Oracle查询慢的原因总结Oracle认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/622/2021_2022_Oracle_E6_9F_A5_E8_c102_622675.htm 查询速度慢的原因很多，常见如下几种： 1、没有索引或者没有用到索引（这是查询慢最常见的问题，是程序设计的缺陷） 2、I/O吞吐量小，形成了瓶颈效应。 3、没有创建计算列导致查询不优化。 4、内存不足 5、网络速度慢 6、查询出的数据量过大（可以采用oracle认证更多详细资料多次查询，其他的方法降低数据量） 7、锁或者死锁（这也是查询慢最常见的问题，是程序设计的缺陷） 8、sp_lock，sp_who，活动的用户查看，原因是读写竞争资源。 9、返回了不必要的行和列 10、查询语句不好，没有优化 可以通过如下方法来优化查询 ： 1、把数据、日志、索引放到不同的I/O设备上，增加读取速度，以前可以将Tempdb应放在RAID0上，SQL2000不在支持。数据量（尺寸）越大，提高I/O越重要。 2、纵向、横向分割表，减少表的尺寸（sp_spaceuse） 3、升级硬件 4、根据查询条件，建立索引，优化索引、优化访问方式，限制结果集的数据量。注意填充因子要适当（最好是使用默认值0）。索引应该尽量小，使用字节数小的列建索引好（参照索引的创建），不要对有限的几个值的字段建单一索引如性别字段 5、提高网速； 6、扩大服务器的内存，Windows 2000和SQL server 2000能支持4-8G的内存。配置虚拟内存：虚拟内存大小应基于计算机上并发运行的服务进行配置。运行 Microsoft SQL Server？ 2000 时，可考虑将虚拟内存大小设置为计算机中安装的物理内存的 1.5 倍。如果另外安装了全文检索功能，并打算运行 Microsoft 搜索

服务以便执行全文索引和查询，可考虑：将虚拟内存大小配置为至少是计算机中安装的物理内存的 3 倍。将 SQL Server max server memory 服务器配置选项配置为物理内存的 1.5 倍（虚拟内存大小设置的一半）。 7、增加服务器 CPU个数；但是必须明白并行处理串行处理更需要资源例如内存。使用并行还是串行程是MsSQL自动评估选择的。单个任务分解成多个任务，就可以在处理器上运行。例如耽搁查询的排序、连接、扫描和GROUP BY字句同时执行，SQL SERVER根据系统的负载情况决定最优的并行等级，复杂的需要消耗大量的CPU的查询最适合并行处理。但是更新操作Update，Insert，Delete还不能并行处理。 8、如果是使用like进行查询的话，简单的使用index是不行的，但是全文索引，耗空间。like ’a% ’使用索引 like ’%a ’不使用索引用 like ’%a% ’查询时，查询耗时和字段值总长度成正比，所以不能用CHAR类型，而是VARCHAR.对于字段的值很长的建全文索引。 9、DB Server 和APPLication Server 分离；OLTP和OLAP分离 10、分布式分区视图可用于实现数据库服务器联合体。联合体是一组分开管理的服务器，但它们相互协作分担系统的处理负荷。这种通过分区数据形成数据库服务器联合体的机制能够扩大一组服务器，以支持大型的多层 Web 站点的处理需要。有关更多信息，参见设计联合数据库服务器。（参照SQL帮助文件 ’分区视图 ’） a、在实现分区视图之前，必须先水平分区表 b、在创建成员表后，在每个成员服务器上定义一个分布式分区视图，并且每个视图具有相同的名称。这样，引用分布式分区视图名的查询可以在任何一个成员服务器上运行。系统操作如同每个成员服务器上都有一个原始表的复

本一样，但其实每个服务器上只有一个成员表和一个分布式分区视图。数据的位置对应用程序是透明的。 11、重建索引 DBCC REINDEX，DBCC INDEXDEFRAG，收缩数据和日志 DBCC SHRINKDB，DBCC SHRINKFILE. 设置自动收缩日志。对于大的数据库不要设置数据库自动增长，它会降低服务器的性能。在T-sql的写法上有很大的讲究，下面列出常见的要点：首先， DBMS处理查询计划的过程是这样的： 1、 查询语句的词法、语法检查 2、 将语句提交给DBMS的查询优化器 3、 优化器做代数优化和存取路径的优化 4、 由预编译模块生成查询规划 5、 然后在合适的时间提交给系统处理执行 6、 最后将执行结果返回给用户其次，看一下SQL SERVER的数据存放的结构：一个页面的大小为8K（8060）字节，8个页面为一个盘区，按照B树存放。 12、Commit和rollback的区别 Rollback：回滚所有的事物。 Commit：提交当前的事物。没有必要在动态SQL里写事物，如果要写请写在外面如： begin tran exec（@s） commit trans 或者将动态SQL 写成函数或者存储过程。 13、在查询Select语句中用Where字句限制返回的行数，避免表扫描，如果返回不必要的数据，浪费了服务器的I/O资源，加重了网络的负担降低性能。如果表很大，在表扫描的期间将表锁住，禁止其他的联接访问表，后果严重。 14、SQL的注释申明对执行没有任何影响15、尽可能不使用光标，它占用大量的资源。如果需要row-by-row地执行，尽量采用非光标技术，如：在客户端循环，用临时表，Table变量，用子查询，用Case语句等等。游标可以按照它所支持的提取选项进行分类： 只进必须按照从第一行到最后一行的顺序提取行。FETCH NEXT 是唯一允许的提取操作，也是默认方

式。可滚动性可以在游标中任何地方随机提取任意行。游标的技术在SQL2000下变得功能很强大，他的目的是支持循环。有四个并发选项 READ_ONLY：不允许通过游标定位更新（Update），且在组成结果集的行中没有锁。 OPTIMISTIC WITH valueS：乐观并发控制是事务控制理论的一个标准部分。乐观并发控制用于这样的情形，即在打开游标及更新行的间隔中，只有很小的机会让第二个用户更新某一行。当某个游标以此选项打开时，没有锁控制其中的行，这将有助于最大化其处理能力。如果用户试图修改某一行，则此行的当前值会与最后一次提取此行时获取的值进行比较。如果任何值发生改变，则服务器就会知道其他人已更新了此行，并会返回一个错误。如果值是一样的，服务器就执行修改。选择这个并发选项OPTIMISTIC WITH ROW VERSIONING：此乐观并发控制选项基于行版本控制。使用行版本控制，其中的表必须具有某种版本标识符，服务器可用它来确定该行在读入游标后是否有所更改。在 SQL Server 中，这个性能由 timestamp 数据类型提供，它是一个二进制数字，表示数据库中更改的相对顺序。每个数据库都有一个全局当前时间戳值：@@DBTS.每次以任何方式更改带有 timestamp 列的行时，SQL Server 先在时间戳列中存储当前的 @@DBTS 值，然后增加 @@DBTS 的值。如果某 个表具有 timestamp 列，则时间戳会被记到行级。服务器就可以比较某行的当前时间戳值和上次提取时所存储的时间戳值，从而确定该行是否已更新。服务器不必比较所有列的值，只需比较 timestamp 列即可。如果应用程序对没有 timestamp 列的表要求基于行版本控制的乐观并发，则游标默认为基于数值的乐观并发控制。SCROLL

LOCKS 这个选项实现悲观并发控制。在悲观并发控制中，在把数据库的行读入游标结果集时，应用程序将试图锁定数据库行。在使用服务器游标时，将行读入游标时会在其上放置一个更新锁。如果在事务内打开游标，则该事务更新锁将一直保持到事务被提交或回滚；当提取下一行时，将除去游标锁。如果在事务外打开游标，则提取下一行时，锁就被丢弃。因此，每当用户需要完全的悲观并发控制时，游标都应在事务内打开。更新锁将阻止任何其它任务获取更新锁或排它锁，从而阻止其它任务更新该行。然而，更新锁并不阻止共享锁，所以它不会阻止其它任务读取行，除非第二个任务也在要求带更新锁的读取。滚动锁根据在游标定义的 Select 语句中指定的锁提示，这些游标并发选项可以生成滚动锁。滚动锁在提取时在每行上获取，并保持到下次提取或者游标关闭，以先发生者为准。下次提取时，服务器为新提取中的行获取滚动锁，并释放上次提取中行的滚动锁。滚动锁独立于事务锁，并可以保持到一个提交或回滚操作之后。如果提交时关闭游标的选项为关，则 COMMIT 语句并不关闭任何打开的游标，而且滚动锁被保留到提交之后，以维护对所提取数据的隔离。所获取滚动锁的类型取决于游标并发选项和游标 Select 语句中的锁提示。锁提示 只读 乐观数值 乐观行版本控制 锁定无提示 未锁定 未锁定 未锁定 更新 NOLOCK 未锁定未锁定未锁定 未锁定 HOLDLOCK 共享 共享 共享 更新 UPDLOCK 错误 更新 更新 更新 TABLOCKX 错误未锁定未锁定更新其它 未锁定 未锁定 未锁定 更新 *指定 NOLOCK 提示将使指定了该提示的表在游标内是只读的。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com