1. ***What is the most complex system architecture you have managed which is not bound by confidentiality to discuss?***

It could be something you architected or just something you supported.

Steps to the question:

1. Whiteboard the full architecture of the system.  
2. Explain the system at a high level how requests flowed and data persistence and caching was handled.  
3. Explain the scaling factors of the system – CPU, memory, disk I/O, etc. – How did you monitor and measure it?  
4. Discuss the potential failure points of the system.  
5. Describe how you tested for potential failures and ensured availability.  
6. What was the biggest failure of this system you had to manage and how did you handle it?

How to prepare:

Understand the full system architectures that you work with, not just individual pieces. Practice explaining them and white boarding them. Know scaling factors and failure points of the architectures you maintain inside and out.

1. ***Tell me about a time that you have implemented an effective monitoring solution for a production system.***

***<***Experience Based : Answer> : Discussing Pain Points and how monitoring helped in reducing unwanted downtimes

1. ***What is different about deploying software to 5,000 nodes vs. deploying that same software to 50 nodes?***

What tools you would need, time to deploy, architectural design on two different system, infrastructure foot prints

1. ***What is continuous integration and how would you implement it?***

* In short, it is a development process where code is integrated directly within the deployment process.
* The software is built, tested, and deployed automatically by various tools. Travis CI, Jenkins, Codeship, and Circle CI are a few that come to mind. Each has their pros and cons, but most are set out to do the same thing.

1. ***What is your process for deployment?***

* This can be a very long answer or short. The process starts with understanding the needs of the software in question to deploy. There may be some requirements specific to the build, ie. Node project will need Node, package manager, and other dependencies.
* Understanding what type of continuous integration also whether to use a container like Docker.
* What backing up data looks like. Database is important to have redundancy.
* What kind of machine you’ll be running on Linux, Ubuntu, Windows.
* If it’s a web app what type of web server you’ll be running Apache2, Nginx, etc.
* How you plan on configuring the web server.
* Implement SSH, and or rewrite rules.
* Caching options, caching proxy like CloudFlare.
* Monitoring of server for downtime or errors.

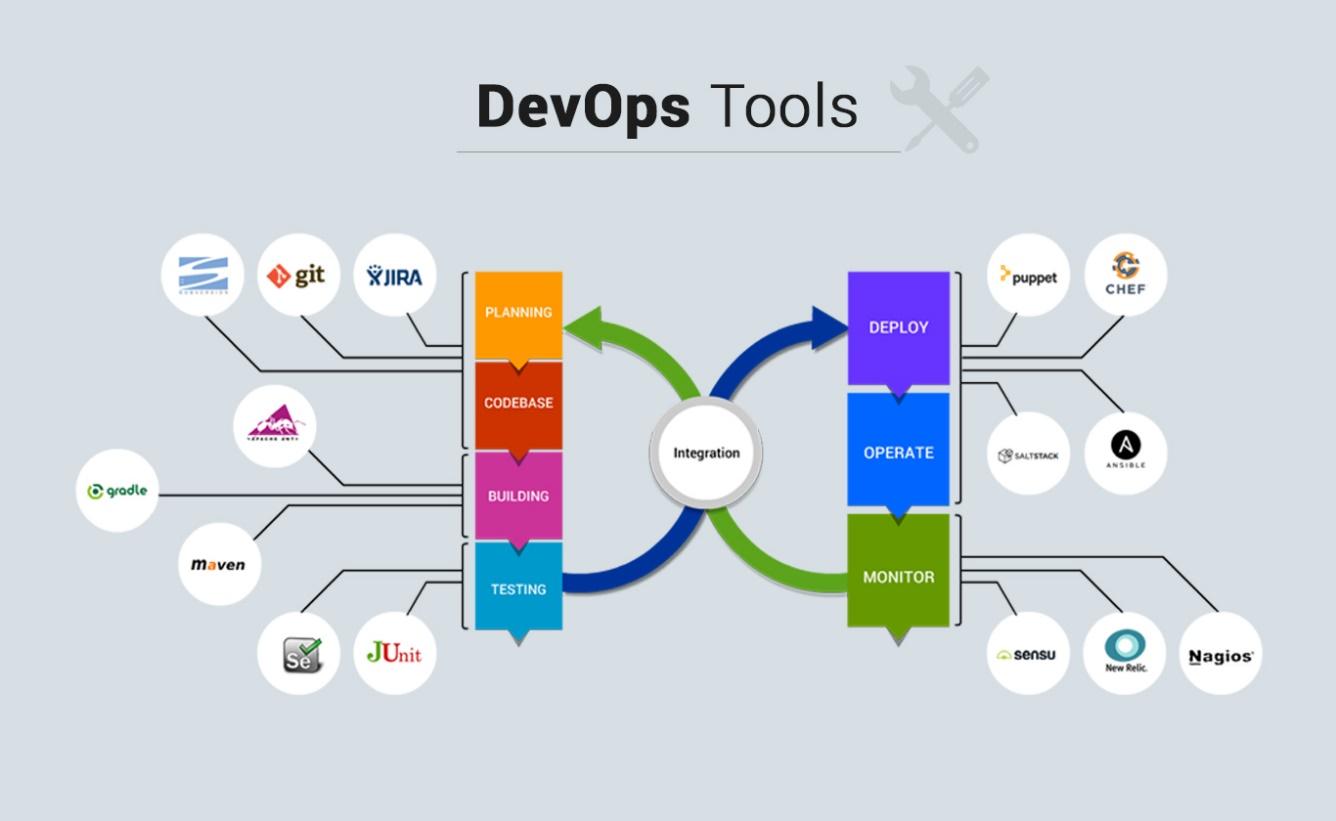
1. ***How do you know if something goes wrong in production and what do you do when it happens?***

* Implementing a monitoring tool for real-time error tracking and downtime is a must (i.e. New Relic, AppDynamics, Sentry, etc.).
* If something goes wrong, you can roll back to a previous state, Backup snapshot. The key is to determine what the problem is. Why did production go down? Running trace routes or checking error logs help to pinpoint issues. Once the problem is located quickly find a solid solution to prevent future issues.

1. ***What are some security measures you consider when deploying?***

* SSH is a must, no reason not to have it.
* Secure all files and directories for minimum permissions necessary.
* Audit and monitor/track logs to identify possible attacks.

1. ***Devops Tools Infographics:***



1. ***Which are the top DevOps tools? Which tools have you worked on?***

The most popular DevOps tools are mentioned below:

* Git : Version Control System tool
* Jenkins : Continuous Integration tool
* Selenium : Continuous Testing tool
* Puppet, Chef, Ansible : Configuration Management and Deployment tools
* Nagios : Continuous Monitoring tool
* Docker : Containerization tool

You can also mention any other tool if you want, but make sure you include the above tools in your answer.  
The second part of the answer has two possibilities:

1. If you have experience with all the above tools then you can say that I have worked on all these tools for developing good quality software and deploying those softwares easily, frequently, and reliably.
2. If you have experience only with some of the above tools then mention those tools and say that I have specialization in these tools and have an overview about the rest of the tools.