Day 1: LVM

- :
  - LVM PV/VG/LV
  - LVM
  - LVM
  - LVM
- :
  - 4
  - Go JSON

(30-40%)

- :
  1. **(Physical Volume - PV)**: LVM RAID pvcreate LVM **(Physical Extents - PE)**PE LVM 4MB
  2. **(Volume Group - VG)**: PV VG (vgcreate) vgextend PV
  3. **(Logical Volume - LV)**: VG LV (lvcreate) VG
- :
  - **LVM** : LVM "" PVVGLV VG **PV** LVM VG PV VG
- :
  - : MySQLPostgreSQL LV
  - : KVM Xen VG LV
  - : LVM

□ □□□□ (40-50%)

- :
  1. :
     - VirtualBox VMware
     - : CentOS 9 Stream Ubuntu Server 22.04 LTS
     - : 2 CPU, 2GB RAM, 20GB
     - : **4** 10GB PV
  2. **LVM** :
     -
     - **CentOS/RHEL**: sudo dnf install -y lvm2
     - **Ubuntu/Debian**: sudo apt-get update && sudo apt-get install -y lvm2
     - : lvm version
  3. :
     - : lsblk
     - : sda sdb, sdc, sdd, sde

```
NAME   MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda      8:0    0  20G  0 disk
├─sda1   8:1    0   1G  0 part /boot
└─sda2   8:2    0  19G  0 part /
sdb      8:16   0  10G  0 disk
sdc      8:32   0  10G  0 disk
sdd      8:48   0  10G  0 disk
sde      8:64   0  10G  0 disk
```

Go (20-30%)

- : lsblk JSON
- : Go
  1. : mkdir -p lvm-manager/cmd/day01 && cd lvm-manager/cmd/day01
  2. **(main.go)**:

```
package main

import (
        "bytes"
        "encoding/json"
```

```
        "fmt"
        "log"
        "os/exec"
        "strings"
)

// BlockDevice  lsblk JSON
type BlockDevice struct {
        Name       string `json:"name"`
        Size       string `json:"size"`
        Type       string `json:"type"`
        MountPoint string `json:"mountpoint"`
}

// LsblkOutput  lsblk  JSON
type LsblkOutput struct {
        BlockDevices []BlockDevice `json:"blockdevices"`
}

// getBlockDevices  lsblk
func getBlockDevices() ([]BlockDevice, error) {
        // -J  JSON , -o
        cmd := exec.Command("lsblk", "-J", "-o", "NAME,SIZE,TYPE,MOUNTPOINT")

        var out, stderr bytes.Buffer
        cmd.Stdout = &out
        cmd.Stderr = &stderr

        log.Println("Executing command:", cmd.String())
        err := cmd.Run()
        if err != nil {
                return nil, fmt.Errorf("lsblk command failed: %v\nStderr: %s", err, stderr.String())
        }

        var report LsblkOutput
        if err := json.Unmarshal(out.Bytes(), &report); err != nil {
                return nil, fmt.Errorf("failed to unmarshal lsblk JSON output: %v", err)
        }

        return report.BlockDevices, nil
}

func main() {
        log.Println("Starting block device discovery...")

        devices, err := getBlockDevices()
        if err != nil {
                log.Fatalf("FATAL: Could not retrieve block device information: %v", err)
        }

        fmt.Println("\n--- System Block Devices ---")
        fmt.Printf("%-15s %-10s %-10s %-s\n", "DEVICE", "SIZE", "TYPE", "MOUNTPOINT")
        fmt.Println(strings.Repeat("-", 60))
        for _, device := range devices {
                mountPoint := device.MountPoint
                if mountPoint == "" {
                        mountPoint = "(none)"
                }
                fmt.Printf("%-15s %-10s %-10s %-s\n", device.Name, device.Size, device.Type, mountPoint)
        }
        fmt.Println(strings.Repeat("-", 60))

        log.Println("Discovery finished successfully.")
}
```

- :
  - : lsblk
    - : 1. 2. rescan-scsi-bus.sh ()
  - : go run main.go exec: "lsblk": executable file not found in $PATH
    - : util-linux sudo dnf install util-linux sudo apt-get install util-linux
- :
  - : Go main getBlockDevices
  - : fmt.Errorf stderr

- : Go lvm-manager
- : discover PV
- :
  - :JSONmain
  - : exec.Command json.Unmarshal
  - :

- :

  1. : “PE LVM ”
  2. : Go -u --unmounted MountPoint
     - : Go flag

- : lvm.conf man page (man lvm.conf) LVM devices filter LVM