使用智能优化器提高Oracle的性能极限 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/246/2021_2022__E4_BD_BF_E7_94_A8_E6_99_BA_E8_c102_246113.htm 消耗在准备新的SQL语句的时间是Oracle SQL语句执行时间的最重要的组成部分。但是通过理解Oracle内部产生执行计划的机制，你能够控制Oracle花费在评估连接顺序的时间数量，并且能在大体上提高查询性能。 一、准备执行SQL语句 当SQL语句进入Oracle的库缓存后，在该语句准备执行之前，将执行下列步骤： 1) 语法检查：检查SQL语句拼写是否正确和词序。 2) 语义分析：核实所有的与数据字典不一致的表和列的名字。 3) 轮廓存储检查：检查数据字典，以确定该SQL语句的轮廓是否已经存在。 4) 生成执行计划：使用基于成本的优化规则和数据字典中的统计表来决定最佳执行计划。 5) 建立二进制代码：基于执行计划，Oracle生成二进制执行代码。 一旦为执行准备好了SQL语句，以后的执行将很快发生，因为Oracle认可同一个SQL语句，并且重用那些语句的执行。然而，对于生成特殊的SQL语句，或嵌入了文字变量的SQL语句的系统，SQL执行计划的生成时间就很重要了，并且前一个执行计划通常不能够被重用。对那些连接了很多表的查询，Oracle需要花费大量的时间来检测连接这些表的适当顺序。 二、评估表的连接顺序 在SQL语句的准备过程中，花费最多的步骤是生成执行计划，特别是处理有多个表连接的查询。当Oracle评估表的连接顺序时，它必须考虑到表之间所有可能的连接。例如：六个表的之间连接有720（6的阶乘，或6 * 5 * 4 * 3 * 2 * 1 = 720）种可能的连接线路。当一个查询中含有超过10个表的连接时

，排列的问题将变得更为显著。对于15个表之间的连接，需要评估的可能查询排列将超过1万亿（准确的数字是1,307,674,368,000）种。 三、使用optimizer_search_limit参数来设定限制 通过使用optimizer_search_limit参数，你能够指定被优化器用来评估的最大的连接组合数量。使用这个参数，我们将能够防止优化器消耗不定数量的时间来评估所有可能的连接组合。如果在查询中表的数目小于optimizer_search_limit的值，优化器将检查所有可能的连接组合。 例如：有五个表连接的查询将有120（5! = 5 * 4 * 3 * 2 * 1 = 120）种可能的连接组合，因此如果optimizer_search_limit等于5（默认值），则优化器将评估所有的120种可能。optimizer_search_limit参数也控制着调用带星号的连接提示的阀值。当查询中的表的数目比optimizer_search_limit小时，带星号的提示将被优先考虑。 另一个工具：参数optimizer_max_permutations 初始化参数optimizer_max_permutations定义了优化器所考虑组合数目的上限，且依赖于初始参数optimizer_search_limit。optimizer_max_permutations的默认值是80,000。 参数optimizer_search_limit和optimizer_max_permutations一起来确定优化器所考虑的组合数目的上限：除非（表或组合数目）超过参数optimizer_search_limit 或者optimizer_max_permutations设定的值，否则优化器将生成所有可能的连接组合。一旦优化器停止评估表的连接组合，它将选择成本最低的组合。 四、使用ordered提示指定连接顺序 你能够设定优化器所执行的评估数目的上限。但是即使采用有很高价值的排列评估，我们仍然拥有使优化器可以尽早地放

弃复杂的查询的重要机会。回想一下含有15个连接查询的例子，它将有超过1万亿种的连接组合。如果优化器在评估了80,000个组合后停止，那么它才仅仅评估了0.000006%的可能组合，而且或许还没有为这个巨大的查询找到最佳的连接顺序。 在Oracle SQL中解决此问题的最好的方法是手工指定表的连接顺序。为了尽快创建最小的解决方案集，这里所遵循的规则是将表结合起来，通常优先使用限制最严格的WHERE子句来连接表。 下面的代码是一个查询执行计划的例子，该例子在emp表的关联查询上强制执行了嵌套的循环连接。注意，我已经使用了ordered提示来直接最优化表的评估顺序，最终它们表现在WHERE子句上。0select /* ordered use_nl(bonus) parallel(e, 4) */e.ename,hiredate,b.comm.fromemp e,bonus bwheree.ename = b.ename 这个例子要求优化器按顺序连接在SQL语句的FROM子句中指定的表，在FROM子句中的第一个表指定了驱动表。ordered提示通常被用来与其它的提示联合起来来保证采用正确的顺序连接多个表。它的用途更多的是在扭转连接表数在四个以上的数据仓库的查询方面。另外一个例子，下面的查询使用ordered提示按照指定的顺序来连接表：emp、dept、sal，最后是bonus。我通过指定emp到dept使用哈希连接和sal到bonus使用嵌套循环连接，来进一步精炼执行计划。0select /* ordered use_hash (emp, dept) use_nl (sal, bonus) */fromemp,dept,sal,bonuswhere . . . 五、实践建议 实际上，更有效率的做法是在产品环境中减小optimizer_max_permutations参数的大小，并且总是使用稳定的优化计划或存储轮廓来防止出现耗时的含有大量连接的查询。一旦找到最佳的连接顺序，您就可以通过增加ordered提

示到当前的查询中，并保存它的存储轮廓，来为这些表手工指定连接顺序，从而使其持久化。 当你打算使用优化器来稳定计划，则可以照下面的方法使执行计划持久化，临时将optimizer_search_limit设置为查询中的表的数目，从而允许优化器考虑所有可能的连接顺序。然后，通过重新编排WHERE子句中表的名字，并使用ordered提示，与存储轮廓一起使变更持久化，来调整查询。在查询中包含四个以上的表时，ordered提示和存储轮廓将排除耗时的评估SQL连接顺序解析的任务，从而提高查询的速度。 一旦检测到最佳的连接顺序，我们就可以使用ordered提示来重载optimizer_search_limit和optimizer_max_permutations参数。ordered提示要求表按照它们出现在FROM子句中的顺序进行连接，所以优化器没有加入描述。 作为一个Oracle专业人员，你应该知道在SQL语句第一次进入库缓存时可能存在重大的启动延迟。但是聪明的Oracle DBA和开发人员能够改变表的搜索限制参数或者使用ordered提示来手工指定表的连接顺序，从而显著地减少优化和执行新查询所需的时间。