Core Workload in YCSB

We have 6 core different workloads included in the YCSB. We can also defined customized new workload to calculate the benchmark of the serving system.

Workload A: Update heavy workload – This performs mix of 50/50 reads and writes.

Workload B: Read mostly workload – This performs 95/5 reads and writes mix.

Workload C: Read Only – This perform only read.

Workload D: Read latest workload – New records inserted and most recently inserted records are the most popular record.

Workload E: Short ranges – This helps to queries the short range instead of single record.

Workload F: Read-modify-write – The client will read a record and modify the record. Write back the changes

Reference:

https://github.com/brianfrankcooper/YCSB/wiki/Core-Workloads

Implementing New or Customized workloads

In YCSB, workload defines both a Data set and Transaction set. Data set – set of records loaded into the database. Transaction set – set of read and writes performed against the database. Core workload provided by YCSB performs both the data and transaction set. We have two options in implementing new workloads.

1. ***New parameter file***

YCSB core workload is defined by a set of parameter file. We can create our own parameter file with new values for the specific parameters. More details about parameters

2. ***New java class***

There are few steps to create a new workload using the java class.

a. *Extend com.yahoo.YCSB.workload*

Extend the base class com.yahoo.YCSB.workload to create a new workload. Extended class must have public no argument constructor.

b. *write a code to initialize the workload class*

we have two options to initialize the workload using init() or initThread() methods.

init() - is used to call all the workload instances also to initialize any object shared by all threads.

initThread() - called once per workload instance in the context of the worker threads and to initialize any object specific to a single workload instance and single worker thread.

c. *write a cleanup()*

This method is called after the workload is completed and called once for all the workload instances.

d. *define the records to inserted*

Implementing doInsert() method helps to create and insert a single record.

e. *define the transactions*

doTransaction() method is used to execute every transaction. This method helps to execute single transaction using the DB objects passed to access the database. Implementation of this method can have different type of transaction and can call database interface layer multiple times. The invocation of the method should be logical transaction. Also using this method, we can perform insert operations but initially the datasets will be loaded by doInsert() and additional records will loaded by doTransaction().

f. measure latency

The YCSB client measures the latency and throughput of the database operations even for the customized workload. For more complex transaction, we can implement the measures and measurements. Measurements are available in Measurement.measure(). There is single instance Measurements.getMeasurements(). We can also add string tag to label the results.

g. *Use it with the YCSB client*

Implemented classes and necessary libraries are available in the classpath. Run the YCSB client, specify the workload property and provide fully qualified class name of the database.

Reference:

https://github.com/brianfrankcooper/YCSB/wiki/Implementing-New-Workloads