[MySQL FAQ]系列 — 为什么InnoDB表要建议用自增列做主键

我们先了解下InnoDB引擎表的一些关键特征：

- InnoDB引擎表是基于B+树的索引组织表(IOT)；
- 每个表都需要有一个聚集索引(clustered index)；
- 所有的行记录都存储在B+树的叶子节点(leaf pages of the tree)；
- 基于聚集索引的增、删、改、查的效率相对是最高的；
- 如果我们定义了主键(PRIMARY KEY)，那么InnoDB会选择其作为聚集索引；
- 如果没有显式定义主键，则InnoDB会选择第一个不包含有NULL值的唯一索引作为主键索引；
- 如果也没有这样的唯一索引，则InnoDB会选择内置6字节长的ROWID作为隐含的聚集索引(ROWID随着行记录的写入而主键递增，这个ROWID不像ORACLE的ROWID那样可引用，是隐含的)。

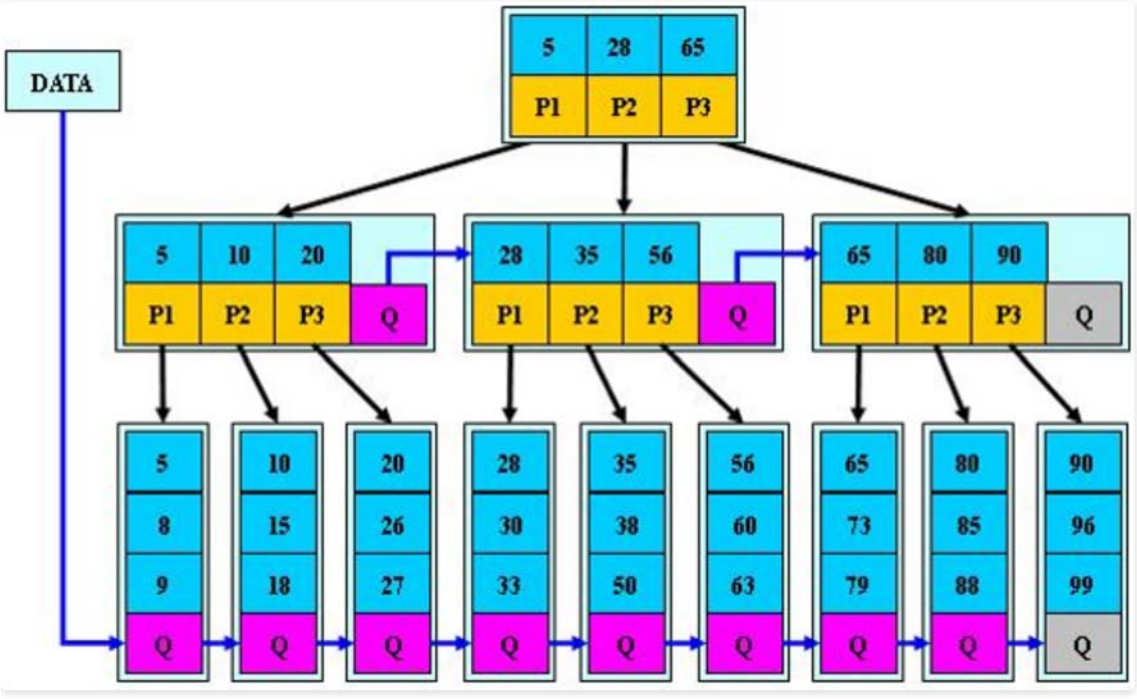
综上总结，如果InnoDB表的数据写入顺序能和B+树索引的叶子节点顺序一致的话，这时候存取效率是最高的，也就是下面这几种情况的存取效率最高：

- 使用自增列(INT/BIGINT类型)做主键，这时候写入顺序是自增的，和B+数叶子节点分裂顺序一致；
- 该表不指定自增列做主键，同时也没有可以被选为主键的唯一索引(上面的条件)，这时候InnoDB会选择内置的ROWID作为主键，写入顺序和ROWID增长顺序一致；
- 除此以外，如果一个InnoDB表又没有显示主键，又有可以被选择为主键的唯一索引，但该唯一索引可能不是递增关系时(例如字符串、UUID、多字段联合唯一索引的情况)，该表的存取效率就会比较差。

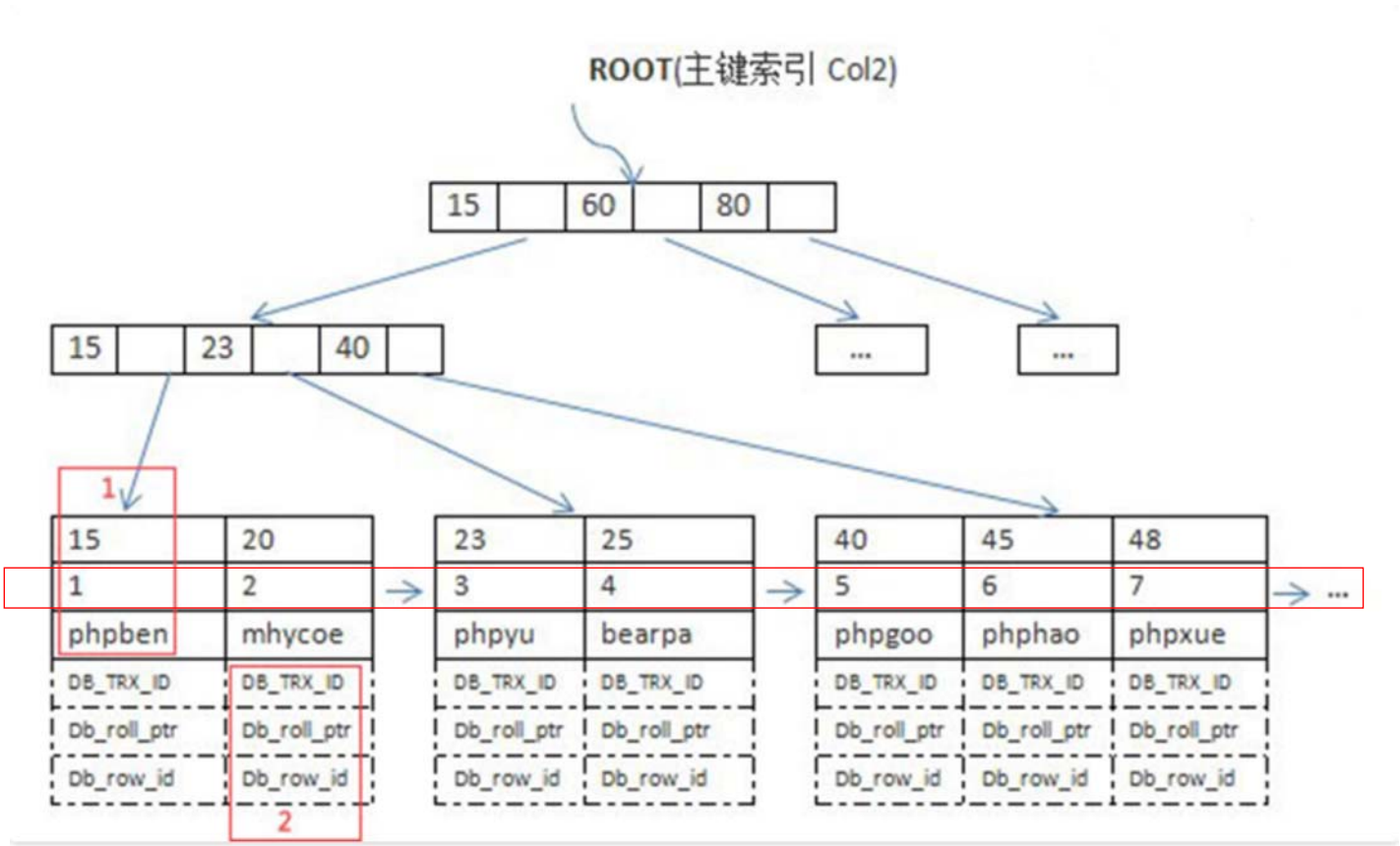
实际情况是如何呢？经过简单TPCC基准测试，修改为使用自增列作为主键与原始表结构分别进行TPCC测试，前者的TpmC结果比后者高9%倍，足见使用自增列做InnoDB表主键的明显好处，其他更多不同场景下使用自增列的性能提升可以自行对比测试下。

附图：

1、B+树典型结构



2、InnoDB主键逻辑结构



延伸阅读：

1、TPCC-MySQL使用手册

2、B+Tree index structures in InnoDB

3、B+Tree Indexes and InnoDB – Percona

4、MySQL官方手册: Clustered and Secondary Indexes

陈佳
2014/11/249:55 下午

我一般也是自增值做主键，我看到有人说用guid做主键，这个场景，应该是分布式环境才用的到。

回复↓

---

yejr 文章作者
2014/11/257:57 下午

分布式场景，也可以用一个发号器的模式，每次取一个号来用，但InnoDB表实际存储时，还是要有一个无业务意义的自增列做主键以保证存储效率。

回复↓