CS6650 Assignment-2 Tan Wang

Git Repository: https://github.com/TanWang-0914/NEU6650-HW
**(Branch: assignment-2)**

## Server Side Design

## Major Classes:

**DynamoDbSource** - class that build dynamoDB client and connect to dynamoDB.This contains following method:

getDataSource(): this method will return the connection to our dynamoDB.

**PurchaseDao** - this is a data access object class, it creates table with given name and contains following method:

createPurchase(String purchaseID, String storeID, String custID, String date, String purchaseBody):
This method will create a records in our purchases Table with given parameters and purchase body.

**PurchaseServlet** - this is our servlet class which are responsible to handle GET and POST, class contains following method:

doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException :
This is a method to handle get request, it will check for validation of request url and parameters. If request url is valid, it will return purchase information in a html type with status code 200. Otherwise, return 404 0r 500 based on error type.

boolean isUrlValid(String[] urlPath):
This is a helper method on validating request url, return true if given url is valid, otherwise false.

boolean allDigits(String s):
This is a helper method on checking if a string contains a digit characters, this method will be called by isUrlValid(),  return true if given string contains only digit characters, otherwise false.

void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException:
This is a method to handle post request, it will check for validation of request url and parameters. If request url is valid, it will first try to write purchase record to database, if writing succeed, it will return purchase information in a html type with status code 201,

if writing to database failed, it will return 200 instead. Otherwise, return 404 0r 500 based on error type.

## Packages:

javax.servlet - javax.servlet-api
This package contains libraries on building our http servlet.

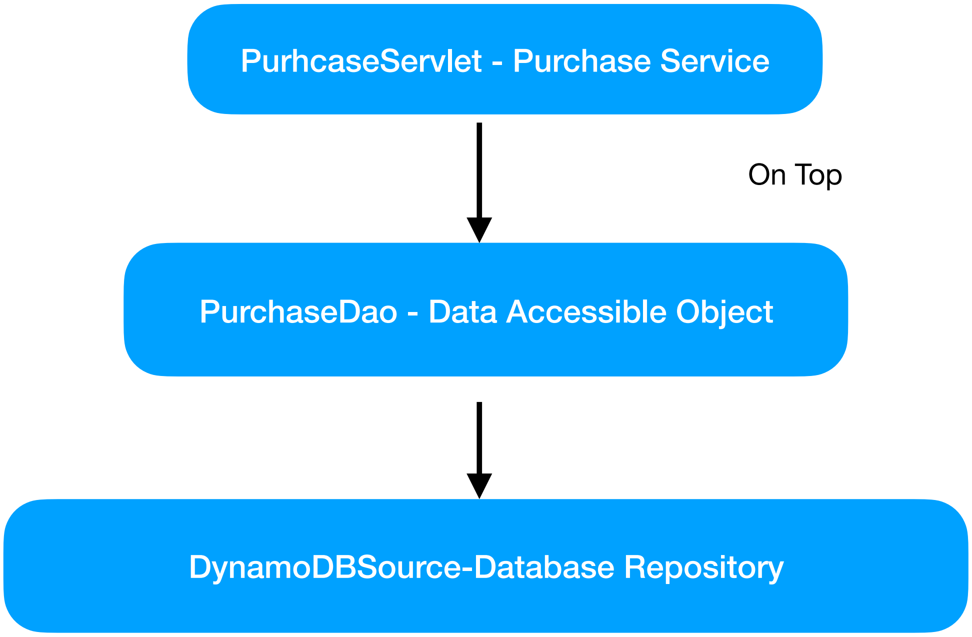
com.amazonaws - aws-java-sdk-dynamodb
This package contains libraries on create dynamoDB, building connection and table.

## Relationship:

DynamoDBSource serve as a repository to our database, it provide connection to DynamoDB.

PurchaseDao serve as a data access class, it provide method on creating item in database.

PurchaseServlet serve as a controller to handle request from client. It will validate the request and perform different task based on request and return response.

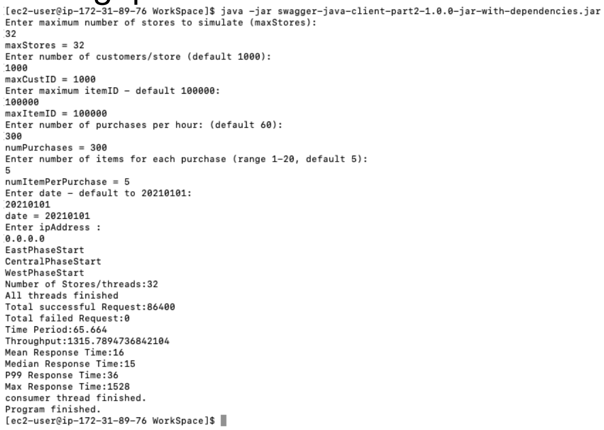

PurhcaseServlet - Purchase Service
On Top
PurchaseDao - Data Accessible Object
DynamoDBSource-Database Repository

## Single Server Test Result

(mean/median/p99/max/throughput are captured in output window)

### 32 thread

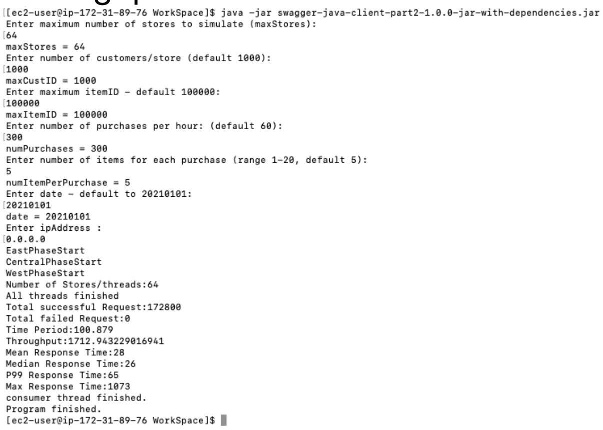
Total Successful request: 86400
Wall Time (sec): 65.66
Mean ResponseTime(mills): 16
Throughput: 1528



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
32
maxStores = 32
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:32
All threads finished
Total successful Request:86400
Total failed Request:0
Time Period:65.664
Throughput:1315.7894736842104
Mean Response Time:16
Median Response Time:15
P99 Response Time:36
Max Response Time:1528
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ █
```

### 64 thread

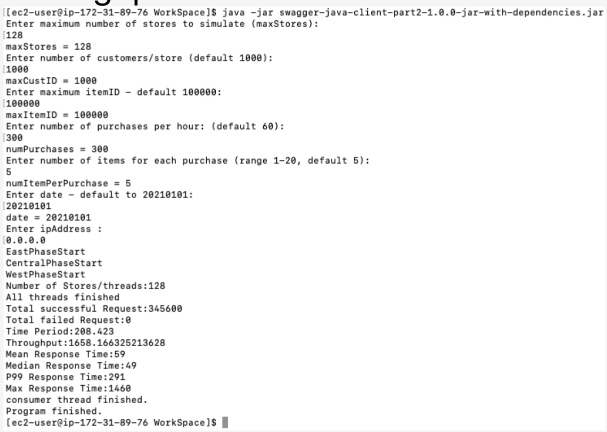
Total Successful request: 172800
Wall Time (sec):100.88
Mean ResponseTime(mills): 28
Throughput: 1712



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
64
maxStores = 64
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:64
All threads finished
Total successful Request:172800
Total failed Request:0
Time Period:100.879
Throughput:1712.943229016941
Mean Response Time:28
Median Response Time:26
P99 Response Time:65
Max Response Time:1073
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ █
```

### 128 thread

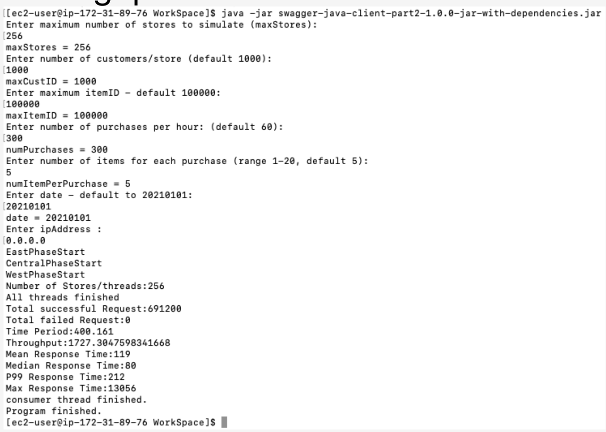
Total Successful request: 345600
Wall Time (sec): 208.42
Mean ResponseTime(mills): 59
Throughput: 1658



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
128
maxStores = 128
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:128
All threads finished
Total successful Request:345600
Total failed Request:0
Time Period:208.423
Throughput:1658.166325213628
Mean Response Time:59
Median Response Time:49
P99 Response Time:291
Max Response Time:1460
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ █
```

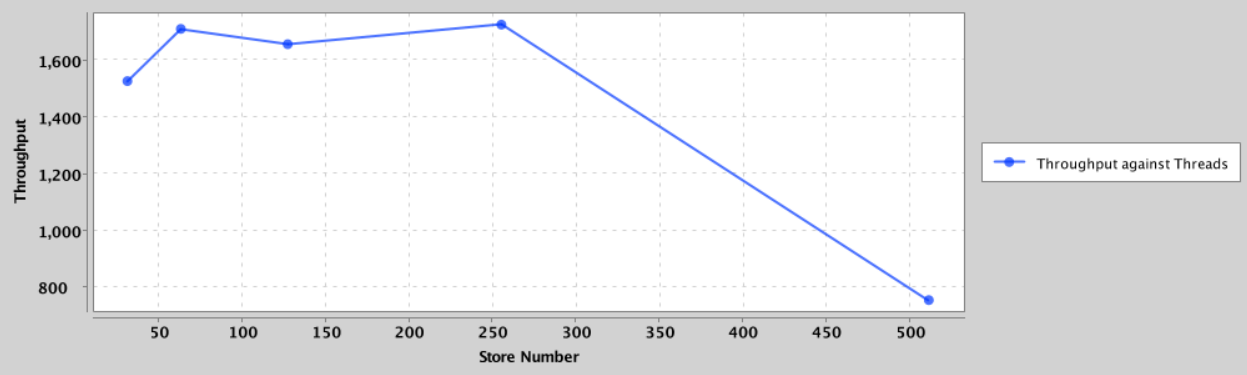
### 256 thread

Total Successful request: 691200
Wall Time (sec): 400.16
Mean ResponseTime(mills): 119
Throughput: 1727



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
256
maxStores = 256
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:256
All threads finished
Total successful Request:691200
Total failed Request:0
Time Period:400.161
Throughput:1727.3047598341668
Mean Response Time:119
Median Response Time:80
P99 Response Time:212
Max Response Time:13056
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ █
```

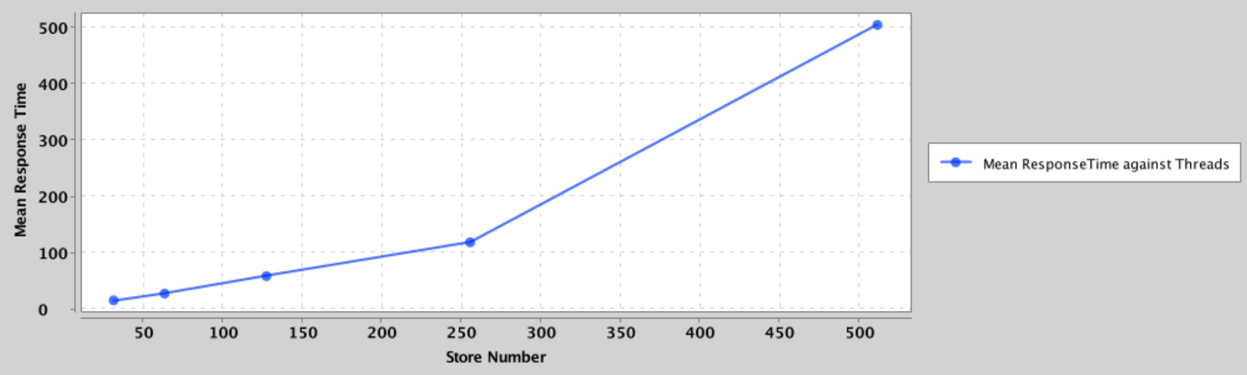
## Throughput against Thread Number (Single Server)



Throughput will increase at first with thread increasing, and then reached maximum of 1700 for single server at 256 store. And then with thread number increasing, throughput on the server will drop down, which might be caused by throttling of database after the burst of high usage.

See detailed on dynamoDB throttling:
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.ReadWriteCapacityMode.html

## Mean Response Time against Thread Number (Single Server)



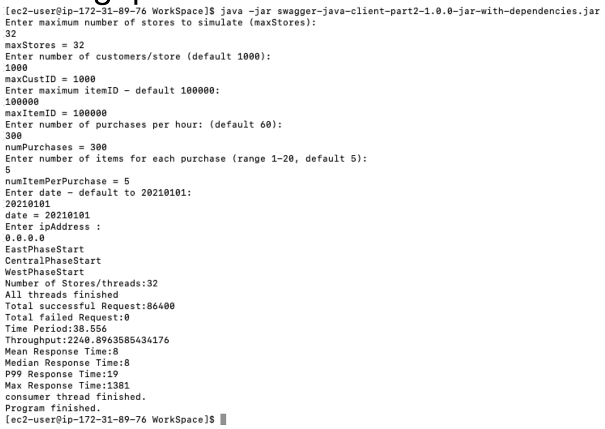
Mean response time will first increase at a steady rate for the server has basically same throughput when thread number is low. When database is subjected to throttling and throughput on the server drop down, mean response time start to increase at a much higher rate due to the long wait at server.

## Load Balanced Server Test Result

(mean/median/p99/max/throughput are captured in output window)

### 32 thread

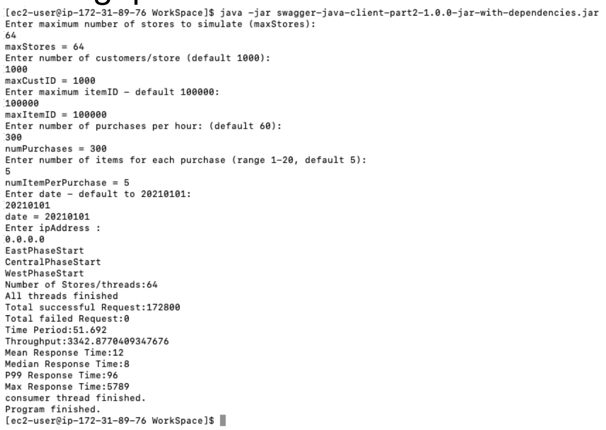
Total Successful request: 86400
Wall Time (sec): 38.55
Mean ResponseTime(mills): 8
Throughput: 2240



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
32
maxStores = 32
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:32
All threads finished
Total successful Request:86400
Total failed Request:0
Time Period:38.556
Throughput:2240.8963585434176
Mean Response Time:8
Median Response Time:8
P99 Response Time:19
Max Response Time:1381
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ 
```

### 64 thread

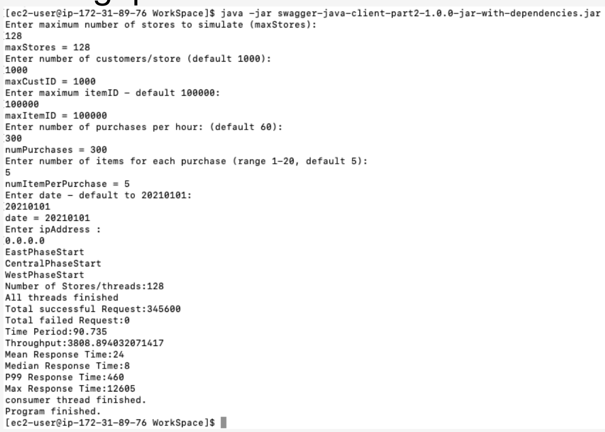
Total Successful request: 172800
Wall Time (sec):51.69
Mean ResponseTime(mills): 12
Throughput: 3342



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
64
maxStores = 64
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:64
All threads finished
Total successful Request:172800
Total failed Request:0
Time Period:51.692
Throughput:3342.8770409347676
Mean Response Time:12
Median Response Time:8
P99 Response Time:96
Max Response Time:5789
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ 
```

### 128 thread

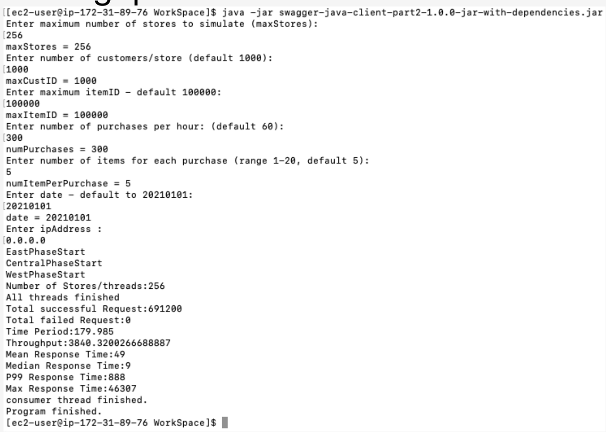
Total Successful request: 345600
Wall Time (sec): 90.73
Mean ResponseTime(mills): 24
Throughput: 3808



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
128
maxStores = 128
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:128
All threads finished
Total successful Request:345600
Total failed Request:0
Time Period:90.735
Throughput:3808.894032071417
Mean Response Time:24
Median Response Time:8
P99 Response Time:460
Max Response Time:12605
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ 
```

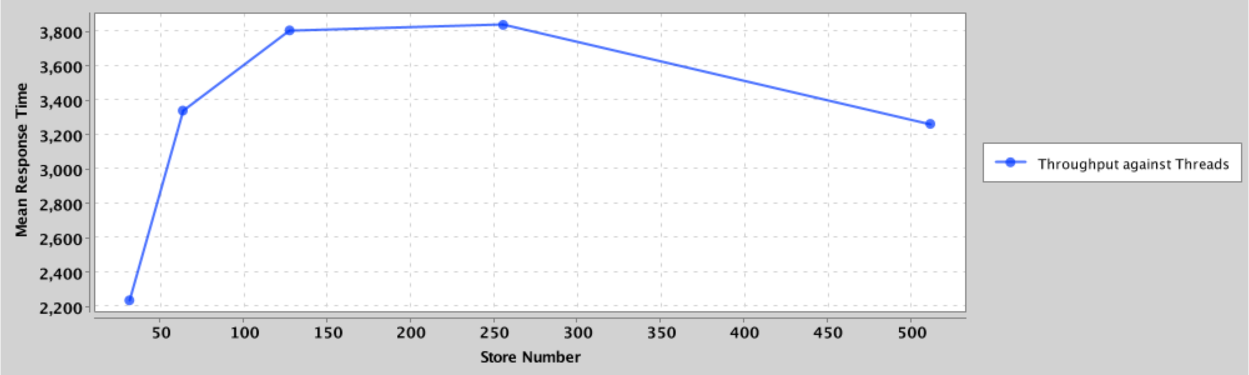
### 256 thread

Total Successful request: 691200
Wall Time (sec): 179.98
Mean ResponseTime(mills): 49
Throughput: 3840



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
256
maxStores = 256
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:256
All threads finished
Total successful Request:691200
Total failed Request:0
Time Period:179.985
Throughput:3840.3200266688887
Mean Response Time:49
Median Response Time:9
P99 Response Time:888
Max Response Time:46307
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$ 
```

## Throughput against Thread Number (Load Balancer Server)

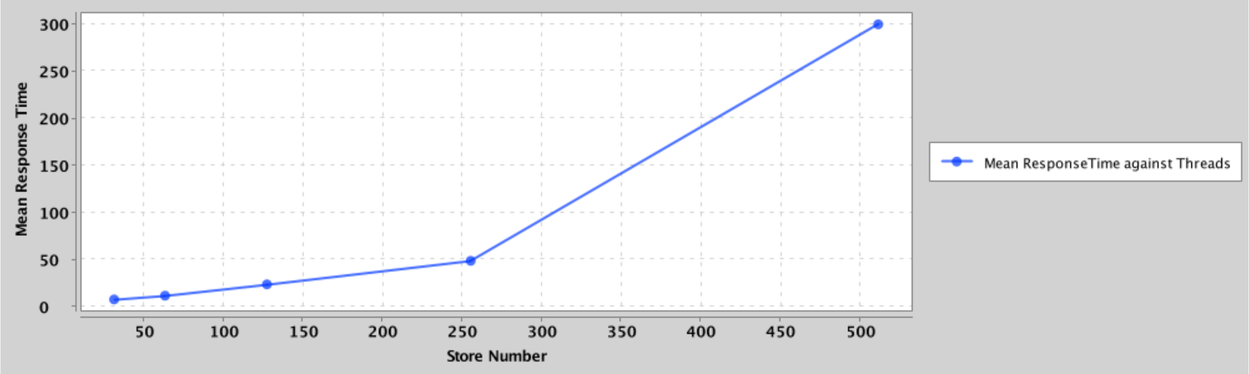


With load balancing, we can see an increase on the maximum throughput, which is possible from distributing request to four servers, and higher usage of database capacity.

Throughput will increase at first with thread increasing, and then reached maximum of 3800 at 256 store. Which is 3 times of single server. And then with thread number increasing, throughput on the server will drop down, which might be caused by throttling of database after the burst of high usage. But average throughput is still higher than single server.

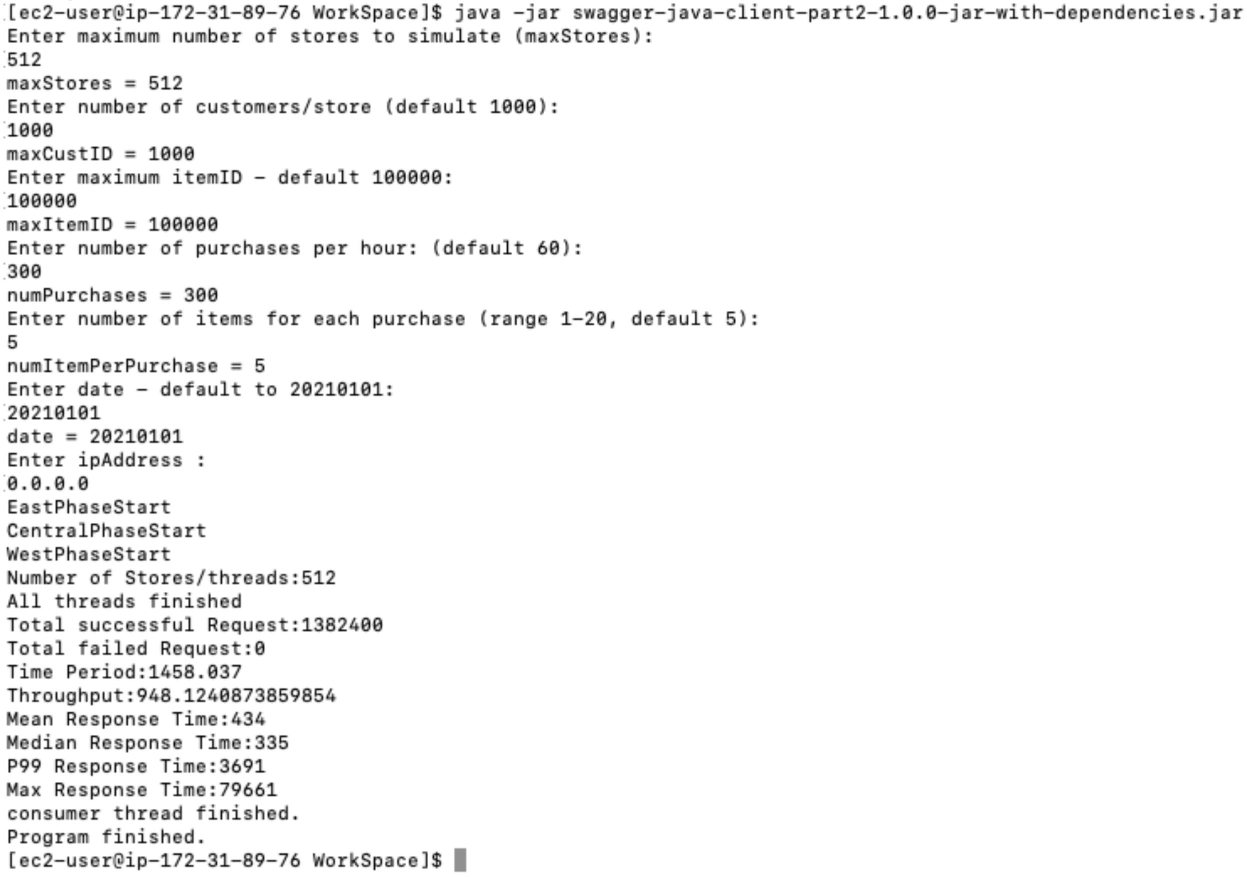
See detailed on dynamoDB throttling:
https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.ReadWriteCapacityMode.html

## Mean Response Time against Thread Number (Load Balancer Server)



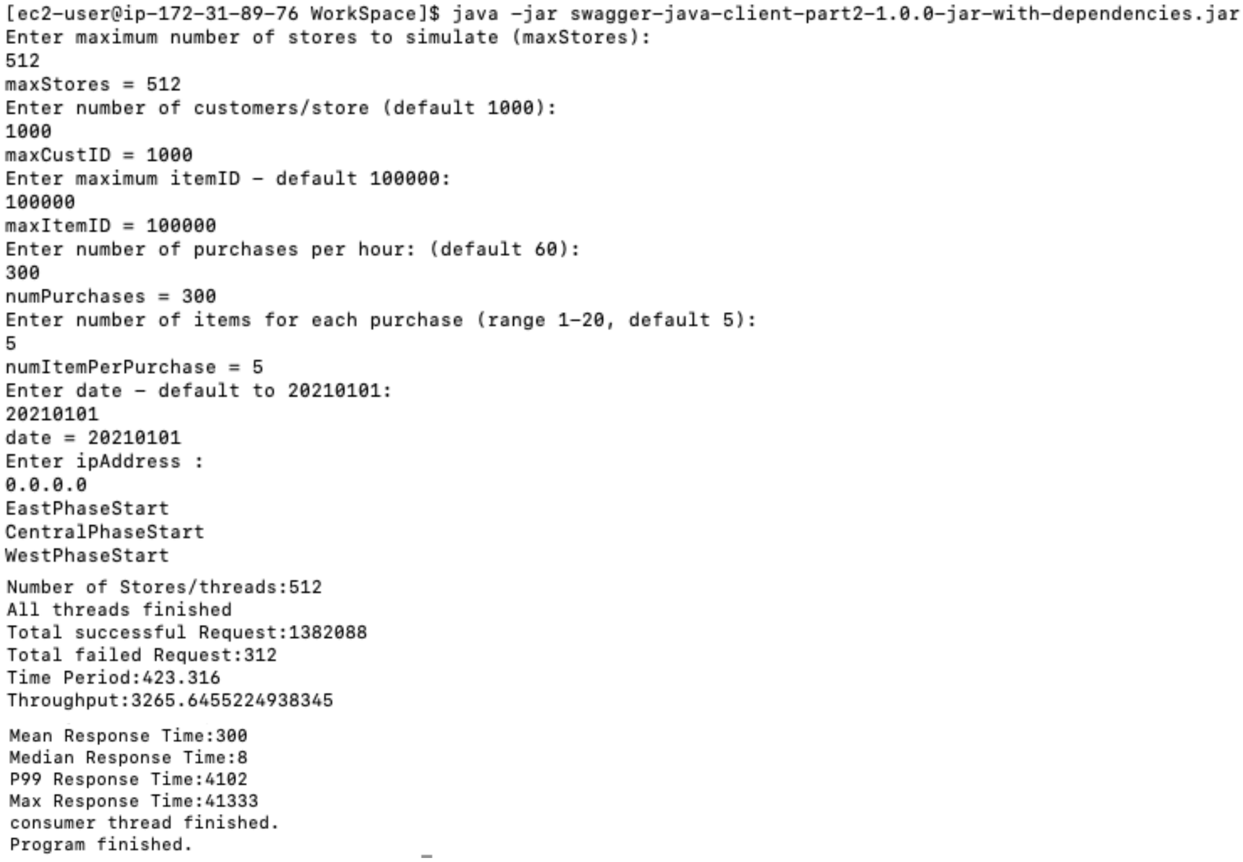
Mean response time will first increase at a steady rate for the server has basically same throughput when thread number is low. When database is subjected to throttling and throughput on the server drop down, mean response time start to increase at a much higher rate due to a longer wait at server. But overall result are better then single server.

## Bonus - Test run with 512 clients as max threads
## Single Server Case:



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
512
maxStores = 512
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:512
All threads finished
Total successful Request:1382400
Total failed Request:0
Time Period:1458.037
Throughput:948.1240873859854
Mean Response Time:434
Median Response Time:335
P99 Response Time:3691
Max Response Time:79661
consumer thread finished.
Program finished.
[ec2-user@ip-172-31-89-76 WorkSpace]$
```

## Load Balancer Case



```
[ec2-user@ip-172-31-89-76 WorkSpace]$ java -jar swagger-java-client-part2-1.0.0-jar-with-dependencies.jar
Enter maximum number of stores to simulate (maxStores):
512
maxStores = 512
Enter number of customers/store (default 1000):
1000
maxCustID = 1000
Enter maximum itemID - default 100000:
100000
maxItemID = 100000
Enter number of purchases per hour: (default 60):
300
numPurchases = 300
Enter number of items for each purchase (range 1-20, default 5):
5
numItemPerPurchase = 5
Enter date - default to 20210101:
20210101
date = 20210101
Enter ipAddress :
0.0.0.0
EastPhaseStart
CentralPhaseStart
WestPhaseStart
Number of Stores/threads:512
All threads finished
Total successful Request:1382088
Total failed Request:312
Time Period:423.316
Throughput:3265.6455224938345
Mean Response Time:300
Median Response Time:8
P99 Response Time:4102
Max Response Time:41333
consumer thread finished.
Program finished.
```

When running 512 store thread on load balancer server, I encountered some bad gateway exceptions on post request , these requests has been counted as failed requests( can be see from above, failed request: 312 , at a failed rate of 0.022%)
Exception messages have been excluded from screenshot.