## 树的逻辑结构

树中的数据元素成为**节点**

### 1树的定义

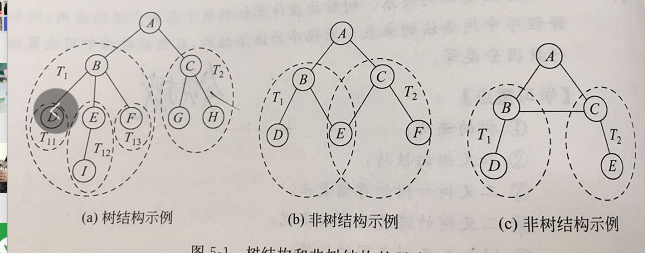
树是n个节点组成的集合。(n>=0)

n=0是空树

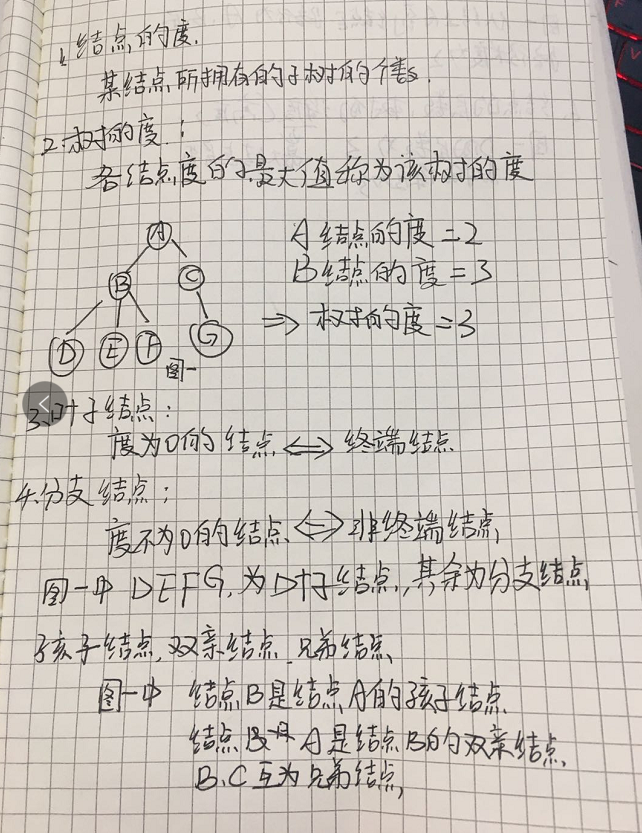
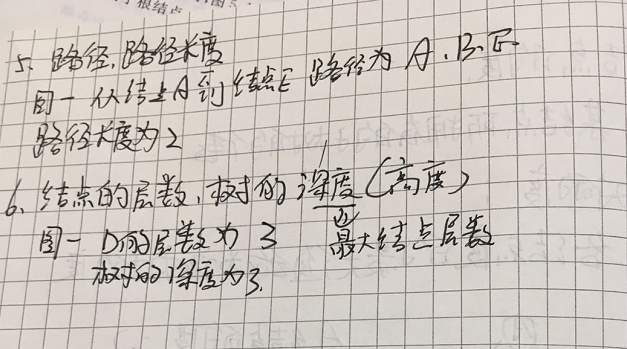
**非空树满足的条件：**

A:有且仅有一个特定的成为根节点

B:当n>1时，除了根节点之外的其余结点能被分为M（m>0）个互不相交的有限集合，其中每一个集合又是一棵树，并成为根节点的子树

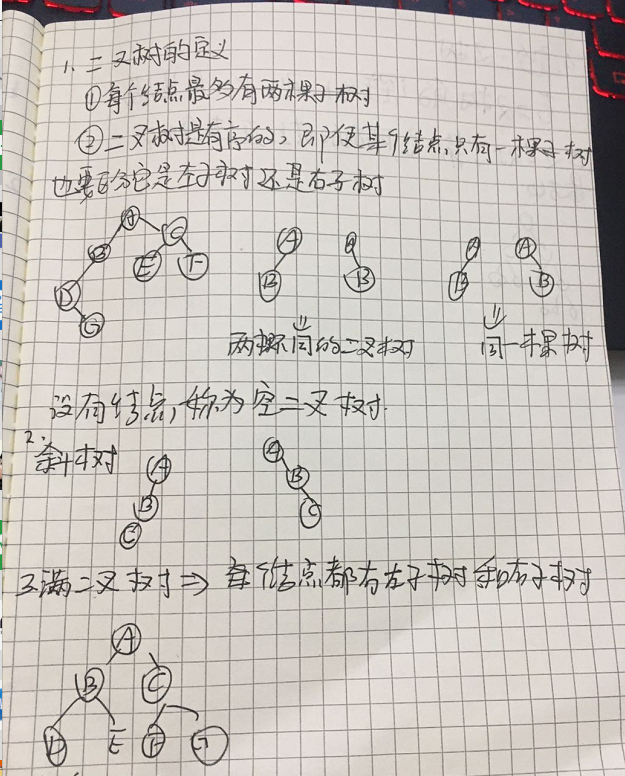


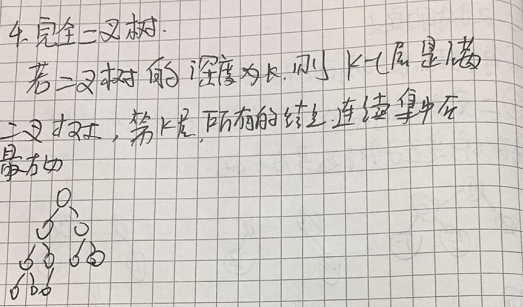
### 1.2树的基本术语

## 二叉树

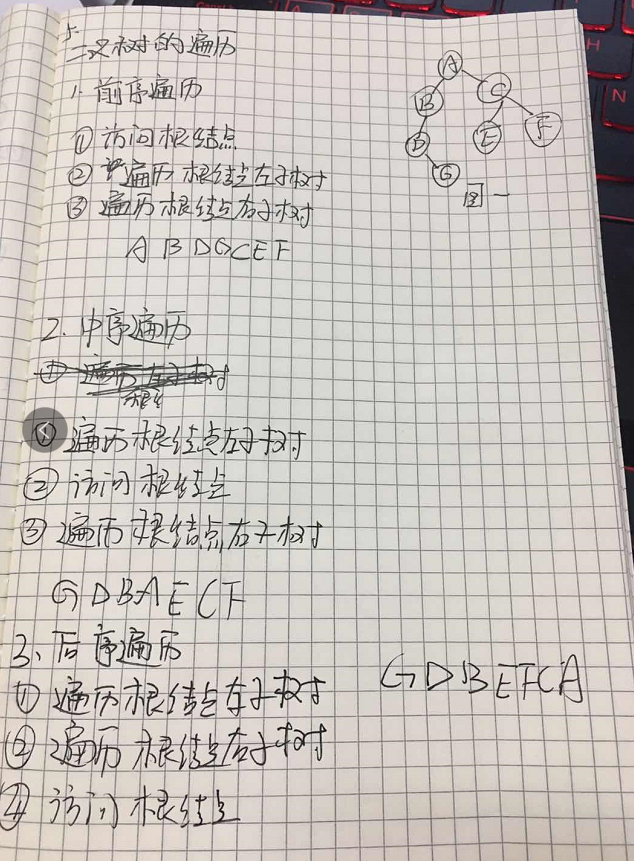
### 1.基础概念







### 二叉树的遍历



这里中序遍历写出了 应该为DGBAECF

#### 层序遍历

从第一层开始，从上到下逐层遍历

ABCDEFG

#### 深度优先遍历

前序遍历

中序遍历

后序遍历

#### 广度优先遍历

也叫层序遍历

## 二叉查找树（二叉排序树）

### 定义

若它的左子树不空，则左子树上的所有节点的值均小于根节点的值

若它的右子树不空，则柚子树上的所有节点的值均大于根节点的值

它的左右子树都是二叉排序树

### 二叉排序树代码实现

### 5.4B+树

<https://zhuanlan.zhihu.com/p/54102723>

在理解B+树之前先要理解B-树

数据库为什么要用树形结构存储，因为树的查询效率高而且可以保持有序，

那为什么没有用二叉排序树来实现呢，二叉查找树查询的复杂度是O（logN）,查找速度与比较次数是最小的，但是数据库的索引是存储在磁盘上的，当数据量比较大的时候，索引可能有几个G甚至更多，不可能吧整个索引全部加载到内存中，只能逐一加载磁盘页，每个磁盘页对应着索引树的节点，



从图中可以看出磁盘IO次数和树的高度有关，因此为了减少磁盘IO次数，我们需要把原来高瘦的树结构变成矮胖的

那么什么是B树呢

#### B树 B-tree

https://zhuanlan.zhihu.com/p/54084335

很多人读成B- b减树 是错的

##### 定义

一个m阶的B树具有如下几个特征：m是树的高度

1.根结点至少有两个子女。

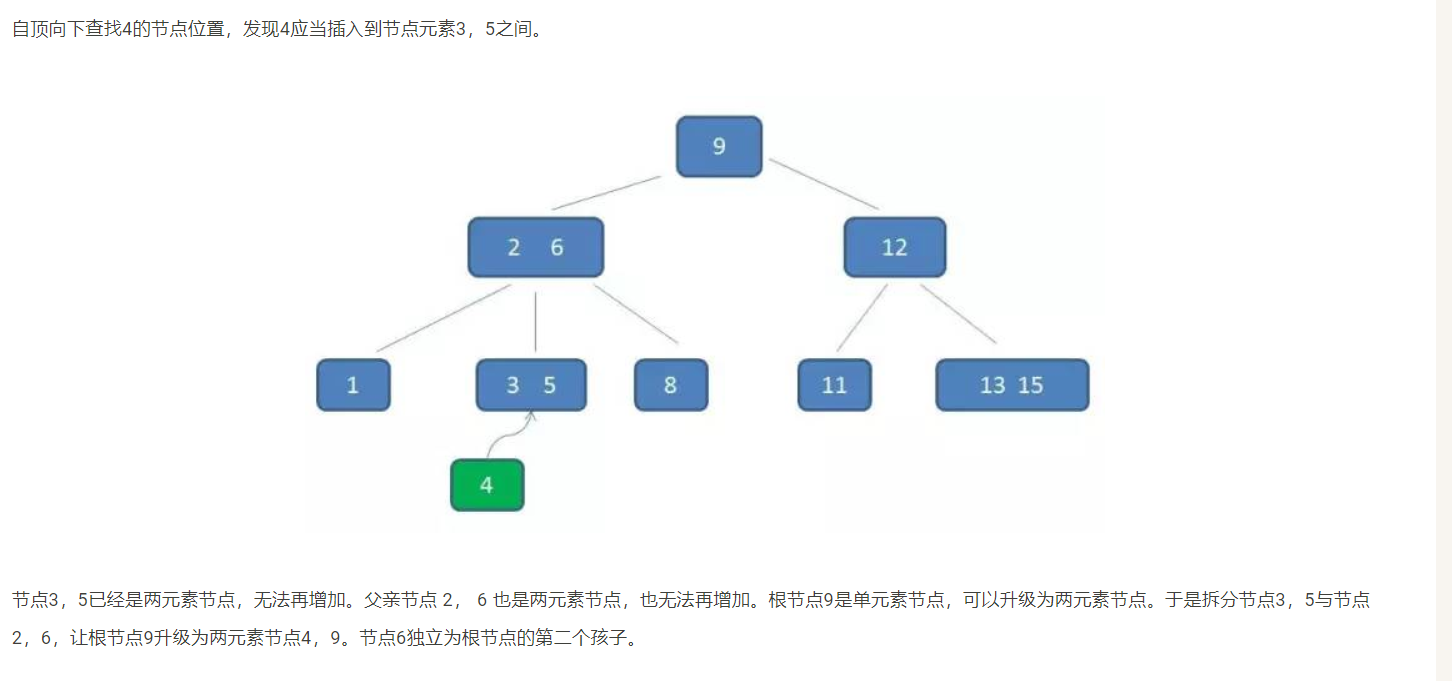
2.每个中间节点都包含k-1个元素和k个孩子，其中 m/2 <= k <= m

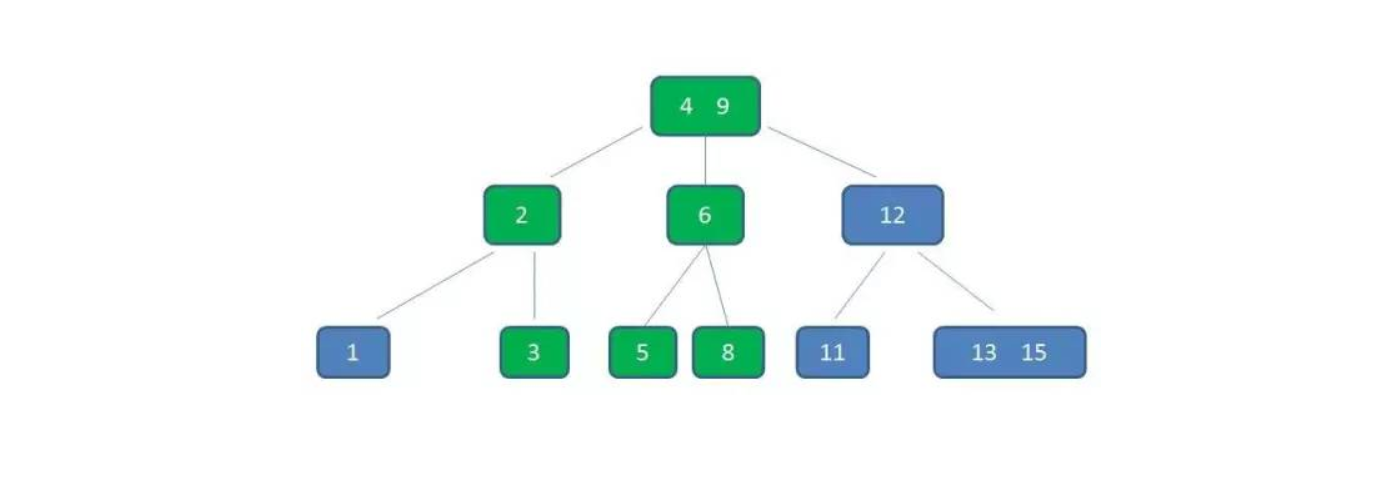
3.每一个叶子节点都包含k-1个元素，其中 m/2 <= k <= m

4.所有的叶子结点都位于同一层。

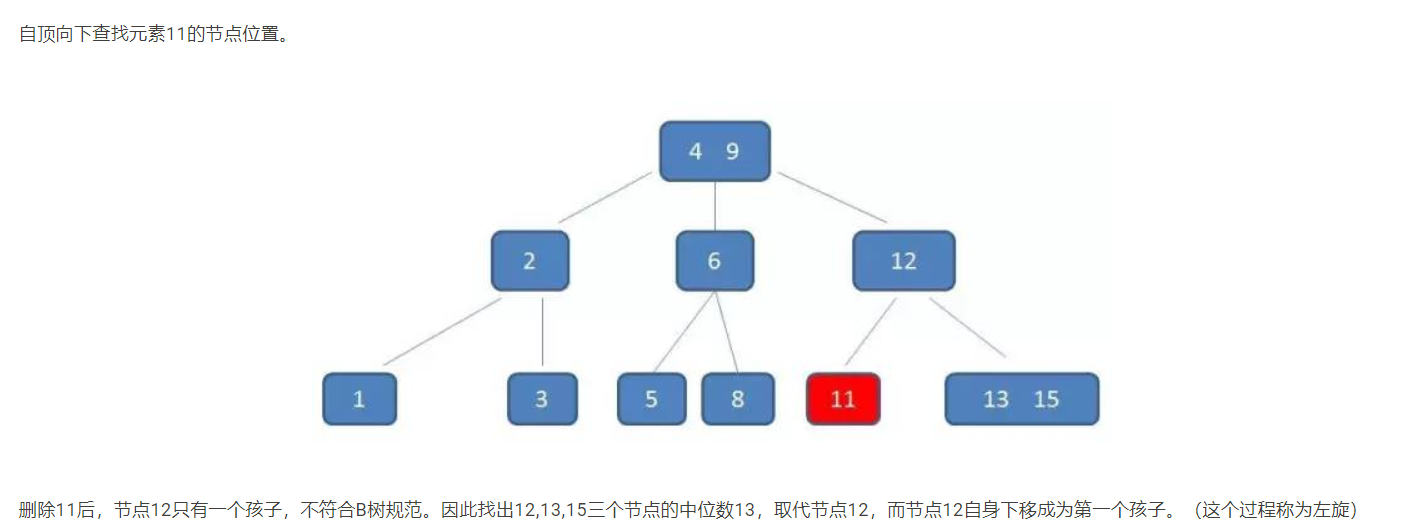
5.每个节点中的元素从小到大排列，节点当中k-1个元素正好是k个孩子包含的元素的值域分划。

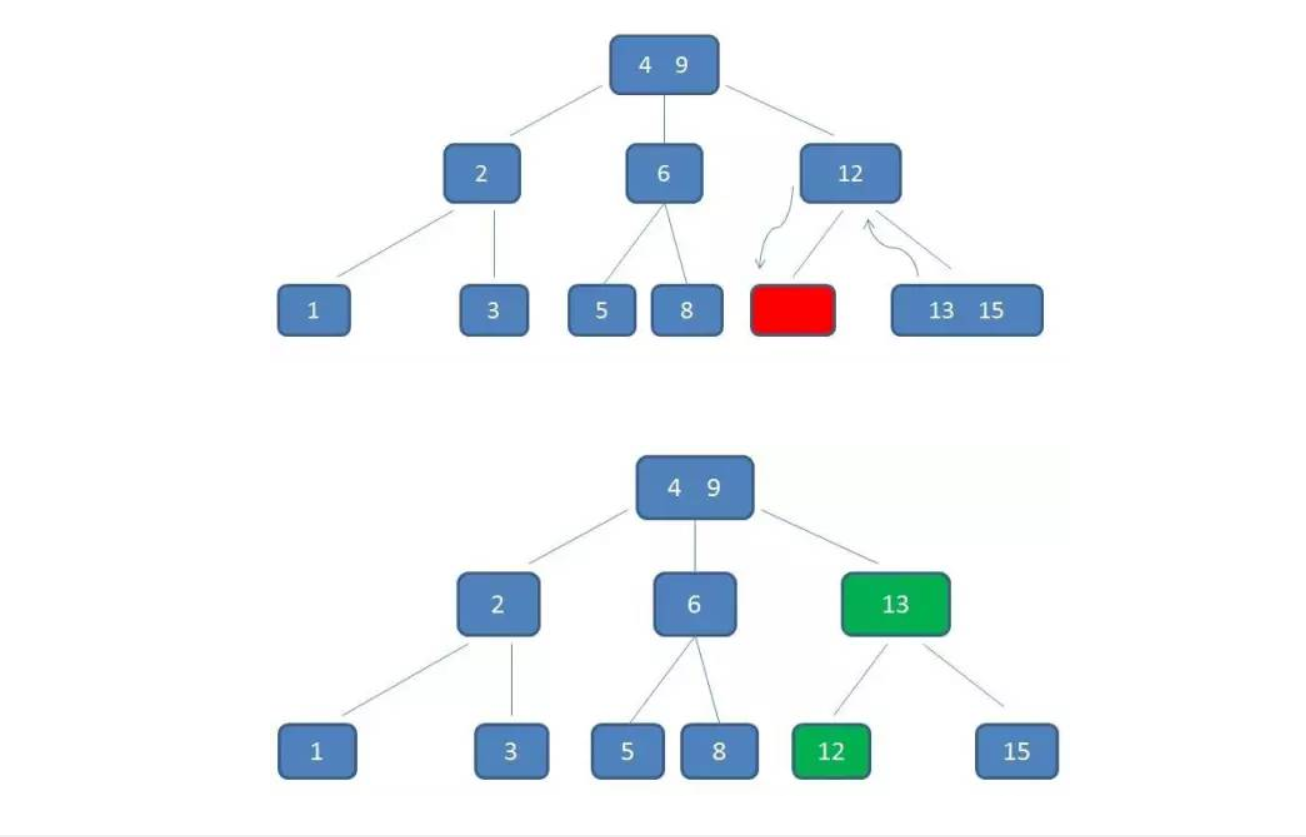
##### 插入新节点





##### 删除节点





##### 应用

B树主要用于文件系统以及部分数据库索引，如关系型数据库MongoDB

##### 代码实现

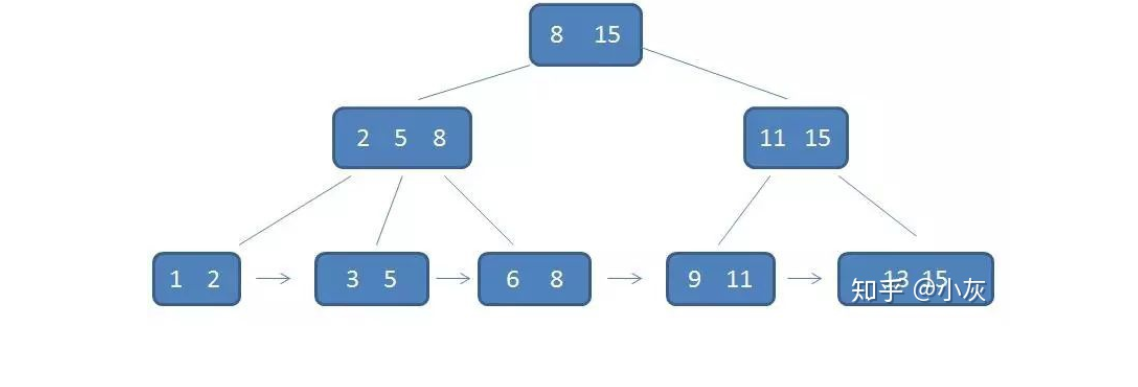
#### B+树

##### 定义

1.有k个子树的中间节点包含有k个元素（B树中是k-1个元素），每个元素不保存数据，只用来索引，所有数据都保存在叶子节点。

2.所有的叶子结点中包含了全部元素的信息，及指向含这些元素记录的指针，且叶子结点本身依关键字的大小自小而大顺序链接。

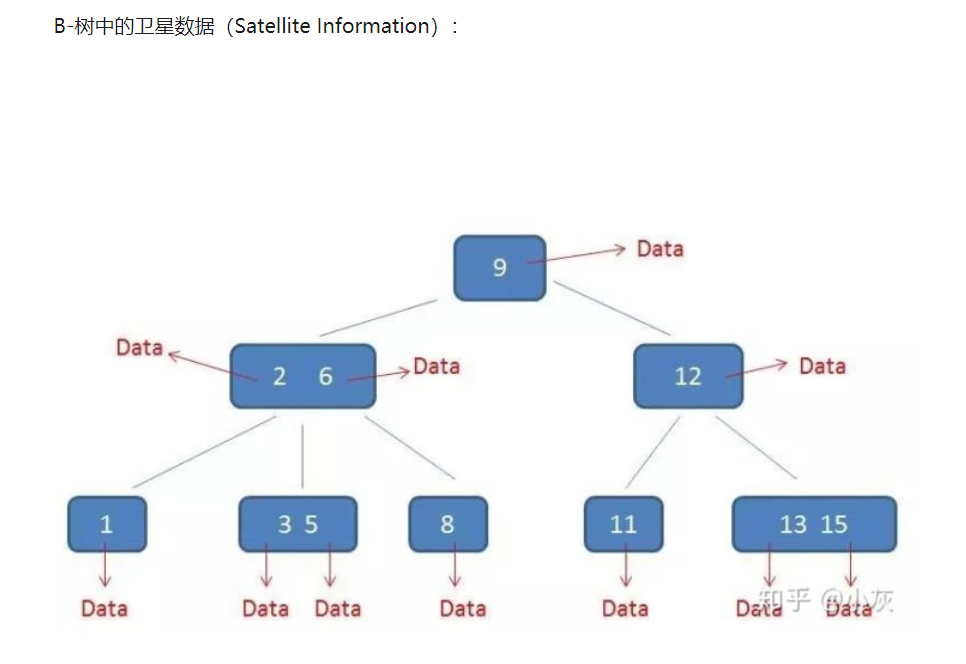
3.所有的中间节点元素都同时存在于子节点，在子节点元素中是最大（或最小）元素。



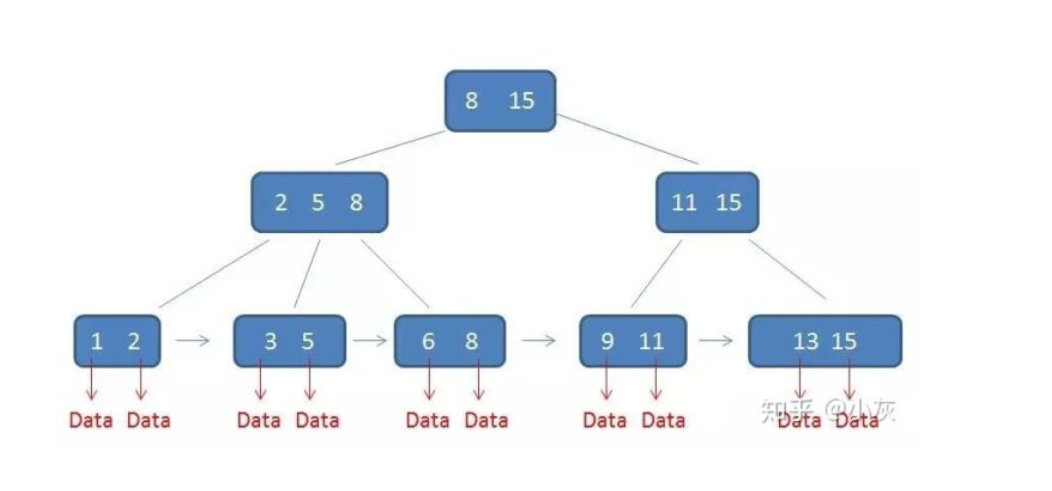
##### 卫星数据

卫星数据是指索引元素所指向的数据记录

在B树中无论是中间节点还是叶子节点都带有卫星数据



在B+树中只用叶子结点带有卫星数据，其余中间节点仅仅是索引，没有任何关联数据



在聚集索引中，叶子节点直接包含卫星数据，在非聚集索引中，叶子节点带有指向卫星数据的指针

因为B+树的中间节点没有卫星数据，所以同样大小的磁盘可以容纳更多的节点元素，这也就意味着在数据量相同的情况下，B+树的结构比B树更加矮胖磁盘的IO次数更少。

B+树比B树的优势：

磁盘IO次数少，查询性能稳定，范围查询简便

#### 二叉查找树（二叉排序树）

##### 定义

若它的左子树不空，则左子树上的所有节点的值均小于根节点的值

若它的右子树不空，则柚子树上的所有节点的值均大于根节点的值

它的左右子树都是二叉排序树

##### 代码实现

###### 二差排序树的插入

小程序后端使用laravel搭建，实现了在线下单购买公章、跟踪订单状态等功能。主要技术是小程序的微信支付，消息订阅和laravel广播实现的实时 聊天、延迟队列完成的自动收货。