Advance map reduce on MovieLens dataset

1] MapReduce to find the top 25 rated movies in the movieLens dataset

Map Function:

```
function()
 {  var value = {count:1};
    var key = this.movieid;  emit(key, value); }
```

Reduce Function:

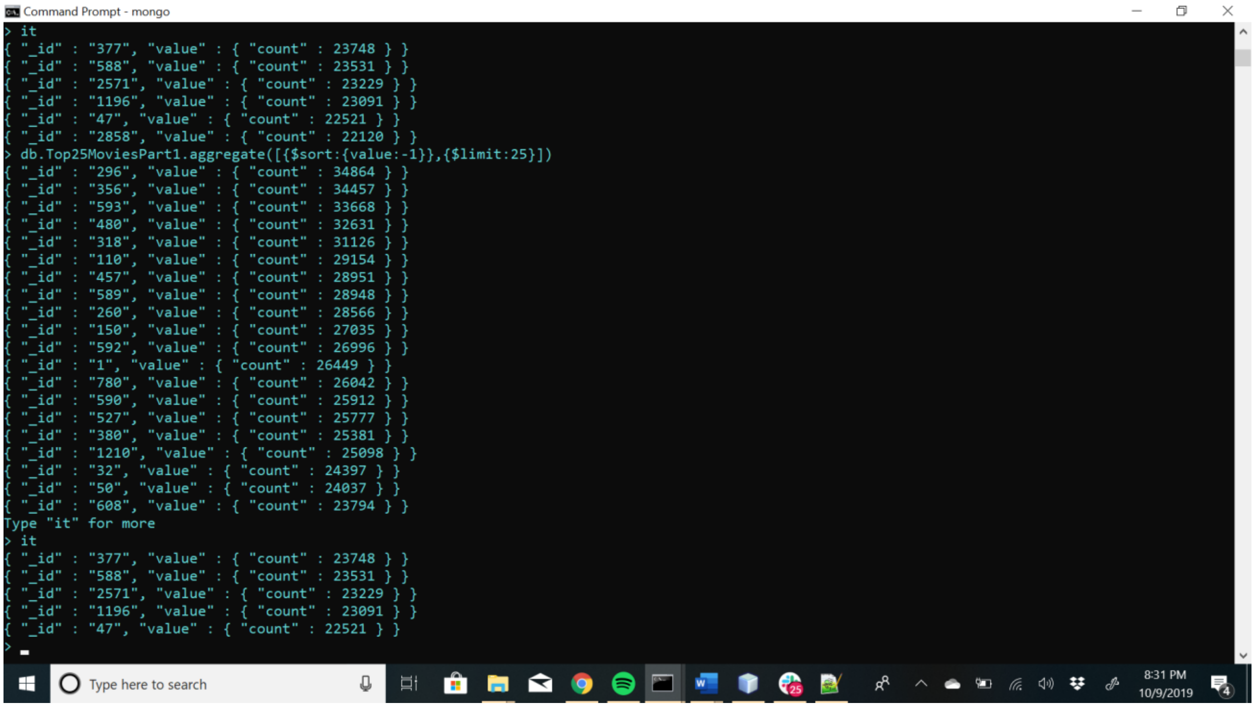
```
function(key, values)
{
                counterVal = { count:0};
                for (var id = 0; id < values.length; id++) {
                counterVal.count  += values[id].count ;
                 }
   return counterVal; }
```

Map reduce and querying:

```
db.ratings.mapReduce(mapFunction1,reduceFunction1,{out:{merge:"Top25RatedMovies"}});
```

```
db.Top25MoviesPart1.aggregate([{$sort:{value:-1}},{$limit:25}])
```

Output:



```
> it
{ "_id" : "377", "value" : { "count" : 23748 } }
{ "_id" : "588", "value" : { "count" : 23531 } }
{ "_id" : "2571", "value" : { "count" : 23229 } }
{ "_id" : "1196", "value" : { "count" : 23091 } }
{ "_id" : "47", "value" : { "count" : 22521 } }
{ "_id" : "2858", "value" : { "count" : 22120 } }
> db.Top25MoviesPart1.aggregate([{$sort:{value:-1}},{$limit:25}])
{ "_id" : "296", "value" : { "count" : 34864 } }
{ "_id" : "356", "value" : { "count" : 34457 } }
{ "_id" : "593", "value" : { "count" : 33668 } }
{ "_id" : "480", "value" : { "count" : 32631 } }
{ "_id" : "318", "value" : { "count" : 31126 } }
{ "_id" : "110", "value" : { "count" : 29154 } }
{ "_id" : "457", "value" : { "count" : 28951 } }
{ "_id" : "589", "value" : { "count" : 28948 } }
{ "_id" : "260", "value" : { "count" : 28566 } }
{ "_id" : "150", "value" : { "count" : 27035 } }
{ "_id" : "592", "value" : { "count" : 26996 } }
{ "_id" : "1", "value" : { "count" : 26449 } }
{ "_id" : "780", "value" : { "count" : 26042 } }
{ "_id" : "590", "value" : { "count" : 25912 } }
{ "_id" : "527", "value" : { "count" : 25777 } }
{ "_id" : "380", "value" : { "count" : 25381 } }
{ "_id" : "1210", "value" : { "count" : 25098 } }
{ "_id" : "32", "value" : { "count" : 24397 } }
{ "_id" : "50", "value" : { "count" : 24037 } }
{ "_id" : "608", "value" : { "count" : 23794 } }
Type "it" for more
> it
{ "_id" : "377", "value" : { "count" : 23748 } }
{ "_id" : "588", "value" : { "count" : 23531 } }
{ "_id" : "2571", "value" : { "count" : 23229 } }
{ "_id" : "1196", "value" : { "count" : 23091 } }
{ "_id" : "47", "value" : { "count" : 22521 } }
> _
```

2] MapReduce to find the number of males and females in the movielens dataset. I have use Movielens 1M dataset to solve this task

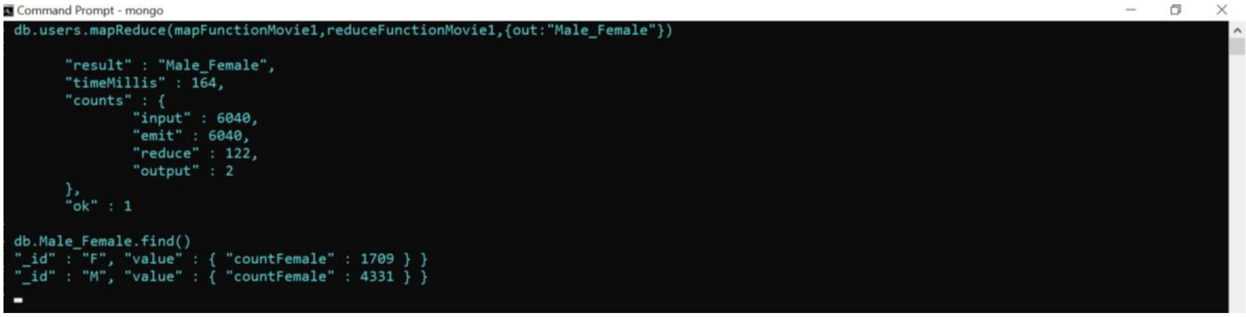
Map function:

```
function()
 {  var value = { countMale:1,countFemale:1};
    var key = this.Gender;
    emit(key, value);
 }
```

Reduce Function:

```
function(key, values)
 {    counterVal1 = { countMale:0};
      counterVal2 = { countFemale:0};
      for (var id = 0; id < values.length; id++)
              {
               if(values[id]=='M')
               { counterVal1.countMale  += values[id].countMale; }
               else
                 { counterVal2.countFemale+=values[id].countFemale; }
} return counterVal1,counterVal2; }
```

OUTPUT



```
db.users.mapReduce(mapFunctionMovie1,reduceFunctionMovie1,{out:"Male_Female"})

       "result" : "Male_Female",
       "timeMillis" : 164,
       "counts" : {
               "input" : 6040,
               "emit" : 6040,
               "reduce" : 122,
               "output" : 2
       },
       "ok" : 1

db.Male_Female.find()
"_id" : "F", "value" : { "countFemale" : 1709 } }
"_id" : "M", "value" : { "countFemale" : 4331 } }
```

3] MapReduce to find the number of movies rated by different users. I have used MovieLens 10M dataset for this.

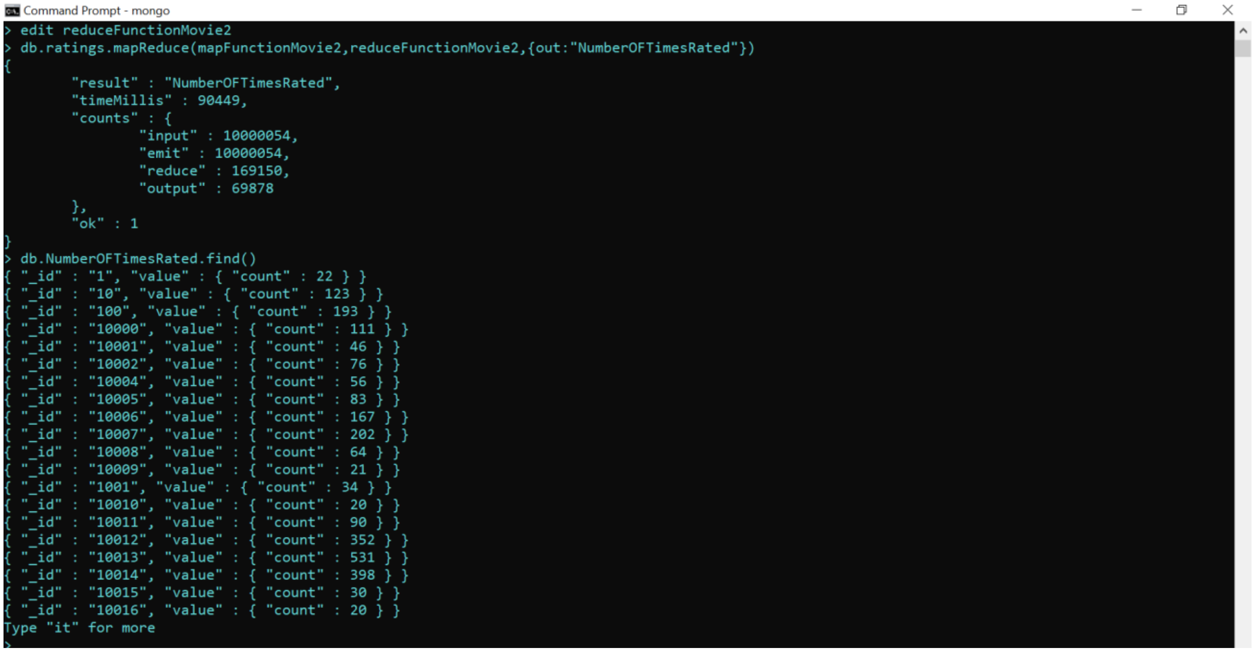
Map Function

```
function()
 {  var value = {count:1};
    var key = this.userid;
    emit(key, value);
 }
```

Reduce Function

```
function(key, values)
{
                counterVal = { count:0};
                for (var id = 0; id < values.length; id++) {
                counterVal.count  += values[id].count ;
                 }
   return counterVal; }
```

OUTPUT



```
> edit reduceFunctionMovie2
> db.ratings.mapReduce(mapFunctionMovie2,reduceFunctionMovie2,{out:"NumberOFTimesRated"})
{
        "result" : "NumberOFTimesRated",
        "timeMillis" : 90449,
        "counts" : {
                "input" : 10000054,
                "emit" : 10000054,
                "reduce" : 169150,
                "output" : 69878
        },
        "ok" : 1
}
> db.NumberOFTimesRated.find()
{ "_id" : "1", "value" : { "count" : 22 } }
{ "_id" : "10", "value" : { "count" : 123 } }
{ "_id" : "100", "value" : { "count" : 193 } }
{ "_id" : "10000", "value" : { "count" : 111 } }
{ "_id" : "10001", "value" : { "count" : 46 } }
{ "_id" : "10002", "value" : { "count" : 76 } }
{ "_id" : "10004", "value" : { "count" : 56 } }
{ "_id" : "10005", "value" : { "count" : 83 } }
{ "_id" : "10006", "value" : { "count" : 167 } }
{ "_id" : "10007", "value" : { "count" : 202 } }
{ "_id" : "10008", "value" : { "count" : 64 } }
{ "_id" : "10009", "value" : { "count" : 21 } }
{ "_id" : "1001", "value" : { "count" : 34 } }
{ "_id" : "10010", "value" : { "count" : 20 } }
{ "_id" : "10011", "value" : { "count" : 90 } }
{ "_id" : "10012", "value" : { "count" : 352 } }
{ "_id" : "10013", "value" : { "count" : 531 } }
{ "_id" : "10014", "value" : { "count" : 398 } }
{ "_id" : "10015", "value" : { "count" : 30 } }
{ "_id" : "10016", "value" : { "count" : 20 } }
Type "it" for more
>
```