

## 第九部分 并行环境与分布式环境

### 第38章 并行查询管理

本章要点：

- 介绍

- 并行加载

- 并行恢复

- 并行传播（复制）

- 并行SQL执行

- 可以并行的SQL操作

- 理解并行度

- 理解查询服务器进程

- 分析对象以更新统计

- 理解9,3,1算法

- 理解并行DML

- OPS环境下的并行执行

- Oracle8i中的改变

- 调整并行查询

- 疑难解答

#### 38.1 介绍

具有大内存与CPU资源的巨型系统已经出现十几年了，然而，这些系统只能使用专有的操作系统，并且并不是非常有效的。使用UNIX开放系统体系结构的多处理器机器在90年代初出现，这些机器具有众多的硬件资源并相对较便宜。Oracle在Oracle7.1版本中引入了并行查询选项（PQO）以充分使用这些系统中可用的硬件资源。Oracle并行查询选项允许长时间运行的SQL操作（主要是查询），以协同方式在多个CPU间运行，这使系统减少了资源密集型SQL操作的运行时间。

并行查询选项使多个服务器进程可以并行执行一定的操作。进程，称为查询协调器，将一条语句的执行调配到多个服务器执行，协同所有服务器的结果，并将结果返回给用户。

虽然这个特性通常称为PQO（并行查询选项），它也包括如下内容：

- 并行加载。

- 并行恢复。

- 并行复制。

- 并行SQL。

## 38.2 并行加载

SQL\*Loader直接路径使你可以使用多个SQL\*Loader会话将数据同步加载到相同的表或分区中，如下例所示：

```
sqlload saledba/saledba control=sales1.ctl parallel=TRUE direct=TRUE  
sqlload saledba/saledba control=sales2.ctl parallel=TRUE direct=TRUE  
sqlload saledba/saledba control=sales3.ctl parallel=TRUE direct=TRUE
```

提示 在SQL\*Loader命令行使用关键字parallel与direct以执行并行数据加载操作。

注意命令行中关键字parallel与direct的使用，还要注意每个会话中三种不同的输入数据文件的使用。并行加载的重要特性是：

每个SQL\*Loader会话分派了一个新的区间并将数据载入新区间中，为优化系统的 I/O 性能，强烈建议在 OPTIONS子句中使用FILE与STORAGE关键字控制新区间的位置与大小。FILE关键字可以在命令行或控制文件中定义，然而，存储子句只能在控制文件中定义。

并行加载要求没有本地或全局索引。如果存在任何索引，会产生一个错误信息并退出加载操作。在加载之前需要手工删除索引，在加载完成后，再重建索引。

每个加载会话需要一个表上的共享锁。

每个加载会话都是独立的，在加载会话之间没有任何联系。

当一个会话完成加载后，Oracle将加载的区间与已有的区间连接到一起，上次加载区间中没有使用的块作为空区间返回，这导致加载之后非标准大小的区间。即使通过加载控制文件中的存储子句选项指定了区间大小，截断也会发生。

如以前所指出的，每个加载会话向新的区间中载入数据，并不需要使用任何现存的区间，即使现存区间中不含有任何数据，因而，一个表的初始区间在并行载入时是从不使用的。

提示 为优化硬盘空间的使用，可以创建一个非常小的初始区间或不使用并行加载选项，将数据加载入初始区间中。

## 38.3 并行恢复

Oracle的基本读-写单位是数据块。每当对块进行修改时，Oracle将这些改变以重做日志的形式记录下来，如果需要时，可以使用这些日志重建这些块。如果由于介质失效或任何其他的原因而造成当前数据文件的内容丢失，那么这些文件可以从适当的备份拷贝中重建，然后进行恢复。恢复过程包括以下步骤：

- 1) 读日志文件并获得数据块的修改序列号。
- 2) 决定哪一个数据块需要改变。
- 3) 在高速缓存中读这些数据块。
- 4) 从重做日志中将相应的改变应用到这些数据块。
- 5) 将修改完成的数据块写回硬盘。

为在并行状态下进行恢复，需要设置初始化参数 RECOVERY\_PARALLELISM。也可以使用RECOVER命令的PARALLEL子句代替。在并行恢复过程中，服务器管理器或 SQLDBA会话读重做日志文件并将改变传递给并行服务器进程。然后这些进程读相应的数据文件并应用

改变。

提示 当在崩溃后重新启动一个Oracle实例时，在实例可以被用户使用之前执行崩溃恢复。如果需要以并行状态执行实例崩溃恢复，设置初始化参数 `RECOVER PARALLELISM`。

## 38.4 并行传播

复制使你可以维护多个数据库中的一个或多个数据库对象映象。Oracle服务器在这些数据库之间通过数据库链传送数据，以传播对复制对象的改变。SNP后台进程执行数据复制。然而，如果要复制的数据卷非常大，同步这个对象会花费很长时间。Oracle8使用并行复制，其中可以使用多个并行服务器进程传播事务。

## 38.5 并行SQL执行

Oracle数据库是一个由不同的进程维护的物理数据文件的集合。这样一套后台进程与共享内存段，统称为Oracle实例，使并发用户可以与数据库交互。当一个用户需要使用（`select`、`insert`、`update`或`delete`）数据库中的数据时，他需要连接到数据库。在大多数UNIX系统中，Oracle使用两任务体系结构。在这个模式中，当一个用户连接到数据库时，派生一个附加的进程，通常称为影子进程、前台等等。影子进程代表用户存取共享内存段与数据文件。

注意 在多线程服务器（MTS）环境中，当用户注册到数据库时，一个共享服务器进程用于代替影子进程。在并行查询操作过程中，这个共享服务器进程用作查询协调器。

Oracle在数据文件的块中存储数据。数据块是Oracle的基本数据输入/输出单元。当用户需要数据时，影子进程读取Oracle的内存高速缓存（共享内存段的一部分）中相应的数据块。然后在高速缓存中的块可以由用户的影子进程进行修改。修改过的数据块（脏缓存器）由DBWR进程写回硬盘。这样，一个影子进程做了用户需要做的主要工作，包括分析SQL语句、排序数据等等操作也由影子进程完成。

这在OLTP系统中工作得很好，在OLTP系统中用户维护的数据总数相对较小，然而在DDS环境中，用户通常需要处理大量数据，影子进程完成必需的工作需要花费一段时间。由于影子进程操纵数据工作时间很长，很辛苦，系统很可能有空闲的CPU与内存资源。这正是Oracle并行查询选项发挥作用的地方。通过使用这个选项，Oracle可以将一个用户的大量数据处理请求分解为多个、相对较小的工作单元，这些工作单元可以由不同的进程并发执行。它使用一套专有后台进程，称为并行查询伺服（服务器），完成这项工作。影子进程被提升为管理的角色，并称为查询协调器，查询协调器的职责如下：

- 1) 将功能分解为并行的小片。
- 2) 确保可以得到足够数量的并行查询伺服器。
- 3) 为给定的工作初始化伺服器进程，并在PQO服务器的进程之间分派工作。
- 4) 从伺服器进程收集输出并将输出返回。
- 5) 当完成预期的工作后，查询伺服器进程被释放，并可由其他的工作获得。

在最有利的情况下，并行执行减少了执行时间，执行时间的因子是使用的查询伺服器的个数。然而，由SQL语句消耗的总时间并没有减少。并行执行比顺序执行使用更多的内存，

所以如果机器没有足够的CPU与内存资源，不会得到预期的伸缩性。

### 38.6 可以并行的SQL操作

当一条SQL语句被执行时，它首先被分析，然后才被执行。在分析完成后与执行开始之前，优化器建立了执行计划。查询协调器进程检查执行计划中的操作以决定单独的操作是否可以被并行化。

在版本8之前，Oracle只能并行具有查询或DDL操作的SQL语句，如create index、create table、as select等等，它的执行计划包括一个现存表的全表扫描。在版本8中，Oracle为插入、更新与删除操作引入了并行执行。

Oracle8服务器概念手册指出Oracle可以并行执行以下的操作：

表扫描。

嵌套循环连结。

排序合并连结。

哈希连结。

“Not in”。

Group by。

Select distinct。

Union与union all。

集合。

从SQL调用的PL/SQL函数。

Order by。

Create table as select。

Create index。

Rebuild index。

Move partition。

Split partition。

Update。

Delete。

Insert...select。

Enable约束（表扫描是并行化的）。

星形变换。

除这些操作之外，Oracle8i还能并行执行cube与rollup操作。

作为一条规则，可以说Oracle可以并行执行任何含有全表（分区）扫描的操作。考虑如下的SQL语句：

```
select cust_id, sales_amt from sales_97 order by sales_amt;
```

执行计划由一个sales\_97表的扫描组成，假设这些列上没有索引，随后是在sales\_amt列上的排序。如图38-1所示，扫描与排序操作被分开并由多个进程执行。扫描与排序都由不同的进程并行完成，因而，PQO不只使一个操作能被多个进程完成，而且使在相同SQL语句中的多个操作可以并行执行而不是顺序执行。这是可能的，原因在于这些操作的创建者与消费者

的本质。在这个例子中，表扫描操作取得的行给予排序操作，不必等待扫描结束。

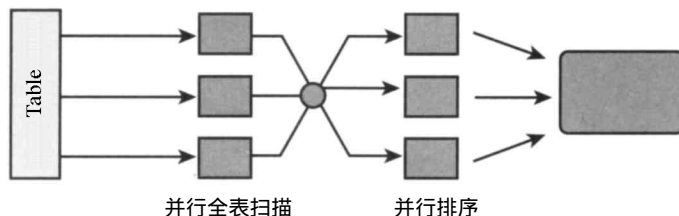


图38-1 并行执行包括一个全表扫描及排序

注意 将一个SQL操作在多个服务器进程之间分隔通常称为内部操作并行，而同步执行多个操作（从相同的SQL语句）称为交互操作并行。

在前例中，扫描与排序操作都有他们自己的服务器进程。因而，交互操作并行为每个操作分配了不同套的服务器。Oracle可以在并行状态下同时执行两个操作。如果SQL语句有三个或更多的可以并行的操作，第三个操作处于等待状态直至第一个操作完成。第一个操作使用的查询伺服器然后被循环由第三个操作使用。

## 38.7 理解并行度

一个SQL操作可以使用的并行查询伺服器的数量称为并行度。Oracle决定并行度依赖于不同的用户输入、实例初始化参数、表所在的数据文件的数量、系统CPU的数量等等。

### 38.7.1 决定并行度

一条给定SQL语句并行度的数量由以下方面按给定的顺序决定：

1) SQL语句级——SQL语句可以通过使用并行提示定义并行度，如下所示：

```
select /*+ PARALLEL (sales_97, 3) */
cust_id, sales_amt from sales_97 order by sales_amt;
```

在这个例子中，Oracle使用三个查询伺服器执行全表扫描操作，排序操作需要另外三个伺服器。因而，SQL语句总共使用七个进程——六个并行查询伺服器，一个影子进程。

2) 对象级——可以通过create或alter命令设置与表、簇、分区等相关的并行度。可以从数据字典的USER\_TABLES视图的degree列查询现有对象的并行度，如下所示：

```
Alter table sales_97 parallel(degree,3);
select degree from user_tables where table_name='SALES_97';
```

第一条语句将sales\_97表的并行度设为3，第二条SQL语句查询sales\_97表的并行度。如果在SQL操作中包括多个表，每个表都有一个不同的并行度，最大的并行度成为这条SQL操作的并行度。在下例中，SQL操作使用并行度5进行操作。

```
alter table sales_97 parallel (degree 3);
alter table warehouse parallel (degree 5);
select /*+ parallel(s) parallel(w) */
w.location, s.sales_date, s.sales_value
from warehouse w, sales_97 s
where c.w_id = s.w_id
order by s.sales_value, w.location ;
```

3) 实例级——如果Oracle没有在前两步获得并行度（即，如果用户没有定义并行度），使用缺省的并行度。

决定缺省并行度的过程与 7.3 版本不同。在 7.3 版本及更高版本中，Oracle 使用可以获得的 CPU 与硬盘数量（文件，如果不能决定硬盘的数量）决定缺省的并行度，如下所示：

```
Default degree of parallelism = Min (#CPU in the system, #disk drives the tables  
is spread on)
```

如果使用 RAID 硬盘与 OS 条带化，那么一个数据文件会跨接多个硬盘条带化，Oracle 无法知道硬盘底层的硬盘条带，并假设文件是在一个硬盘上。

在 7.1 版与 7.2 版中，Oracle 使用缺省的最小并行度，如下所示：

```
Table size in number of blocks / PARALLEL_DEFAULT_SCANSIZE  
PARALLEL_DEFAULT_MAX_SCANS (init.ora parameter)
```

应该使用的并行度依赖于机器资源（包括 CPU、内存及系统的 I/O 带宽），数据如何散布、SQL 语句并发执行的数量及其他系统上的负载。在决定并行度时需要进行充分考虑。具有给定并行度的足够的系统资源非常重要，否则会产生问题，如过多的页面调度、I/O 瓶颈等等，那样反而会达不到目标。

### 38.7.2 当没有足够的查询伺服器时

当决定了一条 SQL 操作的并行度时，查询协调器试图征募可用（已创建但处于空闲状态）的服务器。如果可用的服务器不够用，查询协调器再创建服务器。如果由于 MAX\_PARALLEL\_SERVERS 的限制不能创建所需数量的查询伺服器，它为并行执行创建尽可能多的伺服器，并使用可用的伺服器。

注意 在 7.3 以前的版本中，Oracle 用尽可能多的查询伺服器执行并行操作。如果得不到查询伺服器，查询由影子进程自己执行，即顺序执行。

在 7.3 版中，可以使用 init.ora 参数 PARALLEL\_MIN\_PERCENT 指定执行查询所需的伺服器的最小数量。这个参数的缺省值是 0，它模拟 7.3 以前的版本行为。可以为这个参数指定 0 到 100 之间的任何数。当这个参数具有非零值时，Oracle 以如下方式决定一个操作所需的查询伺服器的最小数量：

```
Minimum slaves required = Parallel_min_percent * Degree of parallelism / 100
```

如果 Oracle 得不到上面的公式所决定的数量的查询伺服器，它向用户发出一个错误信号，退出 SQL 操作。

## 38.8 理解查询服务器进程

当一个 Oracle 实例启动时，也启动初始化参数 PARALLEL\_MIN\_SERVERS 确定的数量的并行查询服务器。如果在一个系统上运行多个 Oracle 实例，每个实例都有它自己的一套独立的并行查询伺服器。与此相似，在 Oracle 并行服务器环境下，每个实例有它自己的一套并行查询伺服器。如果在任何时间实例需要更多的查询服务器，就创建它们。然而，任何时间创建的并行查询伺服器的最大数量都不能超过 PARALLEL\_MAX\_SERVERS 初始化参数。空闲服务器在空闲了 PARALLEL\_SERVER\_IDLE\_TIME 所规定的时间（以分为单位）后被关闭。然而，任何时间服务器的最小数量不能低于 PARALLEL\_MIN\_SERVERS。

在Oracle并行服务器环境下，一个SQL操作可以在多个实例中执行。在一条并行SQL执行中可以参与的实例的最大数量，由实例启动时的PARALLEL\_MAX\_INSTANCE init.ora参数决定。查询协调器也驻留在SQL语句开始时的实例上。

## 38.9 分析对象以更新统计

在并行执行过程中Oracle使用基于成本的优化。如果在数据字典中找到与表相关的统计，就使用它们。如果没有统计，就立即收集它们。如果统计的对象已过时，Oracle也许会使用错误的执行计划。需要经常分析对象以便保持最新的数据字典统计。而且，在表上进行任何主要的数据维护操作之后，都需要重新分析这个表。

## 38.10 理解9,3,1算法

Oracle对并行执行使用不同的算法。在版本8之前，它主要基于全表扫描进行并行操作。并行一个表的扫描的最简单的模式包括如下步骤：

- 1) 如前面一节中所描述的那样决定并行度。
- 2) 找到表的最高水印，因而找出表所拥有的数据块的总数。
- 3) 将要扫描的块分为与总的块数除以可用的查询服务器数相等的部分，现在已准备好为每个查询服务器分配等量的工作了。然而，如果数据不能被平分或对部分数据的存取速度很慢，一些服务器会先于另一些服务器完成工作。为指出这个问题并优化性能，Oracle使用9,3,1算法。

4) Oracle进一步将每项工作分区分成三大块，第一部分是整个的9/13，第二部分与第三部分是整个的3/13与1/13。因而整个表被分成并行度为\*3的分区。

5) 查询协调器现在将所有的9/13部分分给每个查询服务器，当查询服务器完成它们的早期分配工作后，接着分配3/13与1/13部分。这样确保几乎相等地使用所有的服务器。

请注意9,3,1是Oracle用于并行执行的一个基本算法。Oracle并行执行还考虑了其他的因素，如下所示：

- 1) 在Oracle并行服务器环境中，Oracle工作划分算法还考虑了硬盘相似性等。
- 2) 当并行分区表与索引时，Oracle由分区进行并行化。

## 38.11 理解并行DML

如前所述，Oracle也可以并行插入、更新与删除操作。为并行执行插入、更新与删除操作，需要执行以下命令：

```
alter session enable parallel dml ;
```

如果不执行这个命令，DML操作顺序执行，即使SQL语句含有清楚的提示或并行子句使其并行化，也要如此。

更新与删除操作只能跨越分区被并行化。然而，插入操作可以在分区内部被并行化。对更新与删除操作来说，最大的并行度还能超过分区的数量。下面的SQL语句从sales\_97表中删除所有小于\$100的销售事务，它有三个分区：

```
delete /*+ parallel (s,3) */  
from sales_97 s where sales_value < 100 ;
```

下面是与并行查询和并行 DML 操作有关的提示：

**PARALLEL** 可以在 SQL 操作中使用 PARALLEL 提示指明并行度。Oracle7 使你可以使用 PARALLEL 提示控制全表扫描操作的并行度。在 Oracle8 中，可以使用提示控制除表扫描操作之外的 INSERT、UPDATE、DELETE 操作的并行度。PARALLEL 提示的语法为：

**PARALLEL** (<table\_name>, m,n)

m 等于用于 SQL 操作的并行服务器的数目，n 等于执行 SQL 操作的实例数目。仅在 OPS 环境下有用。

**NOPARALLEL** 如果想要忽略与一个对象相关的并行度，并想让 SQL 操作以顺序方式执行，使用 NOPARALLEL 提示。

**APPEND** 可以使用 APPEND 提示在 INSERT 操作过程中控制并行执行。这是并行插入操作的缺省模式，它对插入的数据使用新的空闲块，不使用块中现有的空闲空间。

**NOAPPEND** 这个提示使你可以在为插入的数据分配新块之前，使用块中现有的空闲空间。

**PARALLEL\_INDEX** Oracle8 允许对分区索引进行索引范围内扫描的并行执行。可以使用 PARALLEL\_INDEX 提示实现这个效果，它的语法与 PARALLEL 提示相似。

## 38.12 OPS 环境下的并行执行

Oracle 允许多个 Oracle 并行服务器（OPS）实例参与一条 SQL 操作的并行执行。如果有四个 Oracle 实例，而你执行并行操作的并行度为 2，每个实例使用两个并行查询伺服器。

可以使用以下方式控制参与的实例数。

Parallel 提示。

Parallel 实例组。

INSTANCE\_GROUPS 允许将实例按预期的方式分组。PARALLEL\_INSTANCE\_GROUP 参数指定了执行实例并行操作的缺省组。PARALLEL\_INSTANCE\_GROUP 是一个动态参数，可以在会话级或系统级使用 ALTER SESSION/SYSTEM... 命令改变 PARALLEL\_INSTANCE\_GROUP 参数的值。当一个并行操作开始时，在执行中包括所有属于 PARALLEL\_INSTANCE\_GROUP 参数定义的 INSTANCE\_GROUP 的实例。

考虑一个使用如下初始化参数的四实例 OPS 配置：

```
Instance 1 -> INSTANCE_GROUPS = g_12, g_13, g_14, g_123  
PARALLEL_INSTANCE_GROUP = g_12
```

```
Instance 2 -> INSTANCE_GROUPS = g_12, g_23, g_24, g_123  
PARALLEL_INSTANCE_GROUP = g_123
```

```
Instance 3 -> INSTANCE_GROUPS = g_13, g_23, g_34  
PARALLEL_INSTANCE_GROUP = g_34
```

```
Instance 4 -> INSTANCE_GROUPS = g_14, g_24, g_34  
PARALLEL_INSTANCE_GROUP = g_12
```

当一个用户注册到第一个实例并执行一个并行 SQL 操作时（没有执行 alter session set parallel\_instance\_group 命令），这个操作由实例 1 与实例 2 的并行服务器进程执行。然而如果用户在执行 SQL 语句之前，执行了

```
alter session set parallel_instance_group g_123 ;
```

操作由实例 1, 2, 3 执行。

### 38.13 Oracle 8i 中的改变

Oracle在Oracle8i中的并行查询选项方面做了一些改变，但是这些改变不会影响并行执行的技术。多数改变用以自动确定并行度及调整进程。Oracle这么做的目的是使这个特性尽可能地对用户透明。将数据库管理员从调整并行操作中解放出来，并让 Oracle在考虑到查询开始时可用的系统资源的情况下迅速完成这项工作。

提示 当你希望 Oracle决定并行执行的行为时，设置 PARALLEL\_AUTOMATIC\_TUNING=TRUE，同时设置 PARALLEL\_ADAPTIVE\_MULTI\_USER=TRUE，以便 Oracle根据空闲的系统资源决定并行操作开始时的并行度。

### 38.14 调整并行查询

并行查询操作是一个强有力的工具，如果有效使用，可以减少一些命令的处理时间。然而，它很消耗资源，并且所获得的性能提升高度依赖于数据分布。需要有足够的 CPU与内存资源以支持活动的并行查询伺服器。而且，适当地计划数据以使在并行处理过程中不会有 I/O 瓶颈。

### 38.15 疑难解答

#### 1. Oracle怎样决定并行操作的并行度？

通常，Oracle用如下方式决定并行度：

- 1) 首先考虑一个并行子句或 SQL语句中的提示。
- 2) 下一个标准是一个对象（表/索引）在数据字典中定义的并行度。
- 3) Oracle使用可用的处理器与对象扩展到的硬盘驱动器数量决定缺省并行度（参见 38.7.1 节以获取详细信息）。

4) 在Oracle 8i中，如果PARALLEL\_ADAPTIVE\_MULTI\_USER设为TRUE，Oracle可以根据查询开始时可用的CPU与硬盘资源减少并行度。

#### 2. 在执行表上的并行操作之前，是否必须分析这些表？

Oracle并行查询操作总是使用基于成本的优化。如果没有表统计，Oracle迅速收集统计。然而，如果有统计，Oracle就使用它们。必须确保在进行主要的 DML操作之后已重新分析了表并更新了数据字典中的统计，过时的统计会导致无效的执行计划。

#### 3. 现在的表与它过去相比，只含有很少的行，使用并行查询的操作是否仍然需要更多的（或相等的）时间？

并行查询操作总是扫描表的当前 HWM、高水位（含有数据的最高格式化块）。从表中删除行并不会降低高水位，因而，不会改进表中所包括的并行 SQL操作的性能。建议在大量删除操作之后重新建表以降低 HWM。