关于业务主键和逻辑主键

这几天对逻辑主键、业务主键和复合主键进行了一些思考，也在网上搜索了一下相关的讨论，相关讨论可以看最下面的参考链接。下面是自己基于 SQL Server 做的一些总结，其他数据库（Oracle、**MySQL**、DB2、.....）应该也类似吧。这个只是自己一时的思考，如有不当请告知，重新思考后再修正。

定义（部分定义来源于 **SQL Server** 联机丛书）：

主键(PRIMARY KEY)：表通常具有包含唯一标识表中每一行的值的一列或一组列。这样的一列或多列称为表的主键 (PK)，用于强制表的实体完整性。

外键(FOREIGN KEY)：外键 (FK) 是用于建立和加强两个表数据之间的链接的一列或多列。在外键引用中，当一个表的列被引用作为另一个表的主键值的列时，就在两表之间创建了链接。这个列就成为第二个表的外键。

聚集索引：聚集索引基于数据行的键值在表内排序和存储这些数据行。每个表只能有一个聚集索引，因为数据行本身只能按一个顺序存储。

非聚集索引：非聚集索引包含索引键值和指向表数据存储位置的行定位器。可以对表或索引视图创建多个非聚集索引。通常，设计非聚集索引是为改善经常使用的、没有建立聚集索引的查询的性能。

自动编号列和标识符列：对于每个表，均可创建一个包含系统生成的序号值的标识符列，该序号值以唯一方式标识表中的每一行。

业务主键（自然主键）：在数据库表中把具有业务逻辑含义的字段作为主键，称为“自然主键(Natural Key)”。

逻辑主键（代理主键）：在数据库表中采用一个与当前表中逻辑信息无关的字段作为其主键，称为“代理主键”。

复合主键（联合主键）：通过两个或者多个字段的组合作为主键。

原理分析：

使用逻辑主键的主要原因是，业务主键一旦改变则系统中关联该主键的部分的修改将会是不可避免的，并且引用越多改动越大。而使用逻辑主键则只需要修改相应的业务主键相关的业务逻辑即可，减少了因为业务主键相关改变对系统的影响范围。业务逻辑的改变是不可避免的，因为“永远不变的是变化”，没有任何一个公司是一成不变的，没有任何一个业务是永远不变的。最典型的例子就是身份证升位和驾驶执照号换用身份证号的业务变更。而且现实中也确实出现了身份证号码重复的情况，这样如果用身份证号码作为主键也带来了难以处理的情况。当然应对改变，可以有很多解决方案，方案之一是做一新系统与时俱进，这对软件公司来说确实是件好事。

使用逻辑主键的另外一个原因是，业务主键过大，不利于传输、处理和存储。我认为一般如果业务主键超过8字节就应该考虑使用逻辑主键了，因为int是4字节的，bigint是8字节的，而业务主键一般是字符串，同样是 8 字节的 bigint 和 8 字节的字符串在传输和处理上自然是 bigint 效率更高一些。想象一下 code == "12345678" 和 id == 12345678 的汇编码的不同就知道了。当然逻辑主键不一定是 int 或者 bigint ，而业务主键也不一定是字符串也可以是 int 或 datetime 等类型，同时传输的也不一定就是主键，这个就要具体分析了，但是原理类似，这里只是讨论通常情况。同时如果其他表需要引用该主键的话，也需要存储该主键，那么这个存储空间的开销也是不一样的。而且这些表的这个引用字段通常就是外键，或者通常也会建索引方便查找，这样也会造成存储空间的开销的不同，这也是需要具体分析的。

使用逻辑主键的再一个原因是，使用 int 或者 bigint 作为外键进行联接查询，性能会比以字符串作为外键进行联接查询快。原理和上面的类似，这里不再重复。

使用逻辑主键的再一个原因是，存在用户或维护人员误录入数据到业务主键中的问题。例如错把 RMB 录入为 RXB ，相关的引用都是引用了错误的数据，一旦需要修改则非常麻烦。如果使用逻辑主键则问题很好解决，如果使用业务主键则会影响到其他表的外键数据，当然也可以通过级联更新方式解决，但是不是所有都能级联得了的。

使用业务主键的主要原因是，增加逻辑主键就是增加了一个业务无关的字段，而用户通常都是对于业务相关的字段进行查找（比如员工的工号，书本的 ISBN No. ），这样我们除了为逻辑主键加索引，还必须为这些业务字段加索引，这样数据库的性能就会下降，而且也增加了存储空间的开销。所以对于业务上确实不常改变的基础数据而言，使用业务主键不失是一个比较好的选择。另一方面，对于基础数据而言，一般的增、删、改都比较少，所以这部分的开销也不会太多，而如果这时候对于业务逻辑的改变有担忧的话，也是可以考虑使用逻辑主键的，这就需要具体问题具体分析了。

昵称：星火spark
园龄：4年2个月
粉丝：0
关注：0
+加关注

< 2016年11月 >

日	一	二	三	四	五	六
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10

搜索

找找看
谷歌搜索

常用链接

我的随笔
我的评论
我的参与
最新评论
我的标签
更多链接

随笔分类

ant
bootstrap
dwz
eos
ext
GIS专业相关(1)
greenetplum
java(8)
javascript(6)
jetty
jquery
luence
maven
mdrill
mysql
oracle(1)
tomcat(1)
weblogic(2)
常见问题(10)
大数据
感悟(1)
工作流
后台(2)
前端(1)
数据库(2)
知识库(2)

随笔档案

使用业务主键的另外一个原因是，对于用户操作而言，都是通过业务字段进行的，所以在这些情况下，如果使用逻辑主键的话，必须要多做一次映射转换的动作。我认为这种担心是多余的，直接使用业务主键查询就能得到结果，根本不用管逻辑主键，除非业务主键本身就不唯一。另外，如果在设计的时候就考虑使用逻辑主键的话，编码的时候也是会以主键为主进行处理的，在系统内部传输、处理和存储都是相同的主键，不存在转换问题。除非现有系统是使用业务主键，要把现有系统改成使用逻辑主键，这种情况才会存在转换问题。暂时没有想到还有什么场景是存在这样的转换的。

使用业务主键的再一个原因是，对于银行系统而言安全性比性能更加重要，这时候就会考虑使用业务主键，既可以作为主键也可以作为冗余数据，避免因为使用逻辑主键带来的关联丢失问题。如果由于某种原因导致主表和子表关联关系丢失的话，银行可是会面临无法挽回的损失的。为了杜绝这种情况的发生，业务主键需要在重要的表中有冗余存在，这种情况最好的处理方式就是直接使用业务主键了。例如身份证号、存折号、卡号等。所以通常银行系统都要求使用业务主键，这个需求并不是出于性能的考虑而是出于安全性的考虑。

使用复合主键的主要原因和使用业务主键是相关的，通常业务主键只使用一个字段不能解决问题，那就只能使用多个字段了。例如使用姓名字段不够用了，再加个生日字段。这种使用复合主键方式效率非常低，主要原因和上面对于较大的业务主键的情况类似。另外如果其他表要与该表关联则需要引用复合主键的所有字段，这就不单纯是性能问题了，还有存储空间的问题了，当然你也可以认为这是合理的数据冗余，方便查询，但是感觉有点得不偿失。

使用复合主键的另外一个原因是，对于关系表来说必须关联两个实体表的主键，才能表示它们之间的关系，那么可以把这两个主键联合组成复合主键即可。如果两个实体存在多个关系，可以再加一个顺序字段联合组成复合主键，但是这样就会引入业务主键的弊端。当然也可以另外对这个关系表添加一个逻辑主键，避免了业务主键的弊端，同时也方便其他表对它的引用。

综合来说，网上大多数人是倾向于用逻辑主键的，而对于实体表用复合主键方式的应该没有多少人认同。支持业务主键的人通常有种误解，认为逻辑主键必须对用户来说有意义，其实逻辑主键只是系统内部使用的，对用户来说是无需知道的。

结论或推论：

1、尽量避免使用业务主键，尽量使用逻辑主键。

2、如果要使用业务主键必须保证业务主键相关的业务逻辑改变的概率为0，并且业务主键不太大，并且业务主键不能交由用户修改。

3、除关系表外，尽量不使用复合主键。

使用逻辑主键的最佳实践指南：

1、足够用就好。系统使用的生命周期以100年为限，逻辑主键数据类型采用下表规则，如果不确定则使用int类型。

数据量	数据类型	数据大小	生成频率	备注
< 128	tinyint	1 字节	1条/年	频率过低，不太靠谱，不建议采用
< 3 万	smallint	2 字节	27条/月	频率较低，慎用
< 21 亿	int	4 字节	40条/分钟	能满足大部分情况
< 922 亿亿	bigint	8 字节	292万条/毫秒	能满足绝大部分情况
>= 922 亿亿	uniqueidentifier	16 字节	100亿用户同时每毫秒生成10亿条，可以连续生成10亿年	可用于分布式、高并发的应用

2、一般采用自增长方式或NewID()方式。

3、主键字段名称一般采用“表名ID”方式，方便识别和表联接。

4、如果表存在分布式应用，则可以考虑采用不同起始值，相同步长方式自增。例如有3个部署在不同地方的库，则可以如下设计：

起始值	步长
1	10
2	10
3	10

步长统一设置10是为了方便日后扩展，这样不同库之间也能保持主键唯一性了，也方便合并。

5、如果存在高并发性需求或数据表迁移需求，可以考虑使用uniqueidentifier类型，并使用NewID()函数。

6、可以考虑对业务主键建立唯一性索引，以实现业务主键唯一性的业务需求。

7、如果需要考虑业务主键的性能需求，可以把业务主键建立聚集索引，而逻辑主键只建立主键约束和非聚集索引即可。

8、关系表可以考虑采用复合主键方式，复合主键不用于实体表。

星火spark
关注 - 0
粉丝 - 0
+加关注

0 0

« 上一篇：关于su和su -的区别
» 下一篇：在Linux下如何用Shell脚本读写XML？现有一个config.xml（转）

posted @ 2016-10-31 14:14 星火spark 阅读(21) 评论(0) 编辑 收藏