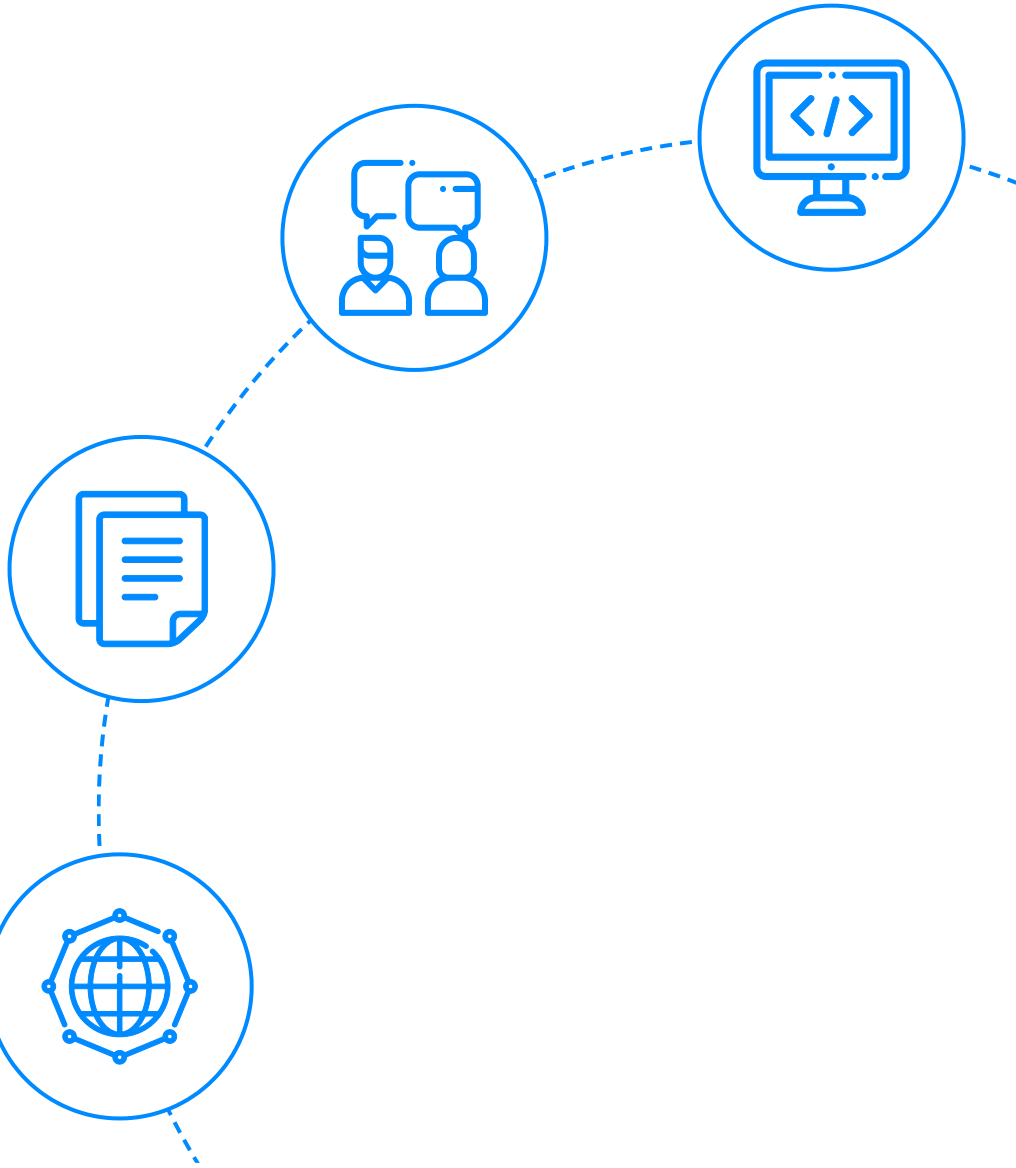
InterviewBit

Performance Testing Interview Questions



To view the live version of the page, click here.

Contents

Performance Testing Interview Questions for Freshers

Performance Testing Interview Questions for Experienced

Performance Testing Interview Questions for Experienced

(.....Continued)

19. Why is it preferred to perform load testing in an automated format?
20. On what kind of values can we perform correlation and parameterization in the LoadRunner tool?
21. How can we identify situations that belong to performance bottlenecks?
22. Can we perform spike testing in JMeter? If yes how?
23. What are the pre-requisites to enter and exit a performance test execution phase?
24. How is load testing different from stress testing?
25. How is endurance testing different from spike testing?
26. What are the best ways for carrying out spike testing?
27. What do you mean by concurrent user hits in load testing?
28. Can the end-users of the application conduct performance testing?
29. What are the metrics monitored in performance testing?
30. What are the common mistakes committed during performance testing?
31. When should we conduct performance testing for any software?
32. What are some of the best tips for conducting performance testing?

Let's get Started

Performance Testing is a process of testing applications for non-functional requirements which helps in determining the speed, stability, responsiveness and scalability of the application under the stipulated workload. A system's performance is one of the main indicators to determine how successfully it is performing in the market. Poor performance results in a bad user experience, leading to a bad reputation and huge losses in revenue which is why it is very much crucial to conduct performance testing.

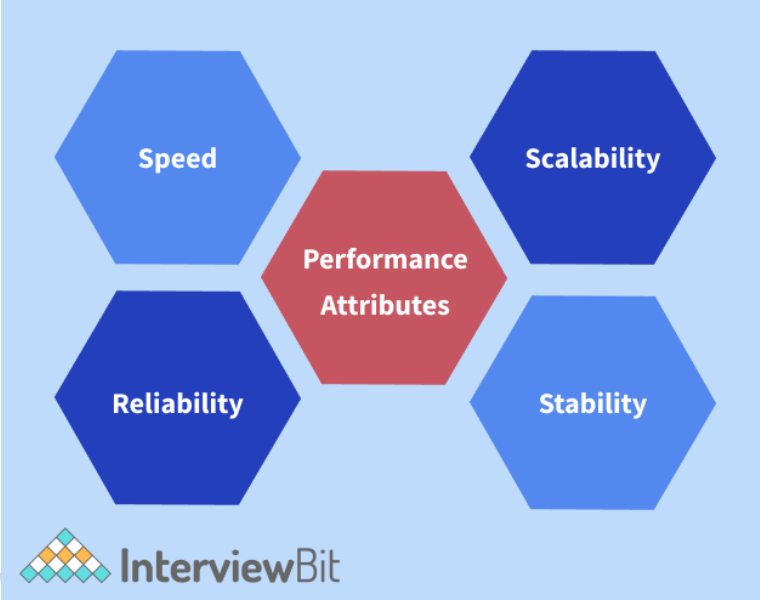
In this article, we will be exploring the most commonly asked performance testing interview questions for both freshers and experienced professionals.

Performance Testing Interview Questions for Freshers

1. What do you understand by Performance Testing?

Performance Testing is a category of software testing that ensures the application performs well under any workload. This type of testing is not done to identify bugs in the application. Its main intention is to eliminate performance issues and bottlenecks by measuring the performance quality attributes of the system.

The following image represents the performance attributes of any system:



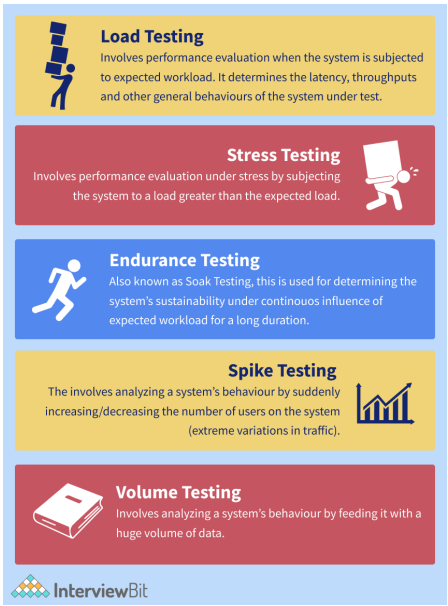
- **Speed** – This determines how fast an application responds to any request.
- **Scalability** – This determines what is the maximum user load an application can handle.
- **Stability** – This determines if an application is stable under differing loads.
- **Reliability** - This determines if an application is consistent under different environmental conditions at a specific period.

2. What are the types of Performance Testing?

The different types of performance testing are:

- **Load testing** – This type of testing checks the ability of an application to perform under known loads. The main goal here is to identify any performance bottlenecks before the application goes into production.
- **Stress testing** – This type of testing involves testing the application's behaviour under extreme stress or workloads for identifying the breaking point of an application.
- **Endurance testing** – This testing is done to ensure that the software can handle the expected load for a continuous period.
- **Spike testing** – This testing is done to ensure that the system works well under the sudden influence of large load spikes.
- **Volume testing** – This testing ensures that the system behaves well under the influence of a large volume of data.

The following image represents the summary of types of performance testing:



3. What are some of the commonly available tools for performance testing?

There are so many tools available for accomplishing performance testing, GUI testing, test management, load testing, functional testing etc. Among various tools, the following are the most commonly used tools for performance testing:

- **QALoad:** This tool is used to perform load tests of web applications, databases and char-based management systems.
- **LoadRunner:** This tool is used to test web applications by using a wide range of platforms and environments. This is used for getting the performance metrics of every component to identify the bottlenecks.
- **WebLOAD (Radview):** This is another tool for running performance and load tests and also comparing the test metrics.
- **Silk performer:** This tool is used for predicting behaviour in terms of complexity and size of e-business before deploying the application.
- **Rational performance tester (IBM):** This tool was developed by IBM and is used to perform automated performance testing of the server and web-based applications.
- **JMeter:** JMeter is a Java-based open source performance testing tool that can be used to test both static and dynamic web applications and resources. It is used for simulating heavy server load for testing the strength and analyzing overall performance under varying load.
- **Flood.io:** This is another load testing tool that is used for executing globally distributed tests on performance from different tools like Selenium, JMeter etc.

4. What are some of the common performance bottlenecks and how do they impact your application?

Bottlenecks are system obstructions that contribute to degradation in the system's performance. They are caused either by hardware issues or coding errors that lead to a reduction of throughput under varying loads. Some of the common bottlenecks in performance are:

- **Processor bottlenecks:** These occur when the processor is overloaded and cannot perform its tasks and respond to requests on time. These can be in 2 forms:
    - CPU running at above 80% capacity for a prolonged period.
    - A long queue of requests for the processor.
      Both of these forms occur when the system has insufficient memory and has continuous interruption from the I/O devices. They can be resolved by adding more RAM, increasing CPU power and improving algorithmic efficiency.
- **Memory Utilization:** This bottleneck occurs when the system does not have fast RAM or does not have sufficient memory. This impacts the speed of serving information to the CPU thereby slowing down the application. Whenever the device does not have enough RAM, the storage will be offloaded to SSD or HDD. These issues can be resolved by increasing the memory capacity or increasing RAM. If the RAM is very slow, then it can be replaced. Sometimes the problem can arise due to memory leak issues wherein a program does not release memory so that the system can use it. Correcting the application program to free memory also can fix the issue.
- **Network bottlenecks:** These occur when two or more devices lack the necessary bandwidth to communicate with each other. Whenever there is an overburdened server or overloaded network that causes the network to lose its integrity. These issues can be resolved by upgrading servers, network hardware like hubs, routers, access points etc.
- **Software bottlenecks:** These bottlenecks are due to the software where programs are built for handling finite tasks so that it doesn't utilize additional RAM or CPU. This makes the program use only a single core or processor despite the availability of resources. These can be resolved by making the software program more efficient so that it can use available resources efficiently.
- **Disk Usage:** The slowest component in a server is long-term storage units like SSDs or HDDs that are unavoidable. The fastest long-term storage units have physical limits which makes it difficult for the programmers to troubleshoot such issues. These can be fixed by increasing RAM caching rates, reducing fragmentation issues or by addressing insufficient bandwidth by switching to faster storage units.

The key to fixing all the bottlenecks are by pinpointing the root cause of the system degradation and then by fixing the code or by adding additional hardware resources depending on the causes.

5. What is the need for conducting performance tests?

Performance testing is conducted for providing information to the stakeholders about the speed, capability, reliability and scalability of the application. These help in identifying what needs to be done for improving the application before it goes to the end-users in the market.

- If the application was released to the market without conducting performance testing, then the issues such as slowness in software, application inconsistencies, application crash under the influence of heavy workloads would not be detected.
- Performance testing helps to determine if the application meets the performance requirements under varying workloads. If an application with poor performance attributes is launched to the market, then it can lead to a bad reputation and loss in sales.
- Whenever we are dealing with mission-critical applications or life-saving systems, then it is very much necessary to conduct performance testing so that the application behaves consistently for a long period.
- Just a 5-minute downtime of **Google.com** in the year 2013 resulted in losses of $545,000. According to **LovetheSales.com**, a downtime of YouTube for 37 minutes cost Google around $1.7m in losses in ad revenues in the year 2020. According to Gremlin, **Amazon.com** loses $13,219,128 per hour due to downtimes. We can see the revenue impact of software downtimes on the companies.

Hence, performance testing is very much crucial before any application is launched to the market.

6. What are some of the common problems that occur due to poor performance?

Following are some of the issues caused due to poor performance:

- **Bottlenecking** – These are obstructions that contribute to system degradation in terms of performance. They occur when there are coding errors or hardware issues that reduces the throughput of the application under certain workloads.
- **Poor response time** – Response time refers to the time taken by the application to respond to any request. For the best user experience, the response times should be very fast. Poor performance of the application leads to slower response times. If response times are slow, then the user loses interest in our application and might turn towards the competitor's application.
- **Long Load time** – Load time refers to the initial time taken by an application to start. This time should be as minimum as possible. Low performance leads to increased load time.
- **Poor scalability** – This impacts software from handling expected user load which makes the application unavailable to some set of users.

7. What do you understand by performance tuning?

Performance Tuning is the process of identifying performance bottlenecks and taking steps to eliminate them. There are two types of tuning, they are:

- **Hardware tuning**: This type of tuning helps in improving the system performance by either replacing, adding or optimizing the hardware (Processor, CPU, RAM etc) of a system to eliminate the bottlenecks caused due to hardware.
- **Software tuning:** This helps identify software bottlenecks by performing code and database profiling. Here, the software code will be modified for resolving the bottlenecks.

8. How is performance testing different from performance engineering?

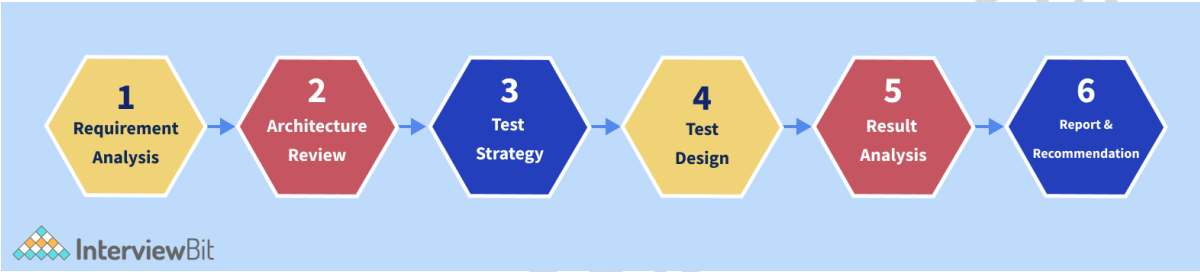
Both of these terms are closely related yet distinct. Performance testing is a subset of performance engineering that primarily deals with gauging the performance of the application under varying loads.

The difference between these two are as follows:

Performance Testing	Performance Engineering
This deals with designing test cases and executing them for determining the performance of an application.	This deals with engineers taking an active part in the SDLC cycle to develop high-performance websites.
It helps in uncovering bugs and performance bottlenecks by providing analysis reports to the developers to resolve them.	This helps developers meet the business requirements and meet the industry standards for reliability, scalability and speed right from the beginning of the application development process.
This makes use of a variety of tools that is only concerned if a website sustains with a given load.	Performance engineering is more of a culture that emphasizes more on using best practices for ensuring that the high-performing applications are developed.
This can be performed using cloud-based tools without requiring the background of programming skills.	To follow best coding practices to deliver high performing apps, it is very much essential to have good programming knowledge.

9. What are the steps involved in conducting performance testing?

Following are the steps involved in the **Performance Testing Lifecycle**:

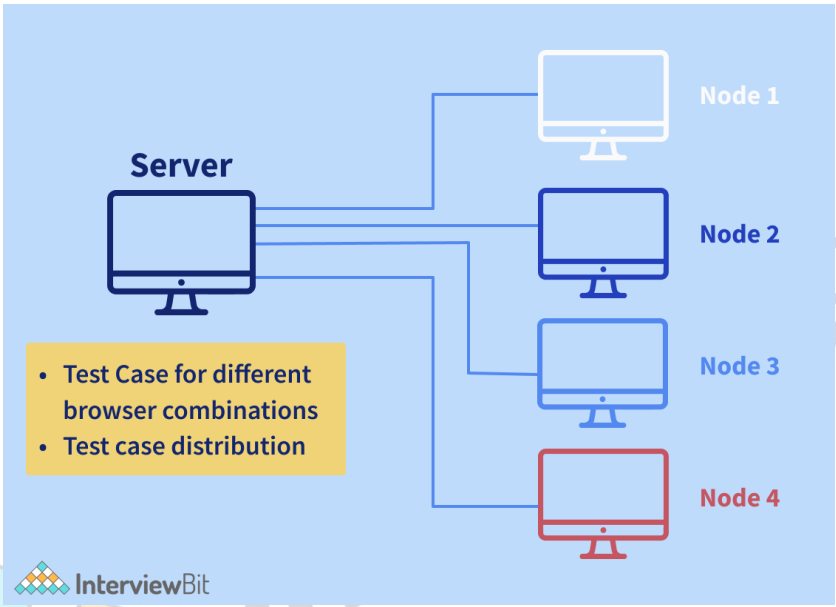


- **Requirement Analysis:** The first step is to determine all the requirements needed for testing after consulting with the clients and developers. This helps in determining the scope and objectives of testing and helps testers to plan out testing accordingly.
- **Architecture Review:** Once all the requirements are gathered and testing is planned, we perform an architectural review of the system under test.
- **Test Strategy:** Once the review is done, carefully layout the strategies required for performance testing that considers the following criteria -
    - Response time of the application
    - Application bottlenecks involving both software and hardware
    - Optimal configuration of the system
    - System capacity and scalability
    - Resource utilization percentage
    - Volume of data and workload capability of the application
- **Test Design:** Once the strategy is ready, the testers have to come up with automation scripts or prepare the testing environment using the tools by following all the best practices and coding standards. The testers have to come up with various test scenarios covering both positive and negative cases.
- **Test Execution:** The scripts prepared in the previous step would be simulated and executed.
- **Results Analysis:** The result of test execution will be analyzed and documented for recording purposes. These metrics will be observed and tracked for any defects.
- **Reports and Recommendations:** The documented results will be presented to the developers along with the recommended resolutions.

10. What do you understand by distributed testing?

Distributed testing is the process of testing applications when lots of users are gaining access to the application from different devices simultaneously. This helps to perform stress testing.

A brief overview of the distributed testing is represented in the image below:



11. What is the metric that determines the data quantity sent to the client by the server at a specified time? How is it useful?

Throughput is the metric that determines the data quantity sent to the client in response to its request by the server. It is calculated in terms of requests per second, hits per second, calls per day, etc. In most cases, it is calculated in bits per second. The value of throughput tells the slowness or fastness of the network and its bandwidth capabilities. The higher the throughput, the higher is the network capability.

12. What do you mean by profiling in performance testing?

Profiling is the process of fine-tuning the performance testing process by helping to identify the system components responsible for most of the issues related to its performance.

13. What is load tuning?

Load tuning is a process of performance improvement technique by conducting necessary modifications to the software configurations depending on the results of load testing.

14. What kind of testing deals with subjecting the application to a huge amount of data?

This kind of testing is known as volume testing. It deals with subjecting the application to a huge amount of data to determine how much data the application can handle when there are a good load of users accessing the application concurrently. It verifies the performance of a system to test whether it can handle a stipulated volume of data by entering huge data volume to the application either incrementally or steadily.

15. What do you know about Scalability testing?

Scalability testing is a type of performance testing that analyzes how well the software is capable of handling complex operational capacity from a simple capacity. Some software takes time to adapt to complex capacities. This testing ensures that the application can scale quickly without any glitches or drawbacks.

16. Why is JMeter used for?

JMeter is a Java-based tool for performing load testing. It helps in analyzing and measuring the performance of web services with the use of plugins. The latest JMeter version is 5.4.2 that requires a Java 8+ version to run.

17. How is performance testing different from functional testing?

Performance Testing	Functional Testing
Helps to validate the system behaviour at different load conditions.	Helps to verify the software accuracy with known inputs and expected output.
This testing is preferred to be done in an automated way.	This testing can be done both manually and automatically.
Several users can perform desired operations.	Here, only one user performs the operations.
Clients, Testers, developers, DBA and Network teams are involved here.	Customer, Tester and Development teams are involved here.
This requires a test environment that is close to the production environment to get the best results.	Here, we do not require a close to the production environment for testing.
Non-functional requirements are tested here.	Functional requirements and business logic is tested here.

Performance Testing Interview Questions for Experienced

18. What are the differences between benchmark testing and baseline testing?

Benchmark Testing is a testing process conducted to compare the system framework performance against set industry standards that are laid by some organizations. Baseline Testing is a type of testing where the tester runs various tests to know the information about the performance. Whenever a change is done in the future, the result of the baseline testing will be considered as a reference point to the next set of testing.

19. Why is it preferred to perform load testing in an automated format?

Performing load testing in a manual way has the following disadvantages:

- Accuracy cannot be predicted easily regarding the application's performance.
- Synchronization among various users becomes challenging to coordinate and maintain.
- In real-time testing, it would require real users to test the application.
- Additionally, manual testing increases the cost of manual effort required.

Due to all the above-mentioned reasons, it is preferred to perform load testing in automated form.

20. On what kind of values can we perform correlation and parameterization in the LoadRunner tool?

Correlation is performed for dynamic values such as session ids, session states, date values etc that are returned from the server in response to any request. Parameterization is conducted upon static data, such as passwords, usernames etc that are usually entered by the user.

21. How can we identify situations that belong to performance bottlenecks?

We can identify performance bottlenecks by monitoring the applications that do not perform well against the stipulated stress and load conditions. We can use LoadRunner software for making use of different monitors that monitor database servers, network delays, firewall monitors etc.

22. Can we perform spike testing in JMeter? If yes how?

Spike Testing is conducted to determine how an application behaves when the number of users accessing the system decreases or increases abruptly. This is because generally when the number of users varies abruptly and suddenly (leading to a spike), then the system behaviour will have unexpected changes. This can be tested in JMeter using Synchronizing Timer. This is simulated by jamming the threads by synchronizing the time until the stipulated number of threads have been blocked and once that is achieved, then release the threads suddenly at once to simulate a large load.

The following steps can be performed:

- Create a performance test plan
- Create a thread group within it
- Add all the JMeter elements specific to business requirements
- Add listeners to view the results
- Run the tests
- Get the results
- Monitor the behaviour.

23. What are the pre-requisites to enter and exit a performance test execution phase?

The necessary **entry criteria** for the execution phase:

- Completed automated scripts
- The test environment should be ready
- Finalized Non-functional requirements (NFR)
- The latest functionally tested code should be deployed
- The test input data should be ready.

The necessary **exit criteria** would be:

- Test cases should cover and meet all the NFRs
- No more performance bottlenecks are present
- All defects are finalized
- The behaviour of the application should be consistent and acceptable in heavy and spiked loads.
- Final reports are submitted and shared.

24. How is load testing different from stress testing?

- **Load testing** is a type of testing that analyzes the software performance when it is saddled with more than normal workloads. The load can be of any kind - data or users accessing the system or the application itself on the server's operating system. When the load is increased, some applications tend to perform slowly (degenerate) and in some other cases, the applications run normally. Load testing determines that the applications run at optimal levels irrespective of the load given to them.
- **Stress testing** on the other hand has a broader approach to the software's performance. It considers the amount of data processed, time taken to process it, network connectivity levels and other applications running in the background. When the stress levels are high, software tends to crash or stop working altogether. This testing imitates the stressful environment to test the resistance of the software to operate correctly.

25. How is endurance testing different from spike testing?

- **Endurance testing** deals with how long the application can endure and perform well irrespective of the loads. Sometimes when an application is used for a long time, it becomes slow or inactive which is why it becomes important to conduct this testing. Endurance testing analyses all changes in the application by simulating lengthy application usage. For example, endurance testing is conducted on a Banking application where we test if it can perform normally under continuous load or large transactions for a long time.
- **Spike testing** deals with pushing the application to the limits by subjecting the software to the highest operation level for identifying the strengths and weaknesses of the application. Spike testing is necessary for instances when eCommerce or shopping sites launch flash sales or holiday discount deals where suddenly a large number of users will be accessing the application. If the application crashes under sudden spike every time, then it would result in a bad user experience and the users would lose faith in the application.

26. What are the best ways for carrying out spike testing?

Spike testing can be carried out by bombarding the application with networking, random connections, data, different operations, firing requests to every single functionality of the application. In this way, the application is pushed to the limits and monitoring can be done to identify if it can work under pressure. The data monitored can be documented and then be analyzed.

27. What do you mean by concurrent user hits in load testing?

Concurrent user hits scenarios arise when more than one user will be hitting or requesting for the same event during the load testing process. This scenario is tested to ensure that multiple users can access the same event requests at the same time in the application.

28. Can the end-users of the application conduct performance testing?

No, end-users cannot conduct performance testing. However, while making use of the software the end-users can discover software bottlenecks. However, that cannot be equated to actual performance testing performed by professional testers. If the end-users want to participate in testing, they can be accommodated in the User Acceptance Testing phase.

29. What are the metrics monitored in performance testing?

Following are the metrics monitored in performance testing:

Metrics	Description
Processor utilization	Time spent by the processor to execute non-idle threads.
Memory usage	Amount of physical memory available so that the server can process.
Disk time	Time taken by the disk to execute read/write request.
Bandwidth	Represented in bits per second (bps) used by network interfaces.
CPU interrupts per second	Average number of hardware interrupts received and processed by the processor each second.
Response time	Time taken to get the first character of the response from the server to the client.
Throughput	Rate at which the server or network receives requests per second.
Amount of connection pooling	Number of user requests met by pooled connections. The higher the number, the better is the performance.
Maximum active sessions	Maximum sessions which are active at once.
Hits per second	Number of hits every second
Thread counts	Number of threads running actively

30. What are the common mistakes committed during performance testing?

Following are some of the mistakes committed during performance testing:

- Unknown or unclear non-functional requirements provided by the business.
- Unclear workload details
- Directly jumping to multi-user tests
- Running test cases for a small duration
- Confusion on an approximation of the concurrent users
- Difference between the test and production environments
- Network bandwidth not stimulated properly
- No clear base-lining of the system configurations.

31. When should we conduct performance testing for any software?

Performance testing is done for measuring the performance of any action in the application. We can run performance tests for checking the performance of the websites and apps. In case we are following waterfall methodology, we can test every time we release a new software's version. If we are using agile methodology, then we need to test continuously.

32. What are some of the best tips for conducting performance testing?

Following are some of the best tips for conducting performance testing:

- The test environment should mirror the production ecosystem as much as possible. If there are any deviations, then the test results might not be accurate and might cause problems when the application goes live.
- It is preferred to have a separate environment dedicated to performance testing.
- The tools selected for testing should automate our test plan in the best possible way.
- Performance tests should be run several times for obtaining a consistent accurate measure of the performance.
- The performance test environment should not be modified in between the testing process.

Conclusion

Performance testing provides in-depth insights regarding the non-functional application requirements like scalability, speed, availability and reliability of the software under test. These help in identifying and resolving the shortcomings and gaps in performance before the application goes live.

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions