**1. What is HTTP? Why is HTTP/2 faster than HTTP/1.1?**

[HTTP](https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/) stands for hypertext transfer protocol, and it is the basis for almost all web applications. More specifically, HTTP is the method computers and servers use to request and send information. For instance, when someone navigates to cloudflare.com on their laptop, their web browser sends an HTTP request to the Cloudflare servers for the content that appears on the page. Then, Cloudflare servers send HTTP responses with the text, images, and formatting that the browser displays to the user.

1. **What is HTTP/1.1?**

The first usable version of HTTP was created in 1997. Because it went through several stages of development, this first version of HTTP was called HTTP/1.1. This version is still in use on the web.

1. **What is HTTP/2?**

In 2015, a new version of HTTP called HTTP/2 was created. HTTP/2 solves several problems that the creators of HTTP/1.1 did not anticipate. In particular, HTTP/2 is much faster and more efficient than HTTP/1.1. One of the ways in which HTTP/2 is faster is in how it prioritizes content during the loading process.

1. **What is HTTP/3?**

HTTP is an essential backbone of the Internet — it dictates how communications platforms and devices exchange information and fetch resources. In short, it is what allows users to load websites.

HTTP/3 is a new standard in development that will affect how web browsers and servers communicate, with significant upgrades for user experience, including performance, reliability, and security.

After the first hypertext transfer protocol ([HTTP](https://www.cloudflare.com/learning/ddos/glossary/hypertext-transfer-protocol-http/)) was released in 1991, subsequent iterations made websites [faster](https://www.cloudflare.com/learning/performance/why-site-speed-matters/) without any changes to the underlying code.

| **Feature** | **HTTP/1.1** | **HTTP/2** | **HTTP/3** |
| --- | --- | --- | --- |
| Multiplexing | No | Yes | Yes |
| Header Compression | No | Yes | Yes |
| Binary Protocol | No | Yes | Yes |
| Server Push | No | Yes | Yes |
| Prioritization | No | Yes | Yes |
| Parallelism | Limited by browser (typically 6-8 connections) | Multiplexed streams with one connection | Multiplexed streams with one connection |
| Latency | Higher due to head-of-line blocking | Reduced due to multiplexing and prioritization | Further reduction with QUIC protocol |
| Compatibility | Widely adopted | Increasing adoption, especially for newer applications | Emerging, adoption expected to increase with time |
| SSL/TLS | Overhead due to multiple connections | Single connection for parallelism | Single connection for parallelism |
| Performance Improvement | Moderate | Significant | Further improvement with QUIC protocol |

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

Objects, in JavaScript, is it’s most important data-type and forms the building blocks for modern JavaScript. These objects are quite different from JavaScript’s primitive data-types(Number, String, Boolean, null, undefined and symbol) in the sense that while these primitive data-types all store a single value each (depending on their types).

Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types.  
An object, is a reference data type. Variables that are assigned a reference value are given a reference or a pointer to that value. That reference or pointer points to the location in memory where the object is stored. The variables don’t actually store the value.

Loosely speaking, objects in JavaScript may be defined as an unordered collection of related data, of primitive or reference types, in the form of “key: value” pairs. These keys can be variables or functions and are called properties and methods, respectively, in the context of an object.

For Eg. If your object is a student, it will have properties like name, age, address, id, etc and methods like updateAddress, updateNam, etc.