# Reasoning on collecting 500M Repositories instead of 100K.

Doing things at a large scale presents a lot of challenges. This document will outline a few things we would need to take into consideration when handling large-scale data collection from 100,000 repositories to 500 million.

1. At the current scale, we can let the program run sequentially with just one data base. It still takes some time to get the 100,000 repositories’ data but doing it for 500 million repos would create new problems:
   1. Github API rate limits a thousand results per query
   2. One single process (crawler) will have trouble handling all this data efficiently
   3. Database performance would take a hit
   4. Recrawling for the whole database regularly would be an issue and we would need to find some way to do incremental updates to avoid full recrawls
2. Doing this using multiple workers in parallel and having them communicate via a message queue like RabbitMQ or Redis Stream
3. Current Github Token has a result limit per query. Using an upgraded token or using multiple tokens would help.
4. Instead of updating all repositories, we can fetch another parameter that tells us when the repository was last updated. If that was before our last crawl, we ignore that repository. Hence, limiting the size of incremental updates.
5. Normalization in DB if we were to include more information for repositories like PRs and comments, etc.
6. Add monitoring and debugging logs, with batch retry mechanism for failed queries.