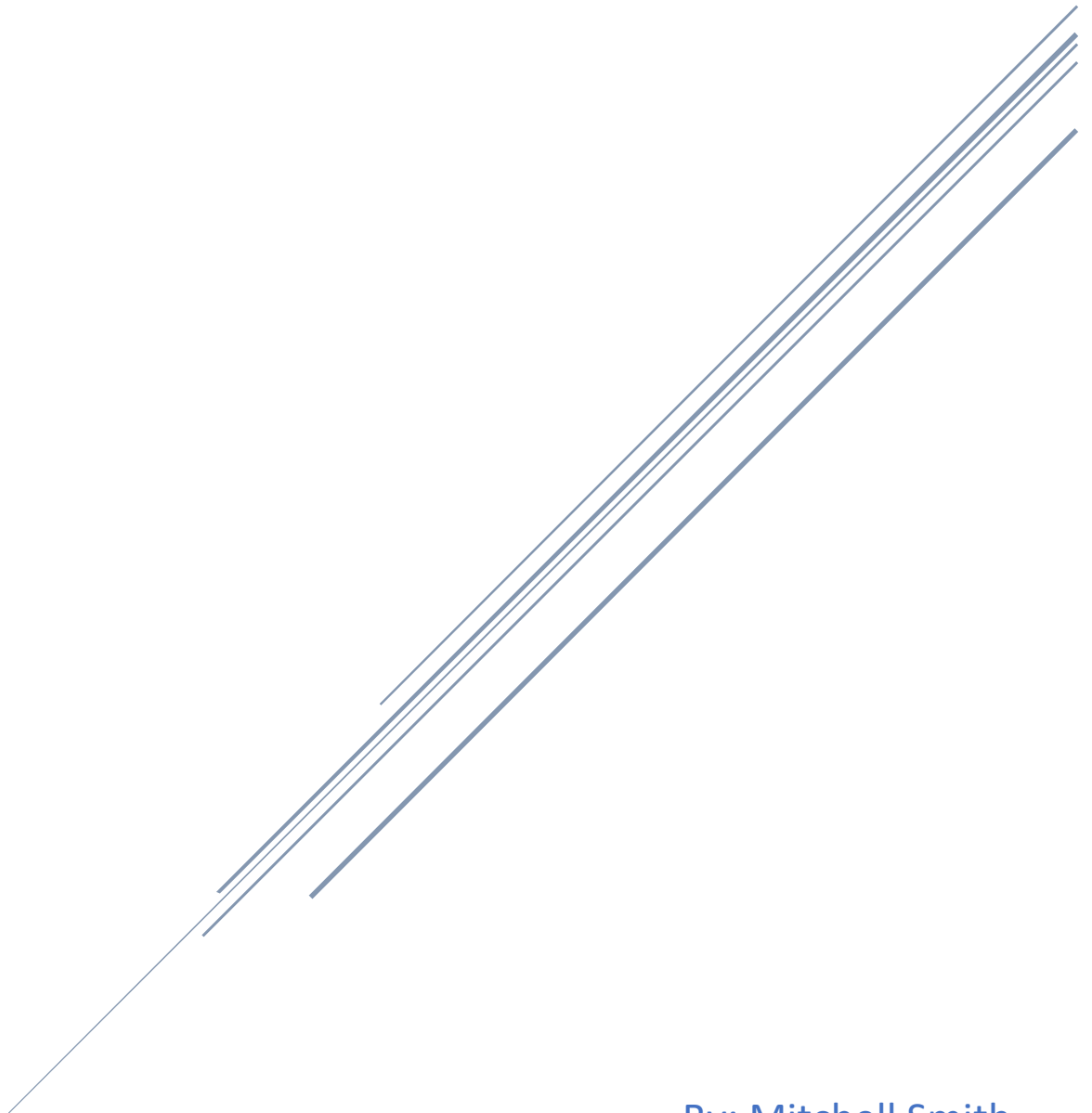


# ASSIGNMENT #4

INET3700 - Automation & Docker



By: Mitchell Smith  
W0440925

## Table of Contents

|  |    |
|--|----|
| Introduction .....   | 2  |
| Task 1: Automation (Cron) .....  | 2  |
| Assignment questions and / or required screenshots provided below. ....  | 2  |
| Step 1. Backup your database (weekly every Friday evening) .....   | 2  |
| Step 2. Setup grandfather, father and son backup jobs for all of your data. ....   | 5  |
| Step 3. On reboot check and output the status of DBMS to file.....   | 9  |
| Step 4. Every Monday at 12:01am create a benchmark report and save it for review with the<br>date/time as part of the name.....                            | 12 |
| Task 2: Docker .....   | 17 |
| Assignment questions and / or required screenshots provided below. ....  | 17 |
| Step 1. Demonstrate a simple Installation & Configuration of Docker within the Ubuntu Server<br>previously created. Clearly demonstrate Docker in use..... | 17 |
| Step 2. Give 6 examples of what you can see Docker being used for, justify your examples. ....   | 19 |
| Task 3: Documentation.....   | 20 |
| Assignment questions and / or required screenshots provided below. ....  | 20 |
| Task 4: Gold Copy .....  | 27 |
| Proof of Gold Copy backed up onto external drive provided below. ....  | 27 |
| Conclusion.....  | 28 |
| References.....  | 29 |
| Website Names, Titles, Authors, Dates, and Links of Articles Used Provided Below.....  | 29 |

## Introduction

In this document, I am going to demonstrate my understanding of the requirements of this assignment pertaining to the topic of automation and docker. I am going to accomplish this through providing detailed explanations for the required questions, as well as by providing labelled screenshots for the technical requirements.

## Task 1: Automation (Cron)

Assignment questions and / or required screenshots provided below.

Step 1. Backup your database (weekly every Friday evening)

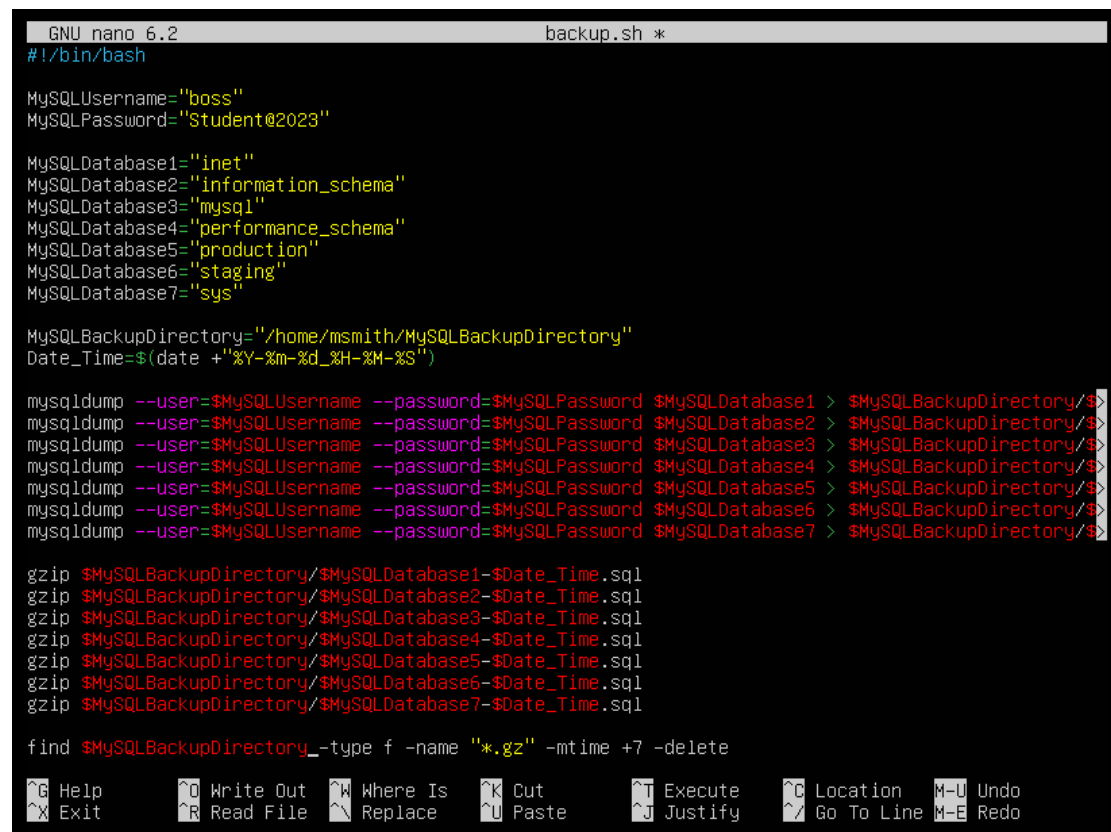
**Screenshot of creating a backup directory called “MySQLBackupDirectory” and navigating to it:**

```
msmith@wbms01:~$ mkdir MySQLBackupDirectory
msmith@wbms01:~$ cd MySQLBackupDirectory
msmith@wbms01:~/MySQLBackupDirectory$
```

**Screenshot of creating a bash script called “backup.sh”**

```
msmith@wbms01:~/MySQLBackupDirectory$ sudo nano backup.sh
[sudo] password for msmith: _
```

**Screenshot of my created bash script:**



```
GNU nano 6.2 backup.sh *
#!/bin/bash

MySQLUsername="boss"
MySQLPassword="Student@2023"

MySQLDatabase1="inet"
MySQLDatabase2="information_schema"
MySQLDatabase3="mysql"
MySQLDatabase4="performance_schema"
MySQLDatabase5="production"
MySQLDatabase6="staging"
MySQLDatabase7="sys"

MySQLBackupDirectory="/home/msmith/MySQLBackupDirectory"
Date_Time=$(date +"%Y-%m-%d_%H-%M-%S")

mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase1 > $MySQLBackupDirectory/$Date_Time.sql
mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase2 > $MySQLBackupDirectory/$Date_Time.sql
mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase3 > $MySQLBackupDirectory/$Date_Time.sql
mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase4 > $MySQLBackupDirectory/$Date_Time.sql
mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase5 > $MySQLBackupDirectory/$Date_Time.sql
mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase6 > $MySQLBackupDirectory/$Date_Time.sql
mysqldump --user=$MySQLUsername --password=$MySQLPassword $MySQLDatabase7 > $MySQLBackupDirectory/$Date_Time.sql

gzip $MySQLBackupDirectory/$MySQLDatabase1-$Date_Time.sql
gzip $MySQLBackupDirectory/$MySQLDatabase2-$Date_Time.sql
gzip $MySQLBackupDirectory/$MySQLDatabase3-$Date_Time.sql
gzip $MySQLBackupDirectory/$MySQLDatabase4-$Date_Time.sql
gzip $MySQLBackupDirectory/$MySQLDatabase5-$Date_Time.sql
gzip $MySQLBackupDirectory/$MySQLDatabase6-$Date_Time.sql
gzip $MySQLBackupDirectory/$MySQLDatabase7-$Date_Time.sql

find $MySQLBackupDirectory -type f -name "*.gz" -mtime +7 -delete

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```

### Screenshot of giving the bash script executable permissions:

```
msmith@wbms01:~/MySQLBackupDirectory$ sudo chmod +x backup.sh
[sudo] password for msmith:
msmith@wbms01:~/MySQLBackupDirectory$ _
```

### Screenshot of executing the bash script:

```
msmith@wbms01:~/MySQLBackupDirectory$ sudo bash backup.sh
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: Dumping 'information_schema' DB content is not supported
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: Got error: 1142: SELECT, LOCK TABLES command denied to user 'boss'@'localhost' for table
'accounts' when using LOCK TABLES
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: [Warning] Using a password on the command line interface can be insecure.
msmith@wbms01:~/MySQLBackupDirectory$ _
```

### Screenshot of MySQL databases backed up to the “MySQLBackupDirectory” directory:

```
msmith@wbms01:~/MySQLBackupDirectory$ ls
backup.sh                               performance_schema-2023-11-08_17-21-47.sql.gz
inet-2023-11-08_17-21-47.sql.gz          production-2023-11-08_17-21-47.sql.gz
information_schema-2023-11-08_17-21-47.sql.gz staging-2023-11-08_17-21-47.sql.gz
mysql-2023-11-08_17-21-47.sql.gz         sys-2023-11-08_17-21-47.sql.gz
msmith@wbms01:~/MySQLBackupDirectory$
```

### Screenshot of modifying the cron job scheduler utility to automatically run the bash script every Friday at 10:00PM:

```
GNU nano 6.2 /tmp/crontab.jj1vdY/crontab *
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh_
```

Help Write Out Where Is Cut Execute Location M-U Undo  
Exit Read File Replace Paste Justify Go To Line M-E Redo

Screenshot confirming that my bash script is set up to execute every Friday evening at 10:00PM:

```
msmith@wbms01:~/MySQLBackupDirectory$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
```

Screenshot of changing the cron job to execute in the next minute for testing:

```
GNU nano 6.2 /tmp/crontab.yMRAzs/crontab *
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
* * * * * /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
@reboot /home/msmith/MySQLStatusReport.sh
0 2 * * * /home/msmith/GFSDDataBackupScript.sh
```

```
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```

Screenshot demonstrating cron job successfully running and backing up all of my MySQL Databases:

```
msmith@wbms01:~/MySQLBackupDirectory$ ls
backup.sh
inet-2023-11-08_17-21-47.sql.gz      performance_schema-2023-11-10_22-00-01.sql.gz
inet-2023-11-10_22-00-01.sql.gz      performance_schema-2023-11-13_01-19-09.sql.gz
inet-2023-11-13_01-19-09.sql.gz      production-2023-11-08_17-21-47.sql.gz
information_schema-2023-11-08_17-21-47.sql.gz  production-2023-11-10_22-00-01.sql.gz
information_schema-2023-11-10_22-00-01.sql.gz  production-2023-11-13_01-19-09.sql.gz
information_schema-2023-11-13_01-19-09.sql.gz  staging-2023-11-08_17-21-47.sql.gz
mysql-2023-11-08_17-21-47.sql.gz      staging-2023-11-10_22-00-01.sql.gz
mysql-2023-11-10_22-00-01.sql.gz      staging-2023-11-13_01-19-09.sql.gz
mysql-2023-11-13_01-19-09.sql.gz      sys-2023-11-08_17-21-47.sql.gz
performance_schema-2023-11-08_17-21-47.sql.gz  sys-2023-11-10_22-00-01.sql.gz
sys-2023-11-13_01-19-09.sql.gz
msmith@wbms01:~/MySQLBackupDirectory$
```

Step 2. Setup grandfather, father and son backup jobs for all of your data.

Screenshot of creating a “DataBackupDirectory” directory:

```
msmith@wbms01:~$ mkdir DataBackupDirectory
msmith@wbms01:~$ _
```

Screenshot of creating a bash backup script called “GFSDDataBackupScript.sh”:

```
msmith@wbms01:~$ sudo nano GFSDDataBackupScript.sh
[sudo] password for msmith: _
```

Screenshots of my “GFSDDataBackupScript.sh” bash script:

```
GNU nano 6.2 GFSDDataBackupScript.sh
#!/bin/bash

Data_Backup="/home /var/spool/mail /etc /root /boot /opt"
File_Path="/home/msmith/DataBackupDirectory"

NameOfWeekDay=$(date +%A)
UbuntuServerName=$(hostname -s)
DayOfMonth=$(date +%d)

if (( $DayOfMonth <= 7 )); then
    FileName_WK="$UbuntuServerName-FirstWeekOfMonth.tgz"
elif (( $DayOfMonth > 7 && $DayOfMonth <= 14 )); then
    FileName_WK="$UbuntuServerName-SecondWeekOfMonth.tgz"
elif (( $DayOfMonth > 14 && $DayOfMonth <= 21 )); then
    FileName_WK="$UbuntuServerName-ThirdWeekOfMonth.tgz"
elif (( $DayOfMonth > 21 && $DayOfMonth <= 31 )); then
    FileName_WK="$UbuntuServerName-FourthWeekOfMonth.tgz"
fi

MonthOfYear=$(date +%m)
OddOrEvenMonth=$(expr $MonthOfYear % 2)

if [ $MonthOfYear -eq 0 ]; then
    FileName_MTH="$UbuntuServerName-EvenMonth.tgz"
else
    FileName_MTH="$UbuntuServerName-OddMonth.tgz"
fi

G Help      W Write Out  W Where Is  K Cut       T Execute   C Location  M-U Undo
X Exit      R Read File  N Replace   U Paste     J Justify   _ Go To Line M-E Redo
```

```
GNU nano 6.2 GFSDataBackupScript.sh

MonthOfYear=$(date +%m)
OddOrEvenMonth=$(expr $MonthOfYear % 2)

if [ $MonthOfYear -eq 0 ]; then
    FileName_MTH="$UbuntuServerName-EvenMonth.tgz"
else
    FileName_MTH="$UbuntuServerName-OddMonth.tgz"
fi

if [ $DayOfMonth == 1 ]; then
    FileName_ARCH=$FileName_MTH
elif [ $NameOfWeekDay != "Saturday" ]; then
    FileName_ARCH="$UbuntuServerName-$NameOfWeekDay.tgz"
else
    FileName_ARCH=$FileName_HK
fi

echo "Backing up $Data_Backup to $File_Path/$FileName_ARCH"
date
echo

tar czf $File_Path/$FileName_ARCH $Data_Backup

echo
echo "Backup has completed"
date

ls -lh $File_Path/
-
```

Screenshot of giving the bash script executable permissions:

```
msmith@wbms01:~$ sudo chmod +x GFSDataBackupScript.sh
[sudo] password for msmith:
msmith@wbms01:~$
```

Screenshot demonstrating my bash script successfully running:

```
msmith@wbms01:~$ sudo /home/msmith/GFSDataBackupScript.sh
Backing up /home /var/spool/mail /etc /root /boot /opt to /home/msmith/DataBackupDirectory/wbms01-Sunday.tgz
Sun Nov 12 10:01:47 PM UTC 2023

tar: Removing leading `/' from member names
tar: Removing leading `/' from hard link targets

Backup has completed
Sun Nov 12 10:03:54 PM UTC 2023
total 713M
-rw-rw-r-- 1 msmith msmith 238M Nov 12 21:43 wbms01-SecondWeekOfMonth.tgz
-rw-r--r-- 1 root root 475M Nov 12 22:03 wbms01-Sunday.tgz
msmith@wbms01:~$
```

Screenshot demonstrating backup files successfully created in “DataBackupDirectory” directory:

```
msmith@wbms01:~/DataBackupDirectory$ ls
wbms01-SecondWeekOfMonth.tgz wbms01-Sunday.tgz
msmith@wbms01:~/DataBackupDirectory$ _
```

Screenshot of creating a cron job for the bash script that runs it every morning at 2:00AM:

```
GNU nano 6.2 /tmp/crontab.H0Xce0/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
@reboot /home/msmith/MySQLStatusReport.sh
0 2 * * * /home/msmith/GFSDDataBackupScript.sh_

[ Wrote 31 lines ]
G Help      O Write Out  W Where Is    K Cut         T Execute     C Location    M-U Undo
X Exit      R Read File   ~ Replace    ~ Paste       J Justify    ~ Go To Line M-E Redo
```

Screenshot confirming that my bash script is set up to execute every morning at 2:00AM:

```
crontab: installing new crontab
msmith@wbms01:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
@reboot /home/msmith/MySQLStatusReport.sh
0 2 * * * /home/msmith/GFSDDataBackupScript.sh
msmith@wbms01:~$
```



## Screenshot of changing the cron job to execute in the next minute for testing:

```
GNU nano 6.2 /tmp/crontab.eLb1B2/crontab *
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
@reboot /home/msmith/MySQLStatusReport.sh
* * * * * /home/msmith/GFSDDataBackupScript.sh

G Help      O Write Out  H Where Is   K Cut        T Execute    C Location   M-U Undo
X Exit      R Read File  N Replace    U Paste      J Justify    G Go To Line W-E Redo
```

## Screenshots demonstrating cron job successfully backing up system data:

```
msmith@wbms01:~/DataBackupDirectory$ ls
wbms01-Monday.tgz  wbms01-SecondWeekOfMonth.tgz  wbms01-Sunday.tgz
msmith@wbms01:~/DataBackupDirectory$ _
```

## Screenshots of contents in “wbms01-Monday.tgz” file that cron job created:

```
msmith@wbms01:~/DataBackupDirectory$ less wbms01-Monday.tgz
drwxr-xr-x root/root      0 2023-11-02 19:43 home/
drwxr-x--- msmith/msmith  0 2023-11-13 00:37 home/msmith/
-rwxr-xr-x root/root      277 2023-11-09 22:27 home/msmith/BenchmarkReport.sh
drwxrwxr-x msmith/msmith  0 2023-11-12 18:47 home/msmith/MySQLStatusDirectory/
-rw-rw-r-- msmith/msmith   1 2023-11-12 18:47 home/msmith/MySQLStatusDirectory/MySQLStatusReport-2
023-11-12_18-47-37.txt
-rw-r--r-- root/root      140 2023-11-10 15:49 home/msmith/MySQLStatusDirectory/MySQLStatusReport-2
023-11-10_15-49-28.txt
-rwxr-xr-x root/root      334 2023-11-10 15:44 home/msmith/MySQLStatusReport.sh
drwxrwxr-x msmith/msmith  0 2023-11-13 00:01 home/msmith/BenchmarkReportDirectory/
-rw-rw-r-- msmith/msmith  976 2023-11-09 22:46 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-09_22-46-21.txt
-rw-r--r-- root/root      808 2023-11-09 22:43 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-09_22-43-11.txt
-rw-rw-r-- msmith/msmith  976 2023-11-09 22:42 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-09_22-42-36.txt
-rw-rw-r-- msmith/msmith  970 2023-11-13 00:01 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-13_00-01-01.txt
-rw-rw-r-- msmith/msmith  809 2023-11-09 22:42 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-09_22-42-36.txt
-rw-rw-r-- msmith/msmith  809 2023-11-13 00:01 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-13_00-01-01.txt
-rw-r--r-- root/root      809 2023-11-10 14:00 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-10_14-00-48.txt
-rw-r--r-- root/root      976 2023-11-10 14:01 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-10_14-00-48.txt
-rw-rw-r-- msmith/msmith  809 2023-11-09 22:46 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-09_22-46-21.txt
-rw-r--r-- root/root      976 2023-11-09 22:43 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-09_22-43-11.txt
-rw-r--r-- msmith/msmith  3771 2022-01-06 16:23 home/msmith/.bashrc
drwxrwxr-x msmith/msmith  0 2023-11-13 01:22 home/msmith/DataBackupDirectory/
-rw-r--r-- root/root 497980604 2023-11-12 22:03 home/msmith/DataBackupDirectory/wbms01-Sunday.tgz
```

Screenshot of contents in “wbms01-SecondWeekOfMonth.tgz” file that cron job created:

```
drwxr-xr-x root/root      0 2023-11-02 19:43 home/
drwxr-xr-x msmith/msmith  0 2023-11-12 21:22 home/msmith/
-rwxr-xr-x root/root     277 2023-11-09 22:27 home/msmith/BenchmarkReport.sh
drwxrwxr-x msmith/msmith  0 2023-11-12 18:47 home/msmith/MySQLStatusDirectory/
-rw-rw-r-- msmith/msmith   1 2023-11-12 18:47 home/msmith/MySQLStatusDirectory/MySQLStatusReport-2
023-11-12_18-47-37.txt
-rw-r--r-- root/root     140 2023-11-10 15:49 home/msmith/MySQLStatusDirectory/MySQLStatusReport-2
023-11-10_15-49-28.txt
-rwxr-xr-x root/root     334 2023-11-10 15:44 home/msmith/MySQLStatusReport.sh
drwxrwxr-x msmith/msmith  0 2023-11-10 15:20 home/msmith/BenchmarkReportDirectory/
-rw-rw-r-- msmith/msmith  976 2023-11-09 22:46 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-09_22-46-21.txt
-rw-r--r-- root/root     808 2023-11-09 22:43 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-09_22-43-11.txt
-rw-rw-r-- msmith/msmith  976 2023-11-09 22:42 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-09_22-42-36.txt
-rw-rw-r-- msmith/msmith  809 2023-11-09 22:42 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-09_22-42-36.txt
-rw-r--r-- root/root     809 2023-11-10 14:00 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-10_14-00-48.txt
-rw-r--r-- root/root     976 2023-11-10 14:01 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-10_14-00-48.txt
-rw-rw-r-- msmith/msmith  809 2023-11-09 22:46 home/msmith/BenchmarkReportDirectory/CPUBenchmarkRep
ort-2023-11-09_22-46-21.txt
-rw-r--r-- root/root     976 2023-11-09 22:43 home/msmith/BenchmarkReportDirectory/MemoryBenchmark
Report-2023-11-09_22-43-11.txt
-rw-r--r-- msmith/msmith 3771 2022-01-06 16:23 home/msmith/.bashrc
drwxrwxr-x msmith/msmith  0 2023-11-12 21:39 home/msmith/DataBackupDirectory/
-rw-rw-r-- msmith/msmith  0 2023-11-12 21:41 home/msmith/DataBackupDirectory/wbms01-SecondWeekOfM
onth.tgz
drwx----- mysql/mysql  0 2023-10-20 17:01 home/msmith/Data/
drwxr-xr-x mysql/mysql  0 2023-10-12 23:09 home/msmith/Data/inet/
-rw-r----- mysql/mysql 114688 2023-10-12 23:31 home/msmith/Data/inet/members.ibd
drwxr-xr-x mysql/mysql  0 2023-10-20 11:48 home/msmith/Data/#innodb_redo/
-rw-r----- mysql/mysql 3276800 2023-10-20 11:48 home/msmith/Data/#innodb_redo/#ib_redo26_tmp
-rw-r----- mysql/mysql 3276800 2023-10-20 11:48 home/msmith/Data/#innodb_redo/#ib_redo29_tmp
wbms01-SecondWeekOfMonth.tgz
```

Step 3. On reboot check and output the status of DBMS to file.

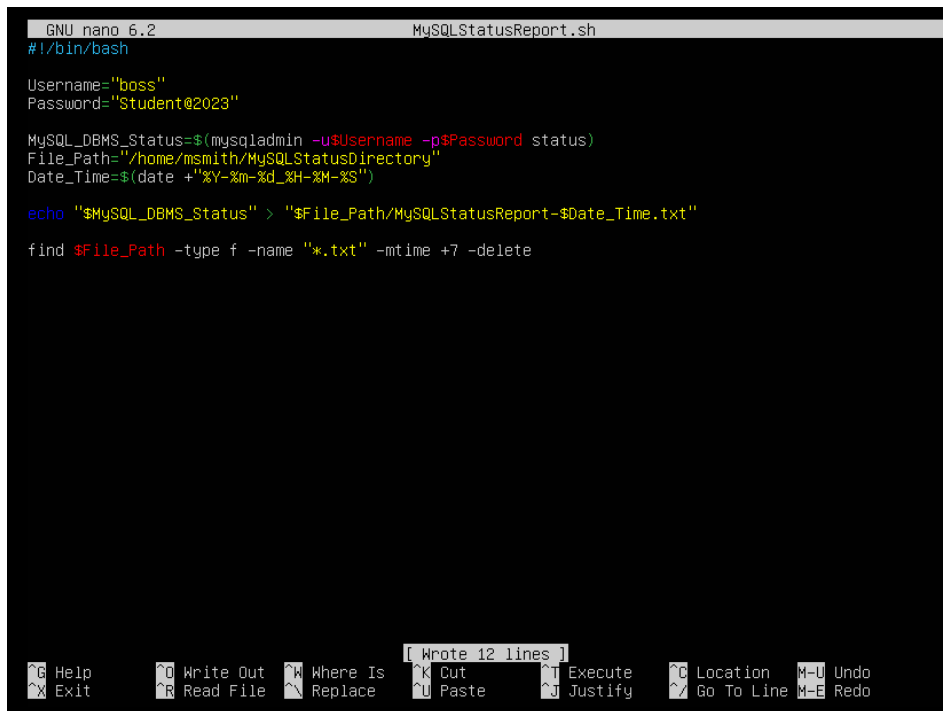
Screenshot of creating “MySQLStatusDirectory” directory:

```
msmith@wbms01:~$ mkdir MySQLStatusDirectory
msmith@wbms01:~$ ls
BenchmarkReportDirectory  Data  MySQLBackupDirectory  restore_file.sql
BenchmarkReport.sh       inet3700_bash_script.sh  MySQLStatusDirectory
msmith@wbms01:~$ _
```

Screenshot of creating “MySQLStatusReport.sh” bash script:

```
msmith@wbms01:~$ sudo nano MySQLStatusReport.sh
[sudo] password for msmith:
```

Screenshot of my created bash script:

A screenshot of a terminal window showing the nano 6.2 text editor editing a file named MySQLStatusReport.sh. The script content is as follows:

```
#!/bin/bash

Username="boss"
Password="Student@2023"

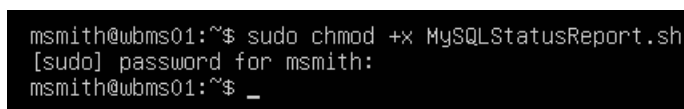
MySQL_DBMS_Status=$(mysqladmin -u$Username -p$Password status)
File_Path="/home/msmith/MySQLStatusDirectory"
Date_Time=$(date +%Y-%m-%d_%H-%M-%S)

echo "$MySQL_DBMS_Status" > "$File_Path/MySQLStatusReport-$Date_Time.txt"

find $File_Path -type f -name "*.txt" -mtime +7 -delete
```


The bottom of the window shows the nano editor's help menu with options like Help, Exit, Write Out, Read File, Where Is, Replace, Cut, Paste, Execute, Justify, Location, Go To Line, Undo, and Redo. A status bar indicates "[ Wrote 12 lines ]".

Screenshot of giving the bash script executable permissions:

A screenshot of a terminal window showing the following commands and output:

```
msmith@wbms01:~$ sudo chmod +x MySQLStatusReport.sh
[sudo] password for msmith:
msmith@wbms01:~$ _
```

Screenshot of script successfully executed after using “sudo bash MySQLStatusReport.sh” command:

A screenshot of a terminal window showing the following commands and output:

```
msmith@wbms01:~$ cd /home/msmith/MySQLStatusDirectory
msmith@wbms01:~/MySQLStatusDirectory$ ls
MySQLStatusReport-2023-11-10_15-49-28.txt
msmith@wbms01:~/MySQLStatusDirectory$
```

**Screenshot of modifying the cron job scheduler utility to automatically check MySQL status every time system boots / reboots:**

```
GNU nano 6.2 /tmp/crontab.i4C3mo/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
@reboot /home/msmith/MySQLStatusReport.sh

[ Wrote 29 lines ]
Help Write Out Where Is Cut Execute Location M-U Undo
Exit Read File Replace Paste Justify Go To Line M-E Redo
```

**Screenshot confirming that my bash script is set up to execute every time my system reboots / boots:**

```
msmith@wbms01:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
@reboot /home/msmith/MySQLStatusReport.sh
```

**Screenshot of changing the cron job to execute in the next minute for testing:**

```
GNU nano 6.2 /tmp/crontab.UzUXk1/crontab.*
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
* * * * *_/home/msmith/MySQLStatusReport.sh
0 2 * * * /home/msmith/GFSDDataBackupScript.sh

^G Help      ^O Write Out  ^H Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_ Go To Line  M-E Redo
```

**Screenshot of cron job successfully creating a file for MySQL Status Report:**

```
msmith@wbms01:~/MySQLStatusDirectory$ ls
MySQLStatusReport-2023-11-10_15-49-28.txt  MySQLStatusReport-2023-11-13_01-42-20.txt
MySQLStatusReport-2023-11-12_18-47-37.txt
msmith@wbms01:~/MySQLStatusDirectory$
```

**Screenshot of contents in “MySQLStatusReport-2023-11-13\_01-42-20.txt” file that cron job created:**

```
msmith@wbms01:~/MySQLStatusDirectory$ cat MySQLStatusReport-2023-11-13_01-42-20.txt
Uptime: 24867 Threads: 2 Questions: 2729 Slow queries: 0 Opens: 7032 Flush tables: 3 Open tabl
es: 439 Queries per second avg: 0.109
msmith@wbms01:~/MySQLStatusDirectory$
```

Step 4. Every Monday at 12:01am create a benchmark report and save it for review with the date/time as part of the name.

**Screenshot of installing sysbench:**

```
msmith@wbms01:~$ sudo apt install sysbench
[sudo] password for msmith:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  liblua5.1-2 liblua5.1-common libmysqlclient21 libpq5
The following NEW packages will be installed:
  liblua5.1-2 liblua5.1-common libmysqlclient21 libpq5 sysbench
0 upgraded, 5 newly installed, 0 to remove and 23 not upgraded.
Need to get 1,845 kB of archives.
After this operation, 8,435 kB of additional disk space will be used.
Do you want to continue? [Y/n] _
```

### Screenshot demonstrating functionality of checking CPU statistics:

```
msmith@woms01:~$ sudo sysbench --test=cpu --cpu-max-prime=20000 run
[sudo] password for msmith:
WARNING: the --test option is deprecated. You can pass a script name or path on the command line without any options.
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
  events per second:   94.96

General statistics:
  total time:          10.0071s
  total number of events: 951

Latency (ms):
  min:                 9.89
  avg:                 10.52
  max:                 24.72
  95th percentile:    11.65
  sum:                 10000.64

Threads fairness:
  events (avg/stddev): 951.0000/0.00
  execution time (avg/stddev): 10.0006/0.00
```

### Screenshot demonstrating sysbench functionality of checking memory statistics (command “sudo sysbench –test=memory run” command was used):

```
Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Running memory speed test with the following options:
  block size: 1KiB
  total size: 102400MiB
  operation: write
  scope: global

Initializing worker threads...

Threads started!

Total operations: 12463059 (1245343.94 per second)

12170.96 MiB transferred (1216.16 MiB/sec)

General statistics:
  total time:          10.0001s
  total number of events: 12463059

Latency (ms):
  min:                 0.00
  avg:                 0.00
  max:                 8.34
  95th percentile:    0.00
  sum:                 4585.26

Threads fairness:
  events (avg/stddev): 12463059.0000/0.00
  execution time (avg/stddev): 4.5853/0.00
```

Screenshot of creating a bash script titled "BenchmarkReport.sh":

```
msmith@wbms01:~$ sudo nano BenchmarkReport.sh
```

Screenshot of my created bash script:

```
GNU nano 6.2 BenchmarkReport.sh *
#!/bin/bash

File_Path="/home/msmith/BenchmarkReportDirectory"
Date_Time=$(date +%Y-%m-%d_%H-%M-%S)

sysbench --test=cpu --cpu-max-prime=20000 run > "$File_Path/CPUBenchmarkReport-$Date_Time.txt"
sysbench --test=memory run > "$File_Path/MemoryBenchmarkReport-$Date_Time.txt" _
```

GNU nano 6.2 BenchmarkReport.sh \*  
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo  
^X Exit ^R Read File ^N Replace ^U Paste ^J Justify ^\_ Go To Line M-E Redo

Screenshot of giving the bash script executable permissions:

```
msmith@wbms01:~$ sudo chmod +x BenchmarkReport.sh
[sudo] password for msmith: _
```

Screenshot of script successfully executing after using "sudo bash BenchmarkReport.sh" command to run script:

```
msmith@wbms01:~$ cd /home/msmith/BenchmarkReportDirectory
msmith@wbms01:~/BenchmarkReportDirectory$ ls
CPUBenchmarkReport-2023-11-09_22-42-36.txt  MemoryBenchmarkReport-2023-11-09_22-42-36.txt
CPUBenchmarkReport-2023-11-09_22-43-11.txt  MemoryBenchmarkReport-2023-11-09_22-43-11.txt
CPUBenchmarkReport-2023-11-09_22-46-21.txt  MemoryBenchmarkReport-2023-11-09_22-46-21.txt
CPUBenchmarkReport-2023-11-10_14-00-48.txt  MemoryBenchmarkReport-2023-11-10_14-00-48.txt
msmith@wbms01:~/BenchmarkReportDirectory$
```

## Screenshot of modifying the cron job scheduler utility to automatically run the bash script every Monday at 12:01AM:

```
GNU nano 6.2 /tmp/crontab.azul00/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh_
```

## Screenshot confirming that my bash script is set up to execute every Monday at 12:01AM:

```
msmith@wbms01:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh
1 0 * * 1 /home/msmith/BenchmarkReport.sh
```



Screenshot of changing the cron job to execute in the next minute for testing:

```
GNU nano 6.2 /tmp/crontab.v8aHuu/crontab *
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 22 * * 5 /home/msmith/MySQLBackupDirectory/backup.sh

* * * * * /home/msmith/BenchmarkReport.sh

@reboot /home/msmith/MySQLStatusReport.sh

0 2 * * * /home/msmith/GFSDataBackupScript.sh

?G Help      ?O Write Out  ?N Where Is   ?K Cut        ?T Execute    ?C Location   M-U Undo
?X Exit      ?R Read File  ?_ Replace    ?U Paste      ?J Justify    ?_ Go To Line  M-E Redo
```

Screenshot demonstrating cron job creating a CPU and Memory Benchmark report:

```
msmith@wbms01:~/BenchmarkReportDirectory$ ls
CPUBenchmarkReport-2023-11-09_22-42-36.txt  MemoryBenchmarkReport-2023-11-09_22-42-36.txt
CPUBenchmarkReport-2023-11-09_22-43-11.txt  MemoryBenchmarkReport-2023-11-09_22-43-11.txt
CPUBenchmarkReport-2023-11-09_22-46-21.txt  MemoryBenchmarkReport-2023-11-09_22-46-21.txt
CPUBenchmarkReport-2023-11-10_14-00-48.txt  MemoryBenchmarkReport-2023-11-10_14-00-48.txt
CPUBenchmarkReport-2023-11-13_00-01-01.txt  MemoryBenchmarkReport-2023-11-13_00-01-01.txt
CPUBenchmarkReport-2023-11-13_01-47-16.txt  MemoryBenchmarkReport-2023-11-13_01-47-16.txt
msmith@wbms01:~/BenchmarkReportDirectory$ _
```

Screenshots of contents in “CPUBenchmarkReport-2023-11-13\_01-47-16.txt” file that the cron job created:

```
msmith@wbms01:~/BenchmarkReportDirectory$ cat CPUBenchmarkReport-2023-11-13_01-47-16.txt
sysbench 1.0.20 (using system LuaJIT 2.1.0-beta3)

Running the test with following options:
Number of threads: 1
Initializing random number generator from current time


Prime numbers limit: 20000

Initializing worker threads...

Threads started!

CPU speed:
  events per second:    95.49

General statistics:
  total time:           10.0028s
  total number of events: 956

Latency (ms):
  min:                  9.98
  avg:                  10.46
  max:                  18.93
  95th percentile:     11.45
  sum:                  9997.38

Threads fairness:
  events (avg/stddev):  956.0000/0.00
  execution time (avg/stddev): 9.9974/0.00

msmith@wbms01:~/BenchmarkReportDirectory$ _
```

**Screenshot of contents in “MemoryBenchmarkReport-2023-11-13\_01-47-16.txt” file that the cron job created:**

```
Running the test with following options:
Number of threads: 1
Initializing random number generator from current time

Running memory speed test with the following options:
  block size: 1KiB
  total size: 102400MiB
  operation: write
  scope: global

Initializing worker threads...

Threads started!

Total operations: 12723731 (1271032.93 per second)

12425.52 MiB transferred (1241.24 MiB/sec)

General statistics:
  total time:                10.0003s
  total number of events:    12723731

Latency (ms):
  min:                       0.00
  avg:                       0.00
  max:                       3.13
  95th percentile:          0.00
  sum:                       4586.04

Threads fairness:
  events (avg/stddev):       12723731.0000/0.00
  execution time (avg/stddev): 4.5860/0.00

msmith@wbms01:~/BenchmarkReportDirectory$ _
```

## Task 2: Docker

Assignment questions and / or required screenshots provided below.

Step 1. Demonstrate a simple Installation & Configuration of Docker within the Ubuntu Server previously created. Clearly demonstrate Docker in use.

**Screenshot of updating package lists before beginning installation:**

```
msmith@wbms01:~$ sudo apt update
Hit:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:2 http://ca.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://ca.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://ca.archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
23 packages can be upgraded. Run 'apt list --upgradable' to see them.
msmith@wbms01:~$ _
```

**Screenshot of installing prerequisite packages that allow apt to use packages over HTTPS:**

```
msmith@wbms01:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-commo
n_
```

### Screenshot of adding the GPG key for the implementation of the official Docker repository:

```
msmith@wbms01:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
msmith@wbms01:~$
```

### Screenshot of adding Docker repository to APT sources:

```
msmith@wbms01:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
focal stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable'
Description:
Archive for codename: focal components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-jammy
.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux
_ubuntu-jammy.list
Hit:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:2 http://ca.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://ca.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://ca.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 https://download.docker.com/linux/ubuntu focal InRelease [57.7 kB]
Get:6 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [33.7 kB]
Fetched 91.4 kB in 5s (19.4 kB/s)
Reading package lists... Done
W: https://download.docker.com/linux/ubuntu/dists/focal/InRelease: Key is stored in legacy trusted.g
pg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
msmith@wbms01:~$ _
```

### Screenshot of installing Docker:

```
msmith@wbms01:~$ sudo apt install docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras docker-compose-plugin
  libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin libslirp0 pigz slirp4netns
0 upgraded, 9 newly installed, 0 to remove and 23 not upgraded.
Need to get 114 MB of archives.
After this operation, 409 MB of additional disk space will be used.
Do you want to continue? [Y/n] _
```

### Screenshot demonstrating Docker successfully installed:

```
msmith@wbms01:~$ sudo systemctl status docker
• docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2023-11-08 15:54:43 UTC; 1min 26s ago
   TriggeredBy: • docker.socket
     Docs: https://docs.docker.com
    Main PID: 3793 (dockerd)
      Tasks: 7
     Memory: 26.3M
        CPU: 1.592s
    CGroup: /system.slice/docker.service
            └─3793 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Nov 08 15:54:40 wbms01 systemd[1]: Starting Docker Application Container Engine...
Nov 08 15:54:40 wbms01 dockerd[3793]: time="2023-11-08T15:54:40.893985396Z" level=info msg="Starting"
Nov 08 15:54:40 wbms01 dockerd[3793]: time="2023-11-08T15:54:40.909248587Z" level=info msg="Detecting"
Nov 08 15:54:41 wbms01 dockerd[3793]: time="2023-11-08T15:54:41.860300094Z" level=info msg="Loading"
Nov 08 15:54:42 wbms01 dockerd[3793]: time="2023-11-08T15:54:42.742536540Z" level=info msg="Loading"
Nov 08 15:54:42 wbms01 dockerd[3793]: time="2023-11-08T15:54:42.866083687Z" level=info msg="Docker >
Nov 08 15:54:42 wbms01 dockerd[3793]: time="2023-11-08T15:54:42.867379500Z" level=info msg="Daemon >
Nov 08 15:54:43 wbms01 dockerd[3793]: time="2023-11-08T15:54:43.034668275Z" level=info msg="API lis
Nov 08 15:54:43 wbms01 systemd[1]: Started Docker Application Container Engine.
lines 1-21/21 (END)
```

### Screenshot demonstrating Docker clearly in use:

```
msmith@wbms01:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:88ec0acaa3ec199d3b7eaf73588f4518c25f9d34f58ce9a0df68429c5af48e8d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

msmith@wbms01:~$ _
```

### Screenshot of running a Ubuntu container with Docker:

```
msmith@wbms01:~$ sudo docker run -it ubuntu bash
[sudo] password for msmith:
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
aece8493d397: Pull complete
Digest: sha256:2b7412e6465c3c7fc5bb21d3e6f1917c167358449fecac8176c6e496e5c1f05f
Status: Downloaded newer image for ubuntu:latest
root@d06d4600f5de:/#
```

Step 2. Give 6 examples of what you can see Docker being used for, justify your examples.

The following list describes the various uses for Docker:

1. Docker could be used to develop and deploy application elements in a more efficient manner in Ubuntu server through the concept of microservices, which would separate the elements of large applications into smaller services and allow for them to be managed more effectively.
2. Docker would certainly be used for the purpose of running applications more efficiently in Ubuntu server through the concept of containerization, which would use a number of separated spaces called containers to store and enclose the elements of each individual application for the purpose of allowing each application to operate more efficiently and to prevent each applications elements from interfering with one another.

3. Docker could potentially be used as a testing environment for applications in Ubuntu server due to its ability to isolate applications from one another through containerization, which would allow for applications to be constructed, configured, and tested without potentially interfering with other applications and / or affecting them.
4. Docker could be used for the purpose of deploying applications on ubuntu server in a quick and highly efficient manner due to its features of rapid deployment and automated deployment that stem from the concept of containerization.
5. Docker could be used for the purpose of enhancing the construction and management procedures for applications by encouraging operations and deployment teams to work together through its adoption of the DevOps concept.
6. Docker could be used for the purpose of effectively scaling applications depending on the demands the applications must meet for the tasks a user performs, which is a result of the Docker environment being resource-efficient due to its foundation of containerization.
7. Docker could be used to perform multiple high-level tasks within a single virtual environment without requiring a large number of resources due to the resource-efficient nature of Docker, which is derived from its foundation of containerization.

### Task 3: Documentation

Assignment questions and / or required screenshots provided below.

| Change Management Log |  |  |
|-----------------------|--|--|
| Activity Number       | Action Taken                           | Description of Action  |
| A001                  | System packages updates / upgraded.    | <p>Following command was used to update system packages:</p> <p><b><i>"sudo apt update"</i></b></p> <p>Following command was used to upgrade system packages:</p> <p><b><i>"sudo apt upgrade"</i></b></p>  |
| A002                  | /etc/apache2/apache2.conf file edited. | <p>/etc/apache2/apache2.conf file edited to incorporate the additional text to the end of the file:</p> <p><b><i>"#Disable Trace HTTP Requests – MS<br/>TraceEnable off"</i></b></p> <p>This task was performed using the following command:</p> |

|      |  |   |
|------|--|---|
|      |  | <b><i>"sudo nano /etc/apache2/apache2.conf".</i></b>  |
| A003 | /etc/apache2/apache2.conf file edited. | <p>/etc/apache2/apache2.conf file edited to incorporate the additional text:</p> <p><b><i>"#Hide Server Tokens and Signatures – MS<br/>ServerTokens Prod<br/>ServerSignature Off<br/>&lt;IfModule mod_headers.c&gt;<br/>Header unset Server<br/>Header unset X-powered-By<br/>&lt;/IfModule&gt;"</i></b></p> <p>using the command:</p> <p><b><i>"sudo nano /etc/apache2/apache2.conf".</i></b></p>  |
| A004 | "apachegroup" group created.           | <p>"apachegroup" group created using the following command:</p> <p><b><i>"sudo groupadd apachegroup"</i></b></p>  |
| A005 | "Apacheuser" user created.             | <p>"apachegroup" user created using the following command:</p> <p><b><i>"sudo useradd -d /var/www/ -g apachegroup -s /sbin/nologin apacheuser"</i></b></p> <p>Password for "apacheuser" created using the following command:</p> <p><b><i>"Sudo passwd apacheuser"</i></b></p> <p>Password was set to <b><i>"Student@2023"</i></b> as per assignment requirements.</p>  |
| A006 | /etc/apache2/envvars file edited.      | <p>/etc/apache2/envvars file edited for the purpose of locating the following lines of text:</p> <p><b><i>"Export APACHE_RUN_USER"<br/>"Export APACHE_RUN_GROUP"</i></b></p> <p>This was done using the following command:</p> <p><b><i>"sudo nano /etc/apache2/envvars"</i></b></p> <p>After locating these lines of text in the text editor, the text was altered to display the following text:</p> <p><b><i>"Export APACHE_RUN_USER=apacheuser"<br/>"Export APACHE_RUN_GROUP=apachegroup"</i></b></p> |

|      |  |   |
|------|--|---|
|      |  |   |
| A007 | Applied text editor changes.   | <p>After making various changes to the files detailed in the change management log, I used the following command to apply the changes:</p> <p><b><i>"sudo systemctl restart apache2"</i></b></p>  |
| A008 | "boss" username created for MySQL.   | <p>The following command was used to create the "boss" user:</p> <p><b><i>"CREATE USER 'boss'@'localhost' IDENTIFIED BY 'Student@2023';"</i></b></p>  |
| A009 | All privileges granted for "boss" user.  | <p>The following command was used to grant all privileges to "boss" user:</p> <p><b><i>"GRANT ALL PRIVILEGES ON *.* TO 'boss'@'localhost' WITH GRANT OPTION;"</i></b></p>   |
| A010 | Created "inet" database in MySQL.  | <p>The following command was used to create the "inet" database in MySQL:</p> <p><b><i>"CREATE DATABASE inet;"</i></b></p>  |
| A011 | Created a "members" table in the "inet" database in MySQL and created columns. | <p>The following command was used to create the "members" table and the columns in the "inet" database:</p> <p><b><i>"CREATE TABLE members ( user_id INT(11) AUTO_INCREMENT, username VARCHAR(50), password VARCHAR(50), PRIMARY KEY (user_id) );"</i></b></p>                        |
| A012 | Inserted data entries into columns of "members" table in "inet" database.      | <p>The following commands were used to implement the data entered into the columns of the "members" table in the "inet" database:</p> <p><b><i>"INSERT INTO members (user_id, username, password) VALUES (1, 'Bruce', MD5('student')), (2, 'Charlotte', MD5('student'));"</i></b></p> |
| A013 | Created a bash script to create databases.                                     | <p>Bash script was created to create two databases, the "staging" database and the "production" database. The "staging" database consisted of the "task" table with the following columns and data:</p> <p><b><i>CREATE DATABASE IF NOT EXISTS staging; USE staging;</i></b></p>      |

|      |                                   |  |
|------|-----------------------------------|--|
|      |                                   | <p><b>CREATE TABLE IF NOT EXISTS tasks ( task_id INT AUTO_INCREMENT PRIMARY KEY, title VARCHAR(255) NOT NULL, start_date DATE, due_date DATE, status TINYINT NOT NULL, priority TINYINT NOT NULL, description TEXT );</b></p> <p><b>INSERT INTO tasks (title, start_date, due_date, status, priority, description) VALUES ('task1', '2020-07-01', '2020-07-31', 1, 1, 'this is the first task'), ('task2', '2020-08-01', '2020-08-31', 2, 2, 'this is the second task'), ('task3', '2020-09-01', '2020-09-30', 1, 1, 'this is the third task'), ('task4', '2020-10-01', '2020-10-31', 1, 1, 'this is fourth task');</b></p> <p>The “production” database consisted of the “completed” table with the following columns and data:</p> <p><b>CREATE DATABASE IF NOT EXISTS production; USE production; CREATE TABLE IF NOT EXISTS completed ( task_id INT AUTO_INCREMENT PRIMARY KEY, task_name VARCHAR(255) NOT NULL, finished_date DATE, status TEXT, description TEXT );</b></p> <p><b>USE production; INSERT INTO completed (task_name, finished_date, status, description) VALUES ('task1', '2020-07-31', 'done', 'task one finished'), ('task2', '2020-08-31', 'completed', 'task two finished'), ('task3', '2020-09-30', 'done', 'task three finished'), ('task4', '2020-10-31', 'done', 'task four finished');</b></p> |
| A014 | Package lists updated.            | <p>The following command was used to update the package lists:</p> <p><b>“sudo apt-get update”</b></p>   |
| A015 | ACL installed onto ubuntu server. | <p>The following command was used to install ACL onto ubuntu server:</p> <p><b>“sudo apt-get install acl”</b></p>  |
| A016 | Created the “Data” directory.     | <p>The following command was used to create the “Data” directory:</p>  |



|      |   |  |
|------|---|--|
|      |   | <b>"mkdir Data"</b>  |
| A017 | Assigned "rw" permissions for my "msmith" user for Data directory.                          | The following command was used to assign "rw" permissions to my "msmith" user for my "Data" directory:<br><br><b>"setfacl -m u:msmith:rw Data"</b>   |
| A018 | Created "ACLpermissiontest" file in "Data" directory.                                       | The following command was used to Create the "ACLpermissiontest" file in "Data" directory:<br><br><b>"~/Data\$ touch ACLpermissiontest"</b>  |
| A019 | Backed up MySQL Databases to a backup file called "restore_file.sql"                        | The following commands were used to back up the MySQL Databases to a backup file called "restore_file.sql":<br><br><b>"mysqldump -u boss -p inet &gt; restore_file.sql"</b><br><b>"mysqldump -u boss -p mysql &gt; restore_file.sql"</b><br><b>"mysqldump -u boss -p production &gt; restore_file.sql"</b><br><b>"mysqldump -u boss -p staging &gt; restore_file.sql"</b><br><b>"mysqldump -u boss -p sys &gt; restore_file.sql"</b> |
| A020 | Set ownership / permissions for the "msmith" user for the "Data" directory.                 | The following commands were used to set ownership / permissions for the "msmith" user for the "Data" directory:<br><br><b>"Sudo chown -R msmith:msmith /home/msmith/Data"</b><br><b>"sudo chmod 750 /home/msmith/Data"</b>   |
| A021 | MySQL data copied to "Data" directory.  | The following command was used to copy the MySQL data to the "Data" directory:<br><br><b>"sudor sync -av /var/lib/mysql/ /home/msmith/Data"</b>  |
| A022 | Edited MySQL Configuration file.  | The following lines of code were implemented into the MySQL configuration file:<br><br><b>"Socket=/home/msmith/Data/mysqld.sock"</b><br><b>"datadir=/home/msmith/Data"</b>   |
| A023 | Edited MySQL Client Configuration file.   | The following lines of code were implemented into the MySQL client configuration file:<br><br><b>"[client]"</b><br><b>"Socket=/home/msmith/Data/mysql.sock"</b>  |
| A024 | Edited "/etc/rsyslog.conf" text file to provide UDP and TCP transport reception on port 514 | The following lines of code were added to the "/etc/rsyslog.conf" text file to provide UDP and TCP transport reception on port 514:  |

|      |   |   |
|------|---|---|
|      |   | <b><code>"\$ModLoad imudp"</code></b><br><b><code>"\$UDPServerRun 514"</code></b><br><b><code>"\$ModLoad imtcp"</code></b><br><b><code>"\$Input TCPServerRun 514"</code></b>  |
| A025 | Enabled UFW.  | The following command was used to enable UFW:<br><br><b><code>"sudo ufw enable"</code></b>  |
| A026 | Configured TCP/UDP ports to receive incoming traffic                                  | The following commands were used to configure TCP/UDP ports to receive incoming traffic:<br><br><b><code>"sudo ufw allow 514/udp"</code></b><br><b><code>"sudo ufw allow 514/tcp"</code></b>  |
| A027 | Installed libuser package.  | The following command was used to install the libuser package:<br><br><b><code>"sudo apt install libuser"</code></b>  |
| A028 | Installed members package   | The following command was used to install the members package:<br><br><b><code>"sudo apt install members"</code></b>  |
| A029 | Created Directory to back up MySQL Databases  | Created a directory called "MySQLBackupDirectory" to back up MySQL databases. The following command was used to create this directory:<br><br><b><code>"Mkdir MySQLBackupDirectory"</code></b>  |
| A030 | Created a bash script to back up databases and gave it executable permissions.        | Created a bash script called "backup.sh" to back up databases. The following commands were used to create this script and give it executable permissions:<br><br><b><code>"sudo nano backup.sh"</code></b><br><b><code>"sudo chmod +x backup.sh"</code></b>                                   |
| A031 | Modified cron job scheduler utility to run "backup.sh" script every Friday at 10:00PM | Modified cron job scheduler utility to run "backup.sh" script every Friday at 10:00PM using the <b><code>"crontab -e"</code></b> command. The following line of code was implemented to execute this:<br><br><b><code>"0 22 * * 5<br/>/home/msmith/MySQLBackupDirectory/backup.sh"</code></b> |
| A032 | Created Directory for grandfather, father, and son backups of system data.            | Created a directory called "DataBackupDirectory" to store grandfather, father, and son backups of system data. The following command was used to create this directory:<br><br><b><code>"Mkdir DataBackupDirectory"</code></b>  |
| A033 | Created a bash script to store grandfather, father, and son                           | Created a bash script called "backup.sh" to store grandfather, father, and son backups of system data.  |

|      |   |  |
|------|---|--|
|      | backups of system data and gave it executable permissions.  | The following commands were used to create this script and give it executable permissions:<br><br><b><i>"sudo nano GFSDDataBackupScript.sh"</i></b><br><b><i>"sudo chmod +x GFSDDataBackupScript.sh"</i></b>   |
| A034 | Modified cron job scheduler utility to run "GFSDDataBackupScript.sh" script every morning at 2:00AM                                       | Modified cron job scheduler utility to run "GFSDDataBackupScript.sh" script every morning at 2:00AM using the <b><i>"crontab -e"</i></b> command. The following line of code was implemented to execute this:<br><br><b><i>"0 2 * * * /home/msmith/GFSDDataBackupScript.sh"</i></b>            |
| A035 | Created a directory to save MySQL status files every time the system reboots  | Created a directory called "MySQLStatusDirectory" to save MySQL status files every time the system reboots. The following command was used to create this:<br><br><b><i>"mkdir MySQLStatusDirectory"</i></b>   |
| A036 | Created a bash script to save the MySQL status to a file and gave it executable permissions.  | Created a bash script called "MySQLStatusReport.sh" to save the MySQL status to a file. The following commands were used to create this script and give it executable permissions:<br><br><b><i>"sudo nano MySQLStatusReport.sh"</i></b><br><b><i>"sudo chmod +x MySQLStatusReport.sh"</i></b> |
| A037 | Modified cron job scheduler utility to run "MySQLStatusReport.sh" script every time the system reboots                                    | Modified cron job scheduler utility to run "MySQLStatusReport.sh" script every time the system reboots using the <b><i>"crontab -e"</i></b> command. The following line of code was implemented to execute this:<br><br><b><i>"@reboot /home/msmith/MySQLStatusReport.sh"</i></b>              |
| A038 | Installed sysbench to create benchmark reports  | Installed sysbench to create benchmark reports. The following command was used to install it:<br><br><b><i>"sudo apt install sysbench"</i></b>   |
| A039 | Created directory to store benchmark report files.  | Created a directory to store CPU and Memory benchmark report files. The following command was used to create it:<br><br><b><i>"mkdir BenchmarkReportDirectory"</i></b>   |
| A040 | Created a bash script to create CPU and Memory benchmark reports and save it to the created directory and gave it executable permissions. | Created a bash script called "BenchmarkReport.sh" to create CPU and Memory benchmark reports and save it to the "BenchmarkReportDirectory" directory. The following commands were used to create this script and give it executable permissions:   |

|      |   |   |
|------|---|---|
|      |   | <b><i>"sudo nano BenchmarkReport.sh"</i></b><br><b><i>"sudo chmod +x BenchmarkReport.sh"</i></b>  |
| A041 | Modified cron job scheduler utility to run "BenchmarkReport.sh" script every Monday at 12:01AM. | Modified cron job scheduler utility to run "BenchmarkReport.sh" script every Monday at 12:01AM. The following command was implemented to execute this:<br><br><b><i>"1 0 * * 1 /home/msmith/BenchmarkReport.sh"</i></b>           |
| A042 | Updated package lists before Docker installation  | Updated package lists before Docker installation. The following command was used to perform this:<br><br><b><i>"sudo apt update"</i></b>  |
| A043 | Installed prerequisite packages that allow apt to use packages over HTTPS                       | Installed prerequisite packages that allow apt to use packages over HTTPS. The following command was used to do this:<br><br><b><i>"Sudo apt install apt-transport-https ca-certificates curl software-properties-common"</i></b> |
| A044 | added the GPG key for the implementation of the official Docker repository                      | added the GPG key for the implementation of the official Docker repository. The following command was used to do this:<br><br><b><i>"curl -fsSL https://download.docker.com/linux/ubuntu/gpg   sudo apt-key add -"</i></b>        |
| A045 | added Docker repository to APT sources.   | added Docker repository to APT sources. The following command was used to do this:<br><br><b><i>"sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"</i></b>                         |
| A046 | Installed Docker  | Installed Docker onto the Ubuntu server. The following command was used to do this:<br><br><b><i>"Sudo apt install docker-ce"</i></b>   |

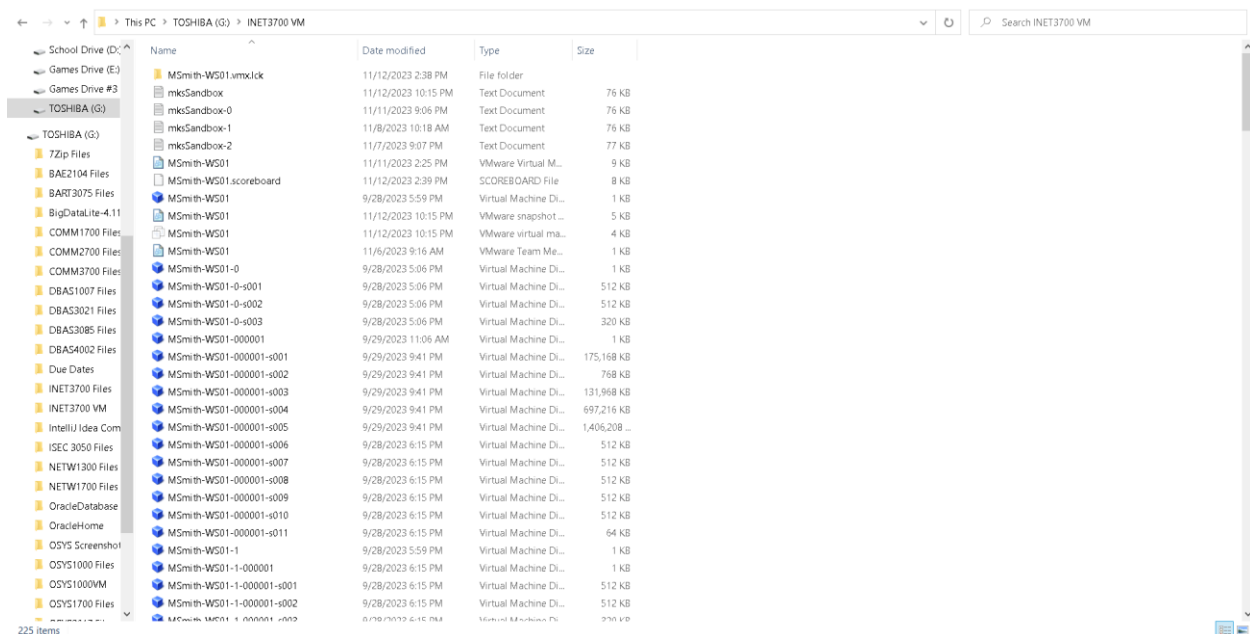
## Task 4: Gold Copy

Proof of Gold Copy backed up onto external drive provided below.

## Screenshot of updated VM Snapshot:



## Screenshot of updated gold copy:



## Conclusion

In this document, I have successfully demonstrated my understanding of the requirements of this assignment pertaining to the topic of automation and docker. I have accomplished this through providing detailed explanations for the required questions, as well as by providing labelled screenshots for the technical requirements.

## References

Website Names, Titles, Authors, Dates, and Links of Articles Used Provided Below.

**Website Name:** The Customize Windows

**Title:** Simple Bash Script For MySQL Database Backup

**Author:** Abhishek Ghosh

**Date:** July 11<sup>th</sup>, 2020

**Link:** [Simple Bash Script For MySQL Database Backup \(thecustomizewindows.com\)](https://thecustomizewindows.com)

**Website Name:** Linux Hint

**Title:** How to Automatically Backup MySQL Database Using Bash Script

**Author:** Abdul Mannan

**Date:** March 2023

**Link:** [How to Automatically Backup MySQL Database Using Bash Script \(linuxhint.com\)](https://linuxhint.com)

**Website Name:** Digital Ocean

**Title:** How To Install and Use Docker on Ubuntu 20.04

**Author:** Brian Hogan

**Date:** May 19<sup>th</sup>, 2020

**Link:** [How To Install and Use Docker on Ubuntu 20.04 | DigitalOcean](https://www.digitalocean.com)

**Website Name:** Simpli Learn

**Title:** Docker Use Cases: Most Common Ways to Use Docker

**Author:** Akshay Badkar

**Date:** July 11<sup>th</sup>, 2023

**Link:** [Docker Use Cases: Most Common Ways to Use Docker \(simplilearn.com\)](https://simplilearn.com)

**Website Name:** Docker Docs

**Title:** Docker overview

**Author:** N/A

**Date:** N/A

**Link:** [Docker overview | Docker Docs](#)

**Website Name:** LinuxConfig.org

**Title:** How to Benchmark Your Linux System

**Author:** Nick Congleton

**Date:** May 29<sup>th</sup>, 2020

**Link:** [How to Benchmark Your Linux System - Linux Tutorials - Learn Linux Configuration](#)

**Website Name:** How To Forge

**Title:** How to Benchmark Your System (CPU, File IO, MySQL) with Sysbench

**Author:** Falko Timme

**Date:** N/A

**Link:** [How to Benchmark Your System \(CPU, File IO, MySQL\) with Sysbench \(howtoforge.com\)](#)

**Website Name:** Computer Hope

**Title:** Linux crontab command

**Author:** N/A

**Date:** November 16<sup>th</sup>, 2019

**Link:** [Linux Crontab Command Help and Examples \(computerhope.com\)](#)

**Website Name:** The Electric Toolbox

**Title:** Get MySQL status information from the command line

**Author:** N/A

**Date:** N/A

**Link:** [Get MySQL status information from the command line - The Electric Toolbox Blog](#)

**Website Name:** Geeks For Geeks

**Title:** echo command in Linux with Examples

**Author:** raghvendra3499

**Date:** July 21<sup>st</sup>, 2023

**Link:** [echo command in Linux with Examples - GeeksforGeeks](#)

**Website Name:** Nix Craft

**Title:** How to run cron job as soon as system reboot

**Author:** Monk

**Date:** December 15<sup>th</sup>, 2021

**Link:** [How to run cron job as soon as system reboot - Linux - nixCraft Linux/Unix Forum](#)

**Website Name:** Tutorials Point

**Title:** Linux tar Command

**Author:** Pradeep Jhuriya

**Date:** November 7<sup>th</sup>, 2023

**Link:** [Linux tar Command \(tutorialspoint.com\)](#)

**Website Name:** Ubuntu

**Title:** Archive rotation shell script

**Author:** N/A

**Date:** July 2023

**Link:** [Archive rotation shell script | Ubuntu](#)

**Website Name:** Ubuntu

**Title:** How to back up using shell scripts

**Author:** N/A

**Date:** July 2023

**Link:** [How to back up using shell scripts | Ubuntu](#)