**Aadhaar Data Analysis using Hadoop/Spark**

**Background**

Aadhaar card is a 12-digit unique identification number provided to each Indian resident. Till date 1.19 billion members are enrolled under Aadhaar as of 30 Nov 2017. To analyze and understand the activity occurring on such a massive scale, a relational SQL database is not enough. Such kind of data is well suited to a massively parallel and distributed system like Hadoop.

In this project we will analyze the aadhaar card data set using Map Reduce/Hive and Spark technologies against different research queries for example total number of aadhaar cards approved by state, total number of aadhaar card applicants by gender, total number of aadhaar card applicants by age type, etc.

**Technologies to use**

* Hadoop /HDFS
* Hive
* Spark
* Amazon - EMR
* FrontEnd Technologies

**Implementation**

Aadhaar Dataset (csv file) is obtained from UIDAI public website and has following columns:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Registrar | Enrollment Agency | State | District | Sub District | Pin Code | Gender | Age | Aadhaar Generated | Enrollment Rejected | Residents providing email | Residents providing mobile no |

This dataset can be fed into various Hadoop Technologies (HDFS or Amazon EMR) and meaningful results can be extracted and analyzed, which can serve as decision support system.

We have few choices to do the analysis:

1. **MapReduce**

In this case, we would have to think from map and reduce point of view. Programmatically, this effort will become quite challenging as lot of custom code is required to successfully execute the business logic even for simplest tasks.

1. **Hive**

Hive is compatible and works with traditional data integration and data analytics tool. We would be able to leverage the SQL capabilities of Hive QL as well as data can be managed in a particular schema. Also, by using Hive, the development time can be significantly reduced.

1. **Spark**

**Spark** is another great choice where along with volume we would also be able to give better performance as all the data is managed within memory.

We would use **Hive** and **Spark** in this project and prove which one is preferable for faster and accurate performance. MapReduce may also be used in one of the queries just get a feel of how MapReduce job is developed and run on a cluster.

Finally, after the implementation is over, the project will be deployed on cloud, Amazon EMR where we’ll leverage readymade cluster creation capabilities of EMR.

Benefits using cloud are:

* Pay as you go
* No hardware cost involved
* Security/firewall and maintenance managed by Amazon EMR

The output generated will be represented using a frontend technology.

**DataFlow**

This dataflow is for MapReduce Vs Hive. Same can be drawn for Spark as well.

Aadhaar Data download from UIDAI website

Data Loaded in HDFS

HIVE Database

Processing data using HIVE query language

Output Results

Create Java Project for Map Reduce analysis

Processing data using Mapper and Reducer

Output Results

Compare Both Results

**Sample reports**

* Count the number of Identities generated in each state
* Count the number of Identities generated by each Enrollment Agency
* Top 10 districts with maximum identities generated for both Male and Female
* Search for the total number of Identities rejected by states
* Find out the total number of Aadhaar applicants by age type
* Find out the total number of Aadhaar applicants state and age group wise who have provided email id and/or mobile number