# Next Big Sound – Data Engineer Challenge

## Why I chose Spark and Hive:

Spark is the fastest distributed system available in today’s technology. This can use the existing Hadoop ecosystem and run as a standalone system. It can use HDFS which is the cheapest means to store the data in the distributed manner. For this use case I have used the Spark-SQL and hive. I chose Scala programming language, as it’s the fastest way of processing the data on the distributed system like spark compared to PySpark.

### Advantages of using Spark-SQL:

* We can execute SQL queries.
* We can interact with Hive.
* Performance is high compared to Hadoop as well as any relational DB.
* We can batch process the hive tables.
* I have ORC file format in this project which enables parallel processing, achieves high level of compression, concurrent read will use separate record readers.

### Advantages of using Hive:

* SQL like queries on large volume of data, by not costing the performance.
* It supports scalability and redundancy at low cost
* It enforces schema on read

## Thought process on working in Wikipedia projects:

In this data pipeline, I have used spark2.4 version. In the first stage I have created the dataframe (DF) and I have loaded the raw data into the DF. In the second stage I have preprocessed the data and loaded the preprocessed data into staging hive table. In the ODS layer I have joined the data with other lookup tables and the final data is loaded in the presentation layer table.

### Points considered:

* I did not force the schema while reading the raw data. As I wanted to have all the data and compare the row count for each stage.
* I have included the staging layer. Where I’m dealing only with the current data with insert overwrite. In this layer preprocessing is done.
* I have included the transformation layer (OSD layer). This layer is a consolidation layer which is created before the presentation layer which all the transformations is made including joins. I have joined the staging table with the ISO table which is master dimension table which validates language code and maps the codes to the language names. The invalid language codes and page titles are filtered out in this layer.
* The final table is in the presentation layer. The table is partitioned by year, month, day and hours as the individual files are hourly files. It’ll help us overwrite the file without having us to load the complete data. Also, here we are loading the data on batches to the historical data.
* As the final table is partitioned the query performance is high as its on individual partition of the data.
* I have designed the code and executed in my local machine. But we can execute this code on cluster mode which can be scaled depending on the volume of the data.
* I have ORC file format which achieved the compression and provided parallelism which improves the performance.
* The final table which has all the historical data is partitioned on ymd and hours. Which again gives better performance for the adhoc queries on the data we loaded.

### Final Queries:

1. Gives the top 10 most viewed pages for each language for the during the first hour of 2012.
2. Give the percentage of views for each project category during the first hour of 2012.

Please go thru the read.me file to see the instructions to run the code.