**1.Which languages can we use with MongoDB?**

**Ans:** At Present, MongoDB offers driver support to C++, Java, PHP, Perl, Python, Go, Scala, and Ruby.

**2.Define Collection?**

**Ans:** The collection is a set of MongoDB documents.

**3.What is the purpose of the save() method?**

**Ans:** We use the save() method for replacing the existing documents with new documents.

**4.What is the purpose of the DB command?**

**Ans:** We use the “DB” command to get the name of the presently selected database.

**5.Which command do we use for creating the backup of the database?**

**Ans:** We use the mongodump command for creating the database backup.

**6.Which command do we use for restoring the backup?**

**Ans:** We use mongorestore for restoring the backup.

**7.What is the syntax of the limit()method?**

**Ans:** Syntax of the limit() method is:

>db.COLLECTION\_NAME.find().limit(NUMBER)

**8.What is the syntax of the sort() method?**

**Ans:** Syntax of the sort() method is:

>db.COLLECTION\_NAME.find().sort({KEY:1})

**9.Which command do we use for dropping a database?**

**Ans:** We use the “DB.drop database” command for dropping a database.

**10.Explain MongoDB Projection?**

**Ans:** In MongoDB, we use Projection for selecting only the required data. It will not select the complete data of a document.

**11.Why do we use the pretty() method?**

**Ans:** We use the pretty() method for displaying the results in a formatted way.

**12.How do we remove a document from the collection?**

**Ans:** By using the remove() method, we remove a document from the collection.

**13.Does MongoDB require plenty of RAM?**

**Ans:** No, MongoDB does not require plenty of RAM. It can run on a small amount of memory. MongoDB dynamically assigns and unassigns RAM according to the needs of other processes.

**14.What are the storage engines used by MongoDB?**

**Ans:** WiredTiger and MMAPv1 are the two storage engines used by MongoDB.

**15.How do we configure the cache size in MongoDB?**

**Ans:** In MongoDB, we cannot configure the cache. MongoDB utilizes the free spaces over the system automatically by using memory-mapped files.

**16.How do we use a primary key in MongoDB?**

**Ans:** “\_id field” is reticent for a primary key in MongoDB. And it is a distinct value. If we do not set anything to the “\_id”, it will systematically fill it with the “MongoDB Id Object”. Yet, we can store any distinct information in that field.

**17.How do we see the connections utilized by MongoDB?**

**Ans:** For seeing the connections utilized by MongoDB, we use db\_adminCommand(”connPoolStats”).

**18.How do applications access the real-time data modifications in MongoDB?**

**Ans:** Applications access the real-time data modifications through the Change streams that serve as the subscriber for every collection operation like delete, insert, and update.

**19.Define BSON?**

**Ans:** Binary JSON or BSON is a binary-encoded format of the JSON. BSON extends the JSON and offers various data fields and types.

**20.How does MongoDB store the data?**

**Ans:** As it is a document-based database, MongoDB stores the documents in Binary Javascript Object Notation or BSON, which is a binary-encoded format of JSON.

**21.Does MongoDB support ACID Transaction? Define ACID Transaction?**

**Ans:** Yes, MongoDB supports ACID Transaction. ACID refers to Atomicity, Consistency, Isolation, and Durability. Transaction manager assures that we handle these attributes.

**22.Can we run more than one Javascript Operation in one MongoDB instance?**

**Ans:** Yes, we can run multiple javascript operations in one MongoDB instance.

23.What is a namespace?

**Ans:** A namespace is the concatenation of the database name and collection name.

**24.Name the two storage engines using MongoDB?**

MMAPv1 and WiredTiger are the two storage engines used by MongoDB.

**25.How do you create and drop a collection in MongoDB?**

**Ans:** Create collection: db.createCollection();

Drop collection: db.collection.drop();

**26.How can you store images, videos and other large files in MongoDB?**

**Ans:** Large files are stored in MongoDB using the GridFS specification.

**27.What is the command to list all the indexes in a collection?**

**Ans:** The command is db.collection.getIndexes();

**28.List some of the data types supported by MongoDB.**

**Ans:** Some data types are numbers, string, arrays, binary data, booleans, date, regular expressions, ObjectId, etc.

**29.Does MongoDB Support foreign key constraints?**

**Ans:** No, MongoDB doesn’t support foreign key constraints. Because of the document structure, MongoDB provides flexible ways to define relationships.

**30.Name the default index created for a new collection.**

**Ans:** The default index created for the new collection is \_id

**31.How do you manage packages in your node.js project?**

**Ans:** It can be managed by a number of package installers and their configuration file accordingly. Out of them mostly use npm or yarn. Both provide almost all libraries of javascript with extended features of controlling environment-specific configurations. To maintain versions of libs being installed in a project we use package.json and package-lock.json so that there is no issue in porting that app to a different environment.

**32.What is Node.js and how it works?**

**Ans:** Node.js is a virtual machine that uses JavaScript as its scripting language and runs Chrome’s V8 JavaScript engine. Basically, Node.js is based on an event-driven architecture where I/O runs asynchronously making it lightweight and efficient. It is being used in developing desktop applications as well with a popular framework called electron as it provides API to access OS-level features such as file system, network, etc

**33.Why is Node.js single-threaded?**

**Ans:** Node.js was created explicitly as an experiment in async processing. This was to try a new theory of doing async processing on a single thread over the existing thread-based implementation of scaling via different frameworks.

**34.What is middleware?**

**Ans:** Middleware comes in between your request and business logic. It is mainly used to capture logs and enable rate limit, routing, authentication, basically whatever that is not a part of business logic. There are third-party middleware also such as body-parser and you can write your own middleware for a specific use case.

**35. Is Node a single threaded application?**

**Ans:** Yes. Node is a single-threaded application with event looping.

**36.What is an asynchronous API?**

**Ans:** All the API's of Node.js library are asynchronous means non-blocking. A Node.js based server never waits for an API to return data. The Node.js server moves to the next API after calling it, and a notification mechanism of Events of Node.js responds to the server for the previous API call.

**37.Does Node.js provide Debugger?**

**Ans:** Yes, Node.js provides a simple TCP based protocol and built-in debugging client. For debugging your JavaScript file, you can use debug argument followed by the js file name you want to debug.

**38.What is a control flow function?**

**Ans:** Control flow function is a generic piece of code that runs in between several asynchronous function calls.

**39.Is it possible to access DOM in Node?**

**Ans:** No, it is not possible to access DOM in Node.

**40.Explain what dynamic routing is in Express.js.**

**Ans:** Shortlisted applicants with extensive knowledge of Express.js should know that dynamic routing involves moving from one page to another depending on the client's request. It works by passing a parameter in a URL or using parameterized URLs to determine how to identify the path.

**41.How would you install Express.js?**

**Ans:** Since Express.js can run on a Node.js runtime environment, applicants should know that additional software is not required to install it. All that’s necessary is to install it as a module and run the Express.js module by entering the following command:

npm install Express --save

**42.What Type Of Web Application Can Built Using Express Js?**

**Ans:** you can build single-page, multi-page, and hybrid web applications.

43. **What is Rest API ?**

**Ans:** A REST API is an application programming interface that adheres to the constraints of REST architectural style and enables interaction with RESTful web services. Interconnected networks make up the web. A web service is a set of open protocols and standards used for exchanging data between client-server applications

44. **What is JSON?**

**Ans:** JSON stands for JavaScript Object Notation.JSON is a lightweight format for storing and transporting data.JSON is often used when data is sent from a server to a web page.JSON is "self-describing" and easy to understand

45. **Uses of JSON:**

**Ans:**

* It is used while writing JavaScript based applications that includes browser extensions and websites.
* JSON format is used for serializing and transmitting structured data over network connection.
* It is primarily used to transmit data between a server and web applications.

Web services and APIs use JSON format to provide public data.

46. **Characteristics of JSON:**

**Ans:**

* JSON is easy to read and write.
* It is a lightweight text-based interchange format.
* JSON is language independent.

47. **JSON - Data Types:**

**Ans:**

In JSON, values must be one of the following data types:

* a string
* a number
* an object (JSON object)
* an array
* a boolean
* null

48. **When to use the GET method?**

**Ans:**

* GET is used to request something from a server with less amount of data to pass.
* When nothing should change on the server because of your action.
* When request only retrieves data from a web server by specifying parameters
* Get method only carries request url & header not request body.

49. **When to use the POST method?**

**Ans:**

* POST should be used when the server state changes due to that action.
* When a request needs its body, to pass a large amount of data.
* When want to upload documents , images , video from client to server

50. **Request Rate limit- Throttling :**

**Ans:** We need to make sure our APIs are running as efficiently as possible. Otherwise, everyone using your database

will suffer from slow performance. Performance isn’t the only reason to limit API requests, either. API limiting, which also known

as rate is limiting, is an essential component of Internet security, as DoS attacks can tank a server with unlimited API requests.

Rate limiting also helps make your API scalable. If your API blows up in popularity, there can be unexpected spikes in traffic,

causing severe lag time.