# Solution M8: Bash Scripts and Automation

One possible solution could be:

1. We create a file named **user\_data.sh** with the following content:

**#!/bin/bash**

**#**

**# user\_data.sh**

**#**

**# Collects user name (first and last) and birth place**

**#**

**echo "Please answer the following questions:"**

**read -p "First name: " FIRST\_NAME**

**read -p "Last name: " LAST\_NAME**

**read -p "Birth place: " BIRTH\_PLACE**

**echo "The information will be stored in /tmp/user\_data.dat"**

**echo "$FIRST\_NAME;$LAST\_NAME;$BIRTH\_PLACE" >> /tmp/user\_data.dat**

1. After three executions we have the following data:

[root@jupiter ~]# **cat /tmp/user\_data.dat**

John;Smith;London

Jane;Hudson;Manchester

Oliver;Stone;New York

1. Then we create a file named **show\_data.sh** with the following structure:

**#!/bin/bash**

**#**

**# show\_data.sh**

**#**

**# Reads a file line by line and displays each row with a prefix**

**#**

**if [ $# -ne 1 ]; then**

**echo "Incorrect set of parameters!"**

**echo "Usage: show\_data.sh file\_to\_read"**

**exit 1**

**fi**

**ROW\_NUM=1**

**echo "File $1 contains:"**

**# The reason for this syntax is that we want no matter are there white spaces**

**# in the line or not to treat it as a whole**

**# For example - if we have line: Oliver;Stone;New York**

**# And we go with the following structure:**

**# …**

**# for row in $(cat $1)**

**# …**

**# Then the output will be:**

**# …**

**# Row #3: Oliver;Stone;New**

**# Row #4: York**

**# …**

**# Instead we want this:**

**# …**

**# Row #3: Oliver;Stone;New York**

**# …**

**# Of course there are other possible solutions :)**

**while read row;**

**do**

**echo "Row #$ROW\_NUM: $row"**

**ROW\_NUM=$(($ROW\_NUM+1))**

**done < $1**

1. Create a new file named **archiver.sh** with the following content:

**#!/bin/bash**

**#**

**# archiver.sh**

**#**

**# Archives a folder to a file**

**#**

**if [ $# -ne 2 ]; then**

**echo "Incorrect amount of parameters!"**

**echo "Usage: archiver.sh /path/to/folder /path/to/file.tar.gz"**

**exit 1**

**fi**

**if [ ! -d $1 ]; then**

**echo "Parameter 1 is not a folder or it does not exists!"**

**echo "Usage: archiver.sh /path/to/folder /path/to/file.tar.gz"**

**exit 2**

**fi**

**if [ -f $2 ]; then**

**echo "Parameter 2 is an existing file and it should be a non-existing one!"**

**echo "Usage: archiver.sh /path/to/folder /path/to/file.tar.gz"**

**exit 3**

**fi**

**tar cvzf $2 $1**

1. We can execute the following steps to solve the task
   1. Database part:

[root@jupiter ~]# **mysql -u root -p**

MariaDB [(none)]> **CREATE DATABASE process\_data;**

MariaDB [(none)]> **USE process\_data;**

MariaDB [process\_data]> **CREATE TABLE processes (observation TIMESTAMP DEFAULT CURRENT\_TIMESTAMP, amount INT);**

MariaDB [process\_data]> **quit**

* 1. Script part:

**#!/bin/bash**

**#**

**# processes.sh**

**#**

**# Get the current amount of running processes and stores it in a database**

**#**

**PROC\_NUM=$(****ps ax --no-header | wc -l)**

**echo "Running processes: $PROC\_NUM"**

**mysql -u root -pPassword1 process\_data -e "INSERT INTO processes (amount) VALUES ($PROC\_NUM)"**

* 1. Test part:
     1. Execute the script several times;
     2. Check the results:

[root@jupiter ~]# **mysql -u root -p**

MariaDB [(none)]> **USE process\_data;**

MariaDB [process\_data]> **SELECT \* FROM processes;**

+---------------------+--------+

| observation | amount |

+---------------------+--------+

| 2017-07-19 11:45:59 | 100 |

| 2017-07-19 11:46:27 | 102 |

| 2017-07-19 11:46:31 | 99 |

| 2017-07-19 11:46:36 | 99 |

| 2017-07-19 11:46:41 | 99 |

+---------------------+--------+

5 rows in set (0.00 sec)

MariaDB [process\_data]> **quit**