1. Node js is Asynchronous
2. Node js is Non-Blocking
3. Node js is single threaded
4. Node js is actually not a framework or a library, but a runtime environment, based on Chrome's V8 JavaScript engine.
5. What is   
   stored procedure:

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

https://www.w3schools.com/sql/sql\_stored\_procedures.asp  
 trigger:

A SQL trigger is a database object which fires when an event occurs in a database. We can execute a SQL query that will "do something" in a database when a change occurs on a database table such as a record is inserted or updated or deleted.

https://www.c-sharpcorner.com/UploadFile/63f5c2/triggers-in-sql-server/  
view:

1. Synchronous means execute top to bottom procedurally its means interpreter wait for completion of block of code and then execute next block.in this may be execution block.
2. Asynchronous means code execute without top to bottom procedurally. Interpreter does not wait for result.if block of code takes more time then this execution goes background and interpreter start execute next block. Node js use function call backs for this purpose. When first block’s execution complete then function call back call to interpreter.
3. **Function call back**:  
   Node. js, being an asynchronous platform, doesn't wait around for things like file I/O to finish - Node. js uses callbacks.  
    A callback is a function called at the completion of a given task; this prevents any blocking, and allows other code to be run in the meantime.

So callback is a function that is passed to another function. When the first function is done, it will run the second function.

Problem:  
Problem is “Call back Hell”.  
**Call back Hell and Solutions:**

The use of nested callbacks makes the code difficult to write and maintain. It also increases the difficulty of identifying the flow of the application.  
**Solution of Call back Hell:**1- Generatorfunction:  
2- Async Waterfall:

https://www.bmc.com/blogs/callback-hell/#:~:text=Callback%20Hell%2C%20also%20known%20as,a%20few%20callbacks%20seem%20harmless.

<https://www.guru99.com/node-js-interview-questions.html>

1. Promises:
2. Async/Await:  
   It is used to get synchronous behavour.

The keyword '**async**' before a function makes the function return a promise, always. And the keyword **await** is **used** inside **async** functions, which makes the program wait until the Promise resolves.

**Call back Vs Promises Vs Aysnc**  
[https://www.loginradius.com/blog/async/callback-vs-promises-vs-async-await/#:~:text=The%20await%20keyword%20is%20used,the%20use%20of%20callbacks%20in%20.&text=In%20using%20async%20and%20await,prepended%20when%20calling%20a%20promise](https://www.loginradius.com/blog/async/callback-vs-promises-vs-async-await/" \l ":~:text=The await keyword is used,the use of callbacks in .&text=In using async and await,prepended when calling a promise).

1. Generator Function:

JavaScript function are expected to run-to-completion – This means once the function starts running, it will run to the end of the function. However, Generators allow us to interrupt this execution and switch to other tasks before returning back to the last interrupted task.

In Call back Hell link it’s defining also

<https://www.guru99.com/node-js-generators-compare-callbacks.html>

1. Arrow function:  
   the arrow functions are only callable and not constructible, i.e arrow functions can never be used as constructor functions. Hence, they can never be invoked with the new keyword.  
   [https://medium.com/better-programming/difference-between-regular-functions-and-arrow-functions-f65639aba256#:~:text=Regular%20functions%20created%20using%20function,be%20used%20as%20constructor%20functions](https://medium.com/better-programming/difference-between-regular-functions-and-arrow-functions-f65639aba256" \l ":~:text=Regular functions created using function,be used as constructor functions).
2. Let VS var   
   var and let are both used for variable declaration in javascript but the difference between them is that var is function scoped and let is block scoped.  
   [https://www.geeksforgeeks.org/difference-between-var-and-let-in-javascript/#:~:text=var%20and%20let%20are%20both,program%20as%20compared%20to%20let](https://www.geeksforgeeks.org/difference-between-var-and-let-in-javascript/" \l ":~:text=var and let are both,program as compared to let).
3. Compare two json objects.  
   JSON.stringify(obj1) === JSON.stringify(obj2)   
   https://www.quora.com/How-can-I-compare-two-JSON-objects-with-their-key-value-pairs-using-JavaScrips  
   https://www.baeldung.com/jackson-compare-two-json-objects
4. If we access particular key in object but this key does not exist then what’s response
5. Destruct object
6. Rest or spread operator  
   spread operator make shallow copy

**Reference Sites For Interview Question:**

1. <https://www.guru99.com/node-js-interview-questions.html>

NARSUN Studios7:01 PM

You need to design a function, which will take an array of integers and will return smallest positive integer in that array.

Constraints: array contains both positive and negative integers, length of array in unspecified, array is not sorted

Example: array = [10, 12, 22, 0, -2, 5, 7, -3]

Answer: 5

function findPositive(array){ let positiveNumber = null; for(let i=0;i<array.length;i++){ if(array[i]>=0 && (array[i] < positiveNumber || positiveNumber == null) ){ positiveNumber = array[i] } } return positiveNumber; } console.log(findPositive([10, 12, 22, 2, -2, 5, 7, -3]));



Write a function to find the length of the non repeating longest subtring in a given string.

e.g abadtca: answer= badtc, length = 5 , abcabb: answer= abc, length=3 , aaaa: answer=a length=1

20. Loops

for (var i =0; i < 10; i++){

console.log(i)

}

for (var i =0; i < 10; i++){

setTimeout(()=>{

console.log(i);

}, 1000);

}

1. Difference between foreach and map .

**14: Node is single threaded but how server for multiple requests.  
https://medium.com/@cinish/nodejs-multiple-client-requests-694d6353218b#:~:text=Multiple%20clients%20make%20multiple%20requests,This%20EventLoop%20is%20single%20threaded.  
https://medium.com/better-programming/is-node-js-really-single-threaded-7ea59bcc8d64**NodeJS Web Server maintains a**limited Thread Pool** to provide services to client requests. Multiple clients make multiple requests to the NodeJS server. NodeJS receives these requests and places them into the **EventQueue .**NodeJS server has an internal component referred to as the **EventLoop**which is an infinite loop that receives requests and processes them. **This EventLoop is single threaded**. In other words, **EventLoop is the listener for the EventQueue**.  
So, we have an event queue where the requests are being placed and we have an event loop listening to these requests in the event queue. What happens next?  
The listener(the event loop) processes the request and if it is able to process the request without needing any blocking IO operations, then the event loop would itself process the request and sends the response back to the client by itself.  
If the current request uses blocking IO operations, the event loop sees whether there are threads available in the **thread pool,**picks up one thread from the thread pool and assigns the particular request to the picked thread. That thread does the blocking IO operations and sends the response back to the event loop and once the response gets to the event loop, the event loop sends the response back to the client.

1. **Cluster in Mongo DB or Database**

23- In which type of applications we use node

24- what is IO into Node

25- how we broadcast in socket

26- how we sent individual in socket

27- what is short and long pulling

28- what we use short pulling or long pulling in socket

29- What is ES6

30- Elastic Search in node.

31- is there index in mongo.

32- difference between in collection and table

33- type script

34- function statement  
 function a(){console.log()}

35- function Expression  
 var b= function (){console.log()}  
function statement and function expression are deferent in respect of hoisting. When code run start “a” was created into memory and contain copy of function while “b” created into memory and contain undefined.  
  
36- function declaration  
function without a name is known as anonymous function.   
it is similar to function statement

37- anonymous function  
 A function without name is known is anonymous function.

function (){}. It will give error.  
 anonymous function just can be used with variable.  
 var b= function (){console.log()}

38- Named function expression.

Anonymous function with named and assigned to variable.  
 var b= function (abc){console.log()}  
39- Difference between parameter and argument  
40- first class function  
41- Arrow function

42- cors in node js

43- call , apply and bind in node js

https://betterprogramming.pub/when-to-use-bind-call-and-apply-in-javascript-1ae9d7fa66d5#:~:text=The%20call()%20and%20apply,that%20can%20be%20executed%20later.