<http://blog.csdn.net/lgb934/article/details/8662956>

### 数据库分区表（一）什么情况下需要分区，准备需要分区的数据

什么数据库需要进行分区？首先看一下我们的案例：2010年6月我们六期IT开发团队接到一个XX全国连锁店的餐饮系统，经过一周的敏捷开发之后，XX餐饮系统正式上线了，由于该软件的功能强大，操作简单，功能灵活等特性，很快在全国各地铺展开来。XX餐饮店的美食也颇受顾客的喜爱，有的店每天的收入高达1W元人民币，每天这么多的收入，那么每天要产生多大的订单呢？< xmlnamespace prefix ="o" ns ="urn:schemas-microsoft-com:office:office" />

这是一个很大的数据量，在刚开始的几个月中，我们还能够正常根据所产生的所有订单，根据产品的类别和销售情况，产生一个商品销售排行报表。可是随着数据量越来越大，现在每次进行商品销售排行汇总都要进行很长时间的等待。而我们的服务器性能是没有问题的，那么我们只有改善数据库的结构来提高数据的检索效率了。改善数据库的结构有两种，一种是采用存储过程代替普通的SQL语句，另外一种就是使用数据库系统中增强索引和规划分区表进行优化，这里我们采用第二种方案来解决问题。

我们这里采用数据分区表的方法来进行数据优化，那么是不是只要数据量足够大之后就要进行数据库分区表来提高查询效率呢？数据多了并不是创建分区表的惟一条件，哪怕你有一千万条记录，但是这一千万条记录都是常用的记录，那么最好也不要使用分区表，说不定会得不偿失。只有你的数据是分段的数据，那么才要考虑到是否需要使用分区表。这里我们的数据是按时间段进行查询的，所以我们可以在这里进行分区。

上面这些就是我们进行数据分区表的前提，如果你有这样的需求，那么还等什么，赶快使用数据分区表吧。这里我们提供一个简单的数据库，方便以后实例的练习，如果你需要的话，可以[点击这里](http://i9ypbq.bay.livefilestore.com/y1pT7UdbRiQaRBHgrenYp8VZxCJeXTcyAJA7fPMW9GC_MmKXfoIMgrib5Bj9Z9Ktkf9XW9m_TbwVmrPbTBcuhWZRVtvL1fvXnI2/%E6%95%B0%E6%8D%AE%E5%BA%93%E5%88%86%E5%8C%BA%E8%A1%A8%E6%A0%B7%E4%BE%8B%E6%95%B0%E6%8D%AE%E5%BA%93.zip?download&psid=1&AVOverride=1)下载。如果你有一个刚刚备份过来的数据库，就是还原不成功怎么办？[这里有解决方案](http://chenxu.wo.blog.163.com/blog/static/50239687201111093851731/)。

### 数据库分区表（二）什么是分区表？

　表分区分为水平分区和垂直分区。水平分区将表分为多个表。每个表包含的列数相同，但是行更少。例如，可以将一个包含十亿行的表水平分区成 12 个表，每个小表表示特定年份内一个月或几个月的数据。任何需要特定月份数据的查询只需引用相应月份的表。而垂直分区则是将原始表分成多个只包含较少列的表。水平分区是最常用分区方式，后面我们以水平分区来介绍具体实现方法。

    简单一点说，分区表就是将一个大表分成若干个小表。这里，我们有一个销售记录表，记录着每个某餐饮店的订单情况，那么你就可以把这个销售记录表按时间分成几个小表，我们这里分成15个小表。2011年以前的记录使用一个表，2011年的记录每两个月使用一个表（2011//01/01-2011/03/01，每个两个月一个表，一共六个表），2012年,2013（）年的的记录同上，2013年的记录使用一个表，2012年以后的记录使用一个表。那么，你想查询哪个年份的记录，就可以去相对应的表里查询，由于每个表中的记录数少了，查询起来时间自然也会减少。

    但将一个大表分成几个小表的处理方式，会给程序员增加编程上的难度。以添加记录为例，以上15个表是独立的15个表，在不同时间添加记录的时候，程序员要使用不同的SQL语句，例如在2011年添加记录时，程序员要将记录添加到2011年那个表里；在2012年添加记录时，程序员要将记录添加到2012年的那个表里。这样，程序员的工作量会增加，出错的可能性也会增加。

    使用分区表就可以很好的解决以上问题。分区表可以从物理上将一个大表分成几个小表，但是从逻辑上来看，还是一个大表。分区表可以将一个销售记录表分成十五个物理上的小表，但是对于程序员而言，他所面对的依然是一个大表，无论是2010年添加记录还是2012年添加记录，对于程序员而言是不需要考虑的，他只要将记录插入到销售记录表——这个逻辑中的大表里就行了。SQL Server会自动地将它放在它应该呆在的那个物理上的小表里。

同样，对于查询而言，程序员也只需要设置好查询条件，OK，SQL Server会自动将去相应的表里查询，不用管太多事了。

　　分区表的准备工作完事了，下一步就是具体创建分区表了，等不急了吧，看下篇博文你就知道了。

### 数据库分区表（三）如何创建分区表1？

创建分区表必须要经过下面五个步骤。

1）创建文件组

2）创建文件

3）创建分区函数

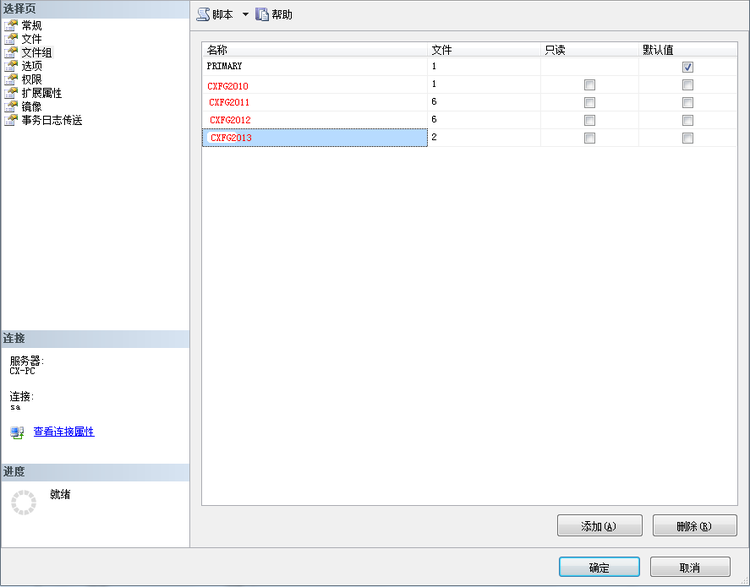
4）创建分区方案

5）创建分区表

（1）创建文件组，有两种方案，一种是通过手动添加，另外一种就是通过SQL脚本进行添加。下面以两种方案来说明：

       方案一：创建文件组，虽然这一步我们可以省略，因为我们可以直接使用Primary文件（也就是系统主文件）。但是为了方便管理，我们还是要创建几个文件组，这样可以将不同的小表（不同时间段，或者不同数据表）放在不同的文件组里，既便于理解又可以提高运行速度。

       打开SQL Server Management Studio，找到分区表所在的数据库，右键单击选择“属性”，选择“文件组”选项，单击下面的“添加”按钮，添加X个文件组，如下图所示：



       方案二：通过查询分析器SQL脚本执行

              ALTER DATABASE CXFunSche ADD FILEGROUP CXFG2010

ALTER DATABASE CXFunSche ADD FILEGROUP CXFG2011

ALTER DATABASE CXFunSche ADD FILEGROUP CXFG2012

ALTER DATABASE CXFunSche ADD FILEGROUP CXFG2013

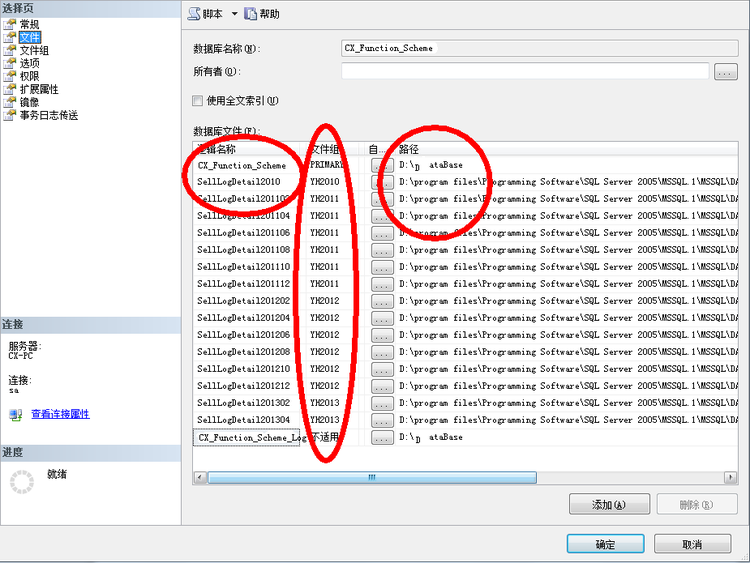
（2）创建数据库文件

    方案一：创建了文件组之后，还要再创建几个数据库文件。为什么要创建数据库文件，这很好理解，因为分区的小表必须要放在硬盘上，而放在硬盘上的什么地方呢？当然是文件里啦。再说了，文件组中没有文件，文件组还要来有啥用呢？还是在上图的那个界面，选择“文件”选项，然后添加几个文件。在添加文件的时候要注意以下几点：

    1、不要忘记将不同的文件放在文件组中。当然一个文件组中也可以包含多个不同的文件。

2、如果可以的话，将不同的文件放在不同的硬盘分区里，最好是放在不同的独立硬盘里。要知道IQ的速度往往是影响SQL Server运行速度的重要条件之一。将不同的文件放在不同的硬盘上，可以加快SQL Server的运行速度。

在本文的实例中，数据库主文件与分区文件就不在同一个目录下，各个分区文件也可以放置在不同的目录下，建议大家在练习时使用。



    方案二：通过查询分析器SQL脚本执行

    ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail2010', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail2010.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2010

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201102', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201102.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2010

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201104', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201104.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2011

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201106', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201106.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2011

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201108', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201108.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2011

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201110', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201110.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2011

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201112', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201112.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2011

ALTER DATABASE CXFunSche ADD FILE ( NAME = N'SellLogDetail201202', FILENAME = N'D:\program files\Programming Software\SQL Server 2005\MSSQL.1\MSSQL\DATA\SellLogDetail201202.ndf' , SIZE = 3072KB , FILEGROWTH = 1024KB ) TO FILEGROUP CXFG2012

.

.

.

由于创建的数据文件太多了，这里就不一一贴出来了，剩下的几个，留给大家自己锻炼一下吧。创建完分区文件后，你就可以在D盘相应的目录下找到你刚刚创建的数据文件。

### 数据库分区表（三）如何创建分区表2？

（3）创建分区函数

创建一个分区函数，创建分区函数的目的是告诉SQL Server以什么方式对分区表进行分区。这一步必须要什么SQL脚本来完成。以上面的例子，我们要将销售记录表按时间分成15个小表。划分的时间为：

第1个小表：2011-01-01以前的数据（不包含2010-01-01）。

    第2个小表：2011-01-01（包含2011-01-01）到2010-01-01之间的数据。

    第3个小表：2011-03-01（包含2011-03-01）到2011-05-01之间的数据。

第4个小表：2011-05-01（包含2011-05-01）到2011-07-01之间的数据。

第5个小表：……

第6个小表：……

--创建分区函数

CREATE PARTITION FUNCTION partfun\_CX (datetime)

AS RANGE RIGHT FOR VALUES('20110101','20110301','20110501','20110701','20110901','20111101','20120101','20120301','20120501','20120701','20120901','20121101','20130101','20130301','20130501','20130701')

1、CREATE PARTITION FUNCTION意思是创建一个分区函数。

    2、partfun\_CX为分区函数名称。

    3、AS RANGE RIGHT为设置分区范围的方式为Right，也就是右置方式。

4、FOR VALUES ('20110101','20110301','20110501','20130101',……)为按这些个值来分区，Values中的值就是分区的条件

（4）创建分区方案

分区方案的作用是将分区函数生成的分区映射到文件组中去。分区函数的作用是告诉SQL Server，如何将数据进行分区，而分区方案的作用则是告诉SQL Server将已分区的数据放在哪个文件组中。

--.创建分区方案

CREATE PARTITION SCHEME partsch\_CX

AS PARTITION partfun\_CX

TO (

    CX2010,

    CX2011, CX2011, CX2011, CX2011, CX2011,CX2011,

   CX2012, CX2012, CX2012, CX2012, CX2012,CX2012,

   CX2013, CX2013, CX2013, CX2013)

1、CREATE PARTITION SCHEME意思是创建一个分区方案。

    2、partsch\_CX为分区方案名称。

    3、AS PARTITION partfun\_CX说明该分区方案所使用的数据划分条件（也就是所使用的分区函数）为partfun\_CX。

    4、TO后面的内容是指partfun\_CX分区函数划分出来的数据对应存放的文件组。

    到此为止，分区函数和分区方案就创建完毕了。创建后的分区函数和分区方案在数据库的“存储”中可以看到

（5）创建分区表

       创建分区表，创建方式和创建普遍表类似，如下所示：

       CREATE TABLE t\_partition3(

    ptId int IDENTITY(1,1) NOT NULL primary key NONCLUSTERED,

    ptName varchar(16) NOT NULL,

    sellTime[datetime] NOT NULL

) ON partsch\_CX(sellTime)

如果你按照上面的代码来实现的话出出现下图所示的错误代码提示：

消息1908，级别16，状态1，第1 行

列'sellTime' 是索引'PK\_\_t\_partition3\_\_671F4F74' 的分区依据列。唯一索引的分区依据列必须是索引键的子集。

消息1750，级别16，状态0，第1 行

无法创建约束。请参阅前面的错误消息。

     这里是不能创建除分区表中除分区字段以外的其它字段为聚集索引，因为聚集索引是在物理上顺序存储的，而分区表是将数据分别存储在不同的表中，这两个概念是冲突的。如果我们创建了其它字段的聚集索引，那么就会按照其它字段在物理上顺序存储，而我们的分区表是根据分区字段进行物理上的顺序存储的。

### 数据库分区表（四）将普通表转换成分区表

2011-02-15 16:27:43|  分类： [数据库学习](http://chenxu.wo.blog.163.com/blog/#m=0&t=1&c=fks_087071092086082070093083080095081086083064093080094068) |  标签： |字号大中小 订阅

　　我们的数据库已经投入使用一段时间了，但是当时没有创建创建分区表，现在我们需要做的是将普通表转换成分区表，但是并不能影响我们数据库里面的数据，那么我们应该如何做呢？只需在该表上创建一个聚集索引，并在该聚集索引中使用分区方案即可。

     说的很简单，但是在实现实现可就没有那么容易了，因为你的数据库中存在主键，外键等约束关系，那么我们在将普通表转换成分区表时，首先就需要解决这些问题。

     我们知道分区表时某个字段为分区条件的，除了这个字段之外的其他字段是不能创建聚集索引的，所以我们将普通表转换成分区表时，必须要删除聚集索引，然后再重新创建一个新的聚集索引，在该聚集索引中使用分区方案。

     但是我们需要修改的t\_sellLog表中的orderId既是主键又是聚集索引，而且还是其它表的外键。因此，我们只能先删除外键关联，再删除主键，然后重新创建orderId为主键，但是设置为非聚集索引，然后将我们的sellTime字段设置为聚集索引，最后添加上我们的外键约束，至此普通表转换成分区表的工作结束,代码如下：

     --查看外键约束

use CX\_Partiton\_Scheme

exec sp\_helpconstraint t\_SellLog

--删除外键约束

alter table t\_sellLog drop constraint FK\_t\_SellLog\_t\_User

--删掉主键

ALTER TABLE t\_SellLog DROP constraint PK\_t\_SellLog

--创建主键，但不设为聚集索引

ALTER TABLE t\_SellLog ADD CONSTRAINT PK\_t\_SellLog PRIMARY KEY NONCLUSTERED (

    orderId ASC

)

 ON [PRIMARY]

 --创建一个新的聚集索引，在该聚集索引中使用分区方案

CREATE CLUSTERED INDEX CT\_SellLog ON t\_SellLog(sellTime)

ON partsch\_CX([sellTime])

 --添加删除掉的外键约束（具体自己根据实际情况自己实现）

     转换成功之后，我们可以通过下面代码查看每个分区表中的记录数：

   --统计所有分区表中的记录总数

select $PARTITION.partfun\_CX([sellTime]) as 分区编号,count(orderId) as 记录数from t\_SellLog group by $PARTITION.partfun\_CX([sellTime])

     我们还可以通过下面的代码，查看数据库库中的数据在哪个分区中：

   --查看数据库表中的数据在哪个分区中

select $PARTITION.partfun\_CX('2010-10-1')  --查询年月日的数据在哪个分区中

select $PARTITION.partfun\_CX('2011-01-1')  --查询年月日的数据在哪个分区中如果你想比较一下我们使用分区方案之后和之前程序有多少效率提高，我们可以通过下面的语句来看看一下脚本的执行时间就OK了，我经过测试的数据是快了0.017秒，一方面由于我们的测试数据量比较小，另一方面我的机器配置还是蛮不错的。

     --查看SQL脚本的执行时间

   select getDate()

   select \* from t\_sellLog

   select getDate()

转自：<http://chenxu.wo.blog.163.com/blog/static/50239687201111542743194/>