1. **Explain why selenium is important in web scraping.**

Ans:- Selenium is a Web Browser Automation Tool.

Primarily, it is for automating web applications for testing purposes, but is certainly not limited to just that. It allows you to open a browser of your choice & perform tasks as a human being would, such as:

- Clicking buttons
- Entering information in forms
- Searching for specific information on the web pages

2. **What's the difference between scraping images and scraping websites? Use an example to demonstrate your point.**

Ans:- **What is Web Scraping?**

Web Scraping refers to the **extraction of data** from a website or webpage. Usually, this data is extracted on to a new file format. For example, data from a website can be extracted to an excel spreadsheet.
Web Scraping can also be done manually, although in most cases automated tools will be used to extract the data.

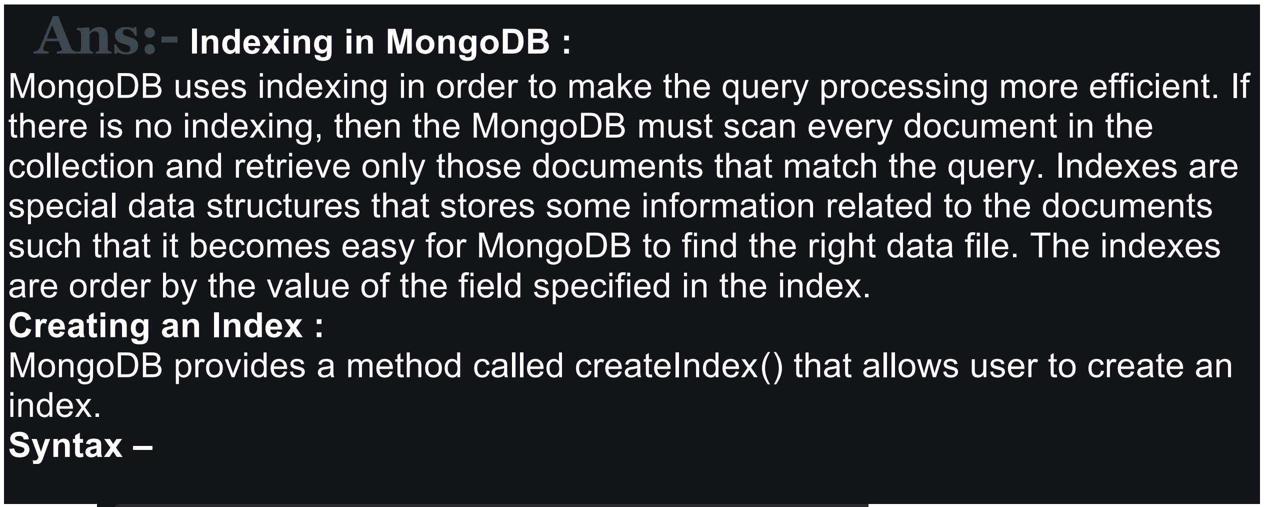
One key aspect of Web Scraping is that it is often done with a focused approach. This means that Web Scraping projects seek to extract specific data sets from a website for further analysis.

For example, a company might extract product details from laptops listed on Amazon in order to figure out how to position their new product in the market.

What is image scraping?

Image scraping is a subset of the web scraping technology. While web scraping deals with all forms of web data extraction, image scraping only focuses on the media side – images, videos, audio, and so on.

3. Explain how MongoDB indexes data.



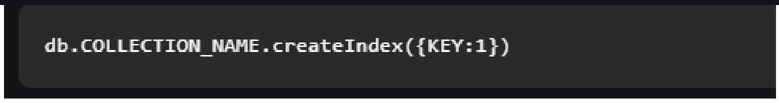
Ans:- **Indexing in MongoDB :**

MongoDB uses indexing in order to make the query processing more efficient. If there is no indexing, then the MongoDB must scan every document in the collection and retrieve only those documents that match the query. Indexes are special data structures that stores some information related to the documents such that it becomes easy for MongoDB to find the right data file. The indexes are order by the value of the field specified in the index.

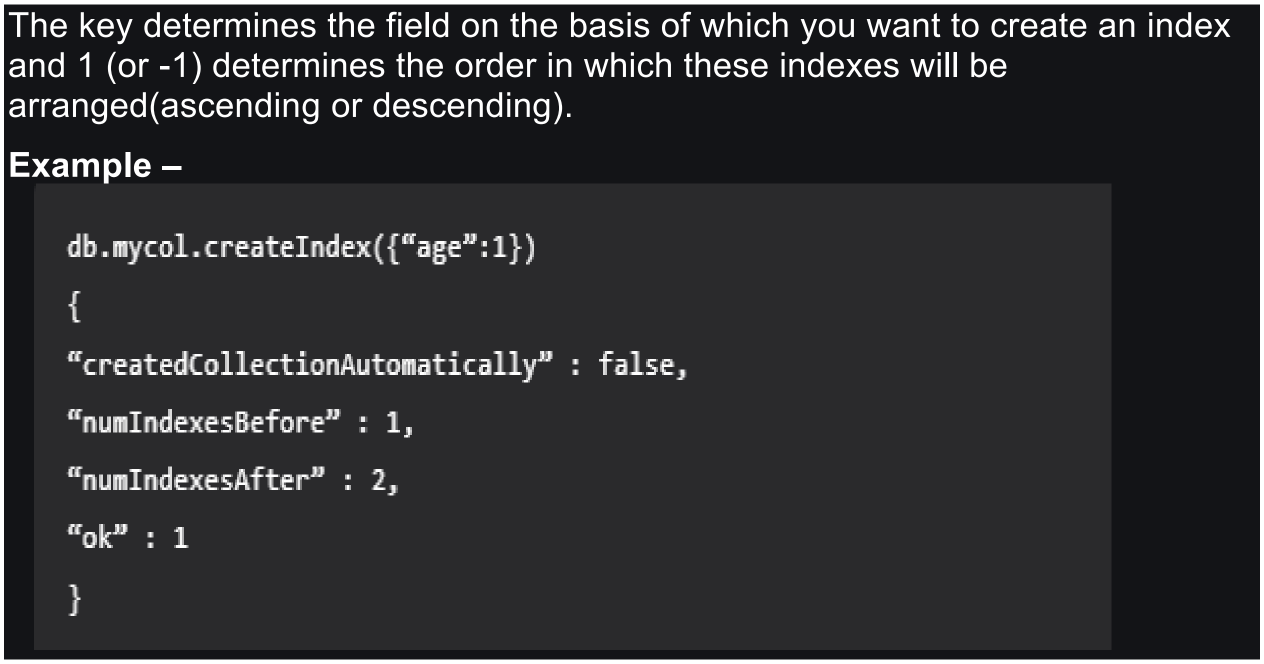
Creating an Index :

MongoDB provides a method called createIndex() that allows user to create an index.

Syntax –



```
db.COLLECTION_NAME.createIndex({KEY:1})
```



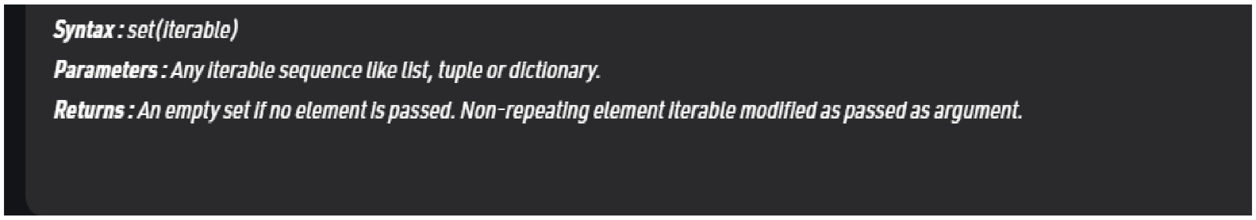
The key determines the field on the basis of which you want to create an index and 1 (or -1) determines the order in which these indexes will be arranged(ascending or descending).

Example –

```
db.mycol.createIndex({“age”:1})
{
“createdCollectionAutomatically” : false,
“numIndexesBefore” : 1,
“numIndexesAfter” : 2,
“ok” : 1
}
```

4.What is the significance of the SET modifier?

Ans:- Set, a term in mathematics for a sequence consisting of distinct language is also extended in its language by Python and can easily be made using set().
set() method is used to convert any of the iterable to sequence of iterable elements with distinct elements, commonly called Set.



Syntax : *set(iterable)*

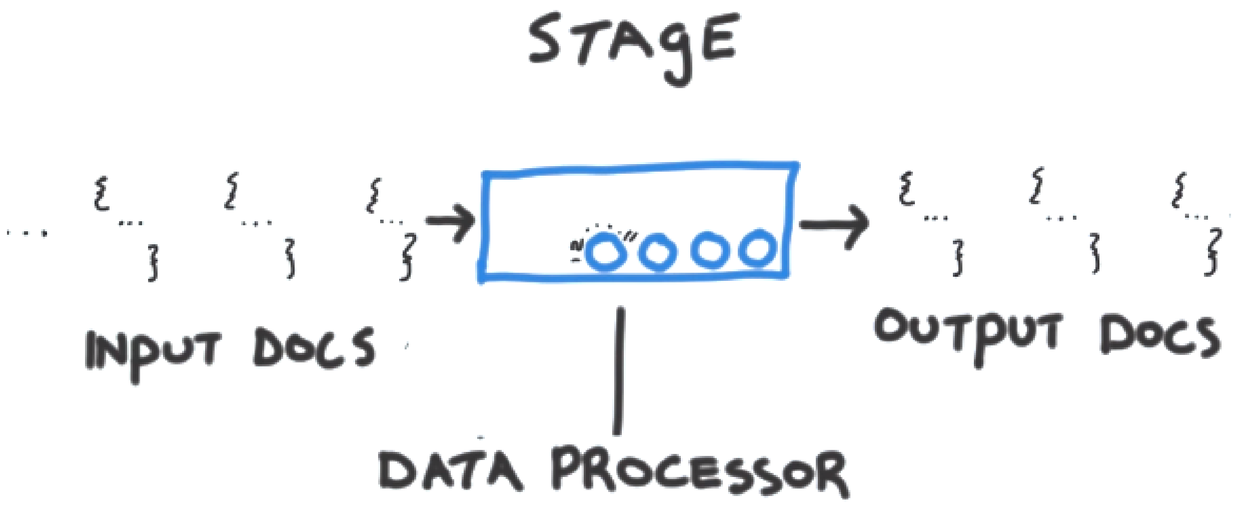
Parameters : *Any iterable sequence like list, tuple or dictionary.*

Returns : *An empty set if no element is passed. Non-repeating element iterable modified as passed as argument.*

Don't worry if you get an unordered list from the set. Sets are unordered. Use sorted(set(sampleList)) to get it sorted

5. Explain the MongoDB aggregation framework?

Ans: The aggregation framework is a set of analytics tools within MongoDB that allows us to run various types of reports or analysis on documents in one or more collections. Based on the idea of a pipeline. We take input from a MongoDB collection and pass the documents from that collection through one or more stages, each of which performs a different operation on it's inputs. Each stage takes as input whatever the stage before it produced as output. And the inputs and outputs for all stages are a stream of documents. Each stage has a specific job that it does. It's expecting a specific form of document and produces a specific output, which is itself a stream of documents. At the end of the pipeline, we get access to the output.



An individual stage is a data processing unit. Each stage takes as input a stream of documents one at a time, processes each document one at a time and produces the output stream of documents. Again, one at a time. Each stage provide a set of knobs or tunables that we can control to parameterize the stage to perform whatever task we're interested in doing. So a stage performs a generic task - a general purpose task of some kind and parameterize the stage for the particular set of documents that we're working with. And exactly what we would like that stage to do with those documents. These tunables typically take the form of operators that we can supply that will modify fields, perform arithmetic operations, reshape documents or do some sort of accumulation task as well as a veriety of other things. Often times, it the case that we'll want to include the same type of stage multiple times within a single pipeline.