1.Github Repo Link

https://github.com/Johnspeanut/cs6650Assignment3

2. Description of Database Design

2.1 Overview

I create two RDSs on AWS as databases for this assignment. RDS is ralational database and is good at joining and query.I set the security group of the databases to accept TCP from anywhere. As long as server send a request to write in the databases, it will write into the databases. After the writing, the message will remove from the RabittMQ queue.

The tables in skier database and resort database share the similar fields:

postId	resortId	seasonId	dayId	skierId	time	Lift
INT AUTO_INCREAMENT	INT	INT	INT	INT	INT	INT

Among the fields, `postId` serves as primary key.

2.2 Primary Packages for Database

2.2.1 Primary Packages

I create three packages for database:

- jdbc: database package for skier alone
- jdbcResort: database package for resort alone
- consumer2DB: database package for both skier and resort Other packages involed:
- client: package to send http request
- servlet: package to receive http post request and push messages into RabbitMQ queue.

2.2.2 Classes

2.2.2.1 `DBCPDataSource` class

The class aims to connect RDS database on AWS.

2.2.2.2 `LiftRide`

This class is to encapsulate a lift ride record. It includes constructor, get methods, and set methods.

2.2.2.3 `Dao`

This class is to insert skier or resort data into the RDS databases.

2.2.2.4 `Main`

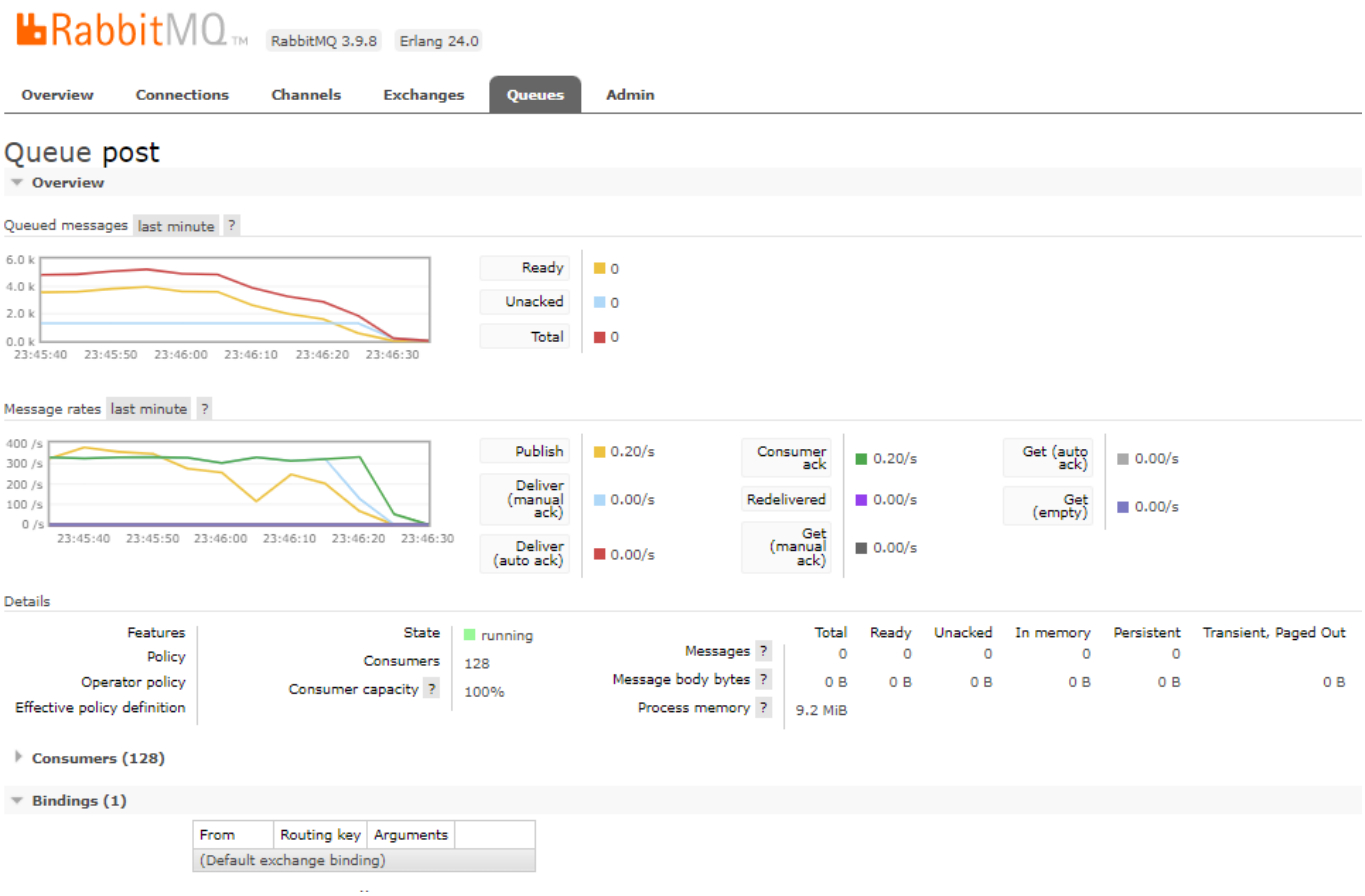
This class is to pull messages from RabbitMQ queue and call Dao class method to write messages into the connected databases.

3. Test Runs for Skier Microservice or Resorts Microservice

- number of databases:1
- number of Tomcat servers: 1
- number of channels in each Tomcat server: 64
- number of threads in the consumer: 128
- basicQos for each channel in consumer: 10

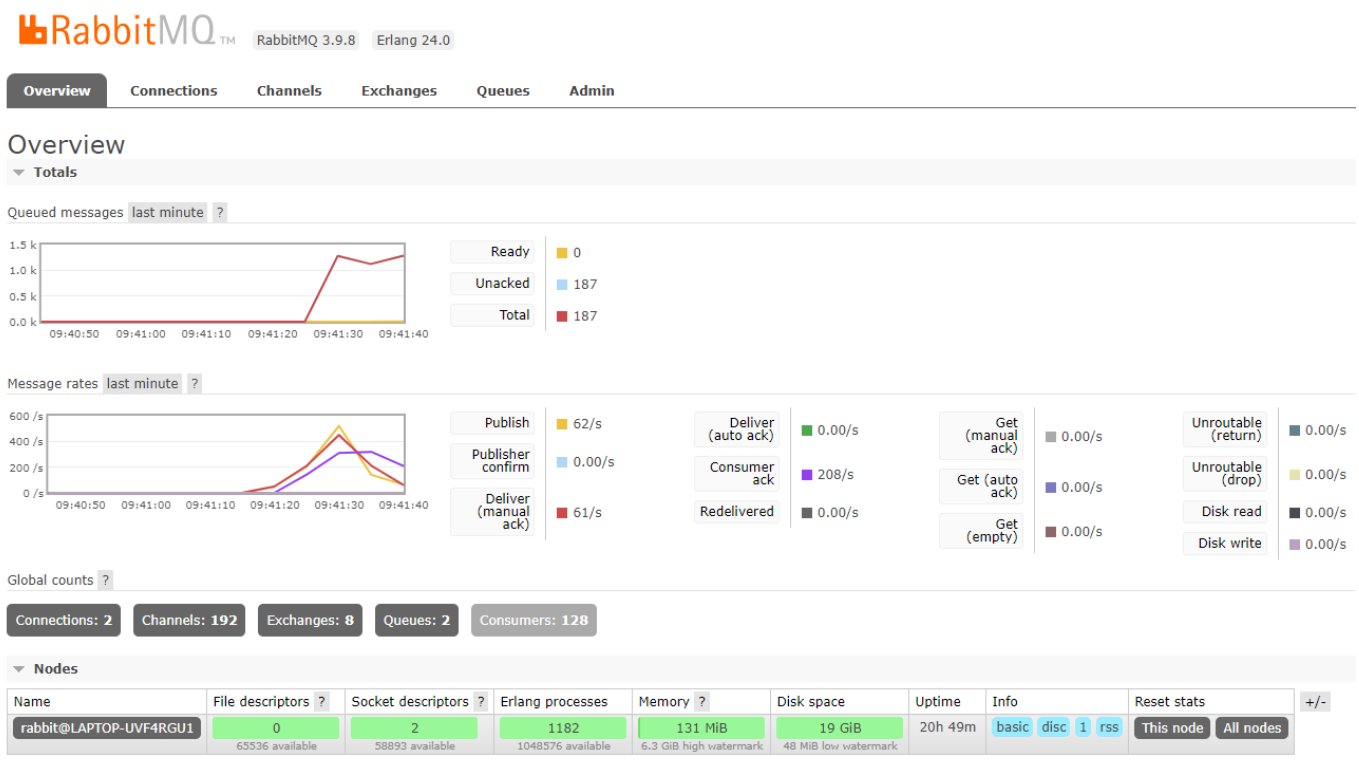
3.1 Test Runs for Skier

3.1.1 128-Threads in client



```
response:Success: /4/seasons/2021/days/100/skiers/44730085 LiftRide@1d151845
response:Success: /3/seasons/2021/days/100/skiers/44732291 LiftRide@ba10c5f
response:Success: /2/seasons/2021/days/100/skiers/44735709 LiftRide@6ed62ce4
response:Success: /9/seasons/2021/days/100/skiers/44739188 LiftRide@58ffef50
response:Success: /6/seasons/2021/days/100/skiers/44743156 LiftRide@620d0da5
response:Success: /9/seasons/2021/days/100/skiers/44746541 LiftRide@6d531a82
response:Success: /9/seasons/2021/days/100/skiers/44748148 LiftRide@3ea103
Number requests: 87173
Number of success: 87173
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 5
Wall time (second): 495
```

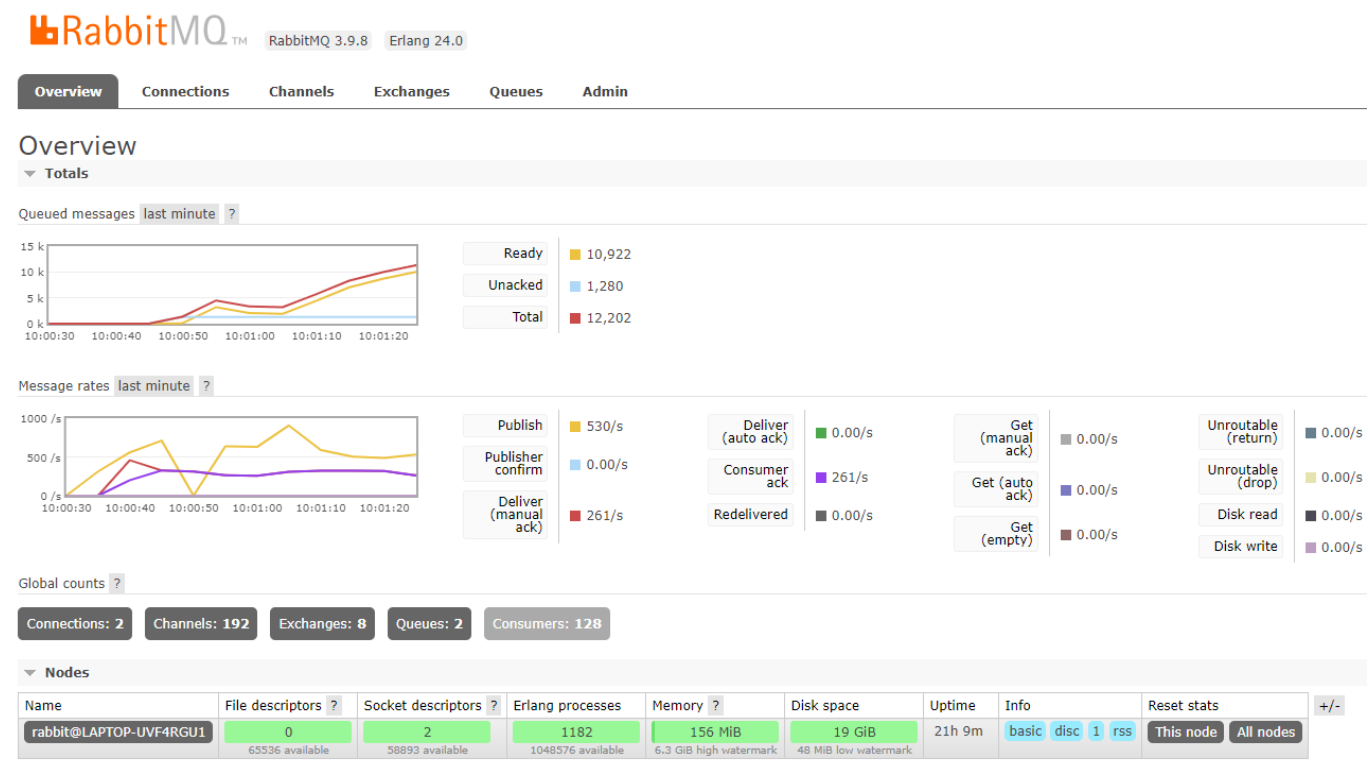
3.1.2 256-Threads in client




```
response:Success: /8/seasons/2021/days/100/skiers/22358643 LiftRide@df213ee
response:Success: /6/seasons/2021/days/100/skiers/22360069 LiftRide@5a58dac7
response:Success: /3/seasons/2021/days/100/skiers/22362886 LiftRide@9fbe70b
response:Success: /1/seasons/2021/days/100/skiers/22364141 LiftRide@7e2ee74
response:Success: /3/seasons/2021/days/100/skiers/22364752 LiftRide@70b75396
response:Success: /4/seasons/2021/days/100/skiers/22366936 LiftRide@6aff0ff7
Number requests: 99287
Number of success: 99287
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 3
Wall time (second): 316
```

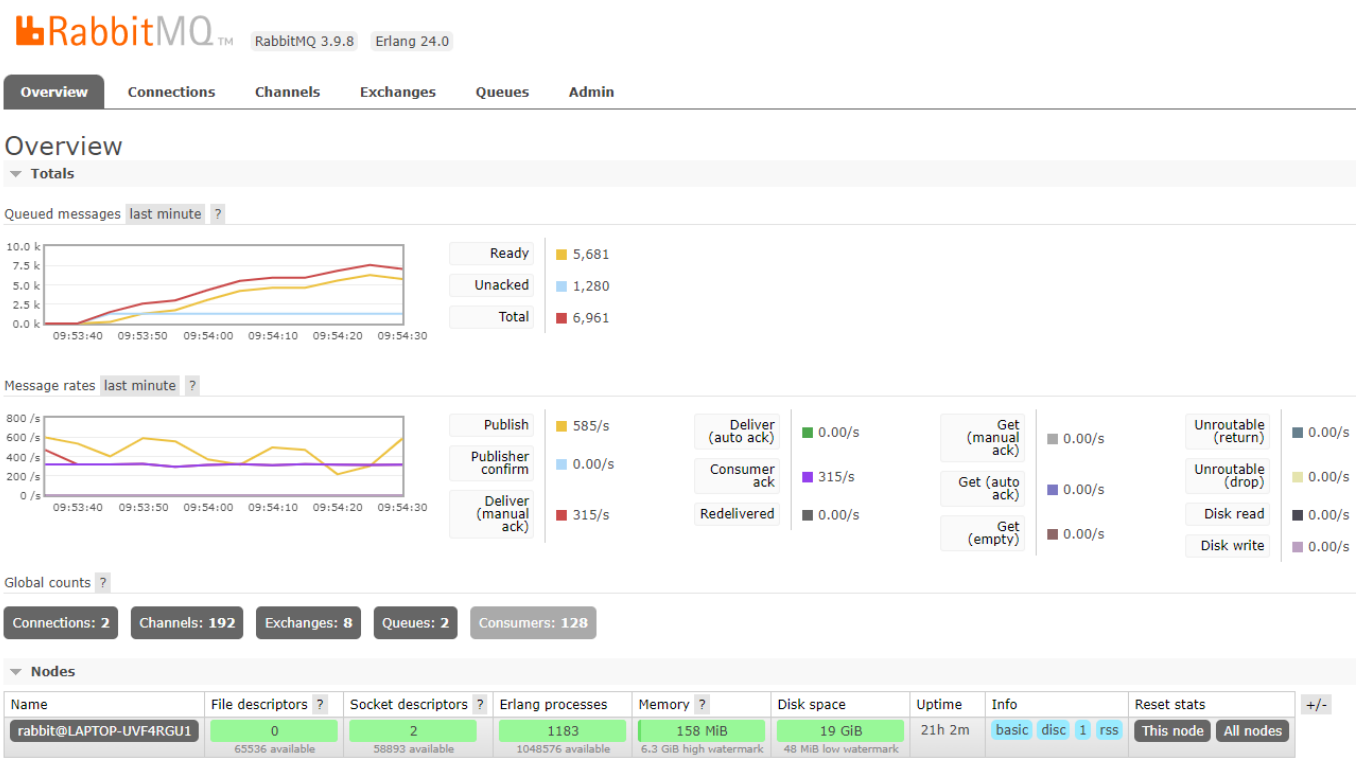
3.2 Test Runs for Resort

3.2.1 128-Threads in client




```
response:Success: /5/seasons/2021/days/100/skiers/44717902 LiftRide@7f40b4eb
response:Success: /6/seasons/2021/days/100/skiers/44718820 LiftRide@7a66656f
response:Success: /4/seasons/2021/days/100/skiers/44723713 LiftRide@71887813
response:Success: /7/seasons/2021/days/100/skiers/44727712 LiftRide@36ba2001
response:Success: /7/seasons/2021/days/100/skiers/44728167 LiftRide@37f361a6
response:Success: /0/seasons/2021/days/100/skiers/44732922 LiftRide@387bfcda
response:Success: /5/seasons/2021/days/100/skiers/44737304 LiftRide@4c45a204
response:Success: /0/seasons/2021/days/100/skiers/44737657 LiftRide@388fc092
response:Success: /9/seasons/2021/days/100/skiers/44742074 LiftRide@6550e3c1
response:Success: /1/seasons/2021/days/100/skiers/44745556 LiftRide@1ce25dff
response:Success: /4/seasons/2021/days/100/skiers/44748957 LiftRide@559d2187
Number requests: 98847
Number of success: 98847
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 3
Wall time (second): 372
```

3.2.2 256-Threads in client

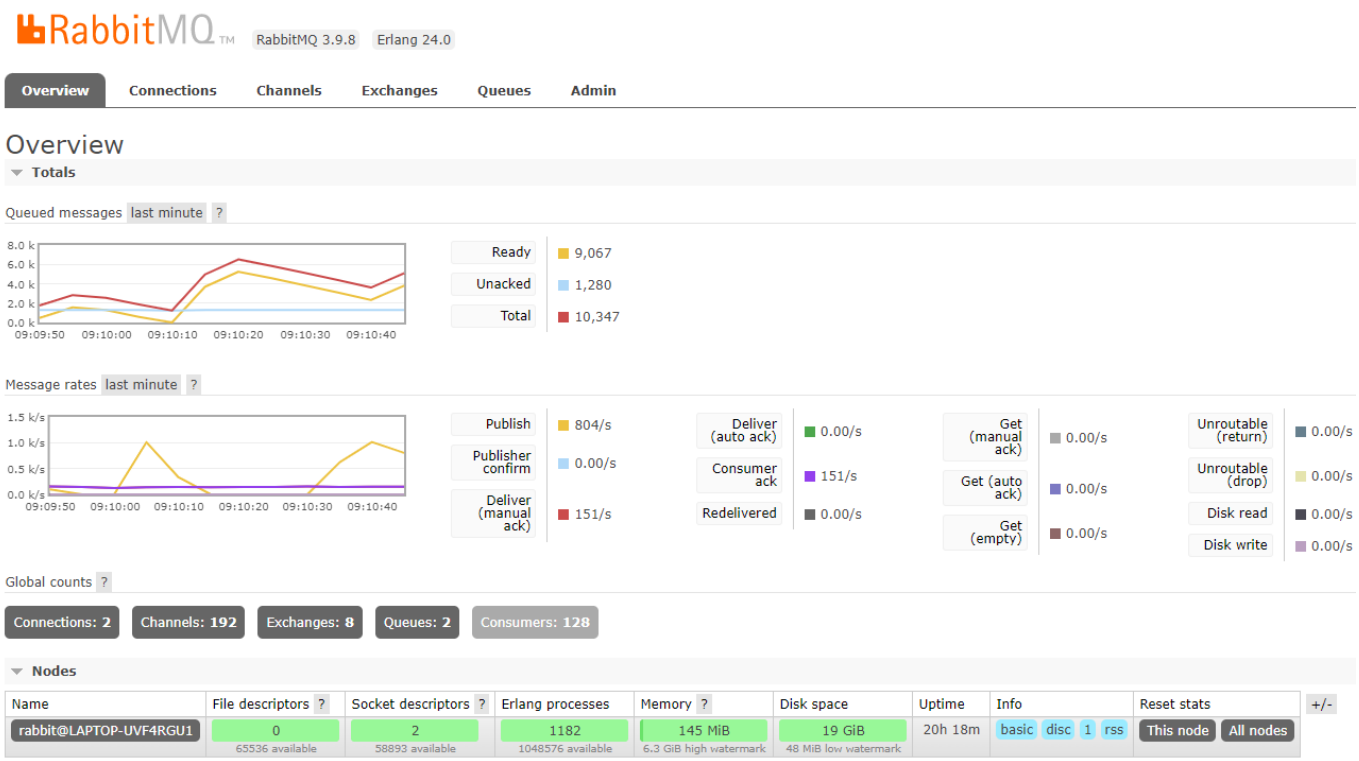



```
response:Success: /6/seasons/2021/days/100/skiers/22358604 LiftRide@60c3edd8
response:Success: /0/seasons/2021/days/100/skiers/22361367 LiftRide@6d6d7a1a
response:Success: /2/seasons/2021/days/100/skiers/22362658 LiftRide@57fe9ae9
response:Success: /0/seasons/2021/days/100/skiers/22363951 LiftRide@3b02fe
response:Success: /2/seasons/2021/days/100/skiers/22366104 LiftRide@9ca706b
response:Success: /6/seasons/2021/days/100/skiers/22366992 LiftRide@3d4fe39f
Number requests: 99856
Number of success: 99856
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 3
Wall time (second): 368
```

4. Test Runs with both Skier and Resorts Microservice

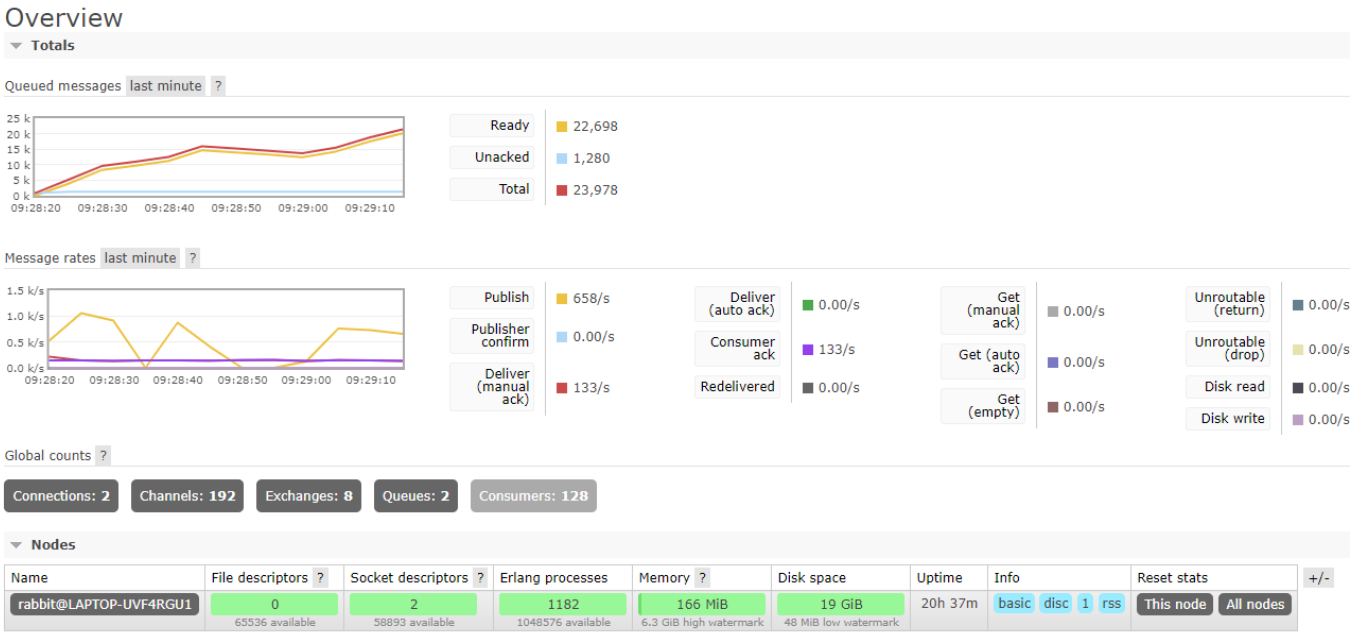
- number of databases:2
- number of Tomcat servers: 1
- number of channels in each Tomcat server: 64
- number of threads in the consumer: 128
- basicQos for each channel in consumer: 10

4.1 128-Threads




```
response:Success: /7/seasons/2021/days/100/skiers/44731537 LiftRide@602a594
response:Success: /1/seasons/2021/days/100/skiers/44735841 LiftRide@6d7edf6c
response:Success: /1/seasons/2021/days/100/skiers/44737791 LiftRide@3225e540
response:Success: /4/seasons/2021/days/100/skiers/44742865 LiftRide@49cb5eec
response:Success: /4/seasons/2021/days/100/skiers/44746267 LiftRide@901362f
response:Success: /8/seasons/2021/days/100/skiers/44748732 LiftRide@10af5f03
Number requests: 99976
Number of success: 99976
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 2
Wall time (second): 292
```

4.2 256-Threads




```
response:Success: /9/seasons/2021/days/100/skiers/22359399 LiftRide@a443008
response:Success: /1/seasons/2021/days/100/skiers/22361400 LiftRide@1dc6f225
response:Success: /5/seasons/2021/days/100/skiers/22362551 LiftRide@473b2c39
response:Success: /2/seasons/2021/days/100/skiers/22363948 LiftRide@5a6aea09
response:Success: /3/seasons/2021/days/100/skiers/22365324 LiftRide@4af77763
response:Success: /3/seasons/2021/days/100/skiers/22366495 LiftRide@7ce93051
Number requests: 96305
Number of success: 96305
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 3
Wall time (second): 359
```

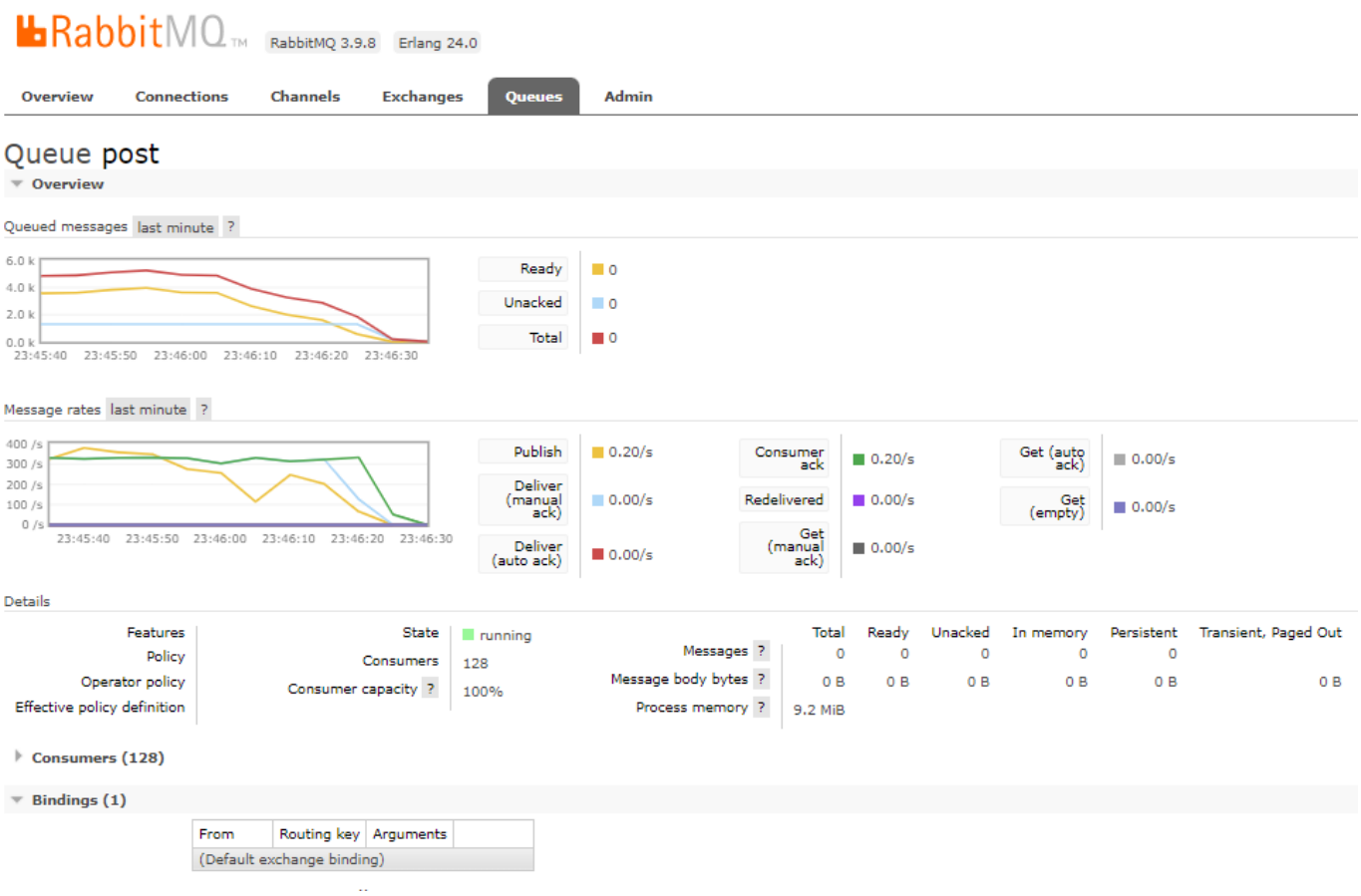
5.Explanation of Mitigation Strategy

5.1 Comparison of results before and after applying mitigation strategy

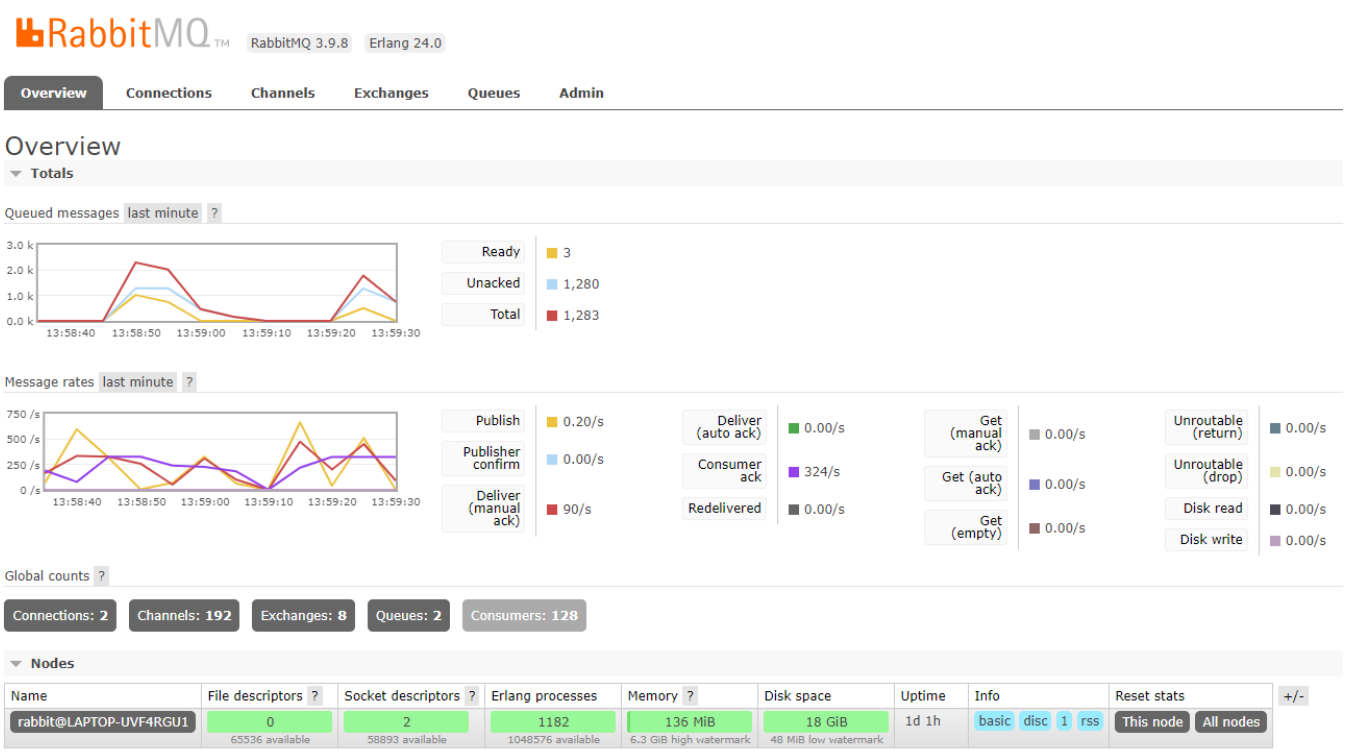
- number of databases:1
- number of Tomcat servers: 1
- number of channels in each Tomcat server: 64
- number of threads in the consumer: 128
- basicQos for each channel in consumer: 10 I use circuit breaker in the client side to deal with unreliabl server. In particular the client will have a rest time of 1 minute if there are 10 errors in 1 minute. After taking circuit breaker mitigation strategy, the wall time increases a little bit. But the throughput rate increases.

5.1.1 128-Threads

5.1.1.1 Result before mitigation strategy



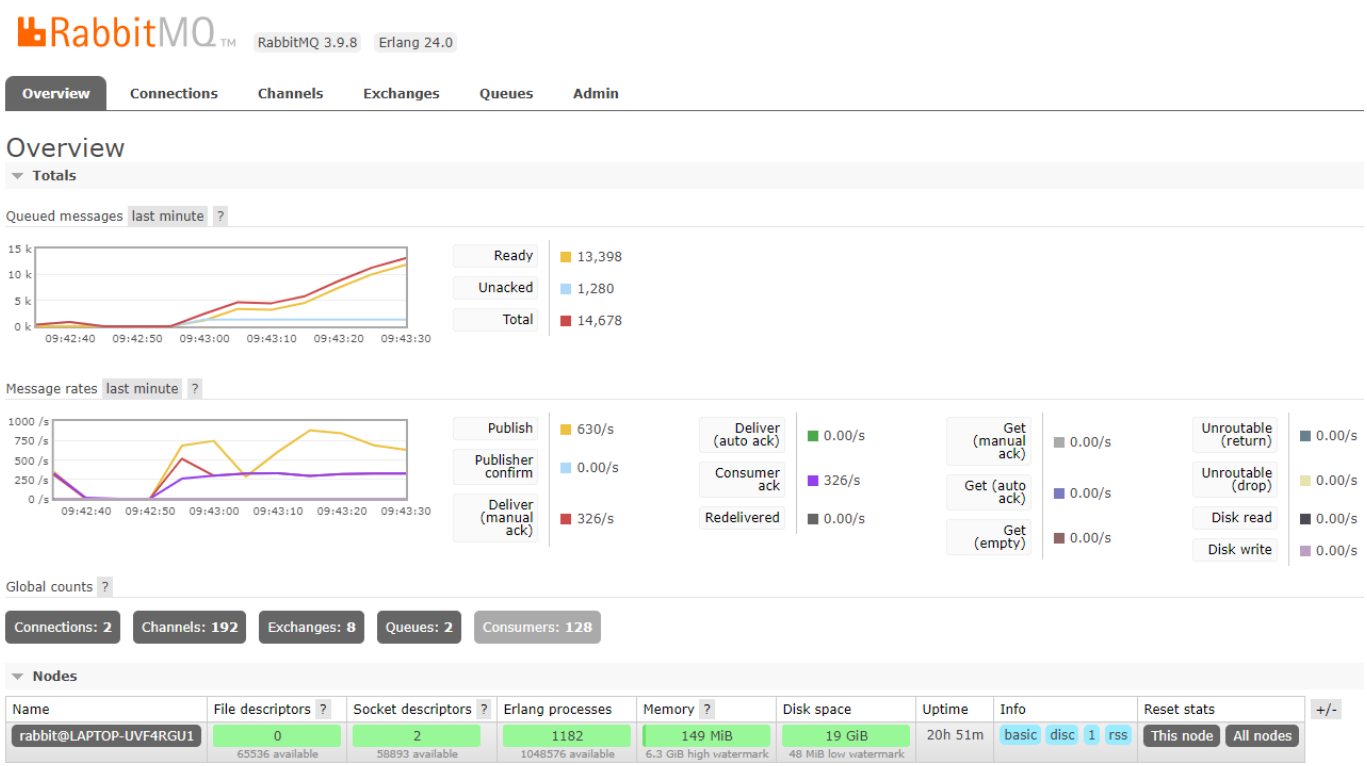
5.1.1.2 Result after mitigation strategy



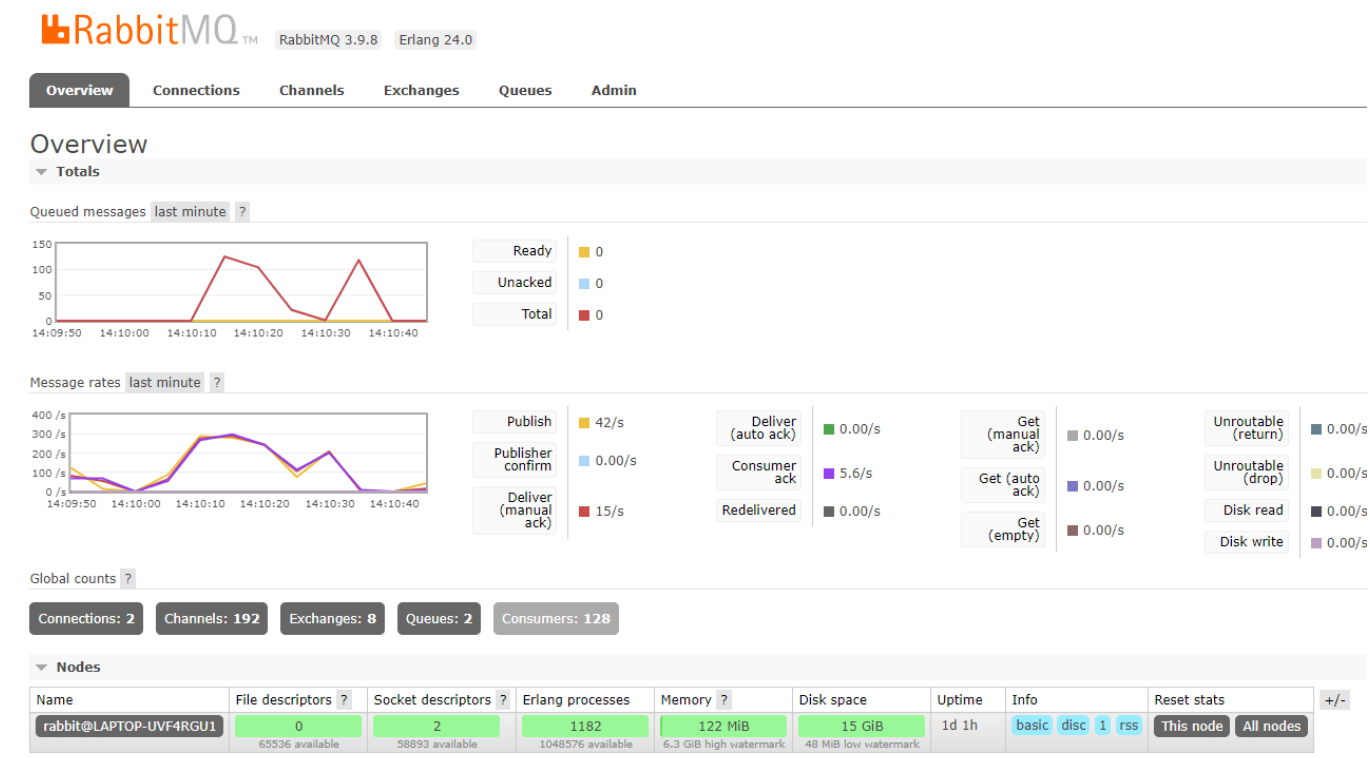

```
response:Success: /2/seasons/2021/days/100/skiers/44721085 LiftRide@3e43b9bd
response:Success: /9/seasons/2021/days/100/skiers/44721913 LiftRide@14c374dc
response:Success: /2/seasons/2021/days/100/skiers/44725227 LiftRide@4e1f0297
response:Success: /5/seasons/2021/days/100/skiers/44729180 LiftRide@3402eede
response:Success: /6/seasons/2021/days/100/skiers/44734127 LiftRide@1329112c
response:Success: /2/seasons/2021/days/100/skiers/44736876 LiftRide@4d50975c
response:Success: /7/seasons/2021/days/100/skiers/44738161 LiftRide@32f91ee8
response:Success: /0/seasons/2021/days/100/skiers/44743711 LiftRide@6318ef97
response:Success: /1/seasons/2021/days/100/skiers/44746436 LiftRide@6fd6c635
response:Success: /0/seasons/2021/days/100/skiers/44747196 LiftRide@4284c462
Number requests: 100000
Number of success: 100000
Number of failure: 0
Throughput in requests per second: Mean response time (milliseconds): 2
Wall time (second): 280
```

5.1.2 256-Threads

5.1.2.1 Result before mitigation strategy



5.1.2.2 Result after mitigation strategy




```
response:Success: /6/seasons/2021/days/100/skiers/44719779 LiftRide@46abdf76
response:Success: /5/seasons/2021/days/100/skiers/44723531 LiftRide@4dee8440
response:Success: /3/seasons/2021/days/100/skiers/44727494 LiftRide@1fa1e09f
response:Success: /5/seasons/2021/days/100/skiers/44730007 LiftRide@5c2ccb58
response:Success: /1/seasons/2021/days/100/skiers/44731898 LiftRide@149ec514
response:Success: /5/seasons/2021/days/100/skiers/44734833 LiftRide@35ca3c0e
response:Success: /6/seasons/2021/days/100/skiers/44738876 LiftRide@79b0c456
response:Success: /8/seasons/2021/days/100/skiers/44740751 LiftRide@27d49b4e
response:Success: /0/seasons/2021/days/100/skiers/44745130 LiftRide@71db83d2
response:Success: /8/seasons/2021/days/100/skiers/44749508 LiftRide@76171e45
Number requests: 100000
Number of success: 77932
Number of failure: 22068
Throughput in requests per second: Mean response time (milliseconds): 5
Wall time (second): 569
```

5.2 Explanation

Circuit breaker allows client pauses sending request to server when there have been too many messages in the RabbitMQ queue. Hence the server would not be down because of its low memory, which may increase throughput rate and wall time.