Perform Incremental Map-Reduce

## Data Setup

The usersessions collection contains documents that log users' sessions each day, for example:

|  |
| --- |
| db.usersessions.insertMany([ |
| { userid: **"a"**, start: ISODate(**'2020-03-03 14:17:00'**), length: 95 }, |
| { userid: **"b"**, start: ISODate(**'2020-03-03 14:23:00'**), length: 110 }, |
| { userid: **"c"**, start: ISODate(**'2020-03-03 15:02:00'**), length: 120 }, |
| { userid: **"d"**, start: ISODate(**'2020-03-03 16:45:00'**), length: 45 }, |
| { userid: **"a"**, start: ISODate(**'2020-03-04 11:05:00'**), length: 105 }, |
| { userid: **"b"**, start: ISODate(**'2020-03-04 13:14:00'**), length: 120 }, |
| { userid: **"c"**, start: ISODate(**'2020-03-04 17:00:00'**), length: 130 }, |
| { userid: **"d"**, start: ISODate(**'2020-03-04 15:37:00'**), length: 65 } |
| ]) |

## Initial Map-Reduce of Current Collection

Run the first map-reduce operation as follows:

1. Define the map function that maps the userid to an object that contains the fields total\_time, count, and avg\_time:

|  |
| --- |
| var mapFunction = function() { |
| var key = this.userid; |
| var value = { total\_time: this.length, count: 1, avg\_time: 0 }; |
|  |
| emit( key, value ); |
| }; |

1. Define the corresponding reduce function with two arguments key and values to calculate the total time and the count. The key corresponds to the userid, and the values is an array whose elements corresponds to the individual objects mapped to the userid in the mapFunction.

|  |
| --- |
| var reduceFunction = function(key, values) { |
|  |
| var reducedObject = { total\_time: 0, count:0, avg\_time:0 }; |
|  |
| values.forEach(function(value) { |
| reducedObject.total\_time += value.total\_time; |
| reducedObject.count += value.count; |
| }); |
|  |
| return reducedObject; |
| }; |

1. Define the finalize function with two arguments key and reducedValue. The function modifies the reducedValue document to add another field average and returns the modified document.

|  |
| --- |
| var finalizeFunction = function(key, reducedValue) { |
|  |
| if (reducedValue.count > 0) |
| reducedValue.avg\_time = reducedValue.total\_time / reducedValue.count; |
|  |
| return reducedValue; |
| }; |

1. Perform map-reduce on the usersessions collection using the mapFunction, the reduceFunction, and the finalizeFunction functions. Output the results to a collection session\_stats. If the session\_stats collection already exists, the operation will replace the contents:

|  |
| --- |
| db.usersessions.mapReduce( |
| mapFunction, |
| reduceFunction, |
| { |
| out: **"session\_stats"**, |
| finalize: finalizeFunction |
| } |
| ) |

1. Query the session\_stats collection to verify the results:

|  |
| --- |
| db.session\_stats.find().sort( { \_id: 1 } ) |