# Is phoenix sutible for ETL and aggregation?

参考：https://stackoverflow.com/questions/52528907/phoenix-and-hive-etl-on-hbase

## Question

Is phoenix sutible for ETL and aggregation?

I am trying to do some ETL on my data. Right now I am using hbase to store my data(footprint on our website). I need to do some aggregations on this data, such as page view on each url ...etc.

From my research I know hive can be used for ETL on hbase data, plus hql provides sql language so that we don't need to write map-reduce codes ourselves. But when I try to query on hbase using hive(pyhive), it takes a long time to finish. Furthermore, if I have phoenix to do sql on hbase, do I still need hive on my hbase.

Right now when I try to use some complicated sql, phoenix will timeout. And hive is very very slow. So what is a good way to do ETL on hbase data?

## Answer

HBase is a suitable choice if you have OLTP type of queries, that is your querying pattern is limited to point gets or small range queries.

Phoenix is just a SQL layer on top of HBase which provides the user capability to query data in HBase using SQL constructs.

When you submit any SQL query, Phoenix will parse it and create an execution plan which can break the query as full scan, range scan or point get depending on what you queried for.

For processing results Phoenix uses Predicate Push down, so HBase performs all the processing (scanning and getting the requisite rows from all regionservers), Phoenix aggregates/collates the resultant rows and return them to the user.

Now to answer your question "Is phoenix suitable for ETL and aggregation?" - NO its not suitable if you need to perform large scans (full or large range scans) with aggregations and expect the results to be processed in seconds or sub-seconds. These queries may work fine if your table footprint is small (few hundred GBs), but as the table size would increase over time you would end up into serious performance issues. This is OLAP scenario and you should look for other alternatives.

I am assuming that when you say using Hive with HBase you plan to create a Hive External Table on top of HBase table that can query the HFiles directly using the HiveHBaseTableInputFormat. This will use MapReduce to process the query and you cannot really leverage performance optimizations through partitioning, bucketing etc.

I would suggest to consider using HBase + Phoenix option for point gets and small range queries, for large aggregation/ETL type queries consider using Spark, it will be better and much faster then the Hive option.

In case your data is append only you can consider using some alternate OLAP supporting storage options like Driud, Clickhouse or Hive on Tez, Hive with Impala or Presto options instead of HBase.

Updated as per comment -

Spark will be faster then Hive (MapReduce) option; Use Spark only for queries using large range or full scan followed by aggregations.

Although Phoenix leverages co-processors and predicate push down to HBase regions, the amount of available resources may become a limitation, Phoenix will parallelize the query in chunks which are decided by GUIDE\_POSTS\_WIDTH, for large tables this can easily range from few 100s to 1000s and create a contention for RPC queues, also during scans HFiles will be read and loaded in regionserver heap, you may run into GC or OOM issues if scan cannot fit results in allocated heap. Scanners may get limited by rpc wait time and available memory resulting in timeouts.

You can tune above situations to an extent but HBase/Phoenix are not meant for OLAP. Using Spark you can directly read HFiles and apply filters thereafter, this will eliminate timeout issues. If same subset of data is queried again and again you can try external rdd/dataframe caching to Ignite.

I would like to add that before looking for an alternate you should analyze how you are storing your data in HBase is it aligned to your query pattern? Revisit your rowkey design if your query plan is creating Full Scans, try to redesign rowkey or query to avoid full scans? Check if your data is well distributed across all regions? Could your high write rate be affecting your read performance?

In case you have clickstream type of data that you need to analyze, do check out Clickhouse, Yandex developed and open sourced it for similar usecase but it does work well for other OLAP usecases too.