Below is a **complete, ready-to-copy Markdown** version of the entire answer set.
You can **export it to PDF in 1 click** using any of these free tools:

How to Export to PDF (Step-by-Step)

Option 1: VS Code (Recommended)

1. Open VS Code
2. Press `Ctrl+Shift+P` → Type **"Markdown: Export to PDF"**
3. Install **"Markdown PDF"** extension if prompted
4. Done! → `devops-interview-answers.pdf`

Option 2: Online (No Install)

1. Go to: https://md-to-pdf.fly.dev
2. Paste the Markdown below
3. Click **"Convert to PDF"** → Download

Option 3: Pandoc (CLI)

```
# Install pandoc (if not)
sudo apt install pandoc

# Save this as devops.md, then:
pandoc devops.md -o devops-interview-answers.pdf --pdf-engine=wkhtmltopdf
```

Full Markdown Content (Copy from here)

```markdown
# DevOps Interview Q&A – Complete Guide
*Prepared for PDF Export | November 2025*

---

## Linux Fundamentals

### 1. What is Linux system?
**Ans:** Linux is an **open-source, Unix-like operating system kernel** combined
with GNU tools forming a complete OS (distribution). It's modular, secure, stable,
and widely used in servers, cloud, and embedded systems.

> **Re-ask ready:** "Linux = Kernel + GNU utilities + Package manager + Shell"

---

### 2. Founder of Linux?
**Ans:** **Linus Torvalds** created the Linux kernel in **1991**. Richard Stallman
```

~~~
founded the **GNU Project** (1983), which provided tools.

> GNU/Linux = GNU tools + Linux kernel

---

### 3. Other operating systems?
| Type | Examples |
|------|---------|
| Unix-like | macOS, FreeBSD, Solaris |
| Windows | Windows Server, Windows 10/11 |
| Mobile | Android (Linux-based), iOS |
| Mainframe | z/OS, AIX |

---

### 4. What is shell and its types?
**Ans:** Shell is a **command-line interpreter** between user and kernel.

| Type | Name | Example |
|------|------|--------|
| Bourne | `sh` | Original |
| **Bourne Again** | `bash` | Default in most distros |
| C Shell | `csh`, `tcsh` | Syntax like C |
| Korn | `ksh` | Advanced scripting |
| Z Shell | `zsh` | Feature-rich, Oh-My-Zsh |

> **Check current shell:** `echo $SHELL` or `ps -p $$`

---

### 5. What is kernel and latest version? How to check?
**Ans:** Kernel is the **core of OS** – manages hardware, memory, processes.

```bash
# Check kernel version
uname -r
# Example output: 5.15.0-105-generic

# Full details
cat /proc/version
~~~

> **Latest stable (as of Nov 2025): ~6.11.x** (check kernel.org)

---

6. Check server configuration (RAM, CPU, sudo)?

Task	Command
CPU Info	`lscpu` or `cat /proc/cpuinfo`
RAM	`free -h` or `cat /proc/meminfo`

Task	Command
Disk	`df -h`, `lsblk`
Sudo Permission	`sudo -l` (lists allowed commands)

---

7. What is CPU core? How to check cores?

Ans: A **CPU core** is an independent processing unit.

```
# Logical cores (with hyper-threading)
nproc
# or
lscpu | grep "CPU(s):"
# or
cat /proc/cpuinfo | grep "processor" | wc -l
```

> **Physical cores:** `lscpu | grep "Core(s) per socket"`

---

8. Where are user properties stored in Linux?

Ans:

- `/etc/passwd` → username, UID, GID, home, shell
- `/etc/shadow` → encrypted password
- `/etc/group` → group membership

```
getent passwd username
```

---

9. Check RAM (with example – free command)

```
free -h
# Output:
#              total        used        free      shared  buff/cache   available
# Mem:          7.8G        2.1G        3.5G        200M        2.2G        5.3G
```

> Alternative: `top`, `htop`, `vmstat`

---

10. What is in `/etc/passwd`?

```
username:x:1001:1001:John Doe:/home/john:/bin/bash
```

Field	Meaning
1	Username
2	Password (`x` → in shadow)
3	UID
4	GID
5	GECOS (full name)
6	Home directory
7	Login shell

---

11. Install OS in VM

Steps:

1. Download ISO (e.g., Ubuntu Server)
2. Create VM in **VirtualBox/VMware**
3. Attach ISO → Boot
4. Follow installer (partition, user, packages)

> Tools: **KVM/QEMU**, **Proxmox**, **Hyper-V**

---

12. Why choose Linux OS?

Reason	Example
Open-source	Free, community
Stability	99.999% uptime
Security	SELinux, AppArmor
Customization	Kernel modules
Cost	No licensing

---

13. Enterprise versions of Linux

Distro	Enterprise Version
Red Hat	**RHEL**
Ubuntu	**Ubuntu Pro (ESM)**
SUSE	**SLES**
CentOS	**CentOS Stream** (community)

---

14. What happens when you create a user?

```
useradd john
# Creates:
# - Entry in /etc/passwd
# - Entry in /etc/shadow
# - Entry in /etc/group
# - Home dir: /home/john
# - Skeleton files copied from /etc/skel
```

> Use `useradd -m -s /bin/bash john` for home + shell

---

15. Major/Critical log files in Linux

File	Purpose
`/var/log/messages`	General system logs (RHEL)
`/var/log/syslog`	General logs (Ubuntu)
`/var/log/auth.log`	Login/auth events
`/var/log/secure`	Auth (RHEL)
`/var/log/kern.log`	Kernel
`/var/log/dmesg`	Boot messages

---

16. What is log rotation?

Ans: Automatic archiving and compression of old logs to save space.

> Tool: **logrotate**
> Config: `/etc/logrotate.conf`, `/etc/logrotate.d/`

```
# Example config
/var/log/apache2/*.log {
    daily
    rotate 7
    compress
    missingok
}
```

---

17. What is cron job? Is there a scheduler?

Ans: Cron is the **task scheduler** in Linux.

```
crontab -e
# Example: Run backup daily at 2 AM
0 2 * * * /backup.sh
```

> Alternatives: `systemd timers`, `anacron`

---

18. What is soft link? How to create? (symlink)

Ans: Symbolic link = pointer to another file.

```
ln -s /path/to/original.txt /path/to/link.txt
```

> Hard link: `ln original.txt hardlink.txt` (same inode)

---

19. What are inodes?

Ans: Index node – stores metadata (permissions, size, location) of a file.
Every file/dir has one inode.

```
# Check inode
ls -i filename
df -i  # inode usage
```

---

20. How to know updates in a package?

```
# RHEL/CentOS
yum check-update httpd

# Ubuntu/Debian
apt list --upgradable | grep httpd
```

---

21. Create shortcut of file

```
# Symbolic link (shortcut)
ln -s /var/www/html/index.html ~/Desktop/index.html
```

---

22. Loops syntax in shell scripting

```
# For loop
for i in 1 2 3; do
    echo $i
done

# While loop
count=1
while [ $count -le 5 ]; do
    echo $count
    ((count++))
done

# Until loop
until [ $count -gt 5 ]; do
    echo $count
    ((count++))
done
```

23. File permissions & special permissions

Permission	Meaning
`rwx`	read, write, execute
`chmod 755 file`	owner: rwx, group/other: r-x

Special Permissions:

Bit	Name	Effect
`u+s`	**SUID**	Run as owner
`g+s`	**SGID**	Run as group / inherit group
`+t`	**Sticky**	Only owner can delete (e.g., /tmp)

```
chmod u+s /usr/bin/passwd    # SUID
chmod +t /tmp                # Sticky bit
```

24. Default file permission for new file

Ans: 666 (rw-rw-rw-) for files, **777** for dirs → minus **umask**

```
umask  # usually 0022
# New file: 666 - 022 = 644 (rw-r--r--)
# New dir: 777 - 022 = 755
```

25. Log file location & types

Type	Location
System	`/var/log/syslog` or `/var/log/messages`
Auth	`/var/log/auth.log`
Kernel	`/var/log/kern.log`
App	`/var/log/apache2/`, `/var/log/mysql/`

26. Package manager: RPM vs YUM

Feature	RPM	YUM/DNF
Level	Low (package)	High (dependency resolver)
Install	`rpm -ivh pkg.rpm`	`yum install httpd`
Repo	No	Yes

YUM install httpd background:

1. Resolves dependencies
2. Downloads from **mirrorlist** (Red Hat Mirror)
3. Installs to `/etc/httpd`, `/var/www`
4. Runs `%post` script

> **Red Hat Repository:** subscription-based
> **Mirror:** public sync (e.g., mirror.centos.org)

27. Reboot Linux server

```
reboot
# or
shutdown -r now
# or
init 6
```

28. Shell scripting, how to run, what is bash

Shell Script: Program written in shell commands.

```
#!/bin/bash
echo "Hello World"
```

Run:

```
chmod +x script.sh
./script.sh
# or
bash script.sh
```

> **Bash** = Bourne Again SHell (most common)

---

29. If first command fails, does second execute?

By default: YES

```
false && echo "This won't print"
# No output

false || echo "This will print"
# Prints
```

> Use `set -e` to exit on failure

---

30. Print date

```
date
# Custom
date '+%Y-%m-%d %H:%M:%S'
```

---

31. Use cases of shell scripting

- Automation (backup, monitoring)
- System admin tasks
- Deployment scripts
- Log parsing
- Cron jobs

Cloud & AWS

32. What is cloud? Where does it exist? How to access?

Ans: Cloud = **remote data centers** offering compute, storage, DB.

- **Exists:** In **provider-owned data centers** (AWS in Virginia, Ireland, etc.)
- **Access:** Web console, CLI (`aws cli`), SDK, SSH/RDP

---

33. Types of cloud

Type	Example
Public	AWS, Azure, GCP
Private	OpenStack, VMware
Hybrid	AWS + On-prem

---

34. Cloud models

Model	Full Form	Example
IaaS	Infrastructure as a Service	EC2, VPC
PaaS	Platform as a Service	Elastic Beanstalk
SaaS	Software as a Service	Gmail, Office 365

---

35. Advantages of cloud

- Scalability
- Pay-as-you-go
- Global reach
- Managed services
- Disaster recovery

---

36. What is Elastic in AWS? (Auto-scaling)

Ans: Elastic = ability to scale automatically.

Auto Scaling Group (ASG):

- Monitors CPU/memory
- Launches/terminates EC2 instances
- Uses **CloudWatch alarms**

> **Background:** Load balancer distributes, new instances register

---

37. Disadvantage of AWS

- Cost unpredictability
- Vendor lock-in
- Complexity
- Outages (rare but happen)

---

38. What is EBS? Can you resize 100GB?

Ans: Elastic Block Store – block storage for EC2.

Yes, resize possible:

```
# AWS Console or CLI
aws ec2 modify-volume --volume-id vol-123 --size 200
```

> Then extend filesystem inside OS (`resize2fs`)

---

39. IAM & Policy

- **IAM** = Identity & Access Management
- **Policy** = JSON document defining permissions

```
{
  "Effect": "Allow",
  "Action": "s3:*",
  "Resource": "*"
}
```

---

40. Private cloud software

Tool	Use
OpenStack	Full IaaS
VMware vSphere	Virtualization
Proxmox	KVM + LXC
oVirt	RHEL-based

---

41. Cloud service categories

Category	Example
Compute	EC2, Lambda
Storage	S3, EBS
Database	RDS, DynamoDB
Networking	VPC, Route53

---

42. Why choose AWS?

- Largest market share
- Most services
- Global regions
- Strong community
- Pay-as-you-go

---

43. EC2 not accessible – possible issues

Issue	Check
Security Group	Port 22 open?
NACL	Allow traffic?
Route Table	Internet Gateway?
Public IP	Assigned?
Key pair	Correct `.pem`?
Instance state	Running?

---

44. Types of storage in AWS

Type	Use
S3	Object storage
EBS	Block (EC2)
EFS	File (shared)
Glacier	Archive

---

45. PEM key – what does it validate?

Ans: `.pem` = private key
Validates with **public key** stored in `~/.ssh/authorized_keys` on EC2

> Authentication: **Key-pair (asymmetric)**

---

46. EC2 authentication types

- **Password** (Windows RDP)
- **Key pair** (Linux SSH)

---

47. Advantages of AWS

- Scalability
- Reliability (99.99% SLA)
- Security (IAM, KMS)
- Global infrastructure

---

48. Subnet – types, uses

Subnet = logical subdivision of VPC

Type	CIDR	Use	Example
Public	Route to IGW	Web servers	10.0.1.0/24
Private	No IGW	DB, app	10.0.2.0/24

---

49. AWS Regions & AZs

- **Regions:** ~30+ (us-east-1, eu-west-1)
- **AZs per region:** 3–6

---

50. When you create VPC, what is created?

- VPC (CIDR)
- Default Route Table
- Default Network ACL
- Default Security Group

---

51. Private vs Public Subnet

Feature	Public	Private
Internet access	Yes (via IGW)	No
NAT Gateway	No	Yes (for outbound)
Use	Web, Bastion	DB, App

SDLC & DevOps Tools

52. SDLC types

Type	Description
Waterfall	Sequential
Agile	Iterative
Spiral	Risk-focused

Type	Description
DevOps	Continuous

---

53. Agile tools

- Jira, Trello, Asana, Azure Boards

---

54. Jenkins – jobs & pipeline

- **Job** = build, test, deploy task
- **Pipeline** = scripted/declarative workflow

```
pipeline {
    agent any
    stages {
        stage('Build') { steps { sh 'mvn clean package' } }
        stage('Test') { steps { sh 'mvn test' } }
    }
}
```

---

55. Jenkins workflow

1. Git webhook → trigger
2. Pull code
3. Build (Maven/Gradle)
4. Test (JUnit)
5. Archive artifacts
6. Deploy (SCP, Docker, Kubernetes)

---

56. CI tools

- Jenkins, GitLab CI, CircleCI, Travis, GitHub Actions

---

57. Code build in CI – detailed

Example: Java + Maven

Step	Action
1	`mvn clean` → delete target/
2	`mvn compile` → .java → .class
3	`mvn test` → run JUnit

Step	Action
4	`mvn package` → create JAR/WAR
5	Archive: `target/app.war`

> Artifacts stored in Jenkins or Nexus

---

58. Maven goals

Goal	Purpose
`clean`	Remove target
`compile`	Compile code
`test`	Run tests
`package`	Build JAR/WAR
`install`	Install to local repo
`deploy`	Push to Nexus

---

59. Integrate Maven with Jenkins

1. Install **Maven Plugin**
2. Configure JDK & Maven in **Global Tool Config**
3. In job: `Invoke top-level Maven targets` → `clean install`

---

60. Java compilation process

```
.java → javac → .class (bytecode) → JVM → machine code
```

---

61. Use Maven in Jenkins

- Freestyle: Maven build step
- Pipeline: `sh 'mvn clean install'`

---

62. Git branch strategy

Strategy	Use
Git Flow	release, hotfix branches
GitHub Flow	feature → PR → main

Strategy	Use
Trunk-based	short-lived branches

---

63. Pull Request in GitHub

- Propose changes
- Code review
- CI runs
- Merge to main

---

64. Staging code

Ans: `git add .` → moves changes to **staging area** (index)

> Local repo → staging → commit → remote

Ansible

65. What is Ansible? Modules?

Ans: Agentless automation tool using YAML playbooks.

Module examples:

- `copy`, `file`, `service`, `package`, `user`

```
- name: Install Nginx
  package:
    name: nginx
    state: present
```

---

66. Ansible components

- **Inventory** (`hosts.ini`)
- **Playbooks** (YAML)
- **Modules**
- **Roles**
- **Ansible Galaxy**

---

67. Host info in Ansible

Ans: `inventory` file:

```
[webservers]
web1 ansible_host=192.168.1.10 ansible_user=ec2-user
```

> Variables: `ansible_port`, `ansible_ssh_private_key_file`

---

68. Validate Ansible playbook

```
ansible-playbook playbook.yml --syntax-check
ansible-playbook playbook.yml --check  # Dry run
```

---

69. Roles in Ansible

Reusable tasks

Structure:

```
roles/
  webserver/
    tasks/main.yml
    handlers/main.yml
    vars/main.yml
```

Use:

```
- hosts: webs
  roles:
    - webserver
```

---

70. Playbook task → use roles

```
- name: Start Nginx
  include_role:
    name: nginx
    tasks_from: start
```

---

71. What is pipeline?

Ans: Sequence of stages (build → test → deploy) in CI/CD

> Jenkins Pipeline, GitLab CI, GitHub Actions

---

72. Declare variables in Ansible

Location	Example
Playbook	`vars: db_name: prod`
Inventory	`db_host=10.0.0.5`
Vars files	`vars/main.yml`
Vault	`ansible-vault create secrets.yml`

---

73. Run Ansible playbook

```
ansible-playbook site.yml -i inventory.ini
# With vault
ansible-playbook site.yml --ask-vault-pass
```

Docker

74. Dockerfile: Deploy cont.jar using COPY

```
FROM openjdk:11-jre
COPY ./target/cont.jar /app/cont.jar
CMD ["java", "-jar", "/app/cont.jar"]
```

> `COPY . .` copies **current directory** to container workdir

You're now ready to face any follow-up!
Save this as `devops-interview-answers.md` and export to PDF using any method above.

```


**Now just copy the above block and paste into any Markdown → PDF converter!**
Your **professional PDF interview guide** is ready in seconds.
```