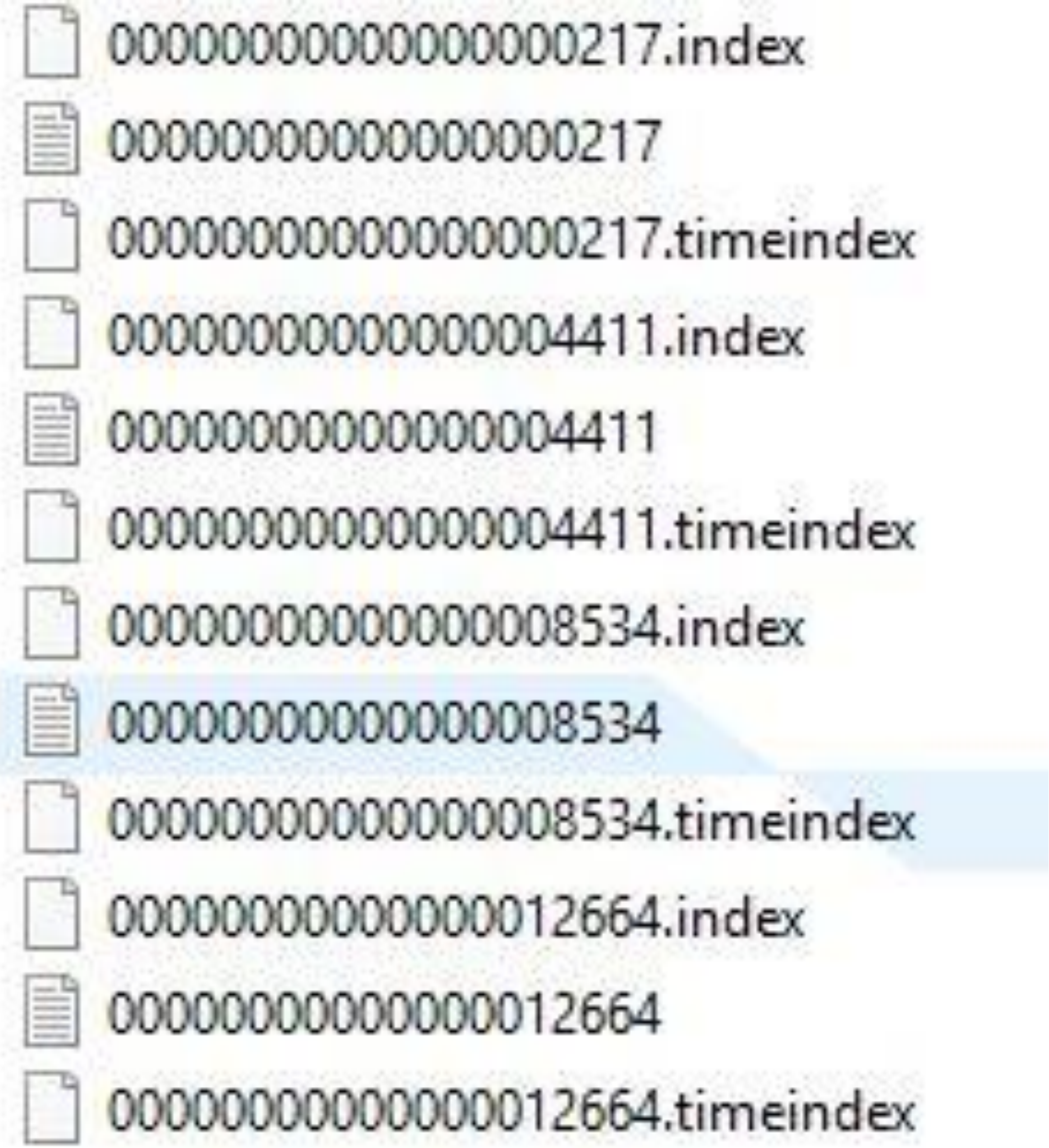


```
amin@DESKTOP-3F1FI9I:~$ kafkacat -b localhost:9092 -L -t numbers
Metadata for numbers (from broker 1: localhost:9092/1):
 3 brokers:
  broker 1 at localhost:9092 (controller)
  broker 2 at localhost:9094
  broker 3 at localhost:9096
 1 topics:
  topic "numbers" with 4 partitions:
    partition 0, leader 3, replicas: 3, isrs: 3
    partition 1, leader 1, replicas: 1, isrs: 1
    partition 2, leader 2, replicas: 2, isrs: 2
    partition 3, leader 1, replicas: 1, isrs: 1
```



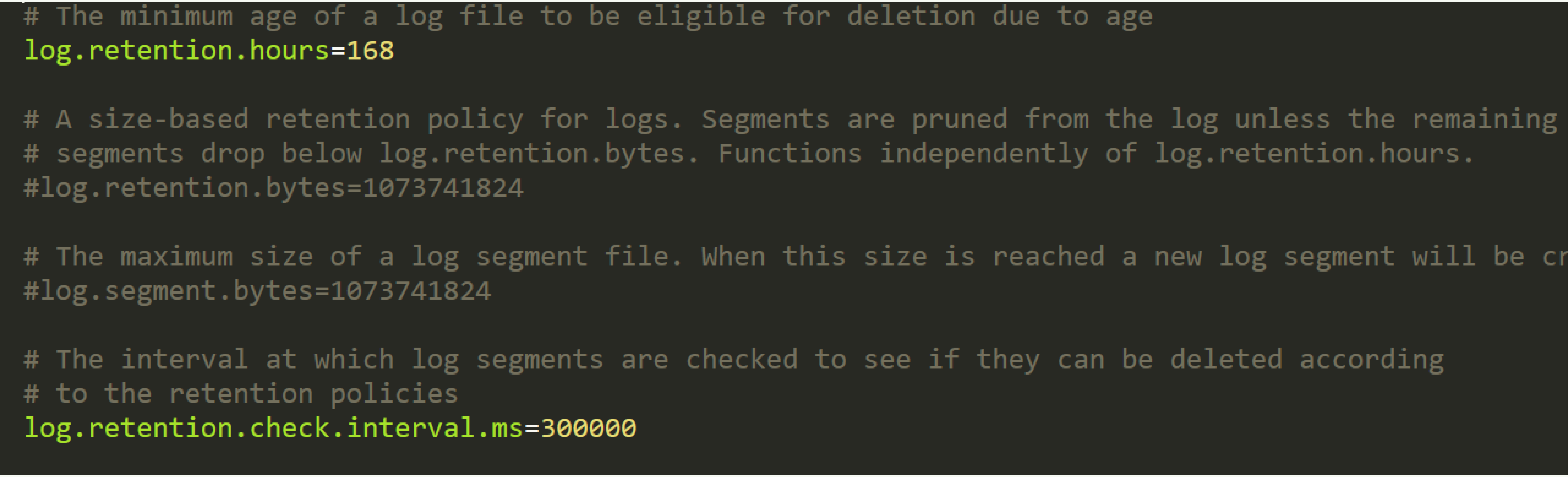


00000000000000000000.index	5/27/2024 12:17 PM	INDEX File	1 KB
00000000000000000000.log	5/27/2024 12:15 PM	Text Document	19 KB
00000000000000000000.timeindex	5/27/2024 12:17 PM	TIMEINDEX File	1 KB
00000000000000000006.snapshot	5/27/2024 12:00 PM	SNAPSHOT File	1 KB
00000000000000000085.snapshot	5/27/2024 12:17 PM	SNAPSHOT File	1 KB
leader-epoch-checkpoint	5/27/2024 12:05 PM	File	1 KB
partition.metadata	5/27/2024 11:46 AM	METADATA File	1 KB

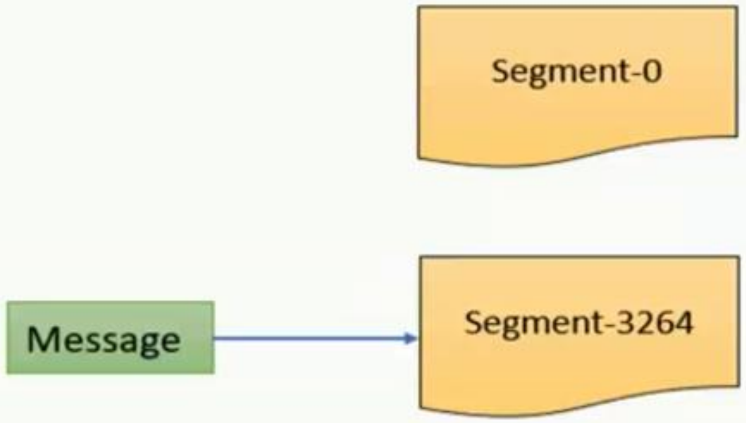


```
00000000000000000217.index
00000000000000000217
00000000000000000217.timeindex
00000000000000004411.index
00000000000000004411
00000000000000004411.timeindex
00000000000000008534.index
00000000000000008534
00000000000000008534.timeindex
00000000000000012664.index
00000000000000012664
00000000000000012664.timeindex
```

1 GB = 1073741824 bytes

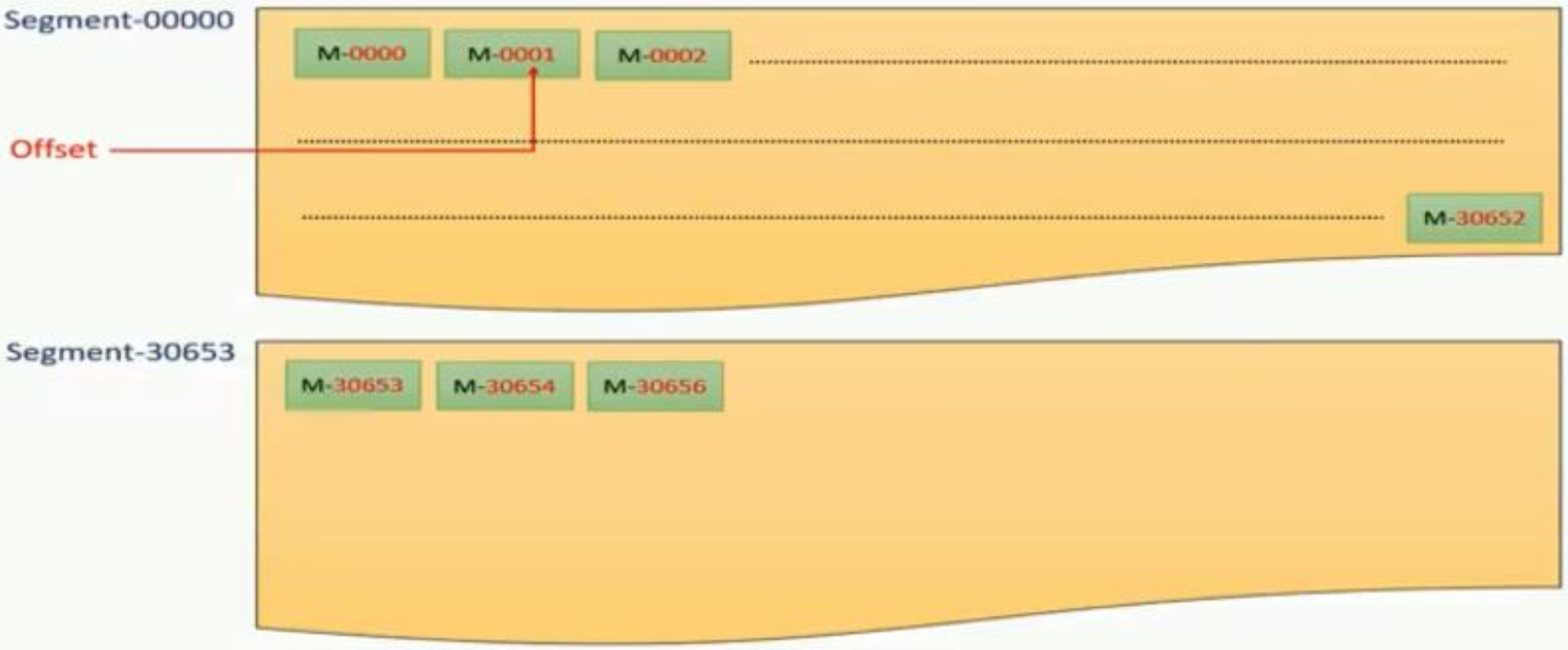


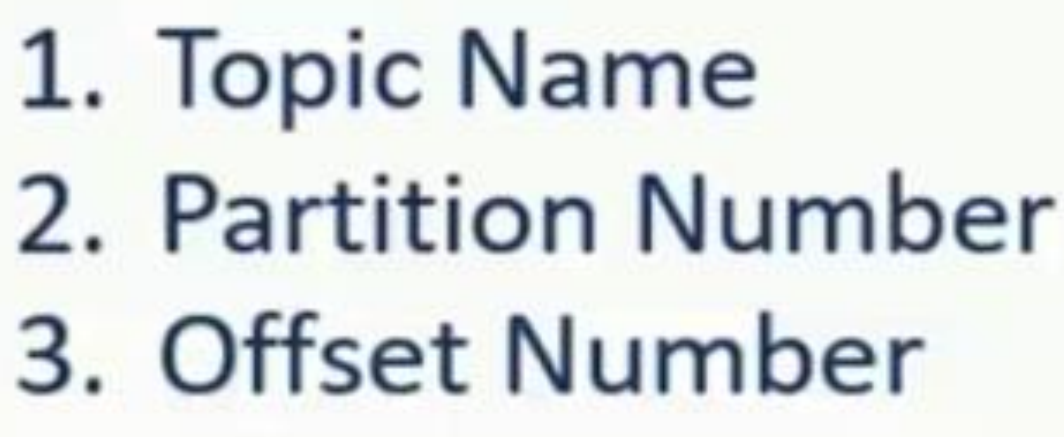
```
# The minimum age of a log file to be eligible for deletion due to age
log.retention.hours=168

# A size-based retention policy for logs. Segments are pruned from the log unless the remaining
# segments drop below log.retention.bytes. Functions independently of log.retention.hours.
#log.retention.bytes=1073741824

# The maximum size of a log segment file. When this size is reached a new log segment will be cr
#log.segment.bytes=1073741824

# The interval at which log segments are checked to see if they can be deleted according
# to the retention policies
log.retention.check.interval.ms=300000
```



Every offset stores in a 64-bit integer numbers that its range is

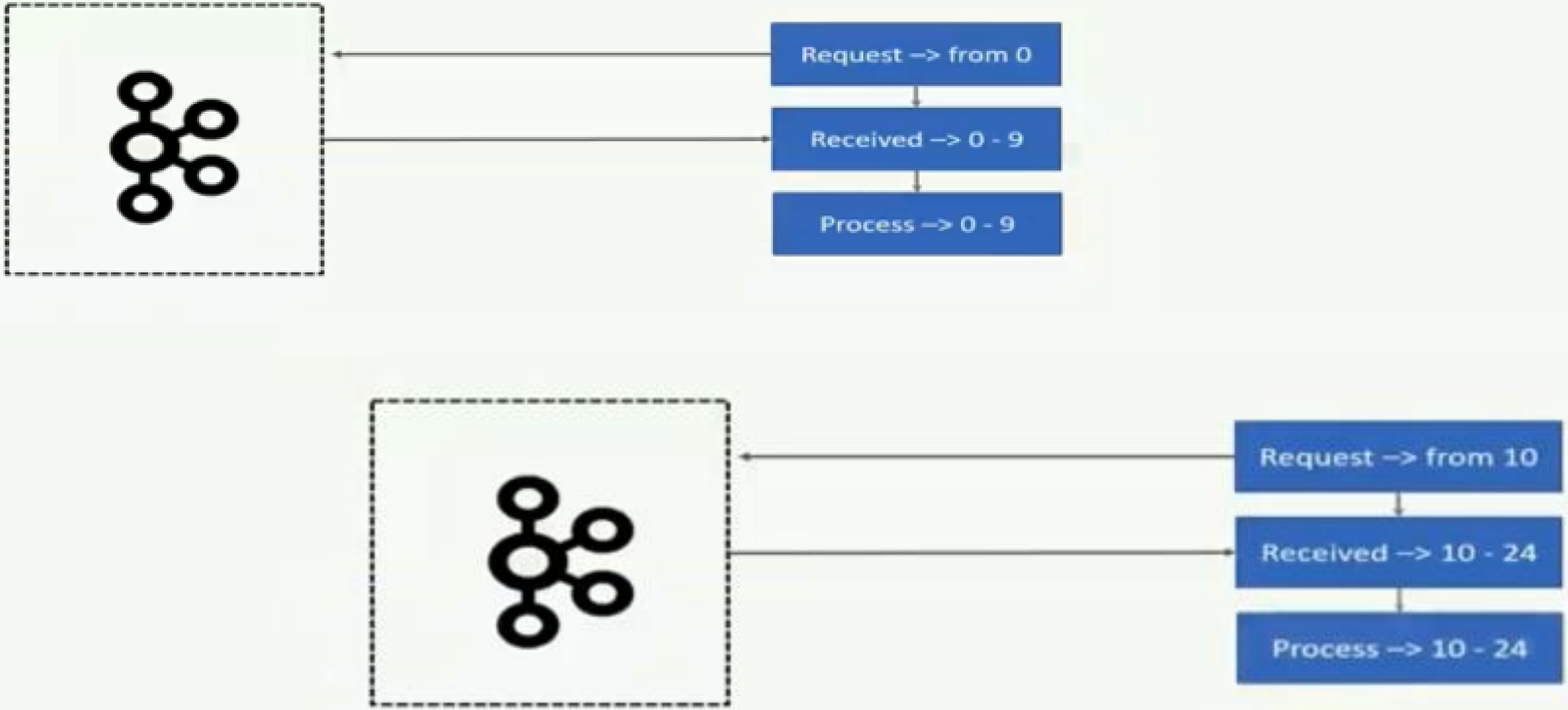
0000000000000000000 till 18/446/744/073/709/551/615





1. Topic Name
2. Partition Number
3. Offset Number

```
SELECT * FROM invoice WHERE customer-id = 091
```



Request –> from 0

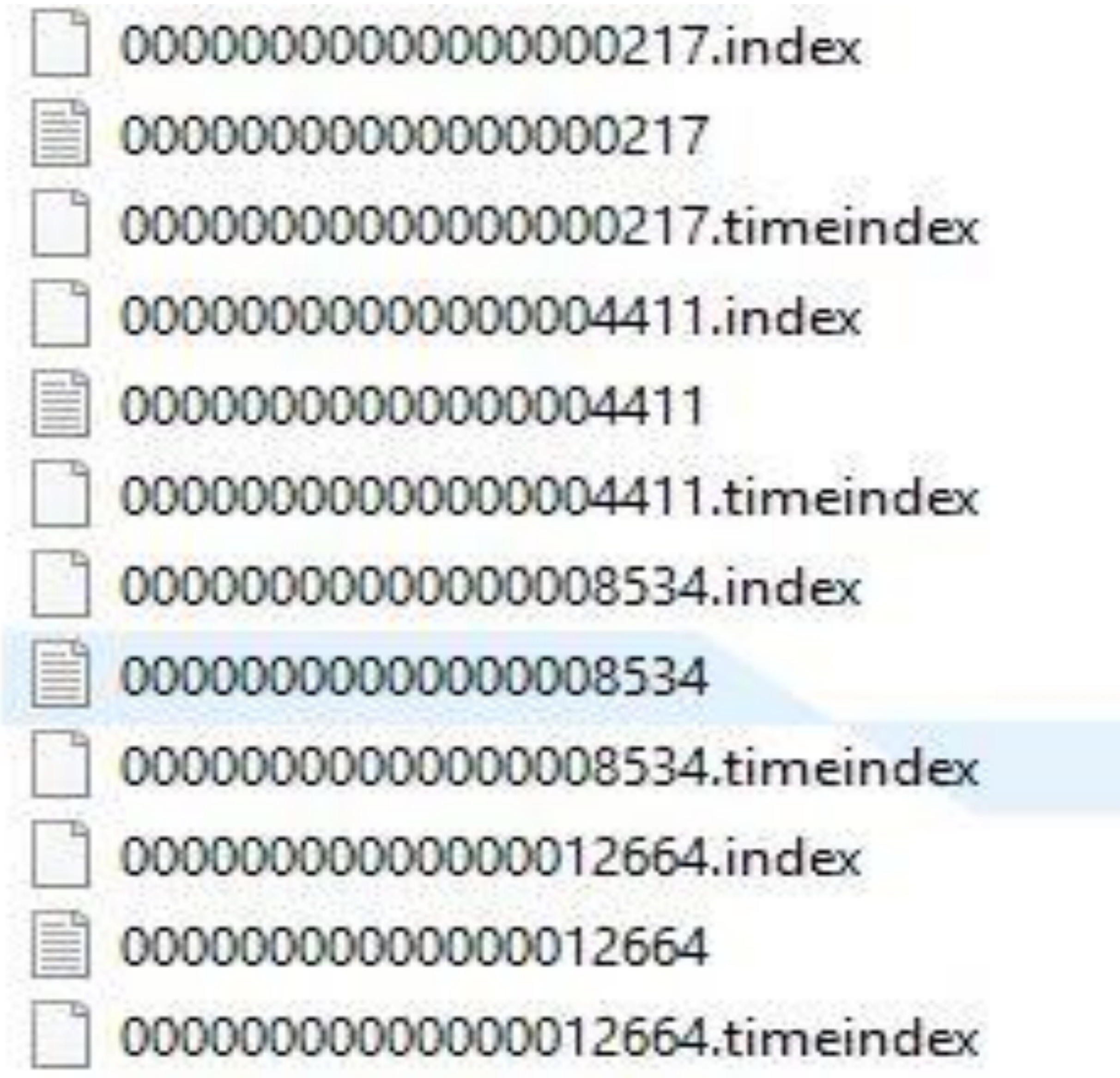
Received –> 0 - 9

Process –> 0 - 9

Request –> from 10

Received –> 10 - 24

Process –> 10 - 24

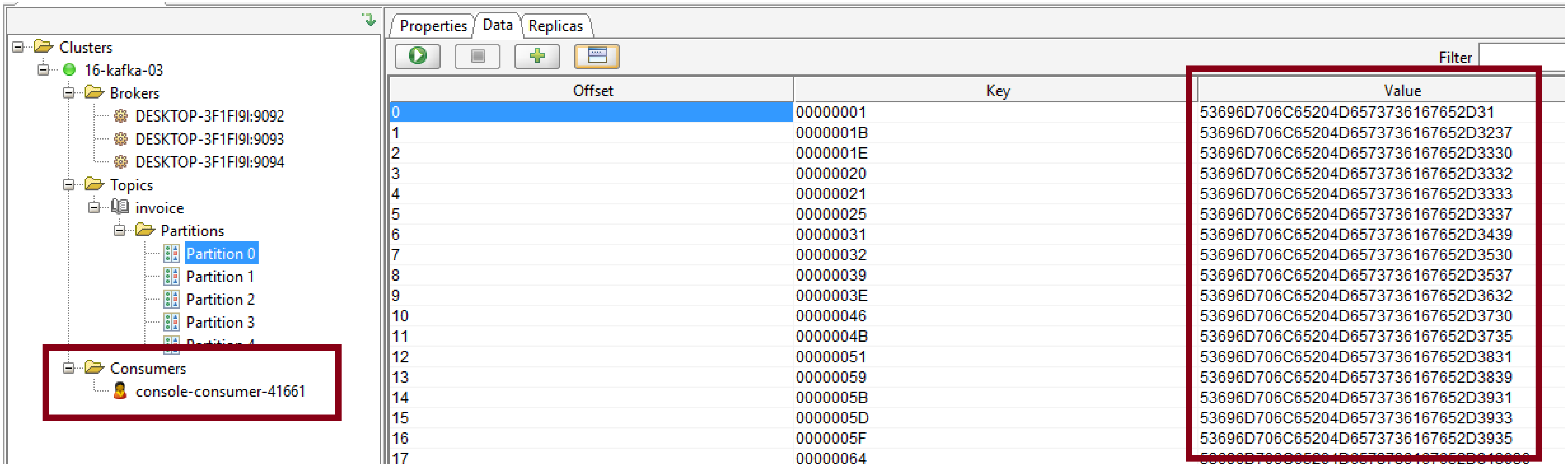


```
00000000000000000217.index
00000000000000000217
00000000000000000217.timeindex
00000000000000004411.index
00000000000000004411
00000000000000004411.timeindex
00000000000000008534.index
00000000000000008534
00000000000000008534.timeindex
00000000000000012664.index
00000000000000012664
00000000000000012664.timeindex
```

Upon the request of clients Kafka with use of .index file can locate the offset which has been requested.

Clients can request messages based on time. So Kafka uses .timeindex.

Kafka stores records in bytes, and when a fetch request comes in from a consumer, Kafka returns records in bytes.



Offset	Key	Value
0	00000001	53696D706C65204D657373616765 2D31
1	0000001B	53696D706C65204D6573736167652D3237
2	0000001E	53696D706C65204D6573736167652D3330
3	00000020	53696D706C65204D6573736167652D3332
4	00000021	53696D706C65204D6573736167652D3333
5	00000025	53696D706C65204D6573736167652D3337
6	00000031	53696D706C65204D6573736167652D3439
7	00000032	53696D706C65204D6573736167652D3530
8	00000039	53696D706C65204D6573736167652D3537
9	0000003E	53696D706C65204D6573736167652D3632
10	00000046	53696D706C65204D6573736167652D3730
11	0000004B	53696D706C65204D6573736167652D3735
12	00000051	53696D706C65204D6573736167652D3831
13	00000059	53696D706C65204D6573736167652D3839
14	0000005B	53696D706C65204D6573736167652D3931
15	0000005D	53696D706C65204D6573736167652D3933
16	0000005F	53696D706C65204D6573736167652D3935
17	00000064	[illegible]