|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **BST** | **AVL** | **B - Trees** | **B+ Trees** | **B\* Trees** |
| **Who** | P.F. Windly, A.D. Booth, A.J.T Colin, T.N. Hibbard  Attributed to Conway Berners-Lee, David Wheeler | Georgy Adelson-Velsky, Evgenii Landis | Rudolf Bayer, Edward M. McCreight | Douglas Comer (no actual person) | Hans Berliner |
| **When** | 1960 | 1962 | 1972 | 1973 | 1979 |
| **What** | also called an ordered or sorted binary tree, is a rooted binary tree data structure with the key of each internal node being greater than all the keys in the respective node's left subtree and less than the ones in its right subtree.  Used to store data hierarchically | is a self-balancing binary search tree (BST). It was the first such data structure to be invented. In an AVL tree, the heights of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property.  Used in databases, sets | B-trees are a way to get better locality and limit the number of seek operations. To access and maintain efficiently an index that is too large to hold in memory. Used in Database Management  Used to index data in large databases. Reduce disk accesses | A B+ tree is an advanced form of a self-balancing tree in which all the values are present in the leaf level. An important concept to be understood before learning B+ tree is multilevel indexing which allows faster data access.  Used to store large amounts of data in secondary memory |  |
| **Properties** | * Non-balancing * Sorted Keys * Right > Root > Left | * Self-balancing * Sorted Keys * Right > Root > Left | * Self-balancing * Sorted Keys * Min = M/2 Children   Max = M Children   * Leaves not connected | * Self-balancing * Sorted Keys  |  |  | | --- | --- | | Max children = order | Max keys = m - 1 | | Min children = | Min keys = - 1 |  * Connected Leaves * Same level leaves * Data repetition | * Self-balancing * Sorted Keys  |  |  | | --- | --- | | Max children = order | Max keys = m - 1 | | Min children = | Min keys = |  * Same level leaves |

**TIME AND SPACE COMPLEXITY**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Time Complexity** | | | | | | | | | |
|  | **BST** | | | **AVL** | | | **B - Trees** | **B+ Trees** | **B\* Trees** |
|  | **Best** | **Average** | **Worst** | **Best** | **Average** | **Worst** | **All** | **All** | None Mentioned |
| **Access** | O (1) | O (log n) | O(n) | O (log n) | O (log n) | O (log n) | O (logm n) | O (log n) |
| **Search** | O (1) | O (1) |
| **Insertion** | O (1) | O (log n) |
| **Deletion** | O (log n) | O (log n) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Space Complexity** | | | | | |
|  | **BST** | **AVL** | **B – Trees** | **B+ Trees** | **B\* Trees** | |
| **Worst** | O(n) | O(n) | O (n) | O (n) | None mentioned | |