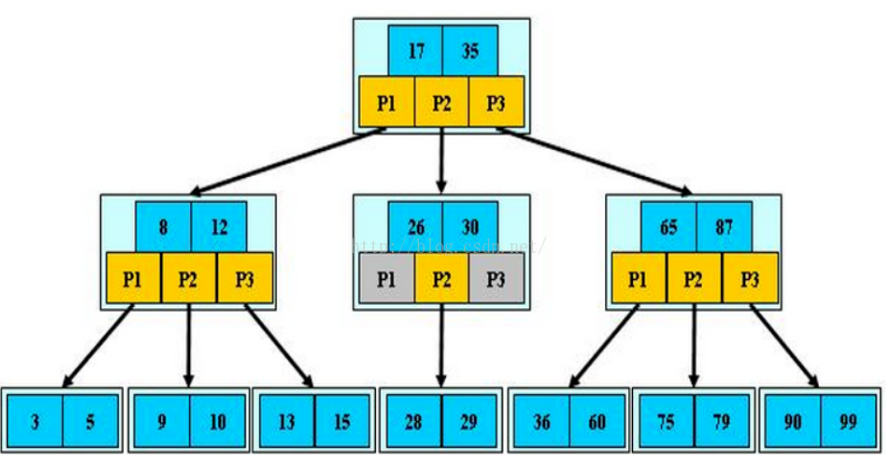
# B树和B+树原理

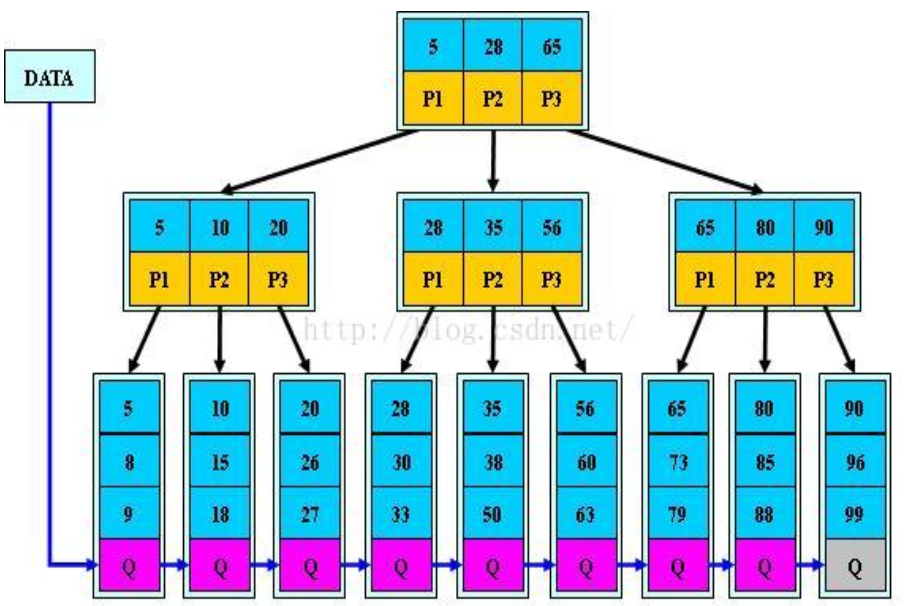
## B树



 1.每个节点中既要存索引信息，又要存其对应的数据，如果数据很大，那么当树的体量很大时，每次读到内存中的树的信息就会不太够。

 2.B树遍历整个树的过程和二叉树本质上是一样的，B树相对二叉树虽然提高了磁盘IO性能，但并没有解决遍历元素效率低下的问题

## B+树



针对以上两个问题，B+树诞生了，B+树相比B树，本质上是一样的，区别就在与B+树的所有根节点都不带有任何数据信息，只有索引信息，所有数据信息全部存储在叶子节点里，这样，整个树的每个节点所占的内存空间就变小了，读到内存中的索引信息就会更多一些，相当于减少了磁盘IO次数，问题1就得到了解决。又由B树的性质可以得到，所有叶子节点都会在同一层，B+树会以一个链表的形式将所有叶子节点的信息全部串联起来，这样，想遍历所有数据信息只需要顺序遍历叶子节点就可以了，方便又高效，问题二就得到了解决。不仅如此，B+树还有一个相应的优质特性，就是B+树的查询效率是非常稳定的，因为所有信息都存储在了叶子节点里面，从根节点到所有叶子节点的路径是相同的。那么，我们总结一下B+树的优点：

（1) B+树的磁盘读写代价更低

  B+的内部结点并没有指向关键字具体信息的指针。因此其内部结点相对B树更小。如果把所有同一内部结点的关键字存放在同一盘块中，那么盘块所能容纳的关键字数量也越多。一次性读入内存中的需要查找的关键字也就越多。相对来说IO读写次数也就降低了。

 （2）B+树的数据信息遍历更加方便； B+树只要遍历叶子节点就可以实现整棵树的遍历，而B树不支持这样的操作（或者说效率太低），而且在数据库中基于范围的查询是非常频繁的，所以数据库索引基本采用B+树

（3) B+树的查询效率更加稳定；由于非终结点并不是最终指向文件内容的结点，而只是叶子结点中关键字的索引。所以任何关键字的查找必须走一条从根结点到叶子结点的路。所有关键字查询的路径长度相同，导致每一个数据的查询效率相当。