Oracle8 系统管理与应用

冯睿江 （西昌卫星发射中心指控站 615621）

摘要：对于大型数据库管理系统Oracle8而言，充分了解其特性，让它高效运行，并开发高质量的应用是关键。本文介绍NT版Oracle8在系统管理和应用中的一些基本原理和常用技术，包括运行环境设置、查询优化以及开发应用。

关键词：Oracle 8 数据库 系统管理 性能优化

Oracle8(NT 版) 是安全、可靠、支持海量数据存储的面向企业级的大型网络数据库管理系统。无论是对数据库管理员，或应用程序开发人员来说，不仅仅是能让Oracle8运行，还要让其高效的运行。因此，更多的了解和掌握Oracle8的内部机制、特性、基本原理和常用技术很有必要。

1 运行环境设置

1.1 设置数据库服务名

在Oracle8中，各种前端工具与后台数据库的连接使用Net8（早期版本称为SQL*Net）服务名，登录时要求输入的“Service”即服务名。Net8实际上也是Oracle自身使用的的一种网络协议。服务名也称为别名，封装了远程数据库的相关信息，如主机名，网络协议，数据库名等,是一个用来标示和连接Oracle 数据库的逻辑名。如果要管理多个数据库，服务名可以确定当前所操作的具体数据库。新建一个数据库后，应当为其设置服务名。设置服务名的工具是Oracle Net8 Easy Config。比如服务名misdb设置如下：

Service Name(数据库服务名)：misdb

Protocal(网络协议)：TCP/IP

Host Name(主机名)：netserver.zkz.net

Database SID(数据库系统标示名)：eims

Post Number(端口号)：1521

1.2 控制数字和日期格式

Oracle8是国际化的产品，支持多种语言，在显示数字、日期格式时不一定符合中文习惯的格式。

Oracle8 输出数字时，缺省情况下如果该值绝对值小于1，则个位数0会被省去，如0.15以.15输出。可用函数to_Char(number, format)完整输出。如：

select to_Char(反修率，‘9.99’) from Device；

此外，即使在安装时选择语言为中文简体，但Oracle8缺省的日期格式还是英式的，即形如“23-5月-99”，修改NLS（国际语言支持）参数NLS_DATE_FORMAT可将日期格式设置为希望的格式，如设置NLS_DATE_FORMAT= YYYY-MM-DD，则允许用户将2001年5月10日以2001-05-10的格式输入。函数to_Char(date, format)则以指定的格式输出日期，如：

select to_Char(生产日期，‘YYYY/MM/DD’) from Device；

要修改默认的日期格式，先在注册表中“HKEY_LOCAL_MACHINE|SOFTWARE|ORAC_LE”下新建字串值：NLS_DATE_FORMAT，数值数据设置为：YYYY-MM-DD，并在初始化文件INIT.ORA中增加“NLS_DATE_FORMAT = YYYY-MM-DD”。

以上设置需要重新启动数据库。

1.3 文件系统设置

如果允许用户操作文件系统，PL/SQL 提供UTL_FILE包进行文件I/O操作。但作为数据库服务器，对文件I/O操作需要严格的安全机制，Oracle通过UTL_FILE_DIR参数限制可访问的目录，形式为：

UTL_FILE_DIR = directory_name

每个可访问的目录需要单独标示。如：

UTL_FILE_DIR = C:\Windows

UTL_FILE_DIR = D:\Source\Output

如果：

UTL_FILE_DIR = *

那么数据库权限就会被屏蔽，即所有目录对UTL_FILE包都是可访问的。

缺省的，Oracle 没有设置任何的UTL_FILE_DIR 值，即所有目录对UTL_FILE 包都是不可访问的。要设置默认的可访问目录，在初始化文件INIT.ORA 中增加以上设置。

以上设置需要重新启动数据库。

2 查询优化

2.1 优化规则

一条SQL语句可有多种执行方式，效率大相径庭。Oracle对各种数据访问方式确定了一个优化规则，级别数越小，优先级越高。Oracel优化器负责分解DML语言(数据管理语言，如Select、Insert、Update或Delete)并根据优化规则选择最有效的方式执行。

优化规则见表1：

表1 优化规则

级别	访问类型
1	通过RowID访问单行数据
2	通过聚簇连接访问单行数据
3	通过使用唯一或主码的聚簇码访问单行数据
4	通过使用唯一或主码访问单行数据
5	聚簇连接
6	散列聚簇码
7	索引聚簇码
8	组合索引
9	单列索引
10	在索引列上的范围查询
11	在索引列上的无范围查询
12	排序合并连接
13	索引列上的MAX()或MIN()
14	索引列上的ORDER BY
15	全表扫描

可见，访问数据库表时，聚簇和索引普遍具有较高的优先级，而对全表的扫描查询应尽量避免。

2.2 设计有效的数据库表

关系数据库的规范化设计就是按照范式理论对数据库表进行逻辑分解，降低数据冗余，提高数据间的一致性。但过多的表间关联也会增加系统的同步维护和开发难度，并且在查询时要对多个表进行连接，一旦这些表中存在一个巨大的表，那么整个连接查询将会非常低效。因此，应当根据实际需求进行规范化设计，甚至对某些应用要考虑适当的冗余。

对需要经常进行统计的常用字段如总和、平均值、最大值等可以加入到数据库表中，并创建触发器，由数据库自动维护。

例如工资表包括职资、基础、生活补贴、职帖、…等字段，计算所有字段的总和得出应发工资，如果工资表有1000行记录，每条记录的应发工资要对10个字段求和，则输出工资报表时，总共要进行10000次计算，才能得出每条记录的应发工资，并且下次输出时又要进行同样的操作。如果在工资表中增加“应发工资”字段，并且使用触发器自动求和，不仅省去了求和程序的编写，应发工资的输出直接通过查询即可，实际的应用效率将大大提高。

对于数据量很大且相对静态的表，可重建一个和原表一致的新表，根据需要将频繁使用的数据提取到新表中，虽然增加了冗余，但对新表的操作能提高系统效率。

或者，对大表进行分区，将数据适当分类，并分区存放。分区表有以下好处：

(1) 可以提高对表的查询性能，因为可以只查询表的一个分区而不是全部表。

(2) 表更易于管理，因为数据被存放在多个相对较小的表中，比大表更容易装载和删除数据；备份和恢复也有更多选择。

2.3 合理使用索引

如果把整个数据库表看作一本书，则索引就是这本书的目录。它可以有效的避免对全表的扫描，只在最小的范围内搜索目标记录，提高查询的效率。索引的使用应遵循几个基本的原则：

(1) 要限制一个表上的索引数量，一般不要超过10个，尽量限制在5个以内。索引会占用数据库空间，索引越多，Insert、Update和Delete语句的执行就越慢，因为要同时修改索引。在只有少数枚举值的字段上(比如“性别”只取“男”、“女”两个值)就不用建立索引，否则会降低数据库的更新速度。

(2) 为要经常查询、排序(如使用Group或Order排序)的字段创建索引，或字段的组合创建组合索引。

(3) 建立组合索引时，应把最经常用到的字段作为前导列(即第一列)，并把要引用到字段都包含到组合索引中。只有当查询条件中出现组合索引的前导列，才会引用该组合索引；而当查询条件引用到字段都包含在组合索引中时，查询性能达到最优。

例如对人员表的“籍贯”、“通信地址”和“邮政编码”建立组合索引，“籍贯”为前导列。

查询语句：

```
Select * from Employee
  Where 邮政编码 = '615621';
```

不会引用该组合索引，不会提高查询性能，因为前导列“籍贯”没有在查询条件中出现；

查询语句：

```
Select * from Employee
  Where (籍贯 = '四川') and(邮政编码 = '615621');
```

的查询性能显著提高，因为该组合索引对查询条件形成索引覆盖。

(4) 为要经常连接进行子查询但没有创建为外部关键字的字段上建立索引。比如在一个嵌套3层的子查询中，如果每一个表有1000条记录，采用顺序查询，一共要查询1000*1000*1000=10亿条记录，这对查询效率的影响是致命的，如果在3个表中的连接字段上都建立索引，将避免嵌套查询时对每个表的全表扫描，查询效率会得到根本性的提高。

(5) 为要经常进行连接的多个表建立索引聚簇。索引聚簇可在一个Oracle块中同时存储两个或多个表中的数据，其聚簇码常用来连接两个或多个表的连接列。如果需要使用多个表中的数据，那么只需访问一个Oracle块，不需先访问一个表，再访问其他表。

3 开发应用

3.1 程序包分解

Oracle 的 PL/SQL 语言类似 Ada 语言，通过程序包来构建应用程序。但 PL/SQL 对程序包的大小有强制性的限制。对此，Oracle 的解释是：PL/SQL 主要为健壮的事务处理而设计，因此 PL/SQL 编译器强制性的限制 PL/SQL 块的大小，这种大小限制依赖于程序的复合程度。根据作者的经验，如果有一定复杂度的单个程序包程序超过 1500 行左右时，使用 SQL*PLUS 或 SQL Worksheet 都可能无法对其进行编译。这样迫使开发人员要对应用程序进行模块化分解，之后，绝大多数情况都能满足 Oracle 的要求。

3.2 使用触发器

触发器是一些过程，当一个特定的数据库事件发生在指定表上时就执行这些过程。使用触发器可以增加附加的安全设置；以及让Oracle 自动的完成某些数据的统计和审计。用户不必记着去执行它们，也可以省去部分程序的编写。触发器可对 Delete、Insert或Update语句定义Before、After或 Instead Of 事件。

例如对前文工资表中“应发工资”字段可创建触发器如下(为简化见，只计算前四项)：

```
Create or Replace Trigger Sum_Salary
Before Insert Or Update Of 职资，基础，生活补贴，职帖
On Salary
For Each Row
Begin
:new.应发工资:= :new.职资+:new.基础 +:new.生活补贴 +:new.职帖；
End;
```

3.3 使用 truncate 删除全表记录

使用 Delete 命令删除一个表的全部记录后，虽然数据段已不存在任何记录，但却依然保留它所分配的物理空间，因为这些数据要被存储在系统回滚段中，以备以后恢复时用。此外如果这个表很大，而回滚段又太小，可能无法提交 Delete 命令。使用 Truncate 命令可以解决以上问题。它既可以快速的删除大量数据，还释放所占用的物理空间。比如：

```
Truncate Table Salary;
```

如果一个表创建了分区，Truncate 命令可以删除该表的指定分区，如：

```
Alter Table Salary
Truncate Partition Part2;
```

但要注意，Truncate 命令不可逆。

3.4 数据迁移

早期单机版数据库应用系统大多采用 FoxPro、dBase、Paradox 等桌面型数据库系统开发，如果为了满足大型网络应用，要把旧系统升级Oracle8中，就可以使用NT 版Oracle8提供的 Oracle Migration Assistant for MS Access 工具。它以向导方式自动将 Access97数据库迁移到Oracle8中，而 Access97 可直接导入FoxPro、dBase、Paradox 等数据库表。所以可先将目标数据库表导入到Access97中，再从 Access97 迁移到 Oracle8 中。

一般的，迁移后的数据库表可直接使用，但客观的说，数据库表在迁移后的结构上，比如表间关联性、索引、关键字段等或多或少会有损失，另外，为了充分利用Oracle8的特性，最好是在 Oracle8 中根据原来数据库的结构重新创建数据库表，然后采用迁移过来的数据，这样数据库的结构或者数据都不会有损失。■