Atitit notin 性能优化算法总结

[1.1. in和exists区别 1](#_Toc23485)

[1.2. notin 能不能走索引？？答案是有些可以有些不可以 1](#_Toc18319)

[1.3. 虽然Notin里面排除多的情况下可以走索引，但数据库具体实现可能没有实现，此时可以使用join代替 2](#_Toc23736)

[1.4. 和  Not Exists的关系 2](#_Toc10960)

## in和exists区别

in和exists执行时，in是先执行子查询中的查询，然后再执行主查询。而exists查询它是先执行主查询，即外层表的查询，然后再执行子查询。

exists 和 in 在执行时效率单从执行时间来说差不多，exists要稍微优于in。在使用时一般应该是用exists而不用in

如果子查询得出的结果集记录较少，主查询中的表较大且又有索引时应该用in,反之如果外层的主查询记录较少，子查询中的表大，又有索引时使用exists。IN时不对NULL进行处理。

不过这个不知是sql标准这样规定，还是只是某一个数据库具体实现这样的算法规范。。

## notin 能不能走索引？？答案是有些可以有些不可以

少量排除情况下 ，notin 基本就是等同遍历了。Notin里面数据量少的情况下，走不走索引差别不大

Notin里面排除数量较多的情况下，走索引还是有很大差别的。

比如表里面十万数据，排除一条，走不走索引效果是一样的。。但如果排除上万数据，那走走索引还是有用的，排除的数据越多，走索引效果越好。

具体的实现实现效果和数据库类型和版本相关。一般越高级数据库与版本会对此优化越好。。

如果遇到相关数据库对此语句优化不好，那就需要利用适当调整sql语句，换个等义的更高性能写法（一般可读性会因此下降） 或者利用存储过程自定义函数等自己实现性能高的notin算法，或者用编程语言实现

## 虽然Notin里面排除多的情况下可以走索引，但数据库具体实现可能没有实现，此时可以使用join代替

代价就是可读性相对会下降写。。优先使用notin，其次join代替 可读性优先保证

## 和  Not Exists的关系

含义上说 notin 基本就是notexist 也有可能是sql标准，待确定

有可能不同的数据库 区分是为了实现俩种不同的查询算法 有些数据库俩种算法几乎一样效率，有些数据库实现的不一样。。

而且数据量大小不同的情况下，可能会走不同的路径，不能一概而论。。

## In的实现过程

　用IN写出来的SQL的优点是比较容易写及清晰易懂，这比较适合现代软件开发的风格。

　　但是用IN的SQL性能总是比较低的，从SQL执行的步骤来分析用IN的SQL与不用IN的SQL有以下区别：

　　SQL试图将其转换成多个表的连接，如果转换不成功则先执行IN里面的子查询，再查询外层的表记录，如果转换成功则直接采用多个表的连接方式查询。由此可见用IN的SQL至少多了一个转换的过程。一般的SQL都可以转换成功