# Problem Statement

To re-implement the Hyperloglog implementation from C / binary compatible SPARK (Java/Scala) to Python, to enable easier pathways, new capabilities and design options unavailable to us today in current implementation.

About Hyper Log log : HyperLogLog - is popular algorithm for count-distinct problem at scale. Hyperloglog is a cardinality (distinct count) estimation algorithm that is comprised of a specialized data type (the estimator) and specialized functions to work with these estimators. The estimators have a unique value that estimators are additive allowing for cardinality estimators to be aggregated across time and dimensionality with no loss in accuracy .

The basis for the algorithm is that we can use hashes to "randomize" our incoming values (really more spread them out) and once this is done, the occurrence of multiple rare events (long runs of leading zeros) becomes more likely as the size of our set increases. If we do this multiple different times and then take the harmonic mean of these estimates, we can get a very accurate estimate.

https://en.wikipedia.org/wiki/HyperLogLog

**Use case with Epsilon :** Traditional finding of Distinct users/companies often use more space & time complexity. The problem with cardinality is that in order to calculate the cardinality of a given dataset you need to go back to base data in order to determine the cardinality at numerous different levels. E.g**. if you want to know the number of users visiting a particular site for a 2 given days (separately) and then want to find the number of users who visited that same site over the 2 day span you need to reanalyze the base data that still has the user id in it.** In our case this is **millions or billions of rows of data**.

With hyperloglog I can create an estimator by day by site of the number of users and if I want to know the number of users who visited the site over the 2 day span I can just sum the estimators together to get an estimate of the number of users. Due to the way the estimators work, there is no loss in accuracy during this operation. This allows for a great deal of flexibility when generating cardinality estimates, and massive performance gains for the database platform as it no-longer needs to re-analyze base data in order to create the higher level estimate.

**Objective** - Port the current postgres HLL implementation to Python implementation such that counters created in postgres can be serialized as text and sent to Python and vice versa

## Project Tasks

### 1. Literature Review & Research

- Study existing Hyperlog log  algorithm and how it works .

### 2. Define Project Scope & Requirements

- Currently our team is migrating to cloud - Databricks & AWS
- Current hyper log log algorithm is having old version of Spark, python libraries
- As per current priority migration of Python is taken highest priority

### 3. Testing & Evaluation

- No Compilation errors.
- Should be able to execute .
- Analyze performance based on simulated results.

### 8. Documentation & Reporting

- Document any issues ,lessons to be learned.

References : Whitepapers referenced in above github link

http://algo.inria.fr/flajolet/Publications/FlFuGaMe07.pdf
https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/40671.pdf

Other informative resources on this topic

Hyperloglog explained - https://speakerdeck.com/citusdata/what-is-hyperloglog-and-why-you-will-love-it-postgresql-conference-eu-2018-burak-yucesoy

Yahoo solution to same problem -

https://datasketches.apache.org/
https://yahooeng.tumblr.com/post/135390948446/data-sketches
https://github.com/apache/datasketches - open sourced from Yahoo.