

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI)

Author: Noel Milton Vega
Rensselaer Technology Group, LTD.

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

Introduction - Initial concepts to remember

A Veritas Volume Manager (VxVM) volume is an object that is manually constructed from a hierarchy of encapsulated "child" objects.¹ From inner to outer most encapsulation, these objects are:

- subdisks (sd)
- plexes (pl)
- volumes (vol)

Using the `vxmake(1m)` and `vxassist(1m)` commands we create **subdisks**, then **plexes**, and finally **volumes**. Volumes are the only objects that are capable of performing I/O operations on (i.e. mountable, read/writable).

Subdisks are created from physical disks that have been initialized using the `vxdisksetup(1m)` command, and then added to a *diskgroup* using the `adddisk` sub option to the `vxldg(1m)` command. One or more subdisks can be created from a single physical disk.

A **Plex** is used to organize and encapsulate one or more subdisks in order to create a larger **virtual disk** with particular RAID properties. A plex is defined by specifying a list of (best practices hand picked) subdisks to be included in it, along with the RAID behavior they should collectively assume. A plex can contain (i) a single subdisk, (ii) a concatenation of subdisks, (iii) striped subdisks, (iii) a concatenation of striped subdisks, or (iv) RAID5 striped subdisks.² So that is what plexes are for: to take simple subdisks and create more complex virtual disks out of them.

A **Volume** is used to encapsulate one or more plexes in order to create an object that is capable of accepting kernel I/O operations (mountable, readable/writable, etc.). Note that the volume object is where you create mirrored devices by specifying two (2) or more (best practices hand picked) plexes (each plex, in turn, with its own internal RAID properties).

With the foundation set, we'll now go into a step by step illustration of building volume manager disk groups and devices.

Important note: *as of Storage Foundation 4.0, the rootdg diskgroup is no longer required. Volume Manager uses "defaultdg" as the default diskgroup it operates on when the "-g" is omitted from commands. This means that step 1, presented next, will be different when configuring Volume Manager for SF4.0 and newer.*

¹ Note that all objects must exist within a Volume Manager diskgroup. (More on diskgroups later).

² Note that virtual mirrored (RAID1) disks are created inside the volume object from two (2) or more plexes. (More on this later).

STEP 1: Initialize the rootdg diskgroup (default disk group).

```
#!/bin/ksh -x

DISK1=c0t0d0s6      # (change me)
DISK2=c0t1d0s6      # (change me)
UNAME=`uname -n`    # (manually set me if you wish)

vxconfigd -m disable
vxdctl init ${UNAME}
vxdg init rootdg
#
vxdctl add disk ${DISK1} type=simple
vxdisk -f init ${DISK1} type=simple
vxdg -g rootdg adddisk ${DISK1}

vxdctl add disk ${DISK2} type=simple
vxdisk -f init ${DISK2} type=simple
vxdg -g rootdg adddisk ${DISK2}
#
vxdctl enable
rm /etc/vx/reconfig.d/state.d/install-d
```

Important note: as of Storage Foundation 4.0, the rootdg diskgroup is no longer required. Volume Manager uses "defaultdg" as the default diskgroup it operates on when the "-g" is omitted from commands. This means that step 1, presented above, will be different when configuring Volume Manager for SF4.0 and newer.

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

STEP 2: Initialize the physical disks as Volume Manager disks (dm_disks) using the `vxdisksetup(1m)` command. We specify the private region length to be 4096 or 8192 sectors so that we don't run out of space to hold diskgroup object data.

```
root# vxdisksetup -i c4t0d0 privlen=8192
root# vxdisksetup -i c5t0d0 privlen=8192
root# vxdisksetup -i c4t1d0 privlen=8192
root# vxdisksetup -i c5t1d0 privlen=8192
root# vxdisksetup -i c4t2d0 privlen=8192
root# vxdisksetup -i c5t2d0 privlen=8192
root# vxdisksetup -i c4t3d0 privlen=8192
root# vxdisksetup -i c5t3d0 privlen=8192
[ ... ]
```

STEP 3: Initialize the (real) data diskgroup with one disk; then populate it with the rest of disks earmarked for that diskgroup. The name of the data diskgroup we'll use in our example is "oracle-dg". All of these disks must have been previously initialized in step 2.

```
root# vxdg init oracle-dg disk01=c4t0d0
root# vxdg -g oracle-dg adddisk disk02=c5t0d0
root# vxdg -g oracle-dg adddisk disk03=c4t1d0
root# vxdg -g oracle-dg adddisk disk04=c5t1d0
root# vxdg -g oracle-dg adddisk disk05=c4t2d0
root# vxdg -g oracle-dg adddisk disk06=c5t2d0
root# vxdg -g oracle-dg adddisk disk07=c4t3d0
root# vxdg -g oracle-dg adddisk disk08=c5t3d0
[ ... ]
```

STEP 4: For each **dm_disk** in the diskgroup (added in step 3), create one or more subdisks. From a single **dm_disk**, you can create a single subdisk of size **maxsize** (see `vxassist(1m) maxsize` sub-command), or several sequential subdisks by specifying **dm_offsets** and **lengths** for each to the `vxmake(1m)` command. We provide examples for each here.

Let: **disk01** be derived from an EMC hypervolume, roughly 4.0GB+ in size.

The following is an example of how to take a **dm_disk**, **disk01**, and create a maximum sized subdisk from it which we will call **disk01-01** (i.e. the first subdisk carved from **disk01**). Note that *maximum size* can mean the maximum of the unused portion of a **dm_disk**, since it may already have had subdisks carved from it (in which case you'll have to take care when specifying the **dm_offset** and **length** parameters to `vxmake`).

```
root# vxassist -g oracle-dg -p maxsize layout=nostripe,nolog disk01
root# vxmake -g oracle-dg sd disk01-01 disk01,0,8370176s
```

Note that the output of the `vxassist` command above yields 8370176 sectors (**maxsize**), which we feed into the `vxmake` command as the **length** (the initial offset being 0).

If **disk02**, **disk03**, ... **disk08** were the same size as **disk01**, and we wanted to make maximum sized subdisks from each of these **dm_disks** as well, we simply repeat the the `vxmake` command for each, but increment the **dm_disk** name and the subdisk name as shown here:

```
root# vxmake -g oracle-dg sd disk02-01 disk02,0,8370176s
root# vxmake -g oracle-dg sd disk03-01 disk03,0,8370176s
root# vxmake -g oracle-dg sd disk04-01 disk04,0,8370176s
root# vxmake -g oracle-dg sd disk05-01 disk05,0,8370176s
root# vxmake -g oracle-dg sd disk06-01 disk06,0,8370176s
root# vxmake -g oracle-dg sd disk07-01 disk07,0,8370176s
root# vxmake -g oracle-dg sd disk08-01 disk08,0,8370176s
```

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

Let: disk01 be derived from an EMC hypervolume, roughly 9.0GB+ in size.

The following is an example of how to take a dm_disk, disk01, and create four (qty. 4) equally sized 2GB volumes, and an additional volume with the remaining unused portion (roughly 1GB).

```
#vxassist -g oracle-dg -p maxsize layout=nostripe,nolog disk01
#
# (Output from vxassist for this 9GB disk: 18874368 sectors)
# (2GB subdisks length =  $1024^3 \times 2 / 512 = 4194304$  sectors each)
# (Subdisk length for remaining space:  $18874368 - (4194304 \times 4) = 2097152$ )
#
#vxmake -g oracle-dg sd disk01-01 dm_name=disk01 dm_offset=0 len=4194304
#vxmake -g oracle-dg sd disk01-02 dm_name=disk01 dm_offset=4194304 len=4194304
#vxmake -g oracle-dg sd disk01-03 dm_name=disk01 dm_offset=8388608 len=4194304
#vxmake -g oracle-dg sd disk01-04 dm_name=disk01 dm_offset=12582912 len=4194304
#vxmake -g oracle-dg sd disk01-05 dm_name=disk01 dm_offset=16777216 len=2097152
#
```

If disk02, disk03, ... disk08 were the same size as disk01, and we wanted to create the same size and number of subdisks from these as we did from disk01, then we simply repeat the the vxmake command for each, but increment the dm_disk name and the subdisk name as shown here:

```
#vxmake -g oracle-dg sd disk02-01 dm_name=disk02 dm_offset=0 len=4194304
#vxmake -g oracle-dg sd disk02-02 dm_name=disk02 dm_offset=4194304 len=4194304
#vxmake -g oracle-dg sd disk02-03 dm_name=disk02 dm_offset=8388608 len=4194304
#vxmake -g oracle-dg sd disk02-04 dm_name=disk02 dm_offset=12582912 len=4194304
#vxmake -g oracle-dg sd disk02-05 dm_name=disk02 dm_offset=16777216 len=2097152
#
#vxmake -g oracle-dg sd disk03-01 dm_name=disk03 dm_offset=0 len=4194304
#vxmake -g oracle-dg sd disk03-02 dm_name=disk03 dm_offset=4194304 len=4194304
#vxmake -g oracle-dg sd disk03-03 dm_name=disk03 dm_offset=8388608 len=4194304
#vxmake -g oracle-dg sd disk03-04 dm_name=disk03 dm_offset=12582912 len=4194304
#vxmake -g oracle-dg sd disk03-05 dm_name=disk03 dm_offset=16777216 len=2097152
#
# [ ... and so on ... ]
#
```

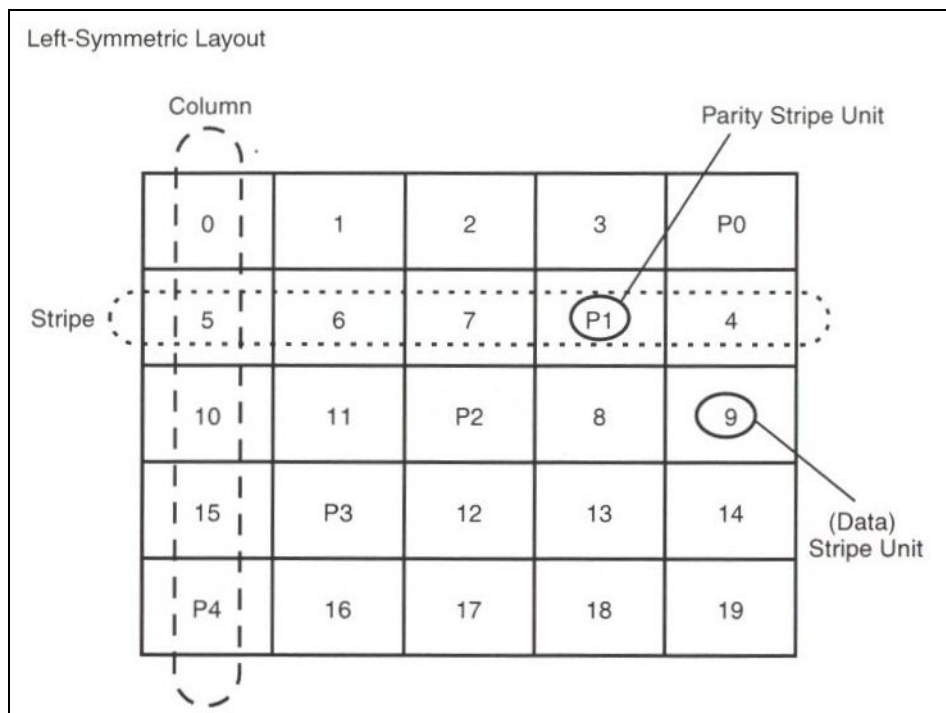
Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

STEP 5: Next we create plexes by specifying a list of subdisks to be included in it, along with the RAID behavior they should collectively assume (single subdisk; raid0 concat of multiple subdisks; raid0 stripe of multiple subdisks; raid5 stripe of multiple subdisks). Here are some examples:

```
vxmake -g oracle-dg plex PLEX01-01 layout=concat ncolumn=0 sd=disk01-01:0
vxmake -g oracle-dg plex PLEX02-01 layout=concat ncolumn=0 sd=disk02-01:0
```

Here we created two plexes for the purposes of **mirroring** data (in a VxVM volume object – not created here). The first plex is created from the single subdisk, disk01-01. The second plex is created from the single subdisk, disk02-01. These two plexes will be combined in a VxVM volume object to create a mirror. Side note: to ensure optimal availability and performance, it is essential that disk01 and disk02 be derived from different physical disks. Moreover, if multipathing is not used, disk01 and disk02 should be visible through different Host Bus Adapters (HBA's).

*As you look at these next examples, remember that Volume Manager striping (raid0 or raid5) occurs **between** columns as shown here, and NOT within a column (at least not until it wraps around). This affects how you place subdisks within a plex.*



```
vxmake -g oracle-dg plex PLEX02-01 layout=STRIPE st_width=128 ncolumn=3 \
sd=disk01-01,disk02-01,disk03-01
```

Here we created a plex consisting of three subdisks, disk01-01, disk02-01, and disk03-01. The parameters to vxmake are such that a STRIPE is created from these three subdisks, each in their own column.

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

```
vxmake -g oracle-dg plex r5PLEX01 layout=raid5 stwidth=32 sd=disk01-01:0,disk02-01:1,disk03-01:2,disk04-01:0,disk05-01:1,disk06-01:2
```

This command creates a raid5 plex which stacks disk01-01 and disk04-01 in column zero (0); stacks disk02-01 and disk05-01 in column one (1); and stacks disk03-01 and disk06-01 in column two (2). This ordering (or exact subdisk column placement) is determined by the number appearing after the colon (:) that suffixes each subdisk name. The order in which subdisks are specified on the command line DOES NOT matter when suffixed like this.

```
vxmake -g oracle-dg plex r5PLEX01 layout=raid5 stwidth=32 sd=disk01-01,disk02-01,disk03-01
```

Here we created a plex consisting of three subdisks, disk01-01, disk02-01, and disk03-01. The parameters to vxmake are such that a RAID5 stripe is created from these three subdisks, each in their own column (because we did not specify a value for **ncolumn**, and therefore defaults to the number of subdisks specified).

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

STEP 6: Finally, we create plex encapsulating volumes on which I/O operations can occur. Here we can create single plex volumes, consisting of one plex; or we can create multi-plex volumes consisting of two or more plexes that are to be mirrored. In either case, the RAID profile (stripe, concat, or single subdisk) of each constituent plex was determined at plex creation time. At volume creation time, we can specify Volume device attributes such as user & group ownership (via the **user** & **group** parameters), permissions (via the **mode** parameter), its length (via the **len** parameter), the log type (via the **log_type** parameter), the read prefer mirror policy (via the **rdpol prefer** parameter – useful for mirrors where one side is geographically local and the other is remote via a SAN), etc.

After creating the volume, we then have to **start** it before I/O operations can occur against it (such as creating a ufs or vxfs filesystem on it, raw read/writes to it, etc). When creating a volume for the purpose of mirroring plexes (data), we first create the volume specifying one participating plex, then start that single plex volume, and finally **attach** all remaining plexes that are to participate in the mirror.

Here are some examples.

```
root# vxmake -U fsgen -g oracle-dg vol VOL01_2GBm read_pol=SELECT \
user=oracle group=oracle mode=0644 log_type=NONE len=4194304 \
plex=PLEX01-01
```

*Note: The trailing 'm' in VOL01_2GBm is my hint that this is a mirror Volume.
An 'r' would indicate a RAID5 Volume.*

```
root# vxvol -g oracle-dg start VOL01_2GBm
```

```
root# nohup vxplex -g oracle-dg -U fsgen att VOL01_2GBm PLEX02-01 &
root# vxvol -g oracle-dg -U fsgen rdpol prefer VOL01_2GBm PLEX01-01
```

Note: Before the last (vxvol - rdpol prefer command).

```
#
root@ultral-a# vxprint -g oracle-dg -qhtv VOL01_2GBm
v VOL01 fsgen ENABLED ACTIVE 8370176 ROUND -
pl PLEX01-01 VOL01 ENABLED ACTIVE 8370176 CONCAT - RW
sd disk01-01 PLEX01-01 disk01 0 8370176 0 c1t2d0 ENA
pl PLEX02-01 VOL01 ENABLED ACTIVE 8370176 CONCAT - RW
sd disk02-01 PLEX02-01 disk02 0 8370176 0 c2t2d0 ENA
```

Note: After the last (vxvol - rdpol prefer command).

```
#
root@ultral-a# vxprint -g oracle-dg -qhtv VOL01_2GBm
v VOL01 fsgen ENABLED ACTIVE 8370176 PREFER PLEX01-01
pl PLEX01-01 VOL01 ENABLED ACTIVE 8370176 CONCAT - RW
sd disk01-01 PLEX01-01 disk01 0 8370176 0 c1t2d0 ENA
pl PLEX02-01 VOL01 ENABLED ACTIVE 8370176 CONCAT - RW
sd disk02-01 PLEX02-01 disk02 0 8370176 0 c2t2d0 ENA
```

```
/usr/sbin/vxmake -U fsgen -g oracle-dg vol VOL01_101.13GB
readpol=SELECT|PREFER|ROUND user=root group=other mode=0644
log_type=NONE len=212098824 plex=PLEX01-01
```

(See vol_pattern(4) man page for read policies)

A sample script that ties it all together:

```
#!/bin/ksh
#
UNAME=`uname -n | sed -e 's/\..*//'\`
DG_NAME=${UNAME}-dg
#
#####
print -n "Creating SUBDISKS... "

MAXSIZE1=`vxassist -g ${DG_NAME} -p maxsize layout=nostripe,nolog
disk001`

NUM_DISKS=64
COUNT=1
while [ ${COUNT} -le ${NUM_DISKS} ]
do
{
  if [ ${COUNT} -lt 10 ]
  then
  {
    COUNT=00${COUNT}
  }
  elif [ ${COUNT} -lt 100 ]
  then
  {
    COUNT=0${COUNT}
  }
  else
  {
    COUNT=${COUNT}
  }
  fi

  print "vxmake -g ${DG_NAME} sd disk${COUNT}-01
disk${COUNT},0,${MAXSIZE1}s"
  vxmake -g ${DG_NAME} sd disk${COUNT}-01 disk${COUNT},0,${MAXSIZE1}s
  (( COUNT = COUNT + 1 ))
}
done
print "done!"
#####
```

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLD).

```
#####  
print -n "Creating PLEXES... "  
#  
# 8GB+  
vxmake -g ${DG_NAME} plex orau01_PLEX layout=STRIPE ncolumn=1  
stwidth=256 sd=disk001-01:0  
#  
# 8GB+  
vxmake -g ${DG_NAME} plex orau02_PLEX layout=STRIPE ncolumn=1  
stwidth=256 sd=disk002-01:0  
#  
# 40GB+  
vxmake -g ${DG_NAME} plex orau03_PLEX layout=STRIPE ncolumn=5  
stwidth=256 sd=disk003-01:0,disk004-01:1,disk005-01:2,disk006-  
01:3,disk007-01:4  
#  
# 48GB+  
vxmake -g ${DG_NAME} plex orau04_PLEX layout=STRIPE ncolumn=6  
stwidth=256 sd=disk008-01:0,disk009-01:1,disk010-01:2,disk011-  
01:3,disk012-01:4,disk013-01:5  
#  
# 56GB+  
vxmake -g ${DG_NAME} plex orau05_PLEX layout=STRIPE ncolumn=7  
stwidth=256 sd=disk014-01:0,disk015-01:1,disk016-01:2,disk017-  
01:3,disk018-01:4,disk019-01:5,disk020-01:6  
#  
# 16GB+  
vxmake -g ${DG_NAME} plex orau06_PLEX layout=STRIPE ncolumn=2  
stwidth=256 sd=disk021-01:0,disk022-01:1  
#  
# 48GB+  
vxmake -g ${DG_NAME} plex orau07_PLEX layout=STRIPE ncolumn=6  
stwidth=256 sd=disk023-01:0,disk024-01:1,disk025-01:2,disk026-  
01:3,disk027-01:4,disk028-01:5  
#  
# 40GB+  
vxmake -g ${DG_NAME} plex orau08_PLEX layout=STRIPE ncolumn=5  
stwidth=256 sd=disk029-01:0,disk030-01:1,disk031-01:2,disk032-  
01:3,disk033-01:4  
#  
# 48GB+  
vxmake -g ${DG_NAME} plex orau09_PLEX layout=STRIPE ncolumn=6  
stwidth=256 sd=disk034-01:0,disk035-01:1,disk036-01:2,disk037-  
01:3,disk038-01:4,disk039-01:5  
#  
# 80GB+  
vxmake -g ${DG_NAME} plex orau10_PLEX layout=STRIPE ncolumn=5  
stwidth=256 sd=disk040-01:0,disk041-01:1,disk042-01:2,disk043-  
01:3,disk044-01:4,disk045-01:0,disk046-01:1,disk047-01:2,disk048-  
01:3,disk049-01:4
```

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

```
#
# 80GB+
vxmake -g ${DG_NAME} plex orau11_PLEX layout=STRIPE ncolumn=5
stwidth=256 sd=disk050-01:0,disk051-01:1,disk052-01:2,disk053-
01:3,disk054-01:4,disk055-01:0,disk056-01:1,disk057-01:2,disk058-
01:3,disk059-01:4
#
# 40GB+
vxmake -g ${DG_NAME} plex orau14_PLEX layout=STRIPE ncolumn=5
stwidth=256 sd=disk060-01:0,disk061-01:1,disk062-01:2,disk063-
01:3,disk064-01:4
#
print "done!"
```

```
print -n "Creating VOLUMES... "
vxmake -Ufsgen -g ${DG_NAME} vol orau01_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau01_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau02_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau02_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau03_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau03_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau04_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau04_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau05_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau05_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau06_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau06_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau07_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau07_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau08_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau08_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau09_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau09_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau10_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau10_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau11_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau11_PLEX
vxmake -Ufsgen -g ${DG_NAME} vol orau14_VOL user=oracle group=dba
mode=0660 logtype=NONE plex=orau14_PLEX
#
print "done!"
```

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLD).

```
print -n "Starting VOLUMES... "
vxvol -g ${DG_NAME} start orau01_VOL
vxvol -g ${DG_NAME} start orau02_VOL
vxvol -g ${DG_NAME} start orau03_VOL
vxvol -g ${DG_NAME} start orau04_VOL
vxvol -g ${DG_NAME} start orau05_VOL
vxvol -g ${DG_NAME} start orau06_VOL
vxvol -g ${DG_NAME} start orau07_VOL
vxvol -g ${DG_NAME} start orau08_VOL
vxvol -g ${DG_NAME} start orau09_VOL
vxvol -g ${DG_NAME} start orau10_VOL
vxvol -g ${DG_NAME} start orau11_VOL
vxvol -g ${DG_NAME} start orau14_VOL
vxvol -g ${DG_NAME} -f startall
print "done!"
#####
```

```
#####
# Now create VxFS filesystems on each of #
# the Volumes created above. Use the prtvtoc(1) #
# to verify the sector sizes used above for each #
# of the volumes: #
# prompt$ prtvtoc -h /dev/vx/rdisk/${DG_NAME}/volname #
#####
print -n "Creating filesystems... "
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau01_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau01_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau02_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau02_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau03_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau03_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau04_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau04_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau05_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau05_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau06_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau06_VOL ${SIZE} > /dev/null
```

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

```
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau07_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau07_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau08_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau08_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau09_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau09_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau10_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau10_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau11_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau11_VOL ${SIZE} > /dev/null
#
SIZE=`prtvtoc -h /dev/vx/rdisk/${DG_NAME}/orau14_VOL | awk '{ print $5
}'`
mkfs -F vxfs /dev/vx/rdisk/${DG_NAME}/orau14_VOL ${SIZE} > /dev/null
#
print "done!"
#####
```

Manually initializing and creating Veritas Volume Manager (VxVM) objects and volumes (CLI).

```
#####  
print -n "Mounting filesystems... "  
mkdir -p /ora/u01 > /dev/null 2>&1  
mkdir -p /ora/u02 > /dev/null 2>&1  
mkdir -p /ora/u03 > /dev/null 2>&1  
mkdir -p /ora/u04 > /dev/null 2>&1  
mkdir -p /ora/u05 > /dev/null 2>&1  
mkdir -p /ora/u06 > /dev/null 2>&1  
mkdir -p /ora/u07 > /dev/null 2>&1  
mkdir -p /ora/u08 > /dev/null 2>&1  
mkdir -p /ora/u09 > /dev/null 2>&1  
mkdir -p /ora/u10 > /dev/null 2>&1  
mkdir -p /ora/u11 > /dev/null 2>&1  
mkdir -p /ora/u14 > /dev/null 2>&1  
#  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau01_VOL /ora/u01  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau02_VOL /ora/u02  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau03_VOL /ora/u03  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau04_VOL /ora/u04  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau05_VOL /ora/u05  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau06_VOL /ora/u06  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau07_VOL /ora/u07  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau08_VOL /ora/u08  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau09_VOL /ora/u09  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau10_VOL /ora/u10  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau11_VOL /ora/u11  
mount -F vxfs -o rw /dev/vx/dsk/${DG_NAME}/orau14_VOL /ora/u14  
print "done!"  
#####
```