1. ***Lista de situaciones donde se presenta la estructura de árbol***

* ***Organización jerárquica de una empresa:*** Cada empleado tiene un superior directo, excepto el CEO, creando una estructura de árbol.
* ***Sistema de archivos en computadoras:*** Los archivos y carpetas están organizados jerárquicamente, donde cada carpeta puede contener archivos u otras carpetas.
* ***Árbol genealógico:*** Muestra las relaciones familiares en una estructura de descendencia y ancestros.
* ***Análisis sintáctico en procesadores de lenguaje:*** Se utilizan árboles para representar la estructura gramatical de las oraciones.
* ***Sistemas de decisión:*** Árboles de decisión utilizados en inteligencia artificial para hacer predicciones o decisiones basadas en varias condiciones.
* ***Rutas de navegación:*** En las redes, los árboles de expansión mínima ayudan a encontrar las rutas más eficientes.
* ***Renderizado de escenas en gráficos por computadora:*** Se utilizan árboles como BSP (Binary Space Partitioning) para determinar qué objetos mostrar en 3D rendering.
* ***Juegos de ajedrez y otros juegos de estrategia:*** Árboles de juegos se utilizan para prever movimientos y sus posibles resultados.
* ***Manejo de expresiones matemáticas:*** Las expresiones se pueden representar y evaluar utilizando árboles de expresión.
* **Clasificación y búsqueda:** Estructuras como árboles de búsqueda binaria ayudan en la rápida búsqueda y clasificación de datos.

1. ***Cuadro comparativo de árboles B, B+ y AVL***

|  |  |  |  |
| --- | --- | --- | --- |
| ***Característica*** | ***Árbol B*** | ***Árbol B+*** | ***Árbol AVL*** |
| **Definición** | Estructura de datos de árbol balanceado con múltiples hijos por nodo. | Variante del árbol B, con todos los datos en hojas y nodos internos como índices. | Árbol binario balanceado en altura. |
| **Balanceo** | Balanceado al dividir nodos cuando se supera un número máximo de claves. | Similar al árbol B, pero reorganiza claves para que solo estén en hojas. | Se reequilibra después de cada inserción o eliminación para mantener factores de equilibrio. |
| **Altura** | Logarítmica respecto al número de elementos. | Logarítmica respecto al número de elementos, generalmente más alto que B. | Estrictamente balanceado para mantener la altura mínima. |
| **Uso** | Ampliamente utilizado en bases de datos y sistemas de archivos para almacenamiento. | Preferido en bases de datos para operaciones de rango y secuenciales por su estructura de hojas. | Utilizado en aplicaciones donde se necesitan inserciones y eliminaciones rápidas y balanceadas. |
| **Ventajas** | Minimiza el acceso al disco, buen rendimiento general | Mejora en operaciones de escaneo secuencial y acceso a rangos. | Alto rendimiento en mantener el árbol balanceado, óptimo para búsqueda. |
| **Desventajas** | Menos eficiente en operaciones de escaneo secuencial que B+. | Mayor espacio y overhead en mantener punteros. | Mayor complejidad de mantenimiento del balanceo. |
| **Operaciones clave** | Inserción, eliminación y búsqueda eficientes, aunque la reorganización puede ser costosa. | Inserción y eliminación requieren redistribución y fusión de nodos. | Inserciones y eliminaciones siempre implican reequilibrio, lo que puede ser más costoso que en B/B+. |

1. ***Trabajo autónomo:***

* **Not using version control**: One of the fundamental errors mentioned is neglecting to use version control systems such as Git. Version control is essential for managing changes to the codebase, allowing multiple developers to work on the same project simultaneously, and keeping a history of all changes in case a rollback is needed.
* **Modeling NoSQL databases as SQL**: Another mistake is treating NoSQL databases like SQL databases. NoSQL databases are designed to handle different data modeling patterns, scalability needs, and consistency requirements. Developers should embrace these differences to fully leverage NoSQL technologies rather than imposing relational structures on them.
* **Insufficient planning**: The video also points out that failing to plan sufficiently for a project can lead to challenges. Adequate planning involves understanding project requirements, setting clear milestones, and preparing for potential setbacks. Without careful planning, projects are more likely to go over budget, exceed timelines, or result in products that don't meet users' needs
* **Lack of Quality Assurance (QA)**: Skipping quality assurance and testing is a critical error. QA is a key component in the development process, helping to catch bugs and issues before the software goes live. Neglecting this phase can compromise the product's reliability and user experience.
* **Modifying code in production**: Directly modifying code in the production environment is highly risky. Such changes can lead to untested code causing immediate and potentially widespread issues for end users. It’s best practice to make changes in a development or staging environment, undergo thorough testing, and then carry out a controlled deployment to production.