Atitit sql优化之性能优化大法 attilax总结

[1. 分析与定位 查找具体问题代码块的大法 1](#_Toc7699)

[1.1. 分类大法 分为复杂分类 和简单分类 问题往往在复杂代码块 1](#_Toc28841)

[1.2. 分类后可能依然很多，需要试试二分查找算法 1](#_Toc6966)

[1.3. 或者不多的情况下走遍历得了 1](#_Toc9106)

[2. 拆分复杂代码 分段优化 oo是个结构化好模式 2](#_Toc26022)

[3. 建立正确的索引以及索引类型 2](#_Toc29796)

[4. 物化视图+试图索引 2](#_Toc21991)

[5. 金字塔模型架构 多级模型 索引链 2](#_Toc8727)

[5.1. 时间维度金字塔 2](#_Toc23954)

[5.2. 地理维度金字塔模型 2](#_Toc13420)

[5.3. 规模维度模型 （分块 分区） 3](#_Toc6055)

[5.4. 其他维度 3](#_Toc27534)

[6. 强制索引 3](#_Toc6213)

[7. 分区 多级分区 3](#_Toc30027)

[8. 自实现索引机制 索引存储，索引新建更新同步等 3](#_Toc24425)

[8.1. 增量索引 3](#_Toc23921)

[8.2. 异步索引 3](#_Toc11875)

[8.3. 分区索引 3](#_Toc6625)

# 分析与定位 查找具体问题代码块的大法

代码量可能很多，需要分类和查找

## 分类大法 分为复杂分类 和简单分类 问题往往在复杂代码块

## 分类后可能依然很多，需要试试二分查找算法

## 或者不多的情况下走遍历得了

最终定位到问题的代码片段位置

# 拆分复杂代码 分段优化 oo是个结构化好模式

每段代码可能需要走不同的索引。 貌似很多情况下走了错误索引，导致性能不是最高。。分段后对方便每段走不同的索引

# 建立正确的索引以及索引类型

索引类型 常见的btree hash fulltext bitmap等

一般在where 等条件字段建立索引

# 物化视图+试图索引

# 金字塔模型架构 多级模型 索引链

## 时间维度金字塔

比如要得到年统计报表，就要扫描365天的记录 如果每天记录有很多的话，数据量就很多。

年统计报表可以建立在季度统计报表上，这样只需要计算4条数句。。

建立一个级联金字塔模型 年度》季度》月度》周》天

## 地理维度金字塔模型

按照地理维度建立层级模型

## 规模维度模型 （分块 分区）

## 其他维度

# 强制索引

有时候执行器没那么智能（ 可能更具一些数据统计信息计算，但数据统计信息有时候不准确了，导致执行器执行了错误的决策路径 ），走了错误的索引，可以强制指定它走正确的索引

重建索引有时候可以也可以解决这个问题

# 分区 多级分区

# 自实现索引机制 索引存储，索引新建更新同步等

## 增量索引

## 异步索引

## 分区索引