YouTube Video Analytics

Kafka Producer --- It reads each record and send it to topic

```
from kafka import KafkaProducer
import json
import time
import pandas as pd
import numpy as np
from json import dumps

class NpEncoder(json.JSONEncoder):
    def default(self, obj):
        if isinstance(obj, np.integer):
            return int(obj)
        elif isinstance(obj, np.floating):
            return float(obj)
        elif isinstance(obj, np.ndarray):
            return obj.tolist()
        elif isinstance(obj, np.bool_):
            return obj.tolist()
        else:
            return super(NpEncoder, self).default(obj)

KAFKA_TOPIC_NAME_CONS = "USvideos_clean_data1"

Transaction_Data=pd.read_csv("USvideos_clean.csv")

def json_serializer(data):
    return json.dumps(data,cls=NpEncoder).encode("utf-8")

if __name__ == "__main__":
    print("Kafka Producer Application Started ... ")
```

```
producer = KafkaProducer(bootstrap_servers=['192.168.81.131:9092'],
                         value_serializer=json_serializer)

message = None

if 1==1:
    for i in range(80000):
        message={}
        message["video_id"] = Transaction_Data['video_id'][i]
        message["trending_date"] = Transaction_Data['trending_date'][i]
        message["title"] = Transaction_Data['title'][i]
        message["channel_title"] = Transaction_Data['channel_title'][i]
        message["category_id"] = Transaction_Data['category_id'][i]
        message["publish_time"] = Transaction_Data['publish_time'][i]
        message["tags"] = Transaction_Data['tags'][i]
        message["views"] = Transaction_Data['views'][i]
        message["likes"] = Transaction_Data['likes'][i]
        message["dislikes"] = Transaction_Data['dislikes'][i]
        message["comment_count"] = Transaction_Data['comment_count'][i]
        message["thumbnail_link"] = Transaction_Data['thumbnail_link'][i]
        message["comments_disabled"] = Transaction_Data['comments_disabled'][i]
        message["ratings_disabled"] = Transaction_Data['ratings_disabled'][i]
        message["video_error_or_removed"] = Transaction_Data['video_error_or_removed'][i]
        message["description"] = Transaction_Data['description'][i]



        print("Message to be sent: ", message)
        time.sleep(5)
        producer.send(KAFKA_TOPIC_NAME_CONS, message)
        time.sleep(1)
else:
    print("END")
```

Kafka Consumer --- It consumes each record from the topic

```
from kafka import KafkaConsumer
import json
from json import loads



if __name__ == "__main__":
    consumer = KafkaConsumer(
        "USvideos_clean_data1",
        bootstrap_servers='192.168.81.131:9092',
        auto_offset_reset='earliest',
        group_id="consumer-group-a",
    )
    print("starting the consumer")

    for msg in consumer:
        print("Video_Data = {}".format(json.loads(msg.value)))
```

FlumeConf for reading data from topic and sending to hdfs –

```
# Naming the components on the current agent.
//source, sink, channel Name config
agent.sources = kafkasource
agent.channels = memoryChannel
agent.sinks = hdfs-sink

# Describing/Configuring the source
agent.sources.kafkasource.type = org.apache.flume.source.kafka.KafkaSource
agent.sources.kafkasource.kafka.bootstrap.servers = 192.168.81.131:9092
agent.sources.kafkasource.topic = USvideos_clean_data1
agent.sources.kafkasource.consumer.group.id = consumer-group-a
agent.sources.kafkasource.batchsize = 100
agent.sources.kafkasource.batchDurationMillis = 200
agent.sources.kafkasource.zookeeperConnect = localhost:2181
```

```
#Use a channel which buffers events in memory
agent.channels.memoryChannel.type = memory
agent.channels.memoryChannel.capacity = 1000
agent.channels.memoryChannel.transactionCapacity = 1000

# Binding the source and sink to the channel
agent.sources.kafkasource.channels = memoryChannel
agent.sinks.hdfs-sink.channel = memoryChannel

# Describing/Configuring the sink
agent.sinks.hdfs-sink.type = hdfs
agent.sinks.hdfs-sink.hdfs.filePrefix = flume-hdfs
agent.sinks.hdfs-sink.hdfs.fileSuffix =.csv
agent.sinks.hdfs-sink.hdfs.fileType = DataStream
agent.sinks.hdfs-sink.hdfs.useLocalTimeStamp = true
agent.sinks.hdfs-sink.hdfs.UseRawLocalFileSystem = true
agent.sinks.hdfs-sink.hdfs.path = hdfs://localhost:9000/kafka2/
agent.sinks.hdfs-sink.hdfs.rollSize = 0
```

Kafka Spark Streaming --- It reads from the local HDFS and displays results

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
from pyspark.sql.window import Window


def print(string):
    pass


if __name__ == "__main__":
    print("PySpark Structured Streaming with Kafka Demo Application Started ...")
```

```
spark = SparkSession \
    .builder \
    .appName("PySpark Structured Streaming with Kafka Demo") \
    .master("local[*]") \
    .getOrCreate()


detail_schema = StructType() \
    .add("video_id", StringType()) \
    .add("trending_date", TimestampType()) \
    .add("title", StringType()) \
    .add("channel_title", StringType()) \
    .add("category_id", IntegerType()) \
    .add("publish_time", StringType()) \
    .add("tags", StringType()) \
    .add("views", IntegerType()) \
    .add("likes", IntegerType()) \
    .add("dislikes", IntegerType()) \
    .add("comment_count", IntegerType()) \
    .add("thumbnail_link", StringType()) \
    .add("comments_disabled", StringType()) \
    .add("ratings_disabled", StringType()) \
    .add("video_error_or_removed", StringType()) \
    .add("description", StringType())

df = spark.read.format("csv").options(header = 'true', inferSchema =
'true').load("hdfs://localhost:9000/kafka2/")
df.printSchema()
```

Insights on Historical Data

- **Top 3 videos for which user interaction (views + likes + dislikes + comments) is the highest.**

  spark.sql("select video_type, video_id,sum(views) as total_views,sum(likes) as total_likes,sum(dislikes) as total_dislikes,sum(comment_count) as total_comment_count from VideosData group by video_id ,video_type order by total_views desc").show(3)

```
scala> spark.sql("select video_type, video_id,sum(views) as total_views,sum(likes) as total_likes,sum(dislikes) as total_dislikes,sum(comment_
count) as total_comment_count from VideosData group by video_id ,video_type order by total_views desc").show(3)
+----------+-----------+-----------+-----------+--------------+-------------------+
|video_type|   video_id|total_views|total_likes|total_dislikes|total_comment_count|
+----------+-----------+-----------+-----------+--------------+-------------------+
|     Music|VYOjWnS4cMY| 3758488765|   96700818|       6054434|           10151289|
|     Music|ffxKSjUwKdU| 1529291326|   49451353|       1877988|            4143608|
|     Music|7C2z4GqqS5E| 1283188291|   71835050|       2389776|           15568561|
+----------+-----------+-----------+-----------+--------------+-------------------+
only showing top 3 rows
```

Insights on Historical Data

- **Bottom 3 videos for which user interaction (views + likes + dislikes + comments) is lowest.**

  spark.sql("select video_type, video_id,sum(views) as total_views,sum(likes) as total_likes,sum(dislikes) as total_dislikes,sum(comment_count) as total_comment_count from VideosData group by video_id ,video_type order by total_views").show(3)

```
scala> spark.sql("select video_type, video_id,sum(views) as total_views,sum(likes) as total_likes,sum(dislikes) as total_dislikes,sum(comment_
count) as total_comment_count from VideosData group by video_id ,video_type order by total_views").show(3)
+---------------+-----------+-----------+-----------+--------------+-------------------+
|     video_type|   video_id|total_views|total_likes|total_dislikes|total_comment_count|
+---------------+-----------+-----------+-----------+--------------+-------------------+
|News & Politics|zeQaJGkFyqQ|        748|          9|             0|                  0|
|      Education|qg0GdM60syI|        773|          2|             0|                  0|
|News & Politics|0devsSCkYRY|       1381|         35|             6|                  4|
+---------------+-----------+-----------+-----------+--------------+-------------------+
only showing top 3 rows
```

- **Top 3 videos of each category in each year**

- By number of views

  In 2018

  spark.sql("select category_id, video_type,year, sum(views) as total_views from (select * from VideosData where year='2018') group by category_id,year, video_type order by total_views desc").show(3)

```
scala> spark.sql("select category_id, video_type,year, sum(views) as total_views from (select * from VideosData where year='2018') group by ca
tegory_id,year, video_type order by total_views desc").show(3)
+-----------+----------------+----+-----------+
|category_id|      video_type|year|total_views|
+-----------+----------------+----+-----------+
|         10|           Music|2018|35461866156|
|         24|   Entertainment|2018|15598541905|
|          1|Film & Animation|2018| 6407893933|
+-----------+----------------+----+-----------+
only showing top 3 rows
```

  In 2017

  spark.sql("select category_id, video_type,year, sum(views) as total_views from (select * from VideosData where year='2017') group by category_id,year, video_type order by total_views desc").show(3)

```
scala> spark.sql("select category_id, video_type,year, sum(views) as total_views from (select * from VideosData where year='2017') group by ca
tegory_id,year, video_type order by total_views desc").show(3)
+-----------+-------------+----+-----------+
|category_id|   video_type|year|total_views|
+-----------+-------------+----+-----------+
|         10|        Music|2017| 4504426631|
|         24|Entertainment|2017| 3877770049|
|         23|       Comedy|2017| 1119573722|
+-----------+-------------+----+-----------+
only showing top 3 rows
```

- By number of comments

  In 2018

  spark.sql("select category_id, video_type,year, sum(comment_count) as total_comment_count from (select * from VideosData where year='2018') group by category_id,year, video_type order by total_comment_count desc").show(3)

```
scala> spark.sql("select category_id, video_type,year, sum(comment_count) as total_comment_count from (select * from VideosData where year='20
18') group by category_id,year, video_type order by total_comment_count desc").show(3)
+-----------+--------------+----+-------------------+
|category_id|    video_type|year|total_comment_count|
+-----------+--------------+----+-------------------+
|         10|         Music|2018|          106512772|
|         24| Entertainment|2018|           44337020|
|         22|People & Blogs|2018|           19736458|
+-----------+--------------+----+-------------------+
only showing top 3 rows
```

  In 2017

  spark.sql("select category_id, video_type,year, sum(comment_count) as total_comment_count from (select * from VideosData where year='2017') group by category_id,year, video_type order by total_comment_count desc").show(3)

```
scala> spark.sql("select category_id, video_type,year, sum(comment_count) as total_comment_count from (select * from VideosData where year='20
17') group by category_id,year, video_type order by total_comment_count desc").show(3)
+-----------+-------------+----+-------------------+
|category_id|   video_type|year|total_comment_count|
+-----------+-------------+----+-------------------+
|         24|Entertainment|2017|           19269872|
|         10|        Music|2017|           18058187|
|         23|       Comedy|2017|            5156204|
+-----------+-------------+----+-------------------+
only showing top 3 rows
```

- By number of likes

  In 2018

  spark.sql("select category_id, video_type,year, sum(likes) as total_likes from (select * from VideosData where year='2018') group by category_id,year, video_type order by total_likes desc").show(3)

```
scala> spark.sql("select category_id, video_type,year, sum(likes) as total_likes from (select * from VideosData where year='2018') group by ca
tegory_id,year, video_type order by total_likes desc").show(3)
+-----------+-------------+----+-----------+
|category_id|   video_type|year|total_likes|
+-----------+-------------+----+-----------+
|         10|        Music|2018| 1207148517|
|         24|Entertainment|2018|  378686601|
|         23|       Comedy|2018|  158775918|
+-----------+-------------+----+-----------+
only showing top 3 rows
```

  In 2017

  spark.sql("select category_id, video_type,year, sum(likes) as total_likes from (select * from VideosData where year='2017') group by category_id,year, video_type order by total_likes desc").show(3)

```
scala> spark.sql("select category_id, video_type,year, sum(likes) as total_likes from (select * from VideosData where year='2017') group by ca
tegory_id,year, video_type order by total_likes desc").show(3)
+-----------+-------------+----+-----------+
|category_id|   video_type|year|total_likes|
+-----------+-------------+----+-----------+
|         10|        Music|2017|  200720125|
|         24|Entertainment|2017|  118115564|
|         23|       Comedy|2017|   56695461|
+-----------+-------------+----+-----------+
only showing top 3 rows
```

- Highest user interaction

  In 2018

spark.sql("select video_type,video_id,year,sum(total_views + total_likes + total_dislikes+ total_comment_count) as Highest_User_interaction FROM (select * from User_interaction where year='2018') group by video_type,video_id,year order by Highest_User_interaction desc").show(3)

```
scala> spark.sql("select video_type,video_id,year,sum(total_views + total_likes + total_dislikes+ total_comment_count) as Highest_User_interac
tion FROM (select * from User_interaction where year='2018') group by video_type,video_id,year order by Highest_User_interaction desc").show(3
)
+----------+-----------+----+-----------------------+
|video_type|   video_id|year|Highest_User_interaction|
+----------+-----------+----+-----------------------+
|     Music|VYOjWnS4cMY|2018|              3871395306|
|     Music|ffxKSjUwKdU|2018|              1584764275|
|     Music|7C2z4GqqS5E|2018|              1372981678|
+----------+-----------+----+-----------------------+
only showing top 3 rows
```

In 2017

spark.sql("select video_type,video_id,year,sum(total_views + total_likes + total_dislikes+ total_comment_count) as Highest_User_interaction FROM (select * from User_interaction where year='2017') group by video_type,video_id,year order by Highest_User_interaction desc").show(3)

```
scala> spark.sql("select video_type,video_id,year,sum(total_views + total_likes + total_dislikes+ total_comment_count) as Highest_User_interac
tion FROM (select * from User_interaction where year='2017') group by video_type,video_id,year order by Highest_User_interaction desc").show(3
)
+-------------+-----------+----+-----------------------+
|   video_type|   video_id|year|Highest_User_interaction|
+-------------+-----------+----+-----------------------+
|Entertainment|FlsCjmMhFmw|2017|               816592094|
|Entertainment|6ZfuNTqbHE8|2017|               694350219|
|        Music|TyHvyGVs42U|2017|               553116684|
+-------------+-----------+----+-----------------------+
only showing top 3 rows
```

- **Top 3 videos in each month**

  Likes or Dislikes ratio is highest – month 11

spark.sql("select a.video_type,a.video_id, (a.total_likes/b.total_dislikes) as highest_ratio from (select video_type,video_id,sum(likes) as total_likes from VideosData where month='11' group by video_id ,video_type)a,(select video_type,video_id, sum(dislikes) as total_dislikes from VideosData where month='11' group by video_id ,video_type)b").show()

```
scala> spark.sql("select a.video_type,a.video_id, (a.total_likes/b.total_dislikes) as highest_ratio from (select video_type,video_id,sum(likes
) as total_likes from VideosData where month='11' group by video_id ,video_type)a,(select video_type,video_id, sum(dislikes) as total_dislikes
 from VideosData where month='11' group by video_id ,video_type)b").show(3)
+----------+-----------+-----------------+
|video_type|   video_id|    highest_ratio|
+----------+-----------+-----------------+
|     Music|YlvCVbfS9M0|61.79863091842556|
|     Music|YlvCVbfS9M0|5701.736842105263|
|     Music|YlvCVbfS9M0|50.27053364269142|
+----------+-----------+-----------------+
only showing top 3 rows
```

Likes or Dislikes ratio is highest – month 12

spark.sql("select a.video_type,a.video_id, (a.total_likes/b.total_dislikes) as highest_ratio from (select video_type,video_id,sum(likes) as total_likes from VideosData where month='12' group by video_id ,video_type)a,(select video_type,video_id, sum(dislikes) as total_dislikes from VideosData where month='11' group by video_id ,video_type)b").show()

```
scala> spark.sql("select a.video_type,a.video_id, (a.total_likes/b.total_dislikes) as highest_ratio from (select video_type,video_id,sum(likes
) as total_likes from VideosData where month='12' group by video_id ,video_type)a,(select video_type,video_id, sum(dislikes) as total_dislikes
 from VideosData where month='11' group by video_id ,video_type)b").show(3)
+----------+-----------+------------------+
|video_type|   video_id|     highest_ratio|
+----------+-----------+------------------+
|    Comedy|2v34qDKvLqM|0.7706788362806617|
|    Comedy|2v34qDKvLqM| 71.10526315789474|
|    Comedy|2v34qDKvLqM|0.6269141531322506|
+----------+-----------+------------------+
only showing top 3 rows
```

- **Top 3 videos of each category in each month**

By number of views
In Month = 12
spark.sql("select video_type,category_id,month,sum(views) as total_views from VideosData where month = '12' group by category_id,month,video_type").show(3)

```
scala> spark.sql("select video_type,category_id,month,sum(views) as total_views from VideosData where month = '12' group by category_id,month,
video_type").show(3)
+----------------+-----------+-----+-----------+
|      video_type|category_id|month|total_views|
+----------------+-----------+-----+-----------+
|          Gaming|         20|   12|   41946209|
|          Sports|         17|   12|  267648081|
|Film & Animation|          1|   12|  388564421|
+----------------+-----------+-----+-----------+
only showing top 3 rows
```

In Month = 11
spark.sql("select video_type,category_id,month,sum(views) as total_views from VideosData where month = '11' group by category_id,month,video_type").show(3)

```
scala> spark.sql("select video_type,category_id,month,sum(views) as total_views from VideosData where month = '11' group by category_id,month,
video_type").show(3)
+--------------------+-----------+-----+-----------+
|          video_type|category_id|month|total_views|
+--------------------+-----------+-----+-----------+
|Nonprofits & Acti...|         29|   11|       7427|
|     Travel & Events|         19|   11|   17151667|
|    Autos & Vehicles|          2|   11|   57830731|
+--------------------+-----------+-----+-----------+
only showing top 3 rows
```

By number of likes

In Month = 12
spark.sql("select video_type,category_id,month,sum(likes) as total_likes from VideosData where month = '12' group by category_id,month,video_type").show(3)

```
scala> spark.sql("select video_type,category_id,month,sum(likes) as total_likes from VideosData where month = '12' group by category_id,month,
video_type").show(3)
+----------------+-----------+-----+-----------+
|      video_type|category_id|month|total_likes|
+----------------+-----------+-----+-----------+
|          Gaming|         20|   12|    1308987|
|          Sports|         17|   12|    6824961|
|Film & Animation|          1|   12|    8274198|
+----------------+-----------+-----+-----------+
only showing top 3 rows
```

In Month = 11
spark.sql("select video_type,category_id,month,sum(likes) as total_likes from VideosData where month = '12' group by category_id,month,video_type").show(3)

```
scala> spark.sql("select video_type,category_id,month,sum(likes) as total_likes from VideosData where month = '11' group by category_id,month,
video_type").show(3)
+--------------------+-----------+-----+-----------+
|          video_type|category_id|month|total_likes|
+--------------------+-----------+-----+-----------+
|Nonprofits & Acti...|         29|   11|          0|
|     Travel & Events|         19|   11|     210251|
|    Autos & Vehicles|          2|   11|    1025045|
+--------------------+-----------+-----+-----------+
only showing top 3 rows
```

By number of dislikes

In Month = 12

spark.sql("select video_type,category_id,month,sum(dislikes) as total_dislikes from VideosData where month = '12' group by category_id,month,video_type order by total_dislikes desc ").show(3)

```
scala> spark.sql("select video_type,category_id,month,sum(dislikes) as total_dislikes from VideosData where month = '12' group by category_id,
month,video_type order by total_dislikes desc ").show(3)
+--------------+-----------+-----+--------------+
|    video_type|category_id|month|total_dislikes|
+--------------+-----------+-----+--------------+
| Entertainment|         24|   12|      13124644|
|         Music|         10|   12|       4932351|
|People & Blogs|         22|   12|       1991136|
+--------------+-----------+-----+--------------+
only showing top 3 rows
```

In Month = 11

spark.sql("select video_type,category_id,month,sum(dislikes) as total_dislikes from VideosData where month = '11' group by category_id,month,video_type order by total_dislikes desc ").show(3)

```
scala> spark.sql("select video_type,category_id,month,sum(dislikes) as total_dislikes from VideosData where month = '11' group by category_id,
month,video_type order by total_dislikes desc ").show(3)
+--------------------+-----------+-----+--------------+
|          video_type|category_id|month|total_dislikes|
+--------------------+-----------+-----+--------------+
|               Music|         10|   11|       2647260|
|       Entertainment|         24|   11|       1299746|
|Science & Technology|         28|   11|        660323|
+--------------------+-----------+-----+--------------+
only showing top 3 rows
```

- **Top 3 channels**
  By number of total views
  spark.sql("select channel_title,sum(likes) as total_likes from VideosData group by channel_title order by total_likes desc").show(3)

```
scala> spark.sql("select channel_title,sum(likes) as total_likes from VideosData group by channel_title order by total_likes desc").show(3)
+-------------------+-----------+
|      channel_title|total_likes|
+-------------------+-----------+
|            ibighit|  199247121|
|ChildishGambinoVEVO|   96700818|
|       Dude Perfect|   60275557|
+-------------------+-----------+
only showing top 3 rows
```

- Likes or Dislikes ratio is highest
  spark.sql("select a.video_type,a.video_id,a.channel_title, (a.total_likes/b.total_dislikes) as highest_ratio from (select channel_title,video_type,video_id,sum(likes) as total_likes from VideosData group by video_id ,video_type,channel_title)a,(select channel_title,video_type,video_id, sum(dislikes) as total_dislikes from VideosData group by video_id ,video_type,channel_title)b order by highest_ratio").show()

```
scala> spark.sql("select a.video_type,a.video_id,a.channel_title, (a.total_likes/b.total_dislikes) as highest_ratio from (select channel_title
,video_type,video_id,sum(likes) as total_likes from VideosData group by video_id ,video_type,channel_title)a,(select channel_title,video_type,
video_id, sum(dislikes) as total_dislikes from VideosData group by video_id ,video_type,channel_title)b").show(3)
+----------+-----------+---------------+------------------+
|video_type|   video_id|  channel_title|     highest_ratio|
+----------+-----------+---------------+------------------+
|     Music|CziCidR4KcY|MariahCareyVEVO| 24.09050279329609|
|     Music|CziCidR4KcY|MariahCareyVEVO|1227.0487804878048|
|     Music|CziCidR4KcY|MariahCareyVEVO| 328.8169934640523|
+----------+-----------+---------------+------------------+
only showing top 3 rows
```

- By number of total comments
  spark.sql("select channel_title,sum(comment_count) as total_comment_count from VideosData group by channel_title order by total_comment_count desc").show(3)

```
scala> spark.sql("select channel_title,sum(comment_count) as total_comment_count from VideosData group by channel_title order by total_comment
_count desc").show(3)
+-------------------+-------------------+
|      channel_title|total_comment_count|
+-------------------+-------------------+
|            ibighit|           31817464|
|ChildishGambinoVEVO|           10151289|
|   jypentertainment|            7575510|
+-------------------+-------------------+
only showing top 3 rows
```

- **Top 3 categories**

  By number of total views
  spark.sql("select category_id,video_type,sum(views) as total_views from VideosData group by channel_title,category_id,video_type order by total_views desc").show(3)

```
scala> spark.sql("select category_id,video_type,sum(views) as total_views from VideosData group by channel_title,category_id,video_type order
by total_views  desc").show(3)
+-----------+----------+-----------+
|category_id|video_type|total_views|
+-----------+----------+-----------+
|         10|     Music| 3758488765|
|         10|     Music| 2235906679|
|         17|    Sports| 1870085178|
+-----------+----------+-----------+
only showing top 3 rows
```

- Likes or Dislikes ratio is highest

  spark.sql("select a.video_type,a.video_id,a.category_id, (a.total_likes/b.total_dislikes) as highest_ratio from (select category_id,video_type,video_id,sum(likes) as total_likes from VideosData group by video_id ,video_type,category_id)a,(select video_type,video_id, sum(dislikes) as total_dislikes from VideosData group by video_id ,video_type)b").show()

```
scala> spark.sql("select a.video_type,a.video_id,a.category_id, (a.total_likes/b.total_dislikes) as highest_ratio from (select category_id,vid
eo_type,video_id,sum(likes) as total_likes from VideosData group by video_id ,video_type,category_id)a,(select video_type,video_id, sum(dislik
es) as total_dislikes from VideosData group by video_id ,video_type)b").show(3)
+----------+-----------+-----------+-----------------+
|video_type|   video_id|category_id|    highest_ratio|
+----------+-----------+-----------+-----------------+
|    Comedy|5qpjK5DgCt4|         23|692.4552196235026|
|    Comedy|5qpjK5DgCt4|         23|63888.10526315789|
|    Comedy|5qpjK5DgCt4|         23|563.2825986078886|
+----------+-----------+-----------+-----------------+
only showing top 3 rows
```

- By number of total comments
  spark.sql("select category_id,video_type,sum(comment_count) as total_comment_count from VideosData group by channel_title,category_id,video_type order by total_comment_count desc").show(3)

```
scala> spark.sql("select category_id,video_type,sum(comment_count) as total_comment_count from VideosData group by channel_title,category_id,v
ideo_type order by total_comment_count desc").show(3)
+-----------+----------+-------------------+
|category_id|video_type|total_comment_count|
+-----------+----------+-------------------+
|         10|     Music|           31817464|
|         10|     Music|           10151289|
|         10|     Music|            7575510|
+-----------+----------+-------------------+
only showing top 3 rows
```

Top 3 videos

- Views on most recent date / (Recent Date - Publish Date)
  spark.sql("select trending_date,publish_time2 ,datediff(trending_date,publish_time2) from VideosData1").show(3)

```
scala> spark.sql("select trending_date,publish_time2 ,datediff(trending_date,publish_time2) from VideosData1").show(10)
+--------------+--------------+--------------------------------------+
|trending_date|publish_time2|datediff(trending_date, publish_time2)|
+--------------+--------------+--------------------------------------+
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
|   2017-11-14|   2017-11-13|                                     1|
+--------------+--------------+--------------------------------------+
only showing top 10 rows
```

- Calculate any 3 videos which got at least 5 comments on every 1000 views

  spark.sql("select a.video_type,a.video_id, (a.total_comment_count/b.total_likes) > 0.005 (select video_type,video_id,sum(comment_count) as total_comment_count from VideosData group by video_id ,video_type)a,(select video_type,video_id, sum(likes) as total_likes from VideosData group by video_id ,video_type)b").show(3)

```
scala> spark.sql("select a.video_type,a.video_id, (a.total_comment_count/b.total_likes)>0.005 from (select video_type,video_id,sum(comment_cou
nt) as total_comment_count from VideosData group by video_id ,video_type)a,(select video_type,video_id, sum(likes) as total_likes from VideosD
ata group by video_id ,video_type)b").show(3)
+----------+-----------+-----------------------------------------------------------------------------------------+
|video_type|   video_id|((CAST(total_comment_count AS DOUBLE) / CAST(total_likes AS DOUBLE)) > CAST(0.005 AS DOUBLE))|
+----------+-----------+-----------------------------------------------------------------------------------------+
|     Music|YlvCVbfS9M0|                                                                                     true|
|     Music|YlvCVbfS9M0|                                                                                     true|
|     Music|YlvCVbfS9M0|                                                                                     true|
+----------+-----------+-----------------------------------------------------------------------------------------+
only showing top 3 rows
```

- Calculate any 3 videos which got at least 4 likes on every 100 views

  spark.sql("select a.video_type,a.video_id, (a.total_likes/b.total_views)>0.04 from (select video_type,video_id,sum(likes) as total_likes from VideosData group by video_id ,video_type)a,(select video_type,video_id, sum(views) as total_views from VideosData group by video_id ,video_type)b").show(3)

```
scala> spark.sql("select a.video_type,a.video_id, (a.total_likes/b.total_views)>0.04 from (select video_type,video_id,sum(likes) as total_like
s from VideosData group by video_id ,video_type)a,(select video_type,video_id, sum(views) as total_views from VideosData group by video_id ,vi
deo_type)b").show(3)
+----------+-----------+-----------------------------------------------------------------------------------+
|video_type|   video_id|((CAST(total_likes AS DOUBLE) / CAST(total_views AS DOUBLE)) > CAST(0.04 AS DOUBLE))|
+----------+-----------+-----------------------------------------------------------------------------------+
|     Music|YlvCVbfS9M0|                                                                               true|
|     Music|YlvCVbfS9M0|                                                                               true|
|     Music|YlvCVbfS9M0|                                                                               true|
+----------+-----------+-----------------------------------------------------------------------------------+
only showing top 3 rows
```

- Number of videos published in each category
  spark.sql("select category_id,count(video_id) from VideosData group by category_id").show(15)

```
scala> spark.sql("select category_id,count(video_id) from VideosData group by category_id").show(30)
+-----------+---------------+
|category_id|count(video_id)|
+-----------+---------------+
|         15|            916|
|         29|             53|
|         22|           3061|
|         28|           2361|
|         43|             57|
|         27|           1642|
|         17|           2125|
|         26|           4140|
|         19|            402|
|         23|           3435|
|         25|           2409|
|         24|           9819|
|          1|           2348|
|         20|            803|
|         10|           6437|
|          2|            386|
+-----------+---------------+
```

Saving Spark job to HDFS

res24.write.format("com.databricks.spark.csv").save("hdfs://localhost:9000/kafka2/")

```
scala> res24.write.format("com.databricks.spark.csv").save("hdfs://localhost:9000/kafka2/")
21/07/26 18:52:28 WARN DataStreamer: Caught exception            (33 + 2) / 200]
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1252)
        at java.lang.Thread.join(Thread.java:1326)
        at org.apache.hadoop.hdfs.DataStreamer.closeResponder(DataStreamer.java:986)
        at org.apache.hadoop.hdfs.DataStreamer.endBlock(DataStreamer.java:640)
        at org.apache.hadoop.hdfs.DataStreamer.run(DataStreamer.java:810)
21/07/26 18:52:30 WARN DataStreamer: Caught exception            (40 + 2) / 200]
```