



# **File System Report**

SEMESTER: 7

BY:

Sujit Upadhyaya(C30101200033)

BACHELOR OF INFORMATION &  
COMMUNICATION TECHNOLOGY SCHOOL OF  
SCIENCE & TECHNOLOGY  
ASIA e UNIVERSITY

## **ACKNOWLEDGEMENT**

I express my gratitude to VIRINCHI COLLEGE for including provision for minor project in the syllabus of BICT 7<sup>TH</sup> semester. I have learned and enjoyed a lot while working on this report. We would like to express our thankfulness to all those who helped us to complete this report. We are deeply indebted to our supervisor, course facilitator too, lecturer whose support, suggestion, encouragement helped us all the time while working on writing project report.

## **ABSTRACT**

In this report we can learn about different file system like NTFS, exFAT, FAT32 and EXT2/3/4. We get knowledge about file system deeply. This report helps us to get information about different file system and how to format different file system to other in terminal.

## Table of Contents

1. INTRODUCTION TO FILE SYSTEM.....	1
2. WHY FILE SYSTEM IS IMPORTANT?.....	2
3. HOW FILE SYSTEM WORKS?.....	3
4. STRUCTURE OF FILE SYSTEM METADATA.....	4
5. HOW THE OS ACCESS FILE SYSTEM.....	5
6. TYPES OF FILE SYSTEM.....	6
7. DIFFERENCE BETWEEN DIFFERENT TYPES OF FILE SYSTEM.....	10
8. HOW TO FORMAT DIFFERENT FILE SYSTEM TO OTHER.....	12
9. CONCLUSION.....	19
10. REFERENCES.....	20

## Table of Figures

Figure 1: File System works.....	3
Figure 2: How os access file system.....	5
Figure 3: NTFS File System Structure.....	6
Figure 4: FAT and FAT32 File System Structure.....	7
Figure 5: exFAT File System Structure.....	8
Figure 6: EXT File System Structure.....	9
Figure 7: checking usb drive.....	13
Figure 8: unmount and format to fat.....	14
Figure 9: Checking usb drive.....	15
Figure 10: unmount and format to exfat.....	16
Figure 11: checking usb drive.....	17
Figure 12: unmount and format to ext 4.....	18
Figure 13: verifying usb drive.....	18

## **1. INTRODUCTION TO FILE SYSTEM**

The file system is a method and data structure that the operating system uses to determine how data is stored and retrieved. By separating the data into chunks and giving each one a name, the data can be easily isolated and identified. Taking its name from the naming convention of a paper-based data management system, each group of data is referred to as a "file". The structure and logical rules used to name the data groups and their names are called the "file system".

## **2. WHY FILE SYSTEM IS IMPORTANT?**

The file system helps the operating system to manage data and files logically. It allows users to easily access, protect, read, write and modify the data on their storage devices. Instead, data on a disk without the file system will be stored in a messy manner, taking up huge storage devices and making it difficult for users to access and search for the desired files. The file system also makes it easy to find lost files

### 3. HOW FILE SYSTEM WORKS?

A file system indexes all data information on a storage device, including file size, attributes, location, and hierarchy in the directory. The file system also specifies the path to a file through the directory structure with a format.

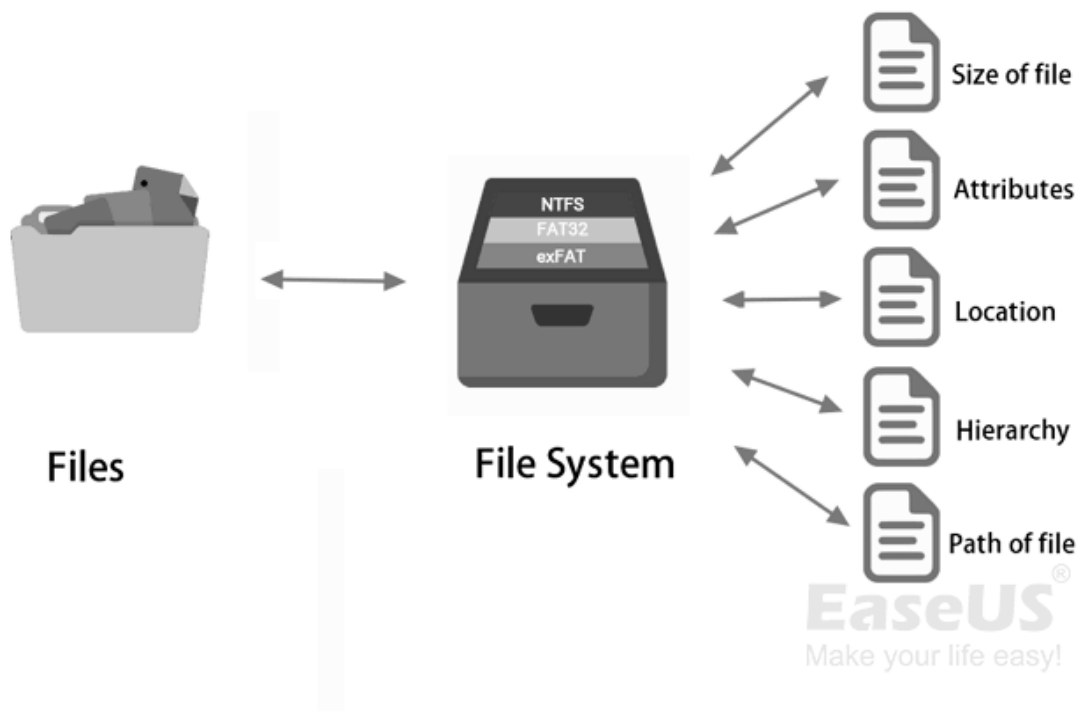


Figure 1: File System works



## **4. STRUCTURE OF FILE SYSTEM METADATA**

All the data information of files on a storage drive is stored in the file system metadata:

- 1) Date created
- 2) Date modified
- 3) Last date of access
- 4) Last backup
- 5) User ID of the file creator
- 6) Access permissions
- 7) File size

## 5. HOW THE OS ACCESS FILE SYSTEM

Here is how the operating system uses the file system to process and access files on a storage device:

- 1) You create a partition on a hard drive or external drive.
- 2) You add a file system format to the drive.
- 3) You store a file in a folder, a directory, or a subdirectory on the drive.
- 4) The file system record the location information of these files.
- 5) OS uses the file system to store and locate these files on your storage devices.

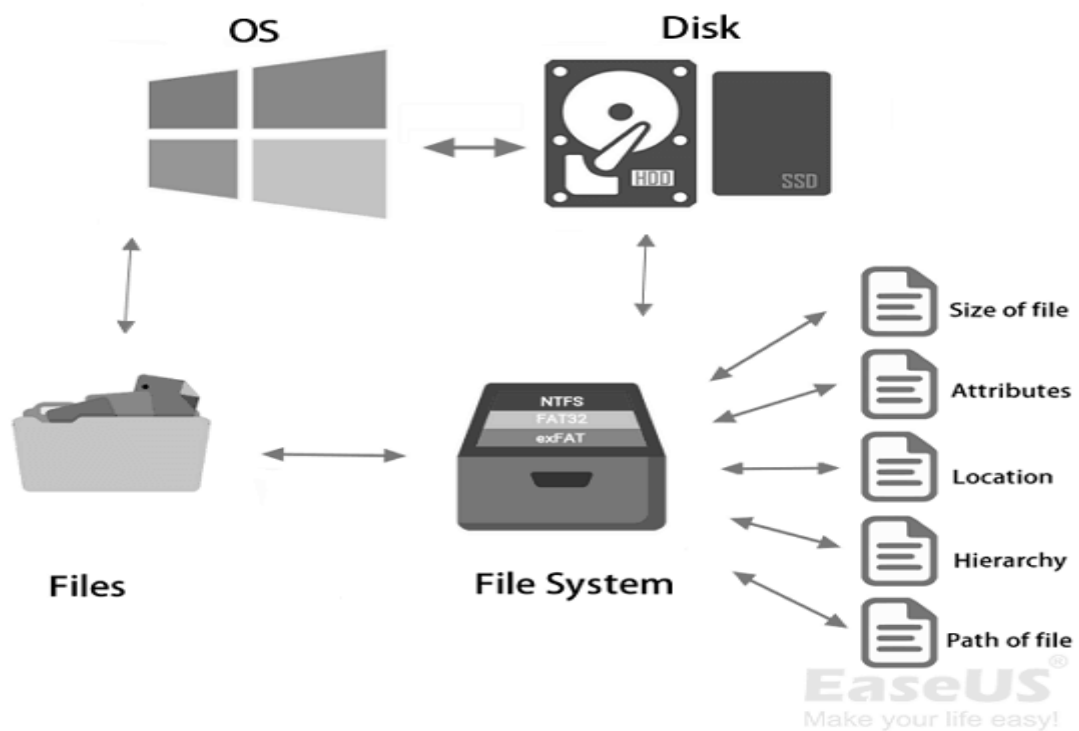


Figure 2: How os access file system

## 6. TYPES OF FILE SYSTEM

- Windows File System: FAT, NTFS, exFAT
- MacOS File System: HFS, APFS, HFS+
- Linux File System: EXT2/3/4, XFS, JFS, Btrfs

### What is NTFS, FAT, exFAT and EXT2/3/4 File System?

#### NTFS:

NTFS, also known as New Technology File System, is a proprietary journaling file system developed by Microsoft. Starting with Windows NT 3.1, this is the default file system of the Windows NT family. It replaced File Allocation Table (FAT) as the preferred file system on Windows and is also supported in Linux and BSD.

It was initial release in 1993 by Tom Miller, Gary Kimura, Brian Andrew, and David Goebel. The last version of this file system is V3.1, (Commonly called NTFS 5.1).

#### NTFS File System Structure:

The NTFS file system has 5 components: O Boot Record , MFT1, MFT Metadata, MFT2, and Data Area.

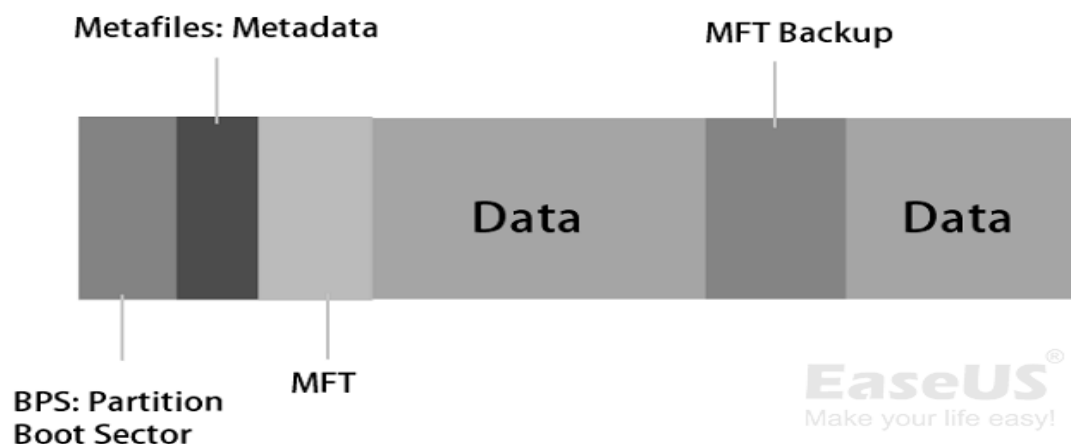


Figure 3: NTFS File System Structure

## FAT or FAT32:

FAT, also known as File Allocation Table, is a file system developed for PC. Originally developed in 1977 for use on floppy disks, it has been adapted for use on hard drives and other devices. It is often supported by today's PC operating systems and many mobile and embedded systems for compatibility reasons, allowing data to be exchanged between different systems.

It was initial release in 1997 by Microsoft, NCR, IBM, Caldera, and more. The last version of this file system was FAT32.

## FAT File System Structure:

The FAT file system consists of four major components: Reserved sectors, FAT Region, Root Directory Region, and Data Region.



**FAT File System Structure**



**FAT32 File System Structure**

**EaseUS**<sup>®</sup>  
Make your life easy!

*Figure 4: FAT and FAT32 File System Structure*

## exFAT:

exFAT is a file system introduced by Microsoft in 2006 and optimized for flash storage devices such as USB drives and SD cards. exFAT was proprietary until August 28, 2019, when Microsoft released its specification.

It was initial release in 2006 by Microsoft. The last version of this file system was exFAT.

### exFAT File System Structure:

The exFAT file system consists of 4 main sections: the Main Boot Region, Backup Boot Region, FAT Region, and Data Region.



### exFAT File System Structure

**EaseUS**<sup>®</sup>  
Make your life easy!

Figure 5: exFAT File System Structure

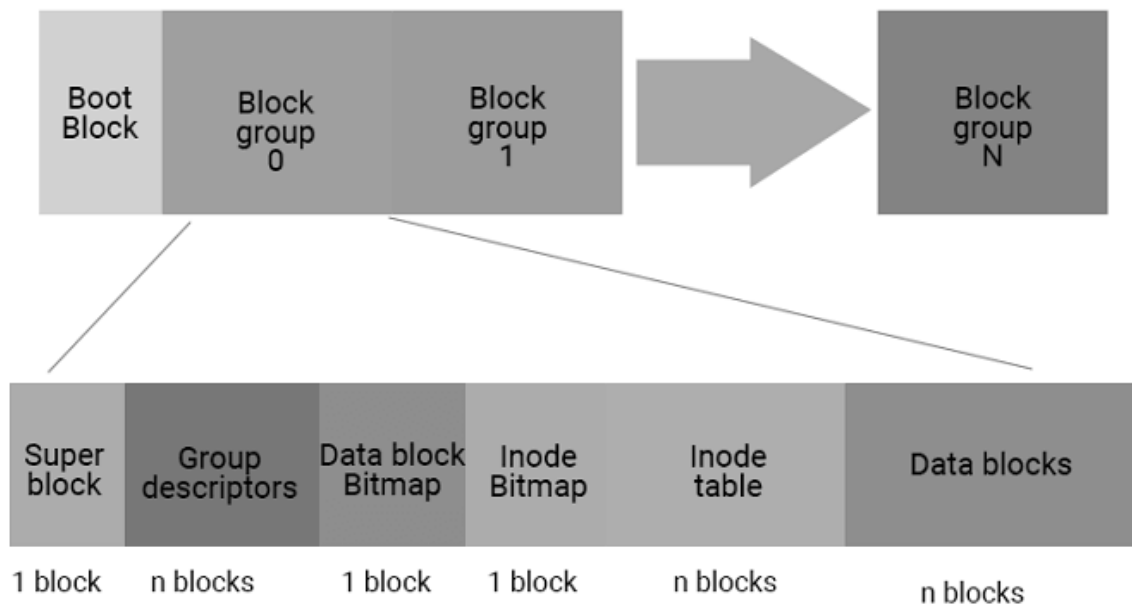
## EXT2/3/4:

EXT was implemented in April 1992 as the first file system built specifically for the Linux kernel. It has a metadata structure inspired by traditional Unix file system principles and was designed by Remy Card to overcome some limitations of the MINIX file system.

It was initial release in 1992 by Remy Card. The last version of this file system is EXT4.

## EXT File System Structure:

The EXT file system can be divided into one block and two groups, including the Boot Block, Block Group 0 (which contains Super Block, Group Descriptors, Data Block Bitmap, Inode Bitmap, Inode Table, and Data Blocks), and Block Group n.



## EXT File System Structure

**EaseUS**<sup>®</sup>  
Make your life easy!

Figure 6: EXT File System Structure

## 7. DIFFERENCE BETWEEN DIFFERENT TYPES OF FILE SYSTEM

Differences	Max File Size	Max Volume Size	Operating System
<b>NTFS</b>	<ul style="list-style-type: none"> <li>• 16EB - 1KB</li> <li>• 16TB - 64KB</li> <li>• 256TB - 64KB</li> <li>• 8PB - 2MB</li> </ul>	<ul style="list-style-type: none"> <li>• 256TB - 64KB</li> <li>• 8PB - 2MB</li> </ul>	<ul style="list-style-type: none"> <li>• Windows NT3.1 and later</li> <li>• macOS X 10.3 and later (Read-only)</li> <li>• Linux kernel 2.6 and later (read-only)</li> <li>• FreeBSD, NetBSD, OpenBSD(read-only), Chrome OS, Solaris, ReactOS(read-only)</li> </ul>
<b>FAT32</b>	<ul style="list-style-type: none"> <li>• 4GB</li> </ul>	<ul style="list-style-type: none"> <li>• 2TB - 512 byte</li> <li>• 8TB - 2KB</li> <li>• 16TB - 4KB</li> </ul>	<ul style="list-style-type: none"> <li>• Windows 95OSR2, Windows 98, XP, 7, 8, 10, and 11.</li> <li>• macOS</li> <li>• Linux</li> </ul>
<b>exFAT</b>	<ul style="list-style-type: none"> <li>• 128 PB</li> </ul>	<ul style="list-style-type: none"> <li>• 128 PB</li> </ul>	<ul style="list-style-type: none"> <li>• Windows XP, Vista, 1/8/10/11, Windows Server 2003/2008/2008 R2</li> <li>• Linux kernal 5.4 and later, FUSE</li> <li>• Mac OS X 6.5 and later</li> </ul>
<b>EXT2/3/4</b>	<ul style="list-style-type: none"> <li>• 4TB - 1KB</li> </ul>	<ul style="list-style-type: none"> <li>• 4TB - 1KB</li> </ul>	<ul style="list-style-type: none"> <li>• Linux kernel 0.96</li> </ul>

- 8TB - 2KB
  - 16TB - 4KB
  - 256PB - 64KB
- 8TB - 2KB
  - 16TB - 4KB
  - 256PB - 64K
- and later



## 8. HOW TO FORMAT DIFFERENT FILE SYSTEM TO OTHER

### NTFS to FAT32:

I am using Linux Operating System so i will do in terminal. Lets go.

Step 1: Open Terminal.

Step 2: Insert pendrive / flash drive.

Step 3: Locate USB Drive using command "df" or "sudo df".

Step 4: Unmount and format Usb drive using command "sudo umount /dev/sdc1"  
and "sudo mkfs.vfat /dev/sdc1".

Step 5: Verify usb drive using command "sudo fsck /dev/sdc1".

```
sonic@SONIC: /media/sonic

(sonic@SONIC)-[/media/sonic]
$ df
Filesystem      1K-blocks    Used Available Use% Mounted on
udev            3979664         0   3979664   0% /dev
tmpfs           803948      1912    802036   1% /run
/dev/sdb2      489634808 92427164 372262136 20% /
tmpfs          4019740         0   4019740   0% /dev/shm
tmpfs           5120         0      5120   0% /run/lock
/dev/loop3      84096      84096         0 100% /snap/whatsdesk/28
/dev/loop4     966784     966784         0 100% /snap/xonotic/64
/dev/loop0      93952      93952         0 100% /snap/gtk-common-themes/1535
/dev/loop1      56960      56960         0 100% /snap/core18/2632
/dev/loop12     50816      50816         0 100% /snap/snapd/17883
/dev/loop2     168832     168832         0 100% /snap/gnome-3-28-1804/161
/dev/loop11    133888     133888         0 100% /snap/drawio/166
/dev/loop6      64768      64768         0 100% /snap/core20/1634
/dev/loop10    150912     150912         0 100% /snap/figma-linux/156
/dev/loop7     224256     224256         0 100% /snap/gnome-3-34-1804/77
/dev/loop9      64768      64768         0 100% /snap/core20/1695
/dev/loop8       128         128         0 100% /snap/bare/5
/dev/loop5      56960      56960         0 100% /snap/core18/2620
/dev/sdb1       523244        160   523084   1% /boot/efi
tmpfs           803948        168   803780   1% /run/user/1000
/dev/sdc1      7862268     39996   7822272   1% /media/sonic/3F2A2CCD42CACAEC
```

Figure 7: checking usb drive

A terminal window titled 'sonic@SONIC: /media/sonic' with search, menu, and window control icons in the title bar. The terminal shows a sequence of commands and their outputs: 1. Command: `sudo umount /dev/sdc1`; Output: `umount: /dev/sdc1: not mounted.` 2. Command: `sudo mkfs.vfat /dev/sdc1`; Output: `mkfs.fat 4.2 (2021-01-31)` 3. Command: `sudo fsck /dev/sdc1`; Output: `fsck from util-linux 2.38.1`, `fsck.fat 4.2 (2021-01-31)`, and `/dev/sdc1: 0 files, 1/1961726 clusters` 4. The prompt `$` is shown at the bottom, indicating the terminal is ready for the next command.

```
(sonic@SONIC)-[/media/sonic]
$ sudo umount /dev/sdc1
umount: /dev/sdc1: not mounted.

(sonic@SONIC)-[/media/sonic]
$ sudo mkfs.vfat /dev/sdc1
mkfs.fat 4.2 (2021-01-31)

(sonic@SONIC)-[/media/sonic]
$ sudo fsck /dev/sdc1
fsck from util-linux 2.38.1
fsck.fat 4.2 (2021-01-31)
/dev/sdc1: 0 files, 1/1961726 clusters

(sonic@SONIC)-[/media/sonic]
$
```

Figure 8: unmount and format to fat

## FAT32 to exFAT:

I am using Linux Operating System so i will do in terminal. Lets go.

Step 1: Open Terminal.

Step 2: Insert pendrive / flash drive.

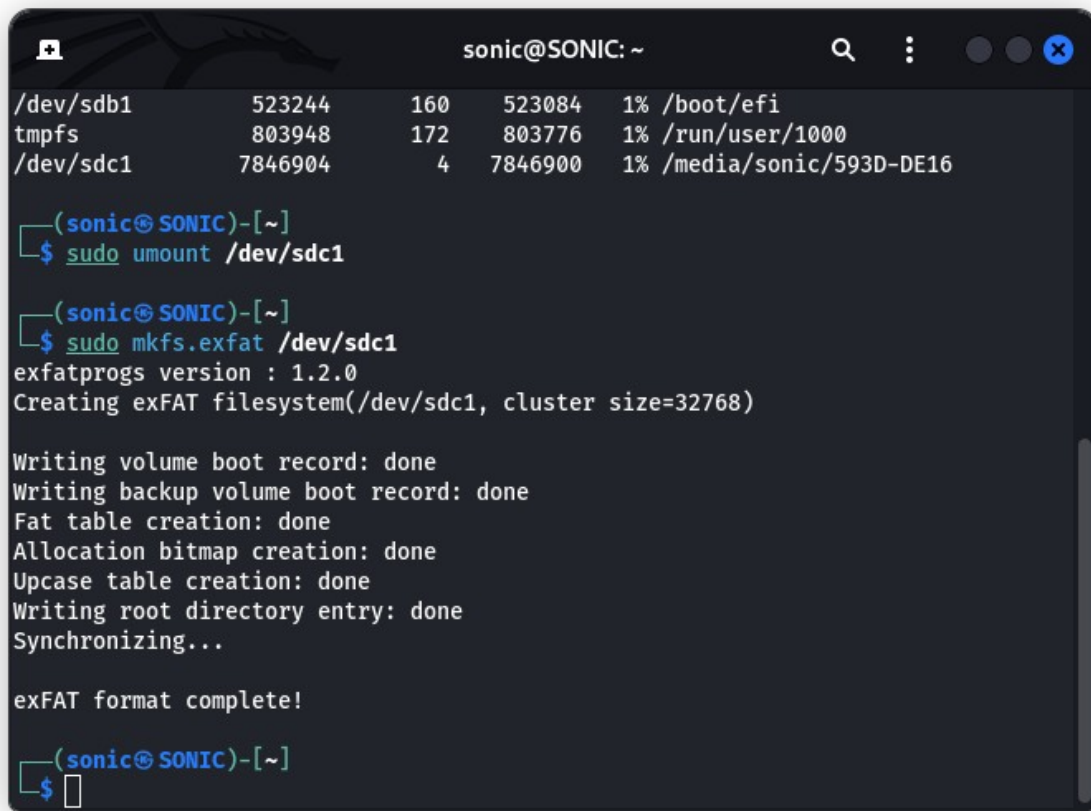
Step 3: Locate USB Drive using command “df” or “sudo df”.

Step 4: Unmount and format Usb drive using command “sudo umount /dev/sdc1” and “sudo mkfs.exfat /dev/sdc1”.

Step 5: Verify usb drive using command ”sudo fsck /dev/sdc1”.

```
(sonic@SONIC)-[~]  
$ df  
Filesystem      1K-blocks    Used Available Use% Mounted on  
udev            3979664         0   3979664   0% /dev  
tmpfs           803948      1924    802024   1% /run  
/dev/sdb2       489634808 92428012 372261288 20% /  
tmpfs          4019740         0   4019740   0% /dev/shm  
tmpfs           5120         0      5120   0% /run/lock  
/dev/loop3       84096      84096         0 100% /snap/whatsdesk/28  
/dev/loop4      966784     966784         0 100% /snap/xonotic/64  
/dev/loop0       93952      93952         0 100% /snap/gtk-common-themes/1535  
/dev/loop1       56960      56960         0 100% /snap/core18/2632  
/dev/loop12      50816      50816         0 100% /snap/snapd/17883  
/dev/loop2     168832    168832         0 100% /snap/gnome-3-28-1804/161  
/dev/loop11     133888    133888         0 100% /snap/drawio/166  
/dev/loop6       64768      64768         0 100% /snap/core20/1634  
/dev/loop10     150912    150912         0 100% /snap/figma-linux/156  
/dev/loop7      224256    224256         0 100% /snap/gnome-3-34-1804/77  
/dev/loop9       64768      64768         0 100% /snap/core20/1695  
/dev/loop8        128        128         0 100% /snap/bare/5  
/dev/loop5       56960      56960         0 100% /snap/core18/2620  
/dev/sdb1       523244        160   523084   1% /boot/efi  
tmpfs           803948        172   803776   1% /run/user/1000  
/dev/sdc1       7846904         4   7846900   1% /media/sonic/593D-DE16
```

Figure 9: Checking usb drive

A terminal window titled 'sonic@SONIC: ~' with standard window controls. It displays the output of the 'df' command, showing three filesystems: /dev/sdb1 (523244 blocks), tmpfs (803948 blocks), and /dev/sdc1 (7846904 blocks). The user then enters 'sudo umount /dev/sdc1' and 'sudo mkfs.exfat /dev/sdc1'. The terminal shows the exfatprogs version (1.2.0) and the creation of an exFAT filesystem on /dev/sdc1 with a cluster size of 32768. It lists several steps: Writing volume boot record, Writing backup volume boot record, Fat table creation, Allocation bitmap creation, Uppercase table creation, Writing root directory entry, and Synchronizing. The process concludes with 'exFAT format complete!'.

```
sonic@SONIC: ~  
df  
/dev/sdb1      523244    160    523084    1% /boot/efi  
tmpfs         803948    172    803776    1% /run/user/1000  
/dev/sdc1     7846904     4   7846900    1% /media/sonic/593D-DE16  
  
(sonic@SONIC)-[~]  
$ sudo umount /dev/sdc1  
  
(sonic@SONIC)-[~]  
$ sudo mkfs.exfat /dev/sdc1  
exfatprogs version : 1.2.0  
Creating exFAT filesystem(/dev/sdc1, cluster size=32768)  
  
Writing volume boot record: done  
Writing backup volume boot record: done  
Fat table creation: done  
Allocation bitmap creation: done  
Uppercase table creation: done  
Writing root directory entry: done  
Synchronizing...  
  
exFAT format complete!  
  
(sonic@SONIC)-[~]  
$
```

Figure 10: unmount and format to exfat

## exFAT to EXT4:

I am using Linux Operating System so i will do in terminal. Lets go.

Step 1: Open Terminal.

Step 2: Insert pendrive / flash drive.

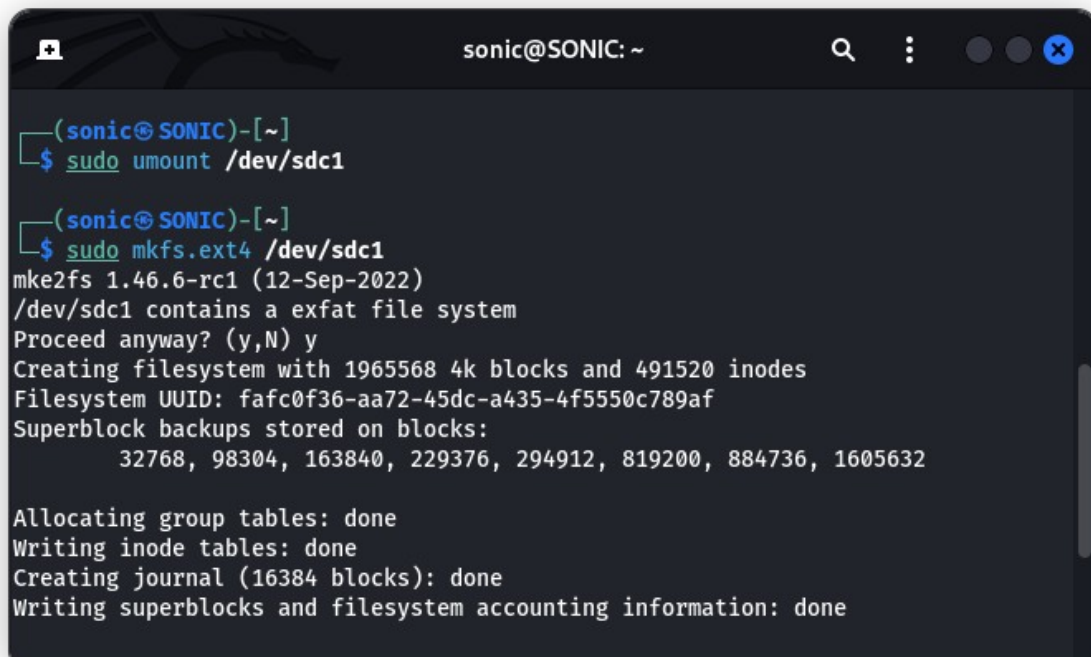
Step 3: Locate USB Drive using command "df" or "sudo df".

Step 4: Unmount and format Usb drive using command "sudo umount /dev/sdc1" and "sudo mkfs.ext4 /dev/sdc1".

Step 5: Verify usb drive using command "sudo fsck /dev/sdc1".

```
sonic@SONIC: ~  
Filesystem      1K-blocks      Used Available Use% Mounted on  
udev            3979664          0    3979664   0% /dev  
tmpfs           803948          1928    802020   1% /run  
/dev/sdb2       489634808 92428676 372260624 20% /  
tmpfs           4019740          0    4019740   0% /dev/shm  
tmpfs           5120            0      5120    0% /run/lock  
/dev/loop3      84096           84096          0 100% /snap/whatsdesk/28  
/dev/loop4      966784          966784          0 100% /snap/xonotic/64  
/dev/loop0      93952           93952          0 100% /snap/gtk-common-themes/1535  
/dev/loop1      56960           56960          0 100% /snap/core18/2632  
/dev/loop12     50816           50816          0 100% /snap/snapd/17883  
/dev/loop2     168832          168832          0 100% /snap/gnome-3-28-1804/161  
/dev/loop11    133888          133888          0 100% /snap/drawio/166  
/dev/loop6      64768           64768          0 100% /snap/core20/1634  
/dev/loop10    150912          150912          0 100% /snap/figma-linux/156  
/dev/loop7     224256          224256          0 100% /snap/gnome-3-34-1804/77  
/dev/loop9      64768           64768          0 100% /snap/core20/1695  
/dev/loop8       128             128          0 100% /snap/bare/5  
/dev/loop5      56960           56960          0 100% /snap/core18/2620  
/dev/sdb1       523244          160    523084   1% /boot/efi  
tmpfs           803948          172    803776   1% /run/user/1000  
/dev/sdc1       7860224          96    7860128   1% /media/sonic/7FAC-F64D  
____(sonic@SONIC)-[~]
```

Figure 11: checking usb drive



```
(sonic@SONIC)-[~]  
$ sudo umount /dev/sdc1  
  
(sonic@SONIC)-[~]  
$ sudo mkfs.ext4 /dev/sdc1  
mke2fs 1.46.6-rc1 (12-Sep-2022)  
/dev/sdc1 contains a exfat file system  
Proceed anyway? (y,N) y  
Creating filesystem with 1965568 4k blocks and 491520 inodes  
Filesystem UUID: fafc0f36-aa72-45dc-a435-4f5550c789af  
Superblock backups stored on blocks:  
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632  
  
Allocating group tables: done  
Writing inode tables: done  
Creating journal (16384 blocks): done  
Writing superblocks and filesystem accounting information: done
```

Figure 12: unmount and format to ext 4



```
(sonic@SONIC)-[~]  
$ sudo fsck /dev/sdc1  
fsck from util-linux 2.38.1  
e2fsck 1.46.6-rc1 (12-Sep-2022)  
/dev/sdc1: clean, 11/491520 files, 55879/1965568 blocks  
  
(sonic@SONIC)-[~]  
$
```

Figure 13: verifying usb drive

## **9. CONCLUSION**

From this report we can get knowledge about file system and we can know how to format drive to different file system. I am using linux operating system so i had used terminal to format the drive to different file system.



## 10. REFERENCES

1. <https://www.techtarget.com/searchwindowsserver/definition/NTFS>
2. <https://www.sweetwater.com/sweetcare/articles/what-fat32/>
3. <https://www.minitool.com/lib/extended-file-allocation-table.html>
4. <https://opensource.com/article/17/5/introduction-ext4-filesystem>
5. <https://www.easeus.com/diskmanager/file-system.html>