Oracle9i/10g/11g编程艺术

第一章：

OSD：Operating System Dependent，操作系统依赖。

总结：

1、理解Oracle体系结构

2、理解锁定和并发控制特性，不同数据库的实现方式不同

第二章：体系结构概述

数据库和实例：数据库是物理性的文件，是磁盘上存储的文件集合；实例是软件上的，它是一组进程和共享内存，一个实例只能打开、装载一个数据库。（参看连接概述图）

专有连接：客户端直接和专有服务器连接，每个连接一个专有服务器；

共享连接：客户端和调度程序连接，调度程序通过队列来管理连接和响应。

第三章：文件

1、与实例相关的文件

A、参数文件(parameter file)：如何找到控制文件，配置一些初始化参数。

B、跟踪文件(trace file)：服务器进程对某种错误异常条件做出响应时创建的诊断文件。

C、警告文件(alert file)：和跟踪文件类似，但是包含了“期望”事件的相关信息。

2、数据库相关文件

A、数据文件(data file)：数据库的主要文件，包含数据表、索引和其他的段(segment)。

B、临时文件(temp file)：用于完成基于磁盘的排序和存储临时表。

C、控制文件(control file)：指明其他文件的存储位置，并包含与文件状态相关的元数据。

D、重做日志文件(redo log file)：事务日志

E、密码文件(password file)：用户认证

从10g开始，提供了两种可选的文件：

1. 修改跟踪文件(change traking file)：便于建立增量备份。
2. 闪回日志文件(flashback log file)：

另外还有一些文件：

1. 转储文件(dump file)
2. 数据泵文件(data pump file)
3. 平面文件(flat file)

其中最重要的是数据文件和重做日志文件。

数据库参数文件：

Init.ora，这是个遗留文件，放在了客户机上，当有多个客户机时管理不便，于是推出了服务器参数文件(server parameter file，SPFILE)，它只能存放在服务器端。此时，只能使用 alter system命令来修改该文件。

数据文件：

表空间（tablespace）是Oracle中的一个逻辑存储容器，位于存储层次体系的顶层，包含一个或多个数据文件。表空间包含段（segment）。

段：占用存储空间的数据库对象，如表、索引、回滚段等。占用存储空间的每一个对象最终都会存储在一个段中。

区段（extent）：段本身由一个或多个区段组成。区段是文件中一个逻辑上连续分配的空间。

块（block）：区段由块组成，块是Oracle中的最小的空间分配单位。

重做日志文件：

在线重做日志文件：分为多个组，每个组中含有N个重做日志成员，这些成员互为镜像。当一个组的成员写完后即会切换到另一个组。

归档重做日志文件：历史

外部表：

利用外部表，可以像读取数据表一样读取平面文件。注意外部表是只读的。

第四章：内存结构

1、进程全局区（Process Global Area，PGA）：特定于进程的一段内存，和其他进程互斥。绝对不会在SGA中分配。它可能包含UGA，此外的其他区用于完成内存中的排序、位图合并和散列。

2、用户全局区（User Global Area，UGA）：即会话状态。如果使用共享连接，则UGA在SGA中分配，如果使用专有连接，则在PGA中分配。

3、系统全局区（System Global Area，SGA）：是一个庞大的共享内存结构。它分为几个池：

A、Java池，为数据库中运行的JVM分配的一段固定大小的内存。

B、大池（Large Pool），共享服务器连接使用大池作为会话内存，并行执行特性使用大池作为消息缓冲池，RMAN可能使用大池作为I/O缓冲池。

C、共享池（Shared Pool），包含共享游标，存储过程，状态对象，字典缓存和诸如此类的大量其他数据。

D、流池（Stream Pool），

E、空池（Null Pool），一块缓冲区（缓存数据库块），重做日志缓冲区，“固定SGA区”。

固定SGA：自描述，用于查找SGA的其他区。

块缓冲区缓存（bolck buffer cache）

默认池，保持池，回收池，跟访问频率有关。

共享池，缓存“程序”数据，比如PL/SQL代码，系统参数，数据字典缓存。根据LRU算法管理。

大池，并不是因为它的体积大，而是因为它管理的内存块大。它是一个回收型的内存空间。

Oracle 10g之后自动SGA内存管理，通过V$SGAINFO视图来查看哪些参数可调节。到11g后，可设置自动内存管理，memory\_target参数表示SGA和PGA能分配的总内存量。

第五章：Oracle进程

1、服务器进程，根据客户请求完成工作，如专用服务器连接和共享服务器连接。

2、后台进程，完成各种维护工作，如将块写至磁盘，维护在线重做日志等。

3、从属进程，类似于Java的守护线程。

连接和会话。一个连接指的是一条物理链路，它上面可以包含多个会话，会话是一个逻辑实体，同一个连接上的各个会话是相互独立的。可以通过V$Session视图查看会话信息。

专用服务器和共享服务器：

专用服务器，客户端和服务器一对一映射，各连接之间相互独立，因此可以执行长时间的事务。共享服务器则相反，由于各个连接相互共享，所执行的事务必须短，否则就可能会造成死锁。因此，共享服务器只适合于OLTP系统（事务短并且频繁）。

共享服务器的好处：

1. 减少操作系统进程/线程数，举例：共有5000个并发用户，但是在任一时刻，最多处理50个，这样就大大减少了线程上下文切换的负担。
2. 刻意地限制并发度，系统支持的并发度是有峰值的，超过峰值，吞吐量不升反降。因此有必要限制并发。
3. 减少内存使用。

我们的J2EE应用一般应当使用专有服务器，因为使用了数据库连接池。

后台进程：

1. 进程监视器（Process Monitor，PMON）：负责在出现异常中止的连接后完成清理。
2. 系统监视器（System Monitor，SMON）：数据库“垃圾收集器”。
3. 检查点进程（Checkpoint Process，CKPT）：更新数据文件的文件首部，以辅助真正建立检查点的进程（DBWn）。
4. 数据库写入器（Database Block Writer，DBWn）：将脏块写入磁盘。
5. 日志写入器（Log Writer LGWR）：将SGA重做日志缓冲区的内容刷新到磁盘。

从属进程：

I/O从属进程：为不支持异步I/O的系统或设备模拟异步I/O

Pnnn：并行查询执行服务器。

第六章：锁和闩

锁用于管理对共享资源的并发访问。数据库锁同时提供数据一致性和完整性。

Oracle的事务和锁

1. 延迟到适当的时刻才提交事务。
2. 只要需要，就应该尽可能长时间保持对数据所加的锁。
3. 行级锁没有相关的开销。
4. 不要随便升级锁，能使用行级锁就不要使用表锁。
5. 可以同时得到并发性和一致性。读取和写入都不会相互阻塞。

锁定问题(事务相关的四个问题)

1. 更新丢失，两个会话同时读取一个数据并修改，后提交的会覆盖掉先提交的，从而导致更新丢失。

悲观锁（pessimistic lock）和乐观锁（optimistic lock）

悲观锁，显式加锁。乐观锁，①Oracle会自动保存前映像，能自动判断②版本号③散列值。其中，版本号是Hibernate的默认实现方式。

Oracle对悲观锁的支持非常好，但是在实际的web应用中，还是应该使用乐观锁，这是因为悲观锁需要一条有状态的和数据库之间的连接，这对于web应用来说是不可承受的。

引起阻塞的DML语句

1. select for update，应该加上nowait
2. insert，这主要表现在唯一性约束上，比如插入相同的主键、唯一性约束

锁类型：

TX锁（事务锁），锁定某行；

TM锁（表锁），防止在修改数据的时候，表结构被修改。

DDL锁

闩

保持极短的一段时间。等待闩的会话不会排队，只会不断地重试。

第八章：事务

Set transaction，开始事务，不是必须；Commit，提交事务；rollback，回滚；savepoint，在事务中做一个标记点；rollback to savepoint，回滚到某一个savepoint

第九章：redo和undo

redo，存放在在线（或归档）重做日志中的内容，万一出现失败时可以利用这些数据来“重放”或者重做事物。undo是记录在undo段中的信息，用来回滚或撤销事务。

实例失败（如主机掉电）、介质失败（如磁盘损坏）、误操作（如删掉一个表），都可以通过在线重做日志（需要的话，配合归档重做日志）恢复到某个恰当的时间点。

undo是逻辑操作而不是物理操作。

Commit操作所需的时间：不要压缩事务，一次提交1000条语句比10次提交100条的效果要好。这是因为在提交操作之前，要做的工作都已经做完，比如修改数据，生成redo和undo并缓存，提交的时候要做的事情所需的时间几乎都是一个数量级的。因此我们commit的时候要以一个业务逻辑为单元。但是rollback就不一样了，它所需的时间跟需要修改的数据量成正比。

一个Oracle实例只有一个LGWR，因此作为开发人员，一定要学会控制生成的redo大小。redo越大，操作越慢。开启NOLOGGING模式要千万小心，搞不好就无法恢复备份了。

块清除（block cleaning）：oracle中，与锁定相关的信息保存在块的首部。这会导致即使在select时也会生成redo，并且会将块弄脏（要清除这些锁定信息），从而要让DBWR刷新磁盘。

临时表中尽量只进行select和delete操作。

索引生成的redo和undo信息量不容小觑。

第十章：表

1、堆组织表（普通表）：空间以堆的方式分配，可重用。

2、索引组织表（Index Organized Table）：按索引结构存储。

3、索引聚簇表（Index Cluster Table）：聚簇是指一个或多个表组成的组，这些表物理地存储在相同的块上。聚簇键以B树索引存储。

4、散列聚簇表（Hash Clustered Table）：和索引聚簇表类似，但是使用的是散列，数据即是索引。

5、有序散列聚簇表

6、嵌套表，和子表的区别在于嵌套表不是独立的表。

7、临时表

8、对象表

9、外部表

堆表、索引表、聚簇表，分别在于对数据的存储方式。堆表存放方便，但是读取的时候可能需要从不同的块上读取，索引和聚簇则将相关的数据存放在一起，读取的时候开销会小些。

第十一章：索引

1. B\*树索引，类似于二叉树，最常见的索引。B代表Balance而不是Binary。它的子类型有：索引组织表，B\*树聚簇索引，降序索引和反向键索引（将键中的字节反转）。
2. 位图索引（bitmap index）：一个索引对应多行，适用与列值范围固定且表用于只读的场景。
3. 位图联结索引
4. 基于函数的索引，就是B\*树索引或者位图索引，只不过保存的是函数值，相当于一个伪列。
5. 应用域索引。用户自己创建的索引。

位图索引：不适用于OLTP系统，如果系统中的数据会由多个并发会话频繁更新，也不适合。一个条目对应多行。如果列的取值范围相对于行数来说接近于0，那么就适合使用位图索引。如果索引所在的列是高频繁写入的，也不适合使用位图索引。

位图联结索引：索引指向的是另一个表，一般用于外键关系的表之间。

null值不会建立索引，因此应该尽量避免null值；外键最好建立索引，以避免死锁（不加索引可能引起全表扫描从而导致死锁）

第十二章：数据类型

char、nchar、varchar2、nvarchar2：定长和变长，带n的支持unicode字符集，存储长度为2000和4000字节。带n的主要是为了存储多字符集。

二进制存储：raw、blob、long raw。raw是变长的，最大可以存储2000字节，不要使用long raw，它是为了向后兼容而保留的。

number，binary float，binary double，number的精度较大（有效位数字较多），但是后两者可以存储的范围更小或更大。number的语法number(p,s)，其中，p表示总位数（精度），s表示小数位数。number和后两者之间还有一个区别，number是一种软件表示，即对number进行四则运算时，oracle使用的是自身的软件实现，而对后两者进行计算时，可以使用硬件来计算。这样的话，在计算上，number的性能不及后两者。

date、timestamp，存储精度可变的固定日期/时间，interval表示时间间隔，两个时间戳之差即为一个时间间隔。timestamp支持小数秒和时区，而date不支持。

date类型是一个7个字节长的定宽日期/时间数据类型，它总是包含7个属性，世纪、世纪中哪一年、月份、月中哪一天、小时、分钟和秒。

clob：会进行数据库和客户端之间的字符集转换。

第十三章：分区

page 538