

# Organización Lógica De Un Disco Duro

Nicolás A. Ortega

# 1. Hoja De Control Del Documento

Cuadro 1: Documento/Archivo

<b>Fecha Última Modificación</b>	14/10/2021	<b>Versión/Revisión</b>	1.0
<b>Fecha Creación</b>	13/10/2021		
<b>Fecha Finalización</b>	14/10/2021		

Cuadro 2: Registro De Cambios

<b>Versión/Revisión</b>	<b>Página</b>	<b>Descripción</b>
1.0	Todas	Fomato básico documento.

Cuadro 3: Autores Del Documento

<b>Apellidos, Nombre</b>	<b>Curso</b>
Ortega Froysa, Nicolás A.	1

<b>Preparado</b>	<b>Revisado</b>	<b>Aprobado</b>
Ortega Froysa, Nicolás A.		

## 2. Particiones En Windows

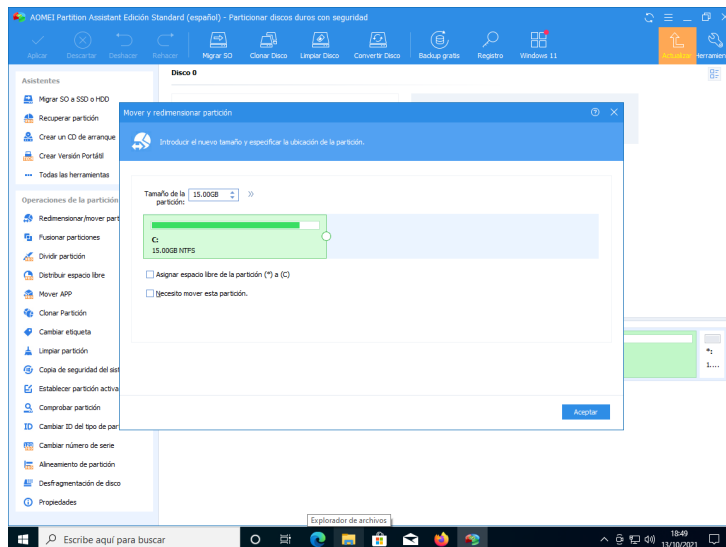


Figura 1: Cambiar tamaño de partición.

Para completar esta tarea, tuve que hacer uso de una máquina virtual de Windows, y decidí usar la herramienta de AOMEI Partition Assistant, ya que tenía más capacidades en su versión gratuita (estándar) que MiniTool.

Al trabajar en una máquina donde el disco virtual entero ya se estaba usando para Windows (volumen C), lo primero que hice fue cambiar el tamaño de aquella partición para tener espacio para trabajar (figura 1). A partir de ahí empecé a crear las nuevas particiones (figura 2). Estas particiones las he creado como particiones primarias, y venía por defecto formateado con NTFS.

A partir de esto, empecé a crear las particiones lógicas. Utilizando el interfaz gráfico pude crear las tres particiones lógicas con las etiquetas **Datos**, **Juegos**, y **Videos** (figura 3). Luego, cambié el formateado de la partición con la etiqueta de **Datos**, de NTFS a FAT32 (figura 4). Esto cuando lo intenté hacer con el MiniTool, era una característica limitada a los programas con licencia profesional, y por eso decidí cambiarme a otra herramienta que tuviera más capacidades en su versión gratuita (i.e. AOMEI Partition Assistant).

Eliminé después la última partición, **Videos** (figura 5), para poder liberar espacio. A partir de ahí pude crear dos particiones lógicas con ese espacio: **ex0** y **ex1**. Éstas las formateé con FAT32 y FAT respectivamente (figura 6).

Luego, intenté fusionar (*merge*) dos particiones – la partición primaria de

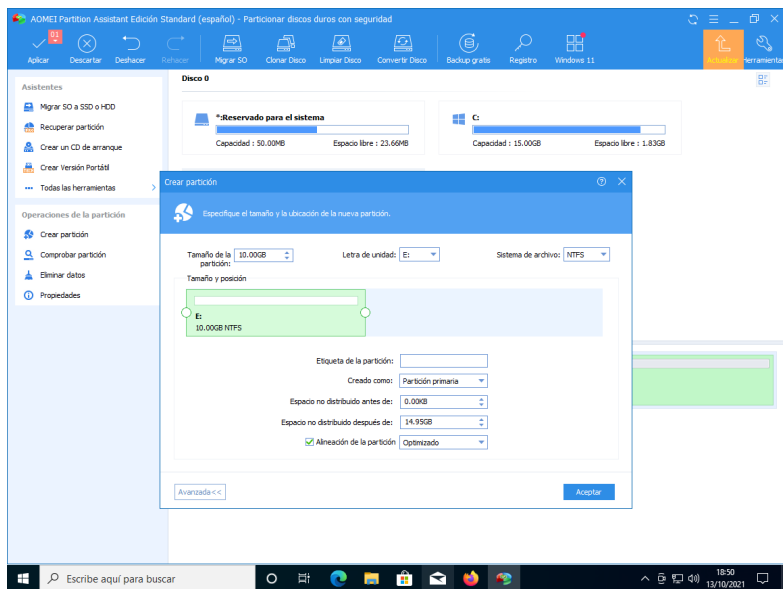


Figura 2: Crear partición primaria.

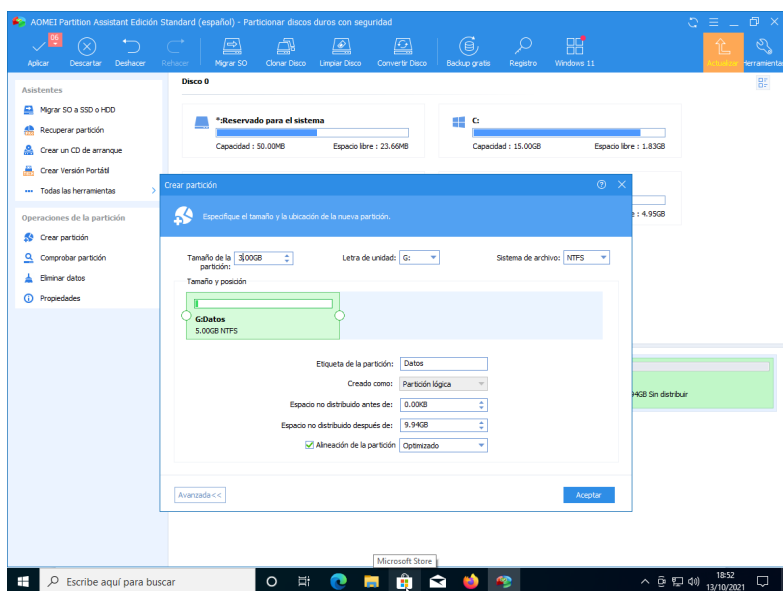


Figura 3: Crear partición lógica.

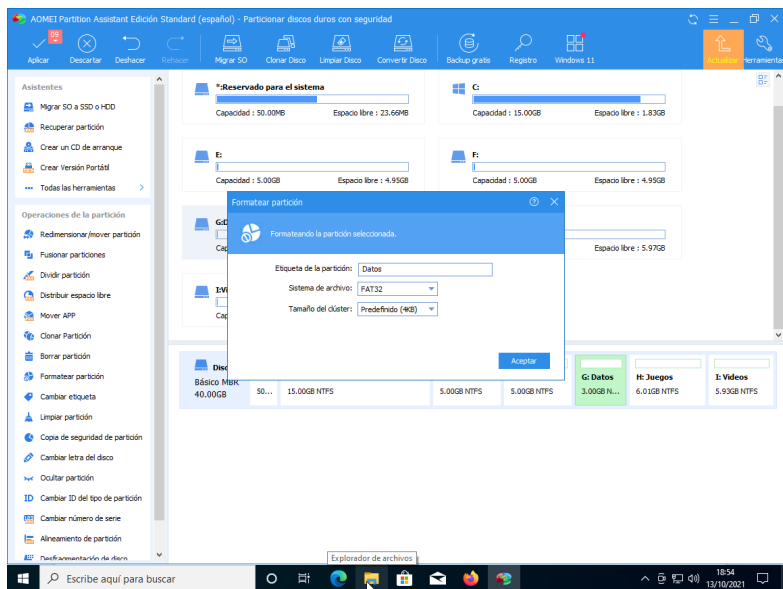


Figura 4: Cambiar formateado de partición.

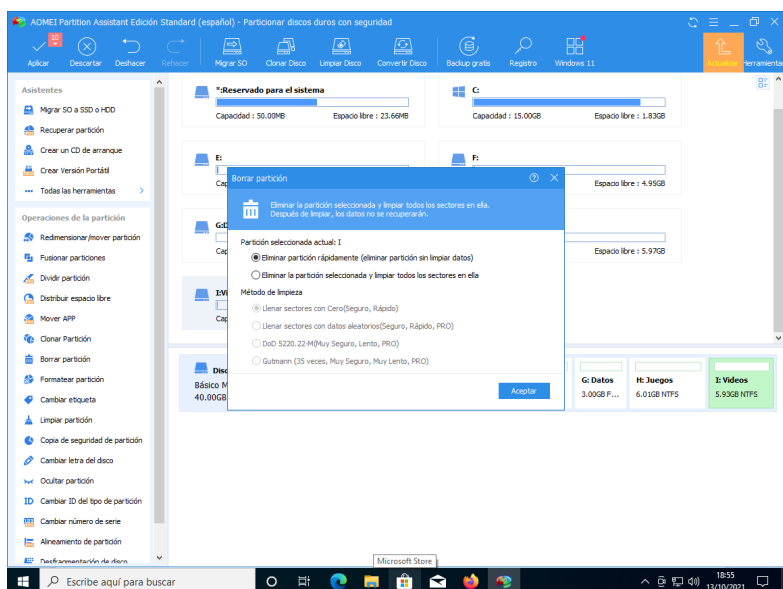


Figura 5: Eliminar partición.

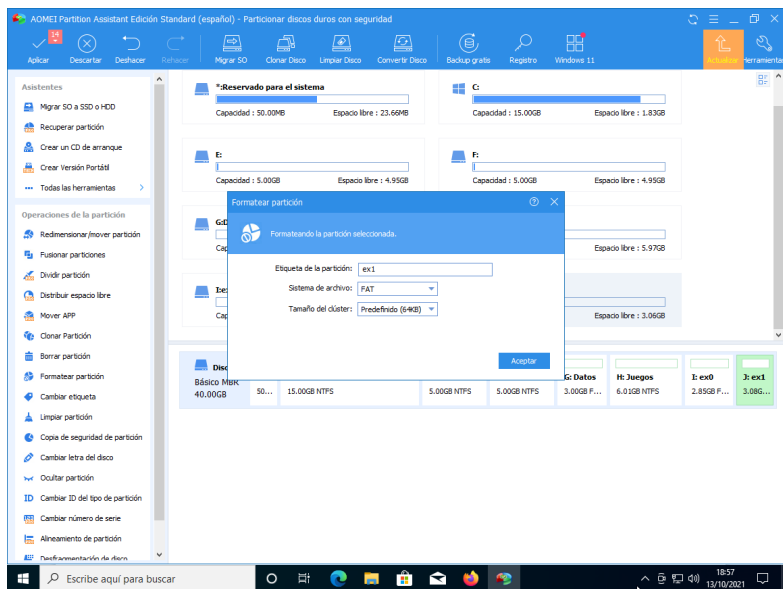


Figura 6: Formatear como FAT.

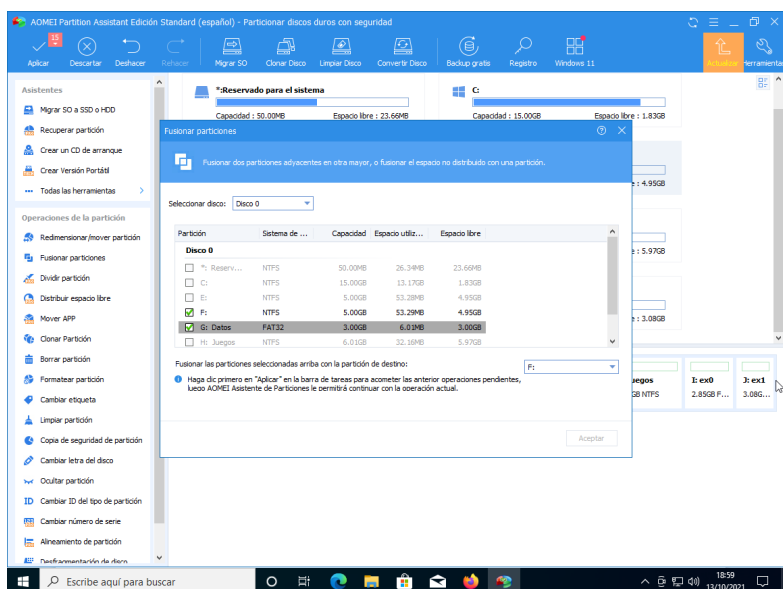


Figura 7: Fusionar particiones.

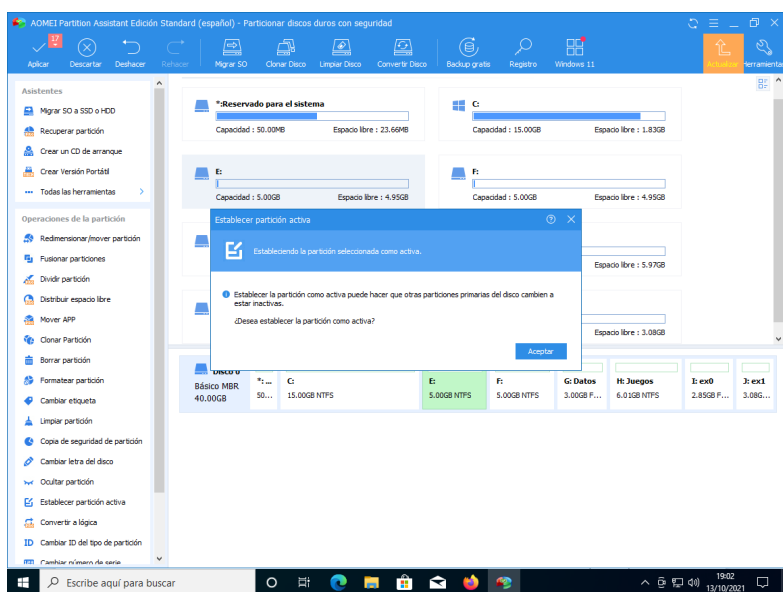


Figura 8: Cambiar partición activa.

F y la partición lógica de Datos (figura 7), y aunque no me decía que fuera directamente imposible, sí que me lo impedía. El mensaje decía que no se puede porque se tienen que aplicar todos los cambios (a.k.a. transacciones) que se hayan acumulado hasta entonces. Esto no lo hice ya que entonces cabe la posibilidad de romper la máquina virtual que la uso para otras asignaturas.

Finalmente cambié la partición activa para la iniciación de la máquina (figura 8). Me mencionó que es posible que esta acción pueda desactivar otras particiones, lo cual tendría sentido ya que sólo se puede iniciar de un sistema operativo a la vez.

### 3. Particiones En UNIX (Linux + fdisk)

Para esta sección, se va a crear primero una imagen virtual sobre la cual podremos hacer los cambios que queremos hacer. Esto se hace para no trabajar con un dispositivo de almacenamiento externo, y simplificar la tarea, además de poder cumplir mejor con los requisitos de la tarea. Esto se hace mediante el comando `dd`, que funciona a bajo nivel, copiando byte por byte la información. Para crear la imagen virtual, corremos el siguiente comando:

```
$ dd if=/dev/zero of=virt-drive.img bs=1M count=30720
30720+0 records in
30720+0 records out
32212254720 bytes (32 GB, 30 GiB) copied, 382,355 s, 84,2 MB/s
```

En esta ejecución vemos varios parámetros del comando. Éstos tienen el significado siguiente:

- `if=/dev/zero`: la primera parte (`if`) significa “input file”, y es el archivo que se usará para copiar. En este caso estamos usando un archivo especial, `/dev/zero`, que simplemente lee ceros.
- `of=virt-drive.img`: los caracteres `of` significan “output file”, y es el archivo destino de los datos que se copian. Lo guardaremos todo en un archivo nombrado `virt-drive.img`.
- `bs=1M`: `bs` quiere decir “block size”, y se refiere al tamaño de los bloques que se van escribiendo al archivo de escritura. En este caso, definimos el tamaño de bloque de 1MB.
- `count=30720`: el número de bloques que se han de copiar. En nuestro caso, como estamos transfiriendo en bloques de 1MB, y queremos crear un disco virtual de 30GB, el número de bloques sería de  $30 \times 1024 = 30720$ .

Este proceso puede tardar mucho tiempo, ya que literalmente está creando un archivo de tamaño de 30GB, y cambiando todos los bytes del disco en esa región a cero. Hay otras herramientas para hacer esto, sobre todo de QEMU que se usa para administrar máquinas virtuales, mas esta herramienta (`dd`) tiene seguridad de estar ya en cualquier máquina UNIX.

Con la imagen virtual, podemos abrirlo como si fuera cualquier otro dispositivo de almacenamiento. Normalmente éstos están en el directorio `/dev/`, mas como este se ha creado, está donde lo creamos. Con el comando siguiente, podemos ver el contenido actual de la imagen (que está vacía):



```
$ fdisk -l virt-drive.img
Disk virt-drive.img: 30 GiB, 32212254720 bytes, 62914560 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Para poder manejar las particiones, usaremos la herramienta `fdisk`, que es la más simple, aunque existan otras. Esto es porque `fdisk` es más universal, y se encontrará en cualquier sistema basado en UNIX. Esto lo hacemos corriendo el comando `fdisk virt-drive.img`.

Entrando el comando `p`, nos dice: `Disklabel type: dos`. Esto es decir que ya al empezar, `fdisk` quiere trabajar con el sistema de particiones de MBR. Esto se usa por defecto ya que es la más universal, aunque hoy en día sería más común utilizar GPT, que se puede crear con el comando `g`.

Para crear las particiones, usaremos el comando `n` (de *new*). Esto nos preguntará por el tipo de partición que queremos crear (primaria o extendida, la segunda siendo la que contiene las particiones lógicas). Al elegir primaria, nos preguntará el número de la partición (sólo del 1 al 4, ya que MBR no puede soportar más). Luego nos preguntará por el primer sector (es normal simplemente elegir el sector por defecto). Cuando se pregunta por el último sector, aquí es donde podemos especificar el tamaño de la partición con el sintaxis `+<tam>G`, donde `<tam>` es el tamaño que queremos. En nuestro caso, diremos `+10GB`. Al terminar nos creará una partición nueva de tamaño de 10GB de tipo Linux.

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p):
```

```
Using default response p.
Partition number (1-4, default 1):
First sector (2048-62914559, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-62914559,
default 62914559): +10G
```

```
Created a new partition 1 of type 'Linux' and of size 10 GiB.
```

Al crear las particiones primarias, se crean las particiones lógicas. Para hacer esto se ha de crear, en primer lugar, la partición extendida. Esta partición (como indica en las instrucciones de `fdisk`) servirá como contenedor

para las particiones lógicas. Para esto, cuando nos pide el tipo de partición introducimos `e`.

```
Partition type
  p  primary (2 primary, 0 extended, 2 free)
  e  extended (container for logical partitions)
Select (default p): e
```

Al crear la partición extendida, que ocupa el resto del disco, al crear una nueva partición sólo nos permitirá crear particiones lógicas:

```
Command (m for help): n
All space for primary partitions is in use.
Adding logical partition 5
First sector (41947136-62914559, default 41947136):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (41947136-62914559,
default 62914559): +2G
```

```
Created a new partition 5 of type 'Linux' and of size 2 GiB.
```

En `fdisk` no existe un comando para cambiar las etiquetas de las particiones. Pero como esto es una función puramente estética, y tampoco importante para la tarea a mano, lo saltamos.

Al acabar con las particiones debemos escribir los cambios. Para esto simplemente corremos el comando `w`.

```
Command (m for help): w
The partition table has been altered.
Syncing disks.
```

Para poder formatear las particiones, debemos de salir de `fdisk`, y usar las herramientas de `mkfs`. Pero como es una imagen, y no un dispositivo de verdad, para acceder las particiones es necesario usar los *dispositivos loop*, que nos permite asociar dispositivos con archivos normales del sistema. Esto lo hacemos mediante el comando siguiente:

```
$ sudo losetup --partscan --show --find virt-drive.img
/dev/loop0
```

Ahora se tratará el dispositivo `/dev/loop0` como si fuera un dispositivo real. Luego entonces podemos acceder a las particiones como `/dev/loop0pX`. Luego para formatear como FAT32 la partición, corremos el comando `mkfs.vfat`. La opción `-F32` nos dice qué tipo de FAT es (en este caso, 32). Esto nos será útil luego cuando queremos formatear como FAT(16).

```
$ sudo mkfs.vfat -F32 /dev/loop0p5
mkfs.fat 4.1 (2017-01-24)
```

Para poder borrar luego las particiones que queramos, tendríamos que entrar otra vez en `fdisk`, mas ahora podemos usar el dispositivo virtual `/dev/loop0`. Luego tan sólo hace falta usar el comando `d` (*delete*) para borrar la partición que queramos.

```
Command (m for help): d
Partition number (1-3,5-7, default 7): 7
```

```
Partition 7 has been deleted
```

Para crear las particiones lógicas, y formatearla como FAT32, simplemente seguimos lo que hemos visto anteriormente. Pero para formatear como FAT(16), simplemente es necesario cambiar `-F32` a `-F16`.

```
$ sudo mkfs.vfat -F16 /dev/loop0p8
mkfs.fat 4.1 (2017-01-24)
```

No se pueden fusionar dos particiones con `fdisk`, ya que realmente lo que se hace es borrar las dos particiones y crear una nueva en su lugar. Así que esto lo he saltado. No se puede encontrar esta funcionalidad, ni siquiera dentro de las opciones avanzadas.

Finalmente, para acabar, como ya hemos acabado modificando el archivo podemos desligar el archivo del dispositivo `/dev/loop0` con el comando `sudo losetup -d /dev/loop0`. Ya no se pueden acceder al archivo por medio de `/dev/loop0`.