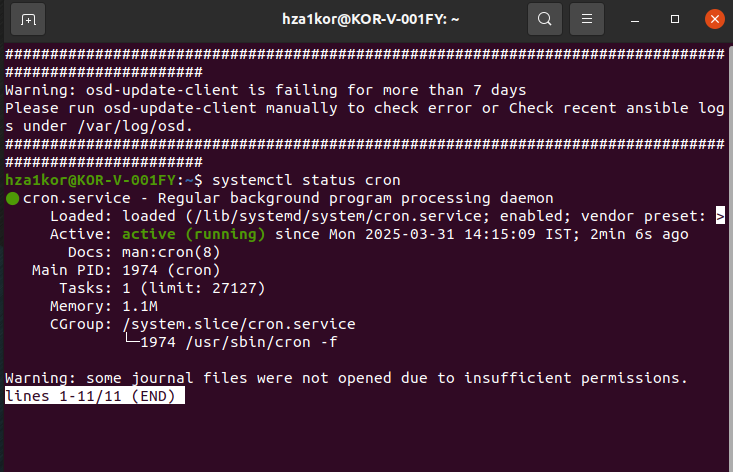
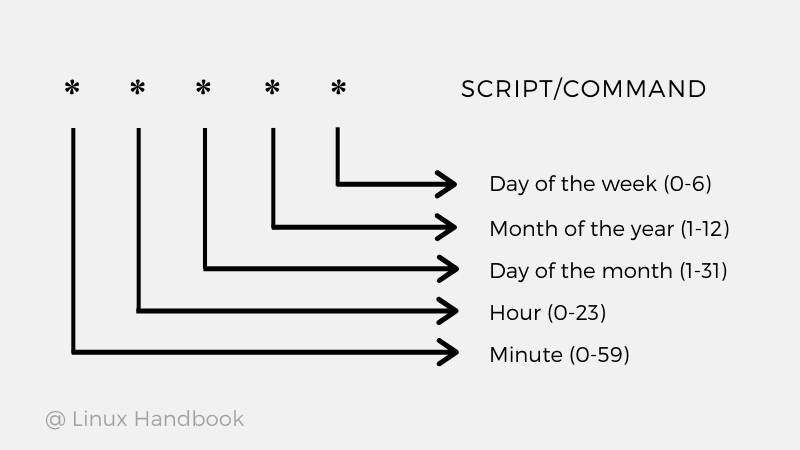
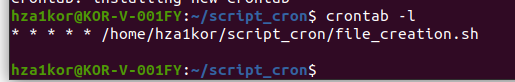
**Understand Cron Job in Linux with Practical Example [HINDI] | MPrashant**

* A cron job is a scheduled task that allows you to run scripts or commands at specific intervals
* Cron jobs are useful when you want to automate repetitive tasks, ensuring they run at scheduled times without manual intervention
* Eg – backing up files, running system maintenance or sending email reports
* **How does cron work –** cron uses a daemon(a background process) called cron to execute scheduled tasks.
* the configuration for these tasks is stored in a file called a crontab
* name of the service is: cron
* To check the status of the cron service, run systemctl status cron
* 
* Cron.service is a regular background program processing daemon
* To list all the existing cron jobs – crontab -l
* Currently there are no cron jobs
* 
* To edit a crontab or create a new crontab use command – crontab -e
* To delete all the cron jobs in one go – crontab -r
* Currently there are no crontabs created, so lets try creating our first cron job
* Remember that when you create a cron job, its specific to your user, like here for **hza1kor** only, not for all users, because of which when we ran the command crontab -l we saw no crontab for **hza1kor**
* Basic format of a cron job is - **\* \* \* \* \*** /executable\_path /script\_path
* Explanation of the syntax is -
  + Minute(0-59)
  + Hour(0-23)
  + Day of the month (1-31)
  + Month(1-12)
  + Day of the week (0-7) (Sunday is 0 or 7)



* We can use crontab guru website to understand each start properly, the values we provide it, it will show you when the task will be executed
* Site - [CRONTAB\_GURU](https://crontab.guru/)
* If I just give \*\*\*\*\*, it means execute every minute
* 
* Use cron guru to properly set timings
* Say I want to create a cron job, which is executed every min
* Note that **cron jobs doesn’t support execution in sec, we can do every minute using \* \* \* \* \*, if I want every 2nd min then - \*/2 \* \* \* \* \***
* I have created a cron job using command crontab -e and added the below data
* **\* \* \* \* \* /home/hza1kor/script\_cron/file\_creation.sh**
* This job will ensure that the job is executed every minute
* We can see that the new cron job is created using crontab -l
* 
* The content inside file\_creation.sh is as follows –

**#!/bin/bash**

**# Script to create a file named cron\_{i} each time it is called.**

**# Change to the directory where the script is located**

**cd "$(dirname "$0")"**

**# Name of the file to store the counter**

**COUNTER\_FILE="cron\_counter.txt"**

**# Initialize the counter file if it doesn't exist**

**if [ ! -f "$COUNTER\_FILE" ]; then**

**echo 0 > "$COUNTER\_FILE"**

**fi**

**# Read the current counter value**

**counter=$(cat "$COUNTER\_FILE")**

**# Increment the counter**

**counter=$((counter + 1))**

**# Save the updated counter back to the file**

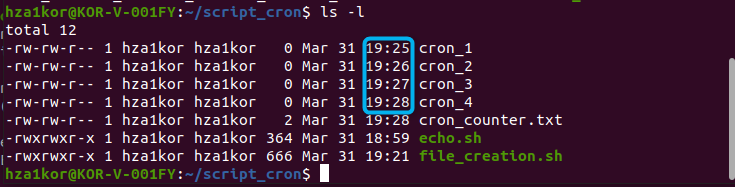
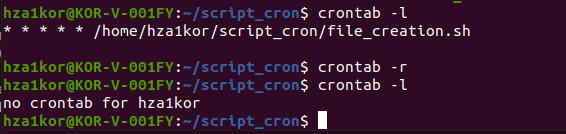
**echo "$counter" > "$COUNTER\_FILE"**

**# Create the file with the name cron\_{counter}**

**file\_name="cron\_${counter}"**

**touch "$file\_name"**

**echo "Created file: $file\_name"**

* Now after every minute a file is created, and the counterfile keeps incrementing the value
* 
* Once you execute crontab -e, it will start the job, so you don’t really have to run any other command to start the job, you edit or create a cron job and it starts executing,
* 
* This will install and start a new job
* Now say I want to remove the crontab, for now I know only one command, but this removed all the cron jobs – crontab -r
* 
* Here earlier there was one crontab which was executing file\_creation.sh, then we used the **crontab -r** command to remove all crontabs, and then when we run **crontab -l** again, we see there are no crontabs
* Say I don’t want a specific time of the day that my script should be executed, just that it should be executed daily, then we can use a special keyword **@daily**
* Example – crontab -e -> @daily /home/hza1kor/script\_cron/system\_logs.sh
* **Note that if your script just has some echo messages, like now I thought, the script will just print the date and time and a quote, but its useless because cron jobs are basically executed in the background so no echo prints will be visible to you on screen**
* The below script will be able to run a few commands like uptime, free and save the o/p in a log file inside system\_logs folder, with the timestamp

**#!/bin/bash**

**# daily\_system\_info.sh**

**# This script logs system information and saves it in a timestamped log file**

**# inside the system\_log directory.**

**# Define the log directory (adjust the path if needed)**

**LOG\_DIR="/home/hza1kor/system\_log"**

**# Create the log directory if it doesn't exist**

**mkdir -p "$LOG\_DIR"**

**# Generate a timestamp for the filename (format: YYYY-MM-DD\_HH-MM-SS)**

**TIMESTAMP=$(date +"%Y-%m-%d\_%H-%M-%S")**

**LOGFILE="${LOG\_DIR}/system\_info\_${TIMESTAMP}.log"**

**# Collect and write system information into the log file**

**{**

**echo "============================================"**

**echo "System Info Report - $(date)"**

**echo "============================================"**

**echo ""**

**echo "Uptime:"**

**uptime**

**echo ""**

**echo "Disk Usage:"**

**df -h**

**echo ""**

**echo "Memory Usage:"**

**free -h**

**echo ""**

**} > "$LOGFILE"**

**echo "Log file created: $LOGFILE"**

* Now I have created a cronjob (crontab -e) with the below content - @daily /home/hza1kor/script\_cron/system\_logs.sh
* Then this script will be executed daily once and save the o/p in /home/hza1kor/system\_logs file with datestamp files