Tutorial 7

Question 0: What computer are you using as a Docker Host for tutorial #7? IS this an EC2 instance? If so, what type? If not an ec2 instance, how many CPUs are available on the DockerHost for the tutorial? How much RAM is available? Can use the command "lscpu" before running sudo docker exec to check the number of CPUs. Can use command "free -m" to check MB memory free on the host.

Answer: I am using ubuntu 18.04
Number of CPU's – 2
Memory free – 4944 MB

Question 1: Without running any test, how much CPU time has been spent in seconds, since this container was created?

Answer: 570744357Nano seconds = 0.570744357 seconds.

Question 2: After running the test, what is the present CPU utilization value in seconds?

Answer: 17891646228 Nano seconds = 17.891646228 seconds.

Question 3: What is the difference in CPU time in seconds that transpired for running the test? (subtract the two values)

Answer: 17.891646228 - 0.570744357 = 17.320901871 seconds.

Question 4: Is it faster to pull the docker image from DockerHub or rebuild the image from scratch locally? Please list the times for pulling vs. building.

Answer: For pulling the docker it took 0m2.238s and for building the docker it took 0m40.170s. **Therefore, for me pulling the docker was faster compared to building the docker.**

QUESTION 5: What happens to the runtime of the test?

Answer: After assigning the cpu-shares of the docker container and performing the stress test, the runtime of the test is 7.82 seconds.

QUESTION 6: What was the CPU utilization for the test? How did it vary from our previous measurement? What could explain the behavior we are seeing?

Answer: CPU utilization before the test was 250912016687 nano seconds = 12.512016687 seconds
CPU utilization after the test was 273902763351 nano seconds = 44.402763351 seconds
CPU utilization = 31.8907467 seconds approx (Difference)
Therefore, the CPU usage time is increased from before assigning the CPU shares to after assigning the CPU shares.

QUESTION 7: What is the performance difference when running the command standalone vs. running two instances at the same time with CPU allocation has been set to 1?

Answer: Performance when running standalone = 18.36 seconds
Performance when running two instances at the same time –

- Container A = 26.10 seconds
- Container B = 26.08 seconds

Therefore, the performance difference = 26.10 - 18.36 = 7.74 seconds

QUESTION 8: What is the memory throughput values (MiB/sec) for both containers A and B?

Answer: Memory throughput for the containers-

- Container A = 83072.00 MiB transferred 8304.55 MiB/sec
- Container B = 82268.00 MiB transferred 8222.94 MiB/sec

QUESTION 9: What is the average memory latency (in ms) for both containers A and B?

Answer: Average memory latency for the containers-

- Container A = 0.12ms
- Container B = 0.12ms

QUESTION 10: How did the memory throughput and memory latency change when comparing the standalone (1 container) test values with the concurrent container test?

Answer:

	Memory Throughput	**Memory Latency**
Standalone	8704.17 MiB/sec	**Avg – 0.12ms**
Concurrent(Acontainer) **Concurrent(Bcontainer)**	8222.94 MiB/sec 8304.55M.B/sec	**Avg – 0.12ms** **Avg -0.12ms**

Memory throughput values have increased when performing standalone to concurrent. Memory latency are same when performing standalone and concurrent.

QUESTION 11: From the test results, do the docker containers appear to provide?
(a) Better memory isolation
(b) Better CPU isolation

Answer: ***(a) Better memory isolation – YES***
(b) Better CPU isolation - NO

QUESTION 12: Provide a plausible explanation for Question #11.

Solution: I got better memory isolation as my latency values for both i.e. running standalone and concurrently are same (0.12ms) and my CPU isolation is not better as my CPU utilization values for standalone is 18.36 seconds and concurrent running are 26.10 seconds and 26.08 seconds i.e. values are different.