Node-Js:

1. What is Node.js, and what are its advantages?
2. What is the difference between Node.js and JavaScript?
3. What are the key features of Node.js?
4. What is an event loop in Node.js, and how does it work?
5. What is a callback in Node.js? Give an example.
6. What are the different types of streams in Node.js?
7. What is package.json in Node.js?
8. What is the role of the Node.js package manager (NPM)?
9. How can you handle errors in Node.js?
10. How can you create a simple HTTP server in Node.js?
11. What is Express.js, and what are its advantages?
12. What is middleware in Express.js?
13. What is RESTful API, and how can you implement it using Express.js?
14. What is the difference between PUT and POST methods in RESTful API?
15. How can you use Socket.io in Node.js to create real-time applications?
16. What is a module in Node.js?
17. What is the difference between require() and import() in Node.js?
18. How can you test Node.js applications?
19. What are some common security concerns with Node.js, and how can you address them? (Authentication, Validation, JWT Token, Refresh token mechanism)
20. Microservices.
21. Passport JS
22. .Dependency v/s dev dependency Script.
23. Use of Next()
24. ORM Tool
25. Timers in Node JS (setTimeout, setImmediate, setInterval, process.nextTick) setTimeout -> Execute after some delay.

* setImmediate -> Execute before anything else.
* setInterval -> Execute after some interval indefinitely.
* process.nextTick -> Execute in between every event loop.

JavaScript:

1. What is the difference between null and undefined in JavaScript?
2. What is the difference between == and === in JavaScript?
3. What are the different data types in JavaScript?
4. What is hoisting in JavaScript?
5. What is closer in JavaScript?
6. What are deep copy and shallow copy in Javascript?
7. What is mutable and immutable in Javascript?
8. ES6 Features (template literal, spread/rest operator, let, const, arrow function, destructing)
9. What is the difference between an arrow and a regular function in JavaScript?
10. What is the difference between var, let, and const in JavaScript?
11. What is the difference between synchronous and asynchronous code in JavaScript?
12. What is a callback function in JavaScript, and why is it useful?
13. What are higher-order functions in JavaScript, and why are they useful?
14. What are the types of scope and scope chain in JavaScript, and how does it work?
15. What is the prototype in JavaScript, and how does it work?
16. What is a promise and async, await in JavaScript, and how does it work?
17. What is the difference between map, filter, and reduce methods in JavaScript?
18. What is the difference between synchronous and asynchronous code in JavaScript?
19. What is the difference between a for loop and a forEach loop in JavaScript?
20. What is the difference between a local and a global variable in JavaScript?
21. What is the difference between Array and Object in Javascript?

Javascript Logic:

1. Write a function that takes in two integers and returns their sum.
2. Write a function that takes in an array of integers and returns the sum of all the even numbers in the array.
3. Write a function that takes in an array of strings and returns the length of the longest string in the array.
4. Write a function that takes in a string and returns the reverse of the string.
5. Write a function that takes in a string and returns true if the string is a palindrome, and false otherwise.
6. Write a function that takes in a sorted array of integers and a target integer, and returns the index of the target integer in the array, or -1 if the target is not in the array.
7. Write a function that takes in a string and returns true if all the characters in the string are unique, and false otherwise.
8. Write a function that takes in an array of integers and returns a new array with all the duplicates removed.
9. Write a function that takes in an array of integers and returns the second-largest number in the array.
10. Write a function that takes in an array of integers and returns the average of all the numbers in the array.
11. Write a function that takes in two arrays of integers and returns a new array that contains only the elements that are common to both arrays.
12. Write a function that takes in a string and returns true if the string contains only unique characters, and false otherwise.
13. Write a function that takes in a sorted array of integers and a target integer and uses binary search to find the index of the target integer in the array. If the target is not in the array, return the index where it would be if it were inserted in the correct order.
14. Write a function that takes in a string and returns the first non-repeating character in the string. If there are no non-repeating characters, return null.
15. Write a function that takes in an array of integers and returns a new array that contains only the elements that appear more than once in the original array.
16. Write a function that takes in two sorted arrays of integers and returns a new array that contains all the elements from both arrays, in sorted order.
17. Write a function that takes in a string and returns the longest palindrome that can be made from any combination of characters in the string.
18. Write a function that takes in an array of integers and returns the second smallest integer in the array.
19. Write a function that takes in a string and returns the first recurring character in the string. If there are no recurring characters, return null.
20. Write a function that takes in an array of integers and returns the sum of the two largest integers in the array.

Typescript:

* 1. What is TypeScript and how does it differ from JavaScript?
  2. How do you define a variable in TypeScript?
  3. What are the basic data types in TypeScript?
  4. How do you define a function in TypeScript?
  5. How do you declare and use interfaces in TypeScript?
  6. What is the difference between an interface and a class in TypeScript?
  7. How do you create and use decorators in TypeScript?
  8. How do you implement inheritance and polymorphism in TypeScript?
  9. How does TypeScript support generics?

[Inheritance](https://www.w3schools.com/java/java_inheritance.asp) lets us inherit attributes and methods from another class.

Polymorphism uses those methods to perform different tasks.

Mongodb:

* 1. What is MongoDB?
  2. What are the advantages of using MongoDB?
  3. What are the disadvantages of using MongoDB?
  4. How does MongoDB differ from a relational database?
  5. What is a document in MongoDB?
  6. What is a collection in MongoDB?
  7. What is the syntax for inserting data into a MongoDB collection?
  8. How do you update a document in MongoDB?
  9. How do you delete a document in MongoDB?
  10. What is an index in MongoDB and why is it important?
  11. What is sharding in MongoDB and how does it work?
  12. What is replication in MongoDB and how does it work?
  13. What is lookup?
  14. What is projection?
  15. How do you perform aggregation in MongoDB?
  16. What is the difference between MongoDB and MySQL?
  17. What is Primary Key?
  18. What is Composite Key?
* Find all documents in the collection products where the price is greater than 100:

db.products.find({price: {$gt: 100}})

* Find the names of all products that start with the letter 'A':

db.products.find({name: {$regex: /^A/}})

* Fetch top 10 documents in the collection products sorted by price in descending order:

db.products.find().sort({price: -1}).limit(10)

* Retrieve the name and price fields in a single document as “ProductInfo”. The first name and the last name must be separated with space:

db.products.find({}, {name: 1, price: 1}).projection({\_id: 0, FullName: {'$concat': ['$name', ' ', '$price']}})

* Fetch the name field from the products collection in the upper case and use the ALIAS name as ProductName:

db.products.find({}, {name: 1}).projection({\_id: 0, ProductName: {$upper: '$name'}})

* Fetch the number of products in the collection products where the category is 'Electronics':

db.products.find({category: 'Electronics'}).count()

* Get the current date:

db.currentDate()

* Retrieve the first four characters of the name field in the products collection:

db.products.find({}, {name: 1}).projection({\_id: 0, name: {$substr: ['$name', 0, 4]}})

* Fetch only the place name(string before brackets) from the address field in the products collection:

db.products.find({}, {address: 1}).projection({\_id: 0, address: {$substr: ['$address', 1, INSTR('$address', '(') - 1]}})

* Create a new collection new\_products that consists of data and structure copied from the products collection:

db.products.copyTo('new\_products')

SQL:

* 1. What is SQL and what are its uses?
  2. What is a database and what are the different types of databases?
  3. What is a primary key and why is it important?
  4. What is a foreign key and how is it used?
  5. What is a join in SQL and what are the different types of joins?
  6. What is a subquery in SQL and how is it used?
  7. What is a view in SQL and how is it used?
  8. What are normalization and denormalization in SQL?
  9. What is an index in SQL and why is it important?
  10. What is a stored procedure in SQL and how is it used?
  11. What is a trigger in SQL and how is it used?
  12. What is the difference between an inner join and an outer join?
  13. What is the difference between char and varchar?
  14. What is the difference between a where clause and a having clause?
  15. What is the constraint?
  16. What is an alias?

**Q1.Write an SQL query to show the last record from a table using sub.**Select \* from Worker where WORKER\_ID = (SELECT max(WORKER\_ID) from Worker);  
**Q2. Write q query to find all the employees whose salary is between 50000 to 100000.**SELECT \* FROM EmployeePosition WHERE Salary BETWEEN '50000' AND '100000';  
**Q2.** **Find the names of employees that begin with 'S'**SELECT name FROM employees WHERE name LIKE 'S%';  
**Q3. Fetch top N records.**SELECT \* FROM employees ORDER BY salary DESC LIMIT 10;  
**Q4. Retrieve the EmpFname and EmpLname in a single column as “FullName”. The first name and the last name must be separated with space:**  
SELECT CONCAT(empfname, ' ', empLname) AS FullName FROM employees;  
**Q5. Fetch the number of employees working in the department 'HR'.**  
SELECT COUNT(\*)FROM employees WHERE department = 'HR';  
**Q6.** **Create a new table that consists of data and structure copied from the other table.**CREATE TABLE new\_table AS SELECT \* FROM old\_table;

**Q15. Write a query to fetch details of employees with the address as “DELHI(DEL)”.**SELECT \* FROM EmployeeInfo WHERE Address LIKE 'DELHI(DEL)%';

**Q16. Write a query to fetch all employees who also hold the managerial position.**SELECT E.EmpFname, E.EmpLname, P.EmpPosition FROM EmployeeInfo E INNER JOIN EmployeePosition P ON E.EmpID = P.EmpID AND P.EmpPosition IN ('Manager');  
  
**Q17.** **Write a query to fetch the department-wise count of employees sorted by department’s count in ascending order.**SELECT Department, count(EmpID) AS EmpDeptCount FROM EmployeeInfo GROUP BY Department ORDER BY EmpDeptCount ASC;

**Q18. Write a query to calculate the even and odd records from a table.**   
SELECT EmpID FROM (SELECT rowno, EmpID from EmployeeInfo) WHERE MOD(rowno,2)=0;

Q19. To retrieve the even records from a table, you have to use the MOD() function as follows:  
SELECT EmpID FROM (SELECT rowno, EmpID from EmployeeInfo) WHERE MOD(rowno,2)=0;

Q20.Write a query to retrieve the odd records from a table.  
  
SELECT EmpID FROM (SELECT rowno, EmpID from EmployeeInfo) WHERE MOD(rowno,2)=1;

Q21. Write a SQL query to retrieve employee details from EmployeeInfo table who have a date of joining in the EmployeePosition table.  
SELECT \* FROM EmployeeInfo AS E WHERE EXISTS(SELECT \* FROM EmployeePosition P WHERE E.EmpId = P.EmpId);Q22. Write a query to retrieve two minimum and maximum salaries from the EmployeePosition table.  
  
SELECT DISTINCT Salary FROM EmployeePosition E1 WHERE 2 >= (SELECTCOUNT(DISTINCT Salary)FROM EmployeePosition E2 WHERE E1.Salary >= E2.Salary) ORDER BY E1.Salary DESC;   
  
Q23.Write a query to retrieve two maximum salaries, you can write a query as below:

SELECT DISTINCT Salary FROM EmployeePosition E1 WHERE 2 >=(SELECTCOUNT(DISTINCT Salary) FROM EmployeePosition E2 WHERE E1.Salary <= E2.Salary) ORDER BY E1.Salary DESC;

**Q21.** **Write a query to find the Nth highest salary from the table without using TOP/limit keyword.**

SELECT Salary FROM EmployeePosition E1 WHERE N-1 = ( SELECT COUNT( DISTINCT ( E2.Salary ) )FROM EmployeePosition E2 WHERE E2.Salary >  E1.Salary );  
 **Q22. Write a query to retrieve duplicate records from a table.**

SELECT EmpID, EmpFname, Department COUNT(\*) FROM EmployeeInfo GROUP BY EmpID, EmpFname, Department HAVING COUNT(\*) > 1;  
 **Q23. Write a query to retrieve the list of employees working in the same department.**

Select DISTINCT E.EmpID, E.EmpFname, E.Department FROM EmployeeInfo E, Employee E1 WHERE E.Department = E1.Department AND E.EmpID != E1.EmpID;

**Q24. Write a query to retrieve the last 3 records from the EmployeeInfo table.**

SELECT \* FROM EmployeeInfo WHERE EmpID <=3 UNION SELECT \* FROM (SELECT \* FROM EmployeeInfo E ORDER BY E.EmpID DESC) AS E1 WHERE E1.EmpID <=3;

**Q25. Write a query to find the third-highest salary from the EmpPosition table.**

SELECT TOP 1 salary FROM( SELECT TOP 3 salary FROM employee\_table ORDER BY salary DESC) AS emp ORDER BY salary ASC;  
  
**Q26. Write a query to display the first and the last record from the EmployeeInfo table.**

-To display the first record from the EmployeeInfo table, you can write a query as follows:

SELECT \* FROM EmployeeInfo WHERE EmpID = (SELECT MIN(EmpID) FROM EmployeeInfo);  
  
-To display the last record from the EmployeeInfo table, you can write a query as follows:

SELECT \* FROM EmployeeInfo WHERE EmpID = (SELECT MAX(EmpID) FROM EmployeeInfo); **Q27. Write a query to add email validation to your database**

SELECT Email FROM EmployeeInfo WHERE NOT REGEXP\_LIKE(Email, ‘[A-Z0-9.\_%+-]+@[A-Z0-9.-]+.[A-Z]{2,4}’, ‘i’);  
 **Q28. Write a query to retrieve Departments who have less than 2 employees working in it.**

SELECT DEPARTMENT, COUNT(EmpID) as 'EmpNo' FROM EmployeeInfo GROUP BY DEPARTMENT HAVING COUNT(EmpD) < 2;  
  
**Q29. Write a query to retrieve EmpPostion along with total salaries paid for each of them.**

SELECT EmpPosition, SUM(Salary) from EmployeePosition GROUP BY EmpPosition;

**Q30. Write a query to fetch 50% records from the EmployeeInfo table.**

SELECT \* FROM EmployeeInfo WHERE EmpID <= (SELECT COUNT(EmpID)/2 from EmployeeInfo);

* ASW LAMDA

1. What is AWS Lambda?
2. What are the benefits of using AWS Lambda?
3. What are the limitations of AWS Lambda?
4. How does AWS Lambda handle cold starts?
5. How does AWS Lambda scale?
6. How do you configure triggers for AWS Lambda functions?
7. How do you monitor AWS Lambda functions?
8. How do you troubleshoot AWS Lambda functions?
9. What are some best practices for using AWS Lambda?
10. What are some common mistakes to avoid when using AWS Lambda?
11. Have you ever used AWS Lambda in a production environment?
12. Can you explain how you would use AWS Lambda to build a serverless application?
13. Can you discuss the trade-offs between using AWS Lambda and traditional server-based architectures?
14. What are some challenges you have faced when using AWS Lambda?
15. How would you improve AWS Lambda?
16. What is AWS Lambda, and how does it fit into the serverless computing model?
17. Can you explain the key components of an AWS Lambda function, including event sources, triggers, and execution environments?
18. How do you handle errors and exceptions in AWS Lambda functions, and what are some best practices for error handling?
19. What are the different programming languages supported by AWS Lambda, and when might you choose one language over another for a specific use case?
20. Explain the concept of cold starts in AWS Lambda. How can you mitigate the impact of cold starts on the performance of your serverless applications?
21. What is the maximum execution timeout for an AWS Lambda function, and how can you extend it for long-running tasks?
22. Describe how you can control access and permissions for AWS Lambda functions using IAM roles and policies.
23. How do you monitor and troubleshoot AWS Lambda functions in a production environment? What AWS services and tools can you use for this purpose?
24. Can you explain the difference between synchronous and asynchronous invocation of AWS Lambda functions? When would you choose one over the other?
25. What is the AWS Lambda Layers feature, and how can it be used to share code and dependencies among multiple Lambda functions?
26. How do you configure and manage environment variables in AWS Lambda, and why are they important?
27. Explain the benefits and use cases of AWS Step Functions in conjunction with AWS Lambda functions.
28. What is the AWS Serverless Application Model (SAM), and how can it simplify the deployment and management of serverless applications?
29. Describe the pricing model for AWS Lambda, including the factors that affect the cost of running Lambda functions.
30. Can you provide an example of a real-world project where you used AWS Lambda to solve a specific problem? What challenges did you face, and how did you address them?

S3

* What is Amazon S3?
* What are the different storage classes available in S3?
* What is the difference between S3 Standard and S3 Glacier?
* What is versioning in S3?
* How can you restore a deleted object in S3?
* What is a lifecycle policy in S3?
* What is object lock in S3?
* How can you encrypt data in S3?
* What are the different ways to access S3?
* What are the best practices for securing S3 buckets?
* What are some of the challenges of using S3?

GIT  
 What is Git?

* What are the benefits of using Git?
* What are the different types of Git branches?
* What is the difference between git pull and git fetch?
* How do you resolve merge conflicts in Git?
* What is a commit?
* What is a tag?
* What is a remote repository?
* What is a stash?
* How do you clone a Git repository?
* How do you push changes to a remote repository?
* How do you create a pull request?
* What are some best practices for using Git?

Here are some additional questions that you may be asked, depending on your experience level and the role you are applying for:

* How would you use Git to collaborate on a project with other developers?
* How would you use Git to manage a large codebase?
* How would you use Git to track changes to your code?
* How would you use Git to deploy your code to production?
* How would you use Git to troubleshoot a code issue?
* How would you use Git to automate your development workflow?