## 1. Difference between HTTP1.1 vs HTTP2 ?

| | **HTTP1.1** | **HTTP2** |
|---|---|---|
| ➢ **Multiplexing** | HTTP1.1 loads resources one after the other,so if one resource cannot be loaded, it blocks all the other resources behind it. | HTTP2 is able to use a single TCP connection to send multiple streams of data at once so that no one resource blocks any other resource. |
| ➢ **Header compression** | Headers are sent on every request leading to a lot of duplicate data being sent uncompressed across the wire. | HTTP2 uses a more advanced compression method called HPACK that eliminates redundant information in HTTP header packets. |
| ➢ **Prioritization** | In HTTP1.1 prioritization is fixed since the page resources load one after the other | HTTP/2 offers a feature called weighted prioritization.<br>This allows developers to decide which page resources will load first, every time. |
| ➢ **Connectivity** | It supports connection reuse i.e. for every TCP connection there could be multiple requests and responses, and pipelining where the client can request several resources from the server at once. | Uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time. |
| ➢ **Authentication** | It is relatively secure since it uses digest authentication | It is better equipped to deal with new TLS features like connection error of type - Inadequate Security. |
| ➢ **Protocol Type** | Text based protocol that is in the readable form. | It is a binary protocol (HTTP requests are sent in the form of 0s and 1s).Needs to be converted back from binary in order to read it. |

**2.** Write a blog about objects and its internal representation in Javascript?

An object is **a standalone entity, with properties and type**. example: A cup is an object, with properties. A cup has a color, a design, weight, a material it is made of, etc. The same way, JavaScript objects can have properties, which define their characteristics.

There are two types of object properties: The ***data*** **property** and the ***accessor*** **property**. Each property has corresponding *attributes*.

- Data properties associate a key with a value.
- Associates a key with one of two accessor functions (`get` and `set`) to retrieve or store a value.

### Create JavaScript Object with Object Literal

One of easiest way to create a javascript object is object literal, simply define the property and values inside curly braces as shown below

Example:

```
let bike = {name: 'SuperSport', maker:'Ducati', engine:'937cc'};
```

### Create JavaScript Object with Constructor

Constructor is nothing but a function and with help of new keyword, constructor function allows to create multiple objects of same flavor as shown below

```
function Vehicle(name, maker) {
   this.name = name;
   this.maker = maker;
}
let car1 = new Vehicle(’Fiesta’, 'Ford’);
let car2 = new Vehicle(’Santa Fe’, 'Hyundai’)
console.log(car1.name);    //Output: Fiesta
console.log(car2.name);    //Output: Santa Fe
```

### Using the JavaScript Keyword new

The following example also creates a new JavaScript object with four properties:

Example:  var person = new Object();

person.firstName = “Jay”;

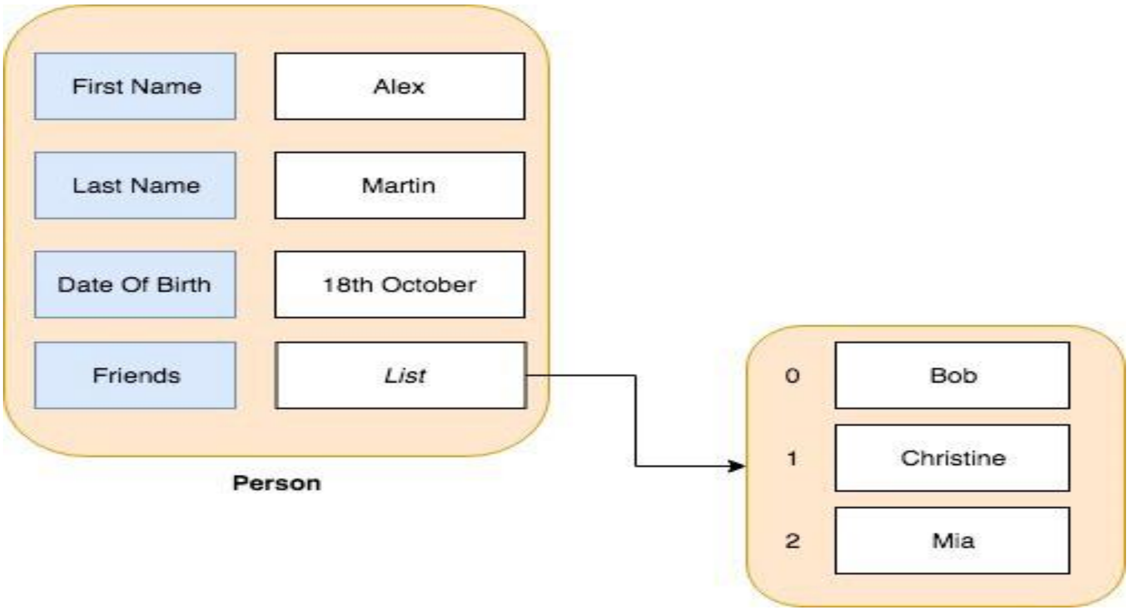
person.lastName = “Doe”;

### Using the `Object.create` method

Objects can also be created using the `Object.create()` method. This method can be very useful, because it allows you to choose the prototype object for the object you want to create, without having to define a constructor function.

```
var Animal = {
  type: ‘wild’, // Default value of properties
  displayType: function() {  // Method which will display Animal
    console.log(this.type);
  }
};


// Create new animal type called Fishes
var fish = Object.create(Animal);
fish.type = 'Fishes';
fish.displayType(); // Output:Fishes
```



**Pictorial representation of Object and internal details**