Bhai 😎 ab samajh lete hain **Streams aur Buffers** ka funda ekdum desi style me:

**1. Concept – Streams aur Buffers kya hai?**

👉 **Socho ek nadi (Stream)**

* Nadi me paani ekdum se nahi aata, **thoda-thoda hissa (chunks)** aata hai.
* Waise hi Node.js me **Stream** data ko ek saath nahi bhejta, **piece-by-piece (chunks)** bhejta hai.

👉 **Buffer kya hai?**

* Buffer ek **balti (temporary memory)** hai jo data ko thoda-thoda store karke rakh leti hai jab tak agla process ready na ho.
* Matlab agar stream me paani aa raha hai aur tum bottle me bhar rahe ho, toh bottle bharne ke liye buffer ek jagah data hold karta hai.

⚡ Iska use hota hai jab data **bahut bada ho** (jaise movies, mp3, large files). Agar pura file ek saath read/write karoge toh RAM blast ho jayegi 🚀.

**2. Syntax & Code – Streams and Buffers ka use kaise kare?**

**Example: File Read with Stream**

const fs = require('fs');

// Read stream create karo

const readStream = fs.createReadStream('file.txt', 'utf8');

// Har chunk aane par 'data' event trigger hoga

readStream.on('data', (chunk) => {

console.log("Naya chunk aaya:");

console.log(chunk);

});

👉 Yaha file.txt ka content thoda-thoda hissa karke milega (chunk by chunk).

**Example: File Copy with Stream (Pipe ka use)**

const fs = require('fs');

// Read and Write Stream

const readStream = fs.createReadStream('file.txt', 'utf8');

const writeStream = fs.createWriteStream('newFile.txt');

// Pipe lagado (direct nadi se nadi)

readStream.pipe(writeStream);

👉 Ye ekdum smart way hai bada file copy karne ka bina RAM overload kiye.

**Buffer Example**

const buffer = Buffer.from('Hello Bhai');

console.log(buffer); // Hexadecimal data

console.log(buffer.toString()); // Normal string

console.log(buffer[0]); // First byte ka ASCII value

**3. Exercise – Practice Time**

**📝 Exercise 1: Large File Reader**

Ek script banao jo ek bigFile.txt ko **stream** se read kare aur har chunk ke size ko print kare.

👉 Hint:

const fs = require('fs');

const readStream = fs.createReadStream('bigFile.txt', 'utf8');

readStream.on('data', (chunk) => {

console.log(`Chunk size: ${chunk.length}`);

});

**📝 Exercise 2: Chat Logger with Stream**

Ek script banao jisme user ke chat messages ek file me **stream ke through write** hote rahe.

👉 Hint:

const fs = require('fs');

const writeStream = fs.createWriteStream('chat.txt');

writeStream.write("User1: Hello Bhai!\n");

writeStream.write("User2: Kya haal hai?\n");

writeStream.write("User1: Sab mast! 😎\n");

console.log("Chat save ho gayi stream ke through.");

**🎯 Desi Summary**

1. **Concept**
   * **Stream** = Data thoda-thoda karke aata hai (jaise nadi ka paani).
   * **Buffer** = Temporary storage (balti) jo chunks hold karta hai.
2. **Syntax & Code**
   * fs.createReadStream()
   * fs.createWriteStream()
   * .pipe() for direct transfer
   * Buffer.from(), .toString()
3. **Exercises**
   * Large File Reader (Stream + chunk size)
   * Chat Logger (Stream + write)

👉 Bhai, chahte ho main tumhe ek **real-world mini project** bana du jisme tum ek **video streaming system** banao Node.js me using streams & buffers?