Kylin 构建Cube时优化方法如下：

**1. Hierarchy(层级) Dimensions优化**

在Fact table(事实表)中的维度满足层级关系时定义这几个维度为Hierarchy。一些列具有层次关系的Dimension组成一个Hierarchy，比如年，月，日组成了一个Hierarchy。

查询时指定group by year ; group by year , month; group by year, month,day;都可以查询，但是如果指定group by month,day就不可以了。

[**2.Derived**](http://2.Derived)**（衍生） Columns优化**

在某张Look up Table(维度表)中有多个维度，改维度表对应的一个或者多个列可以和维度表的主键是一对一的，那么可以将其设置为Derived Dimension。

在Kylin内部会将其用统一的主键来替换，以此来达到降低维度组合的数据。当仍然在一定程度上会降低查询效率。

[**3.Mandatory**](http://3.Mandatory)**（固定） Columns优化**

如果每次查询的group by中都会携带某维度，那么我们可以将这个dimension设置为Mandatory，可以将维度组合减少一半。

[**4.Aggregation**](http://4.Aggregation) **Group（聚合组） 优化**

将维度分组达到降低维度组合的手段。维度的设置主要是让不出现在同一个查询中的两个维度怒计算cuboid。其实相当于把一个cube的树结构划分成了多个不同的树。在不降低

查询性能的情况下减少cuboid的计算量。

[**5.Joint**](http://5.Joint)**(联合维度) Dimensions 优化**

如果两个或者更多的维度要么同时出现在查询中，要么同时不出现在查询中时，这几个维度可以设置成Joint(联合维度) Dimensions.

[**6.Cube**](http://6.Cube)**定义中RowKey顺序：**

Mandatory 维度，Where过滤条件中出现较多的维度，高基数维度，低基数维度