**Explanation and Debugging Steps:**

1. **debugger;:**
   * Insert debugger; within the analyzeData() function, as shown in the commented code.
   * Run your script using node <your\_script.js>.
   * The debugger will pause at the breakpoint.
   * You can then use the following commands in your terminal:
     + **n:** Step to the next line.
     + **s:** Step into a function call.
     + **c:** Continue execution until the next breakpoint.
     + **o:** Step out of the current function.
     + **repl:** Enter the REPL mode to evaluate expressions.
     + **watch:** Add an expression to monitor its value during debugging (e.g., watch total).
2. **node inspect:**
   * Run your script using node inspect <your\_script.js>.
   * This will open a debugger console in your terminal.
   * You can use the same commands listed above (n, s, c, o, repl, etc.) in the debugger console.
   * **Additional Features:** You can also set breakpoints by typing setBreakpoint(<line\_number>) in the debugger console.
   * **Example:**
     + setBreakpoint(26): Sets a breakpoint at line 26 of the script.
     + c: Continue execution until the breakpoint is hit.
3. **Chrome DevTools:**
   * Run your script using node inspect <your\_script.js>.
   * Open Chrome and navigate to chrome://inspect/#devices.
   * Click the link to your Node.js application in the 'Remote Targets' section.
   * This will launch the Chrome DevTools with the Node.js debugger.
   * Use the familiar DevTools interface (Source tab, Breakpoints, Watch expressions, etc.) to debug your code.

**Important Notes:**

* The data.json file should be in the same directory as your script for this example to work correctly.
* The setTimeout(main, 2000) line delays the execution of main() for 2 seconds, giving you time to attach the debugger.
* This example showcases complex relationships within the code, demonstrating the usefulness of the debugger for navigating through nested objects, analyzing data structures, and understanding the execution flow.
* The debugger is a versatile tool that can be used in many ways, and the techniques shown here are just a few examples. Experiment with different approaches and features to find what works best for you.