Short Video Sharing Platform Big Data Analytics

**Business Challenge / Requirement**

A leading short video company is facing challenges in enhancing its revenue and understanding the users so it wants to take help of Big Data Ecosystem to analyze large amounts of data received from varieties of sources, namely through mobile app and website. This analysis will help them to track the behavior of users so that to customize offers for them to buy paid subscription and also calculate royalties to the video creators do that to make them create more videos, this in turn will enhance their revenues.

**The goal of the Project**

The goal the project is to create data pipelines for the short video company which will make the company make appropriate business strategies to enhance their revenue by analyzing users and video data to send offers and royalties to users respectively.

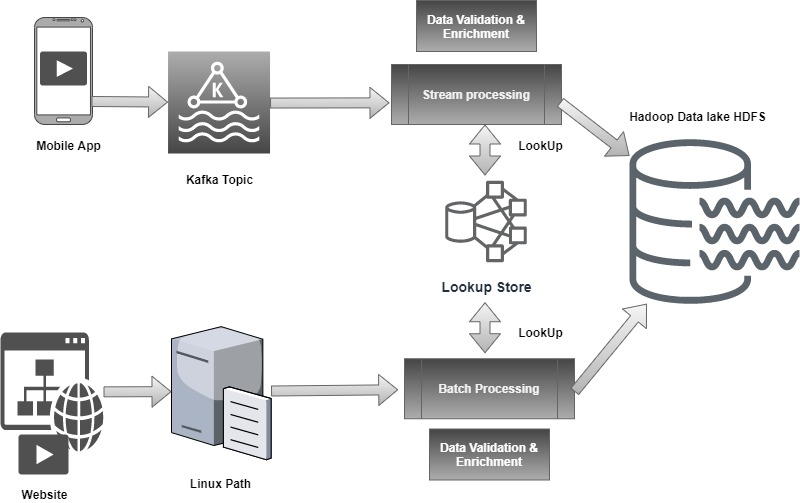
**Use – case**

Primary use case is user behavior analysis and video views data analysis for an on-demand video organization.

**Data Flow Architecture / Process Flow**

1. A Linux file server receives data files in form of xml and csv periodically after every 2 hours. These two files are coming from the website based on user interaction methodology.
2. There is a real time stream following as well from the mobile devices from the mobile app logs in form of json going to kafka topic.
3. The file data and stream data is validated, enriched and processed before loading into HDFS. There is a lookup table in this processing logic as well.
4. Finally data landed to HDFS needs to be analyzed by some analytical queries.

A schematic flow of operations with the best suited components is shown below



1. A Linux file server receives data files periodically after every 2 hours. (For mocking up this step we shall use data generator)
2. There is a real time stream following as well from the mobile devices from the mobile app logs.- KAFKA is most suited for this.
3. The file data and stream data is validated, enriched and processed before loading into HDFS. There is a lookup table in this processing logic as well.- SPARK Streaming is the most suited with lookups on Hbase. Spark streaming because we can use the Spark framework for both batch and real time queries that can make the solution operationally easier. HBASE is used for lookup due to the ability to fetch very fast by rowkey based lookups.
4. Finally data landed to HDFS needs to be analyzed by some analytical queries.-SPARK- Analytical queries can be done on Spark due to better performance and ability to control the jobs.

**Dataset Explanation & Schema**

1. Data coming from web applications reside in local directory (generated by data-generator) and has csv and xml format.

2. Data coming from mobile applications stream has comma delimited format.

3. Data present in lookup directory should be used in HBase.

**Fields present in the data files and streaming data-**

Data files contain below fields.

**Column Name/Field Name Column Description/Field Description**

* User\_id Unique identifier of every user
* Video\_id Unique identifier of every video
* Creator\_id Unique identifier of the lead creator of the video
* Timestamp Timestamp when the video was generated
* minutes\_played minutes between video started to play video was stopped
* Geo\_cd Can be 'A' for America region, 'AP' for Asia Pacific region, 'J' for Japan region, 'E' for europe and 'AU' for australia region
* Creator\_id Unique identifier of the Creator from where the video was played
* Video\_end\_type How the video was terminated.

0 means completed successfully

1 means video was skipped

2 means video was paused

3 means other type of failure like device issue, network error etc.

* Liked-

-true means video was liked (the like button was clicked)

-false means video was not liked (the like button was not clicked)

* Disliked

-true means video was disliked (the dislike button was clicked)

-false means video was not disliked (the dislike button was not clicked)

**LookUp Tables**

There are some existing look up tables present in NoSQL databases. They play an important role in

data enrichment and analysis.

**Table Name Description**

* Creator\_geo\_map (Creator-geocd.txt) Contains mapping of a geo\_cd with Creator\_id
* subscribed\_users(user-subscn.txt) Contains user\_id, subscription\_start\_date andsubscription\_end\_date. Contains details only for subscribed users
* video\_creator\_map (video-creator.txt) Contains mapping of video\_id with creator\_id
* users\_creator(user-creator.txt) Contains an array of creator\_id(s) followed by a user\_id

**Target Environment**

The target Hadoop environment is Cloudera Version of Hadoop distribution.

**Problem Statements / Tasks**

**Problem 1- Data Pre-processing & Enrichment (Spark-Hbase)**

* Parse and Infer schema of the given xml and csv formats data is ingested.
* You are expected to do general data cleaning steps like empty string replacements with actual NULL, data type checks (including date format) and corrections/ rejections, file name checks , empty file checks, malformed record checks and rejection etc.

Learners must apply below rules for data enrichment process:

* If any of like or dislike is NULL or absent, consider it as 0.
* If fields like Geo\_cd and creator\_id are NULL or absent, consult the lookup tables for fields Creator\_id and Video\_id respectively to get the values of Geo\_cd and creator \_id.
* If corresponding lookup entry is not found, consider that record to be invalid.

|  |  |  |
| --- | --- | --- |
| **NULL or absent field** | **Look up field** | **Look up table (Table from which record can be updated)** |
| Geo\_cd | Creator\_id | Creator\_Geo\_Map |
| Creator\_id | Video\_id | Video\_creator\_Map |

**Problem 2 - Data Analysis (Spark)**

Once we have made the data ready for analysis, we have to perform below analysis on a

batch basis.

1. Get the most popular Creators by the criteria of maximum number of videos played also liked by unique users.The output can be in form of a file with columns

creator\_id ,

total\_distinct\_videos\_played ,

distinct\_user\_count

2. Determine total duration of videos played by each type of user, where type of user can be

**'subscribed'** or **'unsubscribed'**. An unsubscribed user is the one whose video is either not

present in **Subscribed\_users** lookup table or has *subscription\_end\_date* earlier than the *timestamp* of the video played by him. The output can be in form of a file with columns

user\_type ,

duration

3. Determine list of connected creators. Connected creators are those whose videos are most

listened by the unique users who follow them.The output can be in form of a file with columns

creator\_id ,

user\_count

4. Determine which videos and creators are generating maximum revenue. Royalty applies to a

video only if it was *liked* or was *completed successfully* or both. The output can be in form of a file with columns

video\_id ,

duration

5. Determine who are the most unsubscribed users who listened to the videos for the longest duration.The output can be in form of a file with columns

user\_id ,

duration

Store the above analyzed results as a separate dataset in HDFS.

**Approach to Solve**

Below steps can be taken to start solving the project problem statements:

* Start by generating batch files in Gateway node location
* Start the Custom Stream data generator.
* Write code and run to take data from /tmp/files and solve Problem 1 in a Scala Class using Spark Batch and HBase API and place the processed data in HDFS directory /data/batch/cleaned/
* Write code and run to take data from /data/batch/cleaned/ and solve Problem 2 in a Scala Class using Spark Batch and HBase Native API
* Write code and run to take data from ‘data’ kafka topic and solve Problem 1 and 2 in a Scala Class using Spark Streaming and Hbase Native APi and finally place the data in HDFS directory /data/stream/

**Considerations / Assumptions**

* LookUp tables are in NoSQL databases. Integrate them with the actual data flow.
* Try to make joins as less expensive as possible.
* Data Cleaning, Validation, Enrichment, Analysis and Post Analysis have to be automated.
* Schedule the jobs using Oozie scheduler.
* Appropriate logs have to be maintained to track the behavior and overcome failures in the pipeline.

**Additional Info**

* Submitted code can run in any given Hadoop cluster of the same Hadoopversion**.**

**Deliverables**

Below are the expected deliverables-

* Code Jar and link to code repo
* Spark-Submit commands to run the code
* Any other script/wrapper required to run the code in any environment

**Business Benefits**

Based on the above analysis the video shortcompany will create a new business strategy to acquire more users, engagement, send offers/ promotions to users and also rewards it most popular video creators.