第3章 Oracle数据库的体系结构

本章学习目标

- 掌握 Oracle 数据库的物理存储结构
- 掌握 Oracle 数据库的逻辑存储结构
- 熟练掌握 Oracle 实例的内存结构、进程结构
- 了解 Oracle 数据库的组成及实例运行过程
- 了解 Oracle 12c 多租户架构

从体系结构上来说，Oracle 数据库服务器可以分为 Oracle 数据库和 Oracle 实例两部分。本章详细介绍了 Oracle 数据库的物理存储结构和逻辑存储结构，以及 Oracle 实例的内存结构和进程结构。

3.1 Oracle数据库的基本结构

Oracle 是一个基于 B/S 模式的关系数据库管理系统(RDBMS)。通常意义上说的 Oracle 数据库是指 Oracle 数据库服务器(Oracle Database Server)，包括 Oracle 实例(Oracle Instance)和 Oracle 数据库(Oracle Database)两部分，即 Oracle Database Server＝Oracle Instance＋Oracle Database。

在启动 Oracle 数据库时，首先要在内存中获取、划分、保留各种用途的区域，运行各种用途的后台进程，即创建一个实例(Instance)；然后再由该例程装载(Mount)、打开(Open)数据库；最后由这个实例来访问和控制数据库的各种物理结构。当用户连接到数据库并使用数据库时，实际上是连接到该数据库的实例，通过实例来连接、使用数据库。因此，实例是用户和数据库之间的中间层。

Oracle 数据库指的是用户存储数据的一些物理文件，包括数据文件、控制文件、重做日志文件、参数文件；还包括密码文件、归档日志文件、备份文件、告警日志文件、跟踪文件等。这些物理文件统称为 Oracle 数据库的物理存储结构。

在实际对数据库进行管理时，操作这些物理文件显然是不方便的。Oracle 通过表空间、段、区、块等逻辑存储结构来更加灵活方便地管理和操作数据库。Oracle 数据库的逻辑存储结构是由“数据库内部”观看其组成的要素，包括表空间、段、区、块及数据库对象。

Oracle 数据库的体系结构如图 3-1 所示。下面分别介绍 Oracle 的物理存储结构、逻辑存储结构和实例。

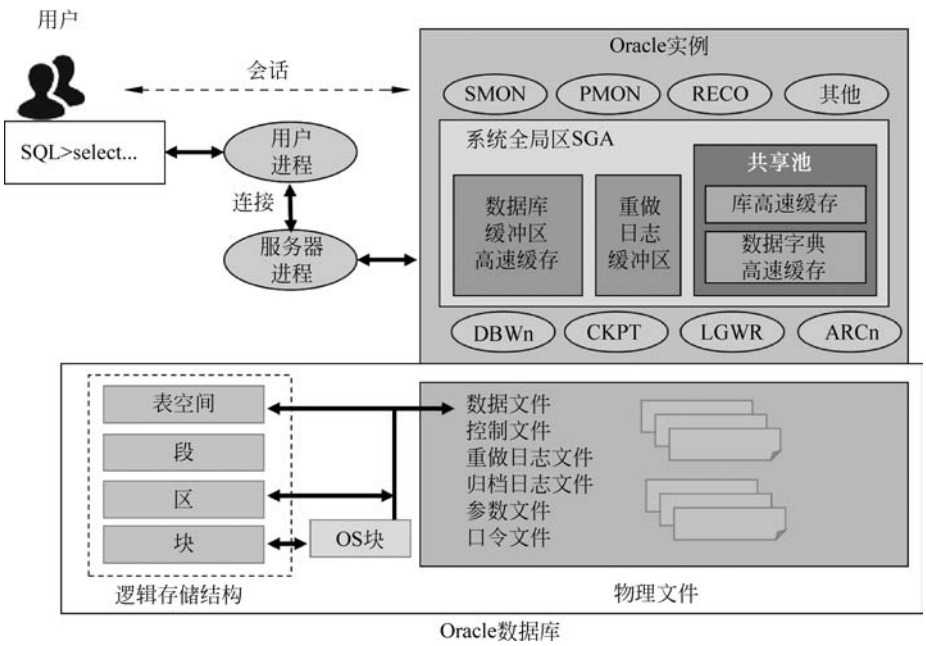

图 3-1　Oracle 数据库的体系结构

一个表空间在物理上对应若干个数据文件，而一个数据文件只能属于一个表空间。在“Oracle 安装路径\Oracleinstall\oradata\orcl”目录下可以看到自动创建的 6 个表空间都有其对应的数据文件。

3.2　Oracle 的数据库结构

3.2.1　Oracle 的物理存储结构

Oracle 的物理存储结构是由存储在磁盘中的操作系统文件所组成的，Oracle 在运行时需要使用这些文件。一般地，Oracle 数据库在物理上主要由 3 种类型的文件组成，分别是数据文件（*.dbf）、控制文件（*.ctl）和重做日志文件（*.log）。在 Oracle 的安装路径“Oracle 安装路径\Oracleinstall\oradata\orcl”的目录下，可以看到当前数据库 ORCL 的物理文件，如图 3-2 所示。

1. 数据文件（DATA FILE）

数据文件是存储数据库数据的操作系统文件，如表的实际数据、索引数据等。通常以“*.dbf”格式命名，如 userCIMS.dbf。数据文件的大小与它们所存储的数据量的大小直接相关，会自动增大。

每个 Oracle 数据库都有一个或多个数据文件，而一个数据文件只能属于一个表空间。数据文件创建后大小可改变，创建新的表空间需要创建新的数据文件。数据文件一旦加入表空间，就不能从这个表空间中移走，也不能和其他表空间发生联系。

如果数据库对象存储在多个表空间中，可以通过把它们各自的数据文件存放在不同的磁盘上来对其进行物理分割。

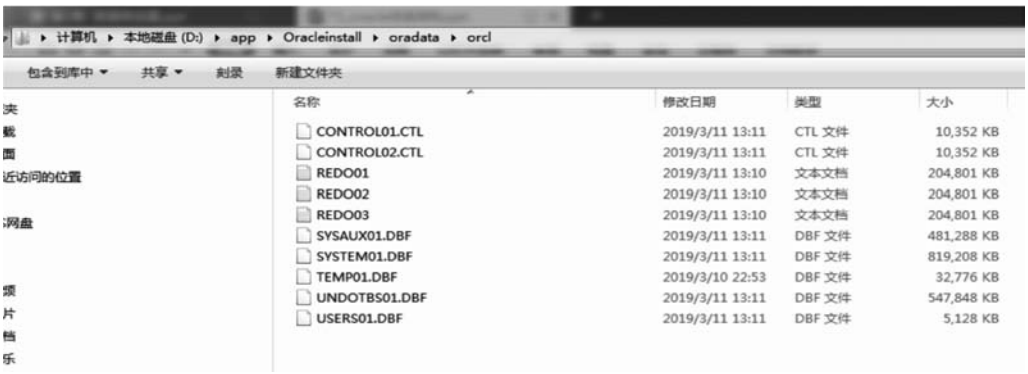

图 3-2　物理存储结构

2. 控制文件(CONTROL FILES)

每个 Oracle 数据库都有相应的控制文件,用以记录、描述数据库的结构,如数据库名、数据库的数据文件和日志文件的名字及位置等信息。控制文件为打开、存取数据库提供必要的信息,其名字后缀通常为".ctl",如 Ctrl1CIMS.ctl。为安全起见,允许控制文件被镜像。通常情况下,控制文件需要多个镜像或备份。

控制文件是一个很小的文件,大小一般在 1～5MB,为二进制文件。但它是数据库中的关键性文件,对数据库的成功启动和正常运行都是至关重要的。当 Oracle 数据库的实例启动时,它的控制文件用于标识数据库和日志文件。当着手数据库操作时它们必须被打开。当数据库的物理组成更改时,Oracle 自动更改其控制文件。当数据恢复时,也要使用控制文件。因此,只有控制文件是正常的,才能正常地装载、打开数据库。

在数据库运行的过程中,每当出现数据库检查点(checkpoint)或修改数据库结构之后,Oracle 就会修改控制文件的内容。DBA 和用户应避免人为地修改控制文件中的内容,否则会破坏控制文件。

3. 重做日志文件(REDO LOG FILES)

重做日志文件用来保存所有数据库事务的日志,名字后缀为".log",如"REDO01.log"。当数据库中的数据遭到破坏时,可以用这些日志来恢复数据库。一个数据库一般有 2～3 个重做日志文件。Oracle 以循环方式向重做日志文件写入,第 1 个日志被填满后,就向第 2 个日志文件写入,依次类推。当所有日志文件都被写满时,就又回到第 1 个日志文件,用新事务的数据对其进行重写。

Oracle 有两种类型的重做日志文件:联机重做日志文件(Online Redo Log File)和归档重做日志文件(Archive Redo Log File)。联机重做日志文件是 Oracle 用来循环记录数据库改变的操作系统文件。归档重做日志文件是为避免联机日志文件重写时丢失重复数据而对联机重做日志文件所做的备份。归档重做日志文件只有数据库运行在归档模式下时才会起作用。

重做日志文件的主要作用是保护数据库,防止数据库故障。为了防止日志文件本身的故障,Oracle 允许镜像日志(Mirrored Redo Log),即可在不同磁盘上维护两个或多个日志副本。重做日志文件中的信息仅在系统故障或介质故障恢复数据库时使用,这些故障阻止将数据库数据写入数据文件。然而,任何丢失的数据在下一次数据库打开时,Oracle 会自

动地应用重做日志文件中的信息来恢复数据库数据文件。

3.2.2 Oracle 的逻辑存储结构

逻辑存储结构是面向用户的，用户使用 Oracle 逻辑存储结构来管理 Oracle 物理存储结构。数据库逻辑存储结构从表空间到数据块形成了不同层次的粒度关系，如图 3-3 所示。一个数据库从逻辑上说是由一个或多个表空间（TABLESPACE）所组成的，表空间是数据库中物理编组的数据仓库，每个表空间是由多个段（SEGMENT）组成的，一个段是由一组区（EXTENT）所组成的，一个区是由一组连续的数据库块（DATABASE BLOCK）组成的。然而，一个数据库块对应硬盘上的一个或多个物理块（OS BLOCK），一个表空间对应一个或多个数据库的物理文件（即数据文件）。

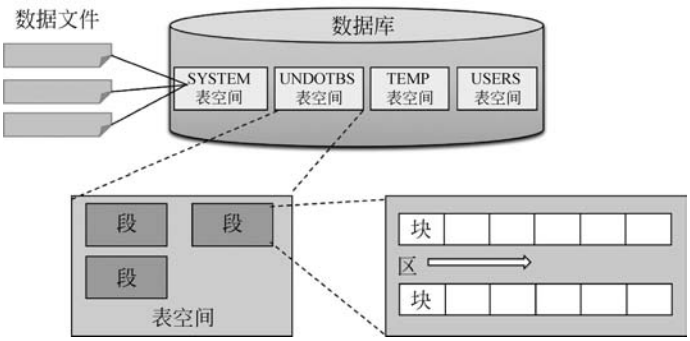

图 3-3　Oracle 逻辑存储结构

1. 表空间（TABLESPACE）

Oracle 数据库由一个或多个称为表空间的逻辑存储单元组成，表空间作为一个整体存储数据库中的所有数据，并且一个表空间只能属于一个数据库。数据库的大小是该数据库中所有表空间大小的总和。

从物理方面讲，Oracle 数据库内的每个表空间由一个或多个数据文件组成，并且一个数据文件只能属于一个表空间。表空间的大小是对应所有数据文件大小的总和。

Oracle 数据库的表空间可以分成两类，系统表空间和非系统表空间。其中，系统表空间是安装数据库时自动建立的，它包含数据库的数据字典、存储过程、包、函数和触发器等对象的定义以及系统回滚段。除此之外，还包含用户数据等信息。系统表空间通常包括 SYSTEM 表空间和 SYSAUX 表空间。

（1）SYSTEM 表空间。SYSTEM 表空间是系统表空间，用于存放 Oracle 系统内部表和数据字典的数据，如表名、列名和用户名等。一般不赞成将用户创建的表、索引等存放在 SYSTEM 表空间中。

（2）SYSAUX 表空间。SYSAUX 表空间是辅助系统表空间，主要存放 Oracle 系统内部的常用样例用户的对象，如存放 CMR 用户的表和索引等，从而减少系统表空间的负荷。SYSAUX 表空间一般不存储用户的数据，由 Oracle 系统内部自动维护。

（3）TEMP 表空间。TEMP 表空间是临时表空间，存放临时表和临时数据，用于排序和汇总等。

(4) UNDOTBS 表空间。UNDOTBS 表空间是重做表空间,存放数据库中有关重做的相关信息和数据。当用户对数据库表进行修改时(包括 INSERT、UPDATE 和 DELETE 操作),Oracle 系统自动使用重做表空间来临时存放修改前的数据。当所做的修改成功完成并提交后,系统根据需要保留修改前数据的时间长短,以及重做表空间的使用率来释放重做表空间的占用空间。

(5) USERS 表空间。USERS 表空间是用户表空间,存放永久性用户对象的数据和私有信息,因此也被称为数据表空间。每个数据库都应该有一个用户表空间,以便在创建用户时将其分配给用户。

(6) EXAMPLE 表空间。EXAMPLE 表空间是示例表空间,用于存放示例数据库的方案对象信息及其培训资料。

系统表 dba_tablespace 存储了数据库表空间的信息。使用 desc dba_tablespaces 语句可以查看系统表 dba_tablespaces 中的字段。使用 Select tablespace_name From dba_tablespaces 语句可以查看 Oracle 系统中当前的所有表空间的名字,结果如图 3-4 所示。

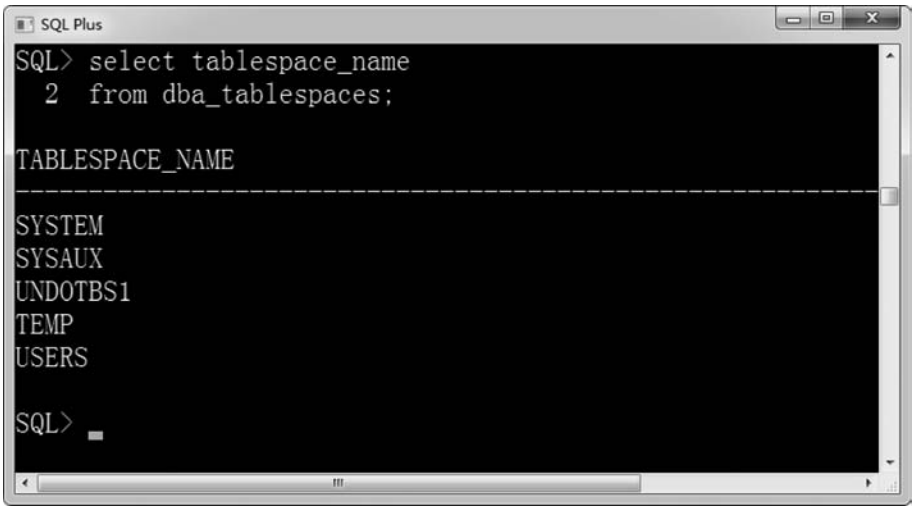

图 3-4 查看当前数据库的表空间

2. 段

根据不同的数据类型(如实际数据、索引数据),需要在数据表空间内划分出不同区域,以存放不同类型的数据,将这些区域称之为“段”(SEGMENT)。例如,存放表中实际数据的区域称为“数据区段”,存放索引的区域称为“索引区段”。

段是由多个数据区构成的,它是为特定的数据库对象分配的一系列数据区。段内包含的数据区可以不连续,并且可以跨越多个文件。使用段的目的是用来保存特定对象。Oracle 的逻辑结构划分如图 3-5 所示。

一个 Oracle 数据库有如下 4 种类型的段。

(1) 数据段:数据段也称为表段,它包含数据并且与表和簇相关。当创建一个表时,系统自动创建一个以该表的名字命名的数据段。

(2) 索引段:索引段包含了用于提高系统性能的索引。一旦建立索引,系统就自动创建

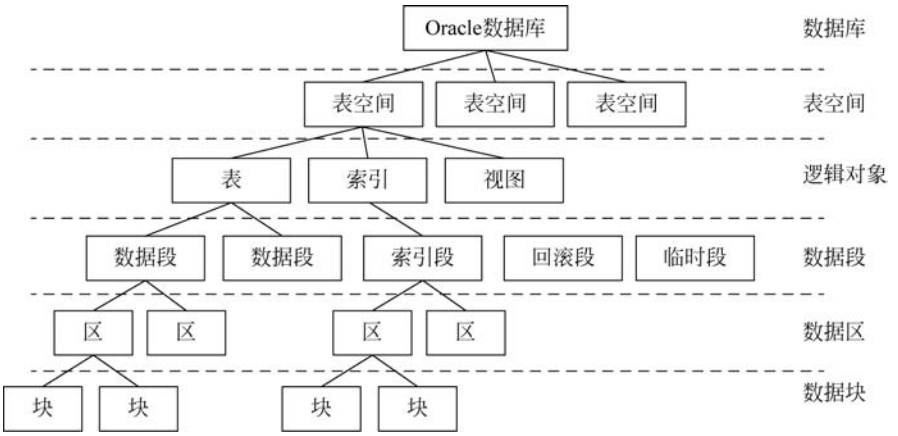

图 3-5　Oracle 数据库的逻辑结构划分

一个以该索引的名字命名的索引段。

(3) 回滚段：回滚段包含了回滚信息，并在数据库恢复期间使用，以便为数据库提供读入一致性和回滚未提交的事务，即用来回滚事务的数据空间。当一个事务开始处理时，系统为之分配回滚段，回滚段可以动态创建和撤销。系统有个默认的回滚段，其管理方式既可以是自动的，也可以是手动的。

(4) 临时段：临时段是 Oracle 在运行过程中自行创建的段。当一个 SQL 语句需要临时工作区时，由 Oracle 建立临时段。一旦语句执行完毕，临时段的区间便退回给系统。

3. 区

区是一组连续的数据块。当一个表、回滚段或临时段创建或需要附加空间时，系统可以为之分配一个新的数据区。一个数据区不能跨越多个文件，因为它包含连续的数据块。

使用区的目的是用来存储特定数据类型的数据，也是表中数据增长的基本单位。在 Oracle 数据库中，分配空间就是以数据区为单位的。一个 Oracle 对象包含至少一个数据区。设置一个表或索引的存储参数包含设置它的数据区的大小。

当在数据库中创建带有实际存储结构的方案对象(如表、索引、簇)时，Oracle 将为该方案对象分配若干个区，以便组成一个对应的段来为该方案对象提供初始的存储空间。当段中已分配的区都被写满后，Oracle 就为该段分配一个新的区，以便容纳更多的数据。

可以在 CREATE TABLE 语句的 STORAGE 子句中，通过设置 3 个存储参数来指定这个表的数据段的存储区大小、第 1 个后续区大小和后续区增加的比例。

```
STORAGE (
    INITIAL 64K
    NEXT 32K
    PCTINCREASE 50
)
```

4. 数据块

在 Oracle 中，数据块是最小的存储单元，Oracle 数据存放在块中，一个块占用一定的磁盘空间。特别需要注意的是，这里的“块”是指 Oracle 数据块，不是操作系统块。

当 Oracle 有数据请求时，系统以块为单位操作数据。也就是说，Oracle 每次请求的数据是块的整数倍。如果 Oracle 请求的数据量不到一块，Oracle 也会读取整个块。所以说，块是 Oracle 读写数据的最小单位或者最基本的单位。

块的标准大小由初始化参数 DB_BLOCK_SIZE 指定。具有标准大小的块称为标准块(Standard Block)。和标准块的大小不同的块叫非标准块(Nonstandard Block)。块的大小是操作系统块大小的整数倍，以 Windows 2000 为例，操作系统块的大小是 4KB，所以 Oracle 块的大小可以是 4KB、8KB、16KB 等。如果块的大小是 4KB，EMP 表每行的数据占 100 字节。如果某个查询语句只返回 1 行数据，那么在将数据读入数据高速缓存时，读取的数据量是 4KB 而不是 100 字节。

5. 逻辑对象

表空间、段、区、块，是 Oracle 数据库的逻辑存储结构。逻辑对象是用户组织和管理数据库中数据的重要形式。主要的逻辑对象包括表、视图、索引、约束、用户、方案、权限和角色等。

1) 表和视图

表是数据库中实际存放用户数据的对象，它包含一组固定的列。表中的列描述该表所跟踪的实体的属性，每个列都有一个名字。视图是从表或其他视图中导出的表，它是虚表，并不存放实际的数据，主要是为了方便用户而存在。

2) 索引

在关系数据库表中，一个行数据的物理位置无关紧要。为了能够找到数据，表中的每行都用一个 RowID 来标识。RowID 告诉数据库这一行的准确位置，包括所在的文件、该文件中的块和该块中的行地址。

索引是帮助用户在表中快速找到记录的数据库结构。它既可以提高数据库性能，又能保证列值的唯一性。当 CREATE TABLE 命令中存在 UNIQUE 或 PRIMARY KEY 约束条件子句时，Oracle 就会自动创建一个索引；也可以通过 CREATE INDEX 命令来创建索引。

3) 约束

可以为一个表创建约束条件。此时，表中的每行都必须满足约束条件定义所规定的条件。约束条件有以下 5 种类型：主键约束、默认值约束、检查约束、唯一性约束及外键约束。

主键约束和外键约束保证关联表的相应行持续匹配，以致它们可以用在后面的关系连接中。在它们被定义为主键约束和外键约束后，不同表的相应列会自动更新，称为引用完整性。数据库的约束性条件有助于确保数据的引用完整性。引用完整性保证数据库中的所有列引用有效且全部约束条件得到满足。

4) 用户

用户账号虽然不是数据库中的一个物理结构，但它与数据库中的对象有着重要的关系，这是因为用户拥有数据库的对象。例如，用户 SYS 拥有数据字典表，这些表中存储了数据库中其他对象的所有信息；用户 SYSTEM 拥有访问数据字典表的视图，这些视图供数据库中其他用户使用。

为数据库创建对象(如表)必须在用户账户下进行。可以对每个用户账户进行自定义，以便将一个特定的表空间作为它的默认表空间。可以把操作系统的账户和数据库账户联系

在一起，这样可以不必既输入操作系统口令，又输入数据库的口令。

5）方案

用户账户拥有的对象集称为用户的方案(SCHEMA)，其可以创建不能注册到数据库的用户账户。这样就为用户账户提供了一种方案，这种方案可以用来保存一组其他用户方案分开的数据库对象。

为了给不同的用户使用数据库对象时提供一个简单的、唯一标识数据库对象的名称，可以为数据库对象创建同义词。同义词有公有同义词和私有同义词两种。

6）权限与角色

为了访问其他账户拥有的对象，必须首先被授予访问这个对象的权限。权限可以授予某个用户或 PUBLIC，PUBLIC 可以把权限授予数据库中的全体用户。

可以创建角色即权限组来简化权限的管理；可以把一些权限授予一个角色，而这个角色又可以被授予多个用户。在应用程序中，角色可以被动态地启用或禁用。

3.3 Oracle 实例

一台计算机上可以创建多个 Oracle 数据库，当同时要使用这些数据库时，就要创建多个实例。为了不使这些实例相混淆，每个实例都要用称为 SID(System IDentifier，系统标识符)的符号来区分，即创建这些数据库时填写的数据库 SID。Oracle 实例是一种数据库访问机制，主要由内存结构和进程结构组成。

3.3.1 内存结构

内存结构是 Oracle 数据库体系结构中最为重要的一部分，内存也是影响数据库性能的第一因素。内存的大小和速度直接影响数据库的运行速度。特别是当用户数增加时，如果内存不足，实例分配不到足够的内存，就会使有些用户连接不到数据库；或者虽然连接上数据库，但是查询的速度明显下降。内存结构主要包括系统全局区(System Global Area，SGA)和进程全局区(Process Global Area，PGA)，即 Oracle Memory Structures＝SGA＋PGA。

1. 系统全局区

当激活 Oracle 数据库时，系统会先在内存中规划一个固定区域，用来存储每位使用者所需存取的数据以及 Oracle 运作时必备的系统信息。这个区域就称为系统全局区(SGA)。SGA 随着数据库实例的启动向操作系统申请分配一块内存结构，又会随着数据库实例的关闭而释放，每个 Oracle 数据库实例有且仅有一个 SGA。

SGA 包含数个重要区域，分别是共享池(Shared Pool)、数据库缓冲区高速缓存(Database Buffer Cache)、重做日志缓冲区(Redo Log Buffer)、Java 池(Java Pool)、大池(Lager Pool)等。

1）共享池(Shared Pool)

共享池是 SGA 中最关键的内存片段，特别是在性能和可伸缩性上。一个太小的共享池会扼杀性能，使系统停止运行，太大的共享池也会有同样的效果，将会消耗大量的 CPU 来管理这个共享池。不正确地使用共享池只会带来灾难。共享池主要分为库高速缓存和数据

字典高速缓存两个部分。

(1) 库高速缓存(Library Cache)。

库高速缓存用于保存最近解析过的 SQL 语句、PL/SQL 过程形成的代码和执行计划。Oracle 在执行一条 SQL 语句、一段 PL/SQL 过程前首先在库高速缓存中搜索,如果查到它们已经解析过了,就利用库高速缓存中的解析结果和执行计划来执行,而不必重新对它们进行解析,从而显著提高执行速度。Oracle 是通过比较 SQL 或 PL/SQL 语句的正文来决定两个语句是否相同的,只有正文完全相同,Oracle 才重用已存在的编译后的代码和执行计划。应该尽量用绑定变量的方式编写 SQL 语句,绑定变量不是在编译阶段赋值的,而是在运行阶段赋值的,因此语句可以不用重新编译。

库高速缓存的管理采用 LRU(Least Recently Used)的队列算法,即最近最少使用的队列算法。刚使用的内存块放在 LRU 队列的头部,而进程每次从队列的尾部获取内存块,获取到的内存块立即移至队列头部。最终使长时间没有使用到的内存块自然移到队列的尾部而被最先使用。

(2) 数据字典高速缓存(Data Dictionary Cache)。

数据字典高速缓存用于存储经常使用的数据字典信息,如表的定义、用户名、口令、权限、数据库的结构等。Oracle 运行过程中经常访问该缓存以便解析 SQL 语句,确定操作的对象是否存在,是否具有权限等。如果需要的信息不在数据字典高速缓存中,服务器进程就从保存数据字典信息的数据文件中将其读入数据字典高速缓存中。数据字典高速缓存中保存的是一条一条的记录(就像是内存中的数据库),而其他缓存区中保存的是数据块信息。

Oracle 没有提供单独设置库高速缓存或数据字典高速缓存空间大小的方法,而是通过设置共享池的大小来间接设置,通过参数 SHARED_POOL_SIZE 可调整,其大小受限于 SGA 的尺寸 SGA_MAX_SIZE 参数。

2) 数据库缓冲区高速缓存(Database Buffer Cache)

数据库缓冲区高速缓存,也叫块缓存区,用于存放从数据文件读取的数据块,其大小由初始化参数 DB_CACHE_SIZE 决定,采用 LRU 队列进行管理。查询时,Oracle 先把从磁盘读取的数据放入此高速缓存供所有用户共享,以后再查询相关数据时不用再次读取磁盘。插入和更新时,Oracle 先在该区域中缓存修改后的数据,之后批量写到硬盘中。通过块缓存区,Oracle 可以提高磁盘的 I/O 性能。

数据库缓冲区高速缓存由许多大小相等的缓冲区组成,这些缓冲区分为如下 3 大类。

(1) 脏缓冲区(Dirty Buffers):脏缓冲区中保存的是被修改过的缓冲区,即当一条 SQL 语句对某个缓冲区中的数据进行修改后,该缓冲区就被标记为脏缓冲区。最后该脏缓冲区被 DBWn 进程写入硬盘的数据文件中永久保存。

(2) 命中缓冲区(Pinned Buffers):命中缓冲区中保存的是最近正在被访问的缓冲区。它始终被保留在数据高速缓存中,不会被写入数据文件。

(3) 空闲缓冲区(Free Buffers):空闲缓冲区中没有数据,等待被写入数据。Oracle 从数据文件中读取数据后,寻找空闲缓冲区,以便写入其中。

Oracle 通过两个列表(DIRTY、LRU)来管理缓冲区。

(1) DIRTY 列表中保存已经被修改但还没有被写入数据文件中的脏缓冲区。

(2) LRU 列表中保存所有的缓冲区(包括还没有被移动到 DIRTY 列表中的脏缓冲区、

空闲缓冲区、命中缓冲区)。当某个缓冲区被访问后,该缓冲区就被移动到 LRU 列表的头部,其他缓冲区就向 LRU 列表的尾部移动。放在最尾部的缓冲区将最先被移出 LRU 列表。

数据库缓冲区高速缓存的工作过程如下。

(1) Oracle 在将数据文件中的数据块复制到数据库缓冲区高速缓存之前,先在此缓存中寻找空闲缓冲区,以便容纳该数据块。Oracle 将从 LRU 列表的尾部开始搜索,直到找到所需的空闲缓冲区为止。

(2) 如果先搜索到的是脏缓冲区,则将该脏缓冲区移动到 DIRTY 列表中,然后继续搜索;如果搜索到的是空闲缓冲区,则将数据块写入,然后将该缓冲区移动到 LRU 列表的头部。

(3) 如果能够搜索到足够的空闲缓冲区,则将所有的数据块写入对应的空闲缓冲区中,搜索写入过程结束。

(4) 如果没有搜索到足够的空闲缓冲区,则 Oracle 就先停止搜索,然后激活 DBWn 进程,开始将 DIRTY 列表中的脏缓冲区写入数据文件中。

(5) 已经被写入数据文件中的脏缓冲区将变成空闲缓冲区,并被放入 LRU 列表中。执行完成这个工作后,再重新开始搜索,直到找到足够的空闲缓冲区为止。

在 Oracle 9i 版本之前,数据库缓冲区高速缓存的大小是由 DB_BLOCK_BUFFER 决定的,缓冲区的大小为 DB_BLOCK_SIZE(Oracle 数据块大小,创建数据库时设定好,后续不能改变)和 DB_BLOCK_BUFFERS(缓冲区块的个数)这两个参数的乘积。之后的版本则是由参数 DB_CACHE_SIZE 及 DB_nK_CACHE_SIZE 确定。

注意:改变参数需重启数据库才能生效。

不同的表空间可以使用不同的块大小。在创建表空间时通过设置参数 BLOCKSIZE 来指定该表空间数据块的大小。如果指定的是 2KB,则对应的缓存大小为 DB_2K_CACHE_SIZE 参数的值,如果指定的是 4KB,则对应的缓存大小为 DB_4K_CACHE_SIZE 参数的值,以此类推。如果不指定 BLOCKSIZE,则默认为参数 DB_BLOCK_SIZE 的值,对应的缓存大小是 DB_CACHE_SIZE 的值。

3) 重做日志缓冲区(Redo Log Buffer)

Oracle 在 DML 或 DDL 操作改变数据写到数据库缓冲区高速缓存之前,先写入重做日志缓冲区,随后 LGWR 后台进程再把日志条目写到磁盘上的联机重做日志文件中。重做日志缓冲区的大小由初始化参数 LOG_BUFFER 决定大小。

4) Java 池(Java Pool)

Java 的程序区。在 Oracle 8i 版本以后,Oracle 在内核中加入了对 Java 的支持。该程序缓存区就是为 Java 程序保留的。如果不用 Java 程序没有必要改变 Java 池的默认大小。

5) 大池(Lager Pool)

可以根据实际业务需要来决定是否在 SGA 区中创建大池。如果没有创建大池,则需要大量内存空间的操作将占用共享池的内存,将对系统性能带来影响。大池没有 LRU 队列,在共享服务器连接时,PGA 的大部分区域(UGA)将放入大池(不包括堆栈区域),并行化的

数据库操作、大规模的 I/O 及备份和恢复操作可能用到大池。大池由初始化参数 LARGE_POOL_SIZE 确定其大小。

2. 进程全局区 PGA

一个 PGA 是一块独占内存区域，Oracle 进程以专有的方式用它来存放数据和控制信息。当 Oracle 进程启动时，PGA 也就由 Oracle 数据库创建了。当用户进程连接到数据库并创建一个对应的会话时，Oracle 服务进程会为这个用户专门设置一个 PGA 区，用来存储这个用户会话的相关内容。当这个用户会话终止时，系统会自动释放这个 PGA 区所占用的内存。这个 PGA 区对于数据库的性能有比较大的影响，特别是对于排序操作的性能。

PGA 主要包含排序区、会话区、堆栈区和游标区 4 个部分。通常情况下，系统管理员主要关注的是排序区，在必要时需要手动调整这个排序区的大小。游标区是一个动态的区域，在游标打开时创建，关闭时释放。故在数据库开发时，不要频繁地打开和关闭游标，这样可以改善数据库的性能。其他分区的内容，管理员只需要了解其用途，日常的维护交给数据库系统来完成即可。

1）为排序设置合理的排序区大小

当用户需要对数据进行排序时，系统会将需要排序的数据保存到 PGA 中的一个排序区内，然后在这个排序区内对这些数据进行排序。如需要排序的数据有 2MB，那么排序区内必须至少要有 2MB 的空间来容纳这些数据；然后排序过程中又需要有 2MB 的空间来保存排序后的结果。由于系统从内存中读取数据比从硬盘中读取数据的速度要快几千倍，因此，如果这个数据排序与读取的操作都能够在内存中完成，无疑可以在很大程度上提高数据库排序与访问的性能。但是，如果 PGA 中的排序区容量不够，不能容纳排序后的数据，系统会从硬盘中获取一个空间，用来保存需要排序的数据。此时，排序的效率就会降低许多。为此在数据库管理中，如果发现用户的很多操作都需要用到排序，则为用户设置比较大的排序区，可以提高用户访问数据的效率。

在 Oracle 数据库中，这个排序区主要用来存放排序操作产生的临时数据。一般来说，这个排序区的大小占据 PGA 程序缓存区的大部分空间，这是影响 PGA 区大小的主要因素。在小型应用中，数据库管理员可以直接采用其默认的值。但是在一些大型应用中，或者需要进行大量记录排序操作的数据库系统中，管理员可能需要手动调整这个排序区的大小，以提高排序的性能，可以通过初始化参数 SORT_AREA_SIZE 来实现。

2）会话区保存着用户的权限等重要信息

会话区保存了会话所具有的权限、角色、性能统计等信息，通常都是由数据库系统自我维护，管理员不用干预。当用户进程与数据库建立会话时，系统会将这个用户的相关权限查询出来，保存在这个会话区内。用户进程在访问数据时，系统会核对会话区内的用户权限信息，确定其是否具有相关的访问权限。

3）堆栈区保存变量信息

堆栈区保存着绑定变量、会话变量、SQL 语句运行时的内存结构等重要的信息。通常都是由数据库系统自我维护，管理员不用干预。这些分区的大小，也是系统根据实际情况来进行自动分配的。当用户会话结束时，系统会自动释放这些区所占用的空间。

4）游标区

游标区是一个动态的区域。当用户执行游标语句并打开游标时，系统会在 PGA 中创

建游标区。当关闭游标时，这个区域就会被释放。创建与释放需要占用一定的系统资源，花费一定的时间，如果频繁地打开和关闭游标，就会降低语句的执行性能。因此，在写语句时，游标最好不要频繁地打开和关闭。

专用服务器模式下，进程和会话是一对一的关系，用户全局区（UGA）被包含在 PGA 中，在共享服务器模式下，进程和会话是一对多的关系，所以 UGA 就不再属于 PGA 了，而会在大池中分配。但如果从大池中分配失败，如大池太小，或是根本没有设置大池，则从共享池中分配。

3.3.2 进程结构

进程是操作系统中一个极为重要的概念。一个进程执行一组操作，完成一个特定的任务。对 Oracle 数据库治理系统来说，进程由用户进程、服务器进程和后台进程组成。

当用户运行一个应用程序时，系统就为它建立一个用户进程。服务器进程处理与之相连的用户进程的请求，它与用户进程进行通信，为相连的用户进程的 Oracle 请求服务。为了提高系统性能，更好地实现多用户功能，Oracle 还在系统后台启动一些后台进程，用于数据库数据操作。数据库的物理结构和存储结构之间的关系是由后台进程来维持的。数据库拥有多个后台进程，其数量取决于数据库的配置。这些进程由数据库管理，用户只需要进行很少的管理。每个进程在数据库中执行不同的任务。Oracle 主要的后台进程与内存结构、数据文件之间的协作关系如图 3-6 所示。

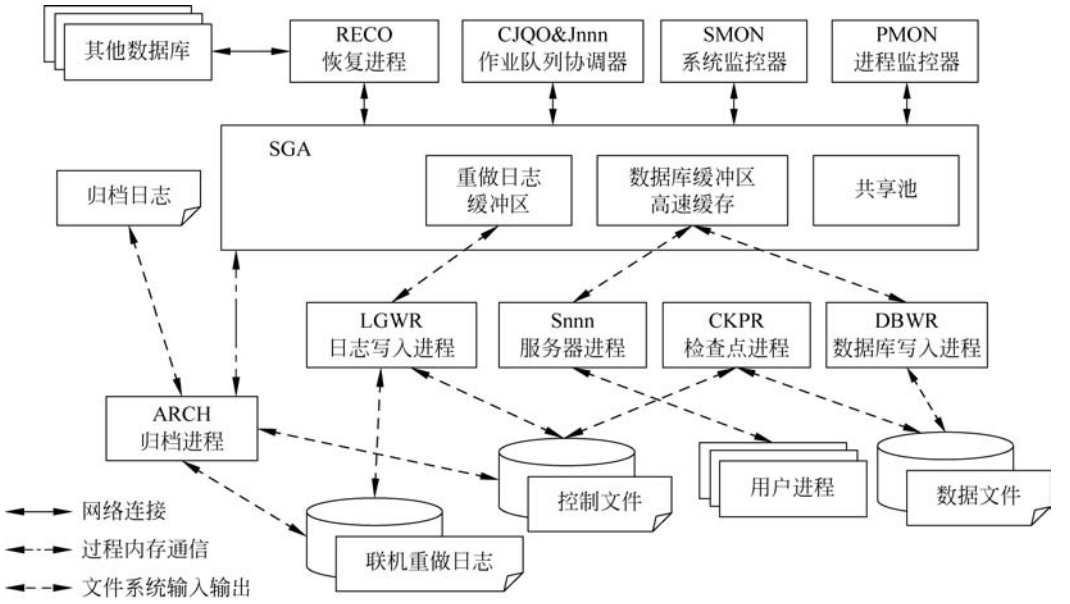

图 3-6　进程、存储结构、数据文件的协作关系

1. DBWR/DBWn（数据库写入进程）

数据库写入进程 DBWR 将数据库缓冲区高速缓存中的数据写入数据文件，是负责缓冲区管理的一个 Oracle 后台进程。当缓冲区中的某个缓冲区被修改时，它被标志为“弄脏”。DBWR 的主要任务是将“弄脏”的缓冲区写入磁盘，使缓冲区保持“干净”。随着缓冲区的使

用，空闲缓冲区的数目逐渐减少。当空闲缓冲区下降到很少，以致用户进程要从磁盘读入块到内存存储区却无法找到空闲缓冲区时，DBWR 将管理缓冲存储区，使用户进程总可得到空闲缓冲区。

Oracle 采用 LRU 算法保持内存中的数据块是最近最少使用的，使 I/O 最小。下列情况预示着 DBWR 要将弄脏的缓冲区写入磁盘。

(1) 当脏缓冲区的数量超过了所设定的限额。

(2) 所设定的时间间隔已到。

(3) 有进程需要数据库高速缓冲区，却找不到空闲缓冲区。

(4) 检查点发生。

(5) 某个表被删除或截断。

(6) 某个表空间被设置为只读。

(7) 对某个表空间进行联机备份。

(8) 某个临时表空间被设置为脱机状态或正常状态。

在前两种情况下，DBWR 将弄脏表中的块写入磁盘，每次可写的块数由初始化参数 DB_BLOCK_WRITE_BATCH 所指定。如果弄脏表中没有该参数指定块数的缓冲区，则 DBWR 从 LRU 表中查找另外一个弄脏缓冲区。

如果 DBWR 在 3s 内未活动，则出现超时。在这种情况下，DBWR 对 LRU 表查找指定数目的缓冲区，将所找到的任何弄脏缓冲区写入磁盘。每当出现超时时，DBWR 就查找一个新的缓冲区组。每次由 DBWR 查找的缓冲区的数目为参数 DB_BLOCK_WRITE_BATCH 的值的两倍。如果数据库空运转，DBWR 最终将全部缓冲区写入磁盘。

在出现检查点时，LGWR 指定一个修改缓冲区表必须写入磁盘。DBWR 将指定的缓冲区写入磁盘。

在有些平台上，一个实例可有多个 DBWR。在这样的实例中，一些块可写入一个磁盘，另一些块可写入其他磁盘。参数 DB_WRITERS 控制 DBWR 的进程个数。Oracle 实例允许启动最多 10 个数据库写进程，即 DBW0～DBW9。

2. LGWR(日志写入进程)

日志写入进程 LGWR 将重做日志缓冲区的数据写入重做日志文件，LGWR 是一个必须和前台用户进程通信的进程。当数据被修改时，系统会产生一个事务日志并记录在重做日志缓冲区内。Oracle 使用快速提交的技术，保证系统的效率，并保证系统崩溃时所提交的数据可以得到恢复。Oracle 引入系统改变号 SCN。SCN 是单调递增的正整数，与 Oracle 内部时间戳对应，保证系统中数据的同步和读一致性。

当 Oracle 发出 commit 命令后，系统的执行过程如下。

(1) 服务器进程把提交的记录连同产生的 SCN 号一起写入重做日志缓冲区。

(2) LGWR 把缓冲区中一直未提交的记录和 SCN 连续的写入联机重做日志文件中。在此之后，Oracle 就能够保证即使在系统崩溃的情况下所有已提交的数据也可以得到恢复。

(3) Oracle 通知用户进程提交已经完成。

(4) 服务器进程修改数据库高速缓冲区中的数据状态，释放资源和打开锁。

写日志要比写数据效率高，记录格式紧凑，I/O 量少，顺序写入。LGWR 写入时机如下。

(1) 事务被提交。

(2) 重做日志缓冲区中变化记录超过 1MB。

(3) 重做日志缓冲区中的记录超过缓冲区容量的 1/3。

(4) DBWR 写入数据文件之前。

(5) 每 3 秒钟。

3. SMON(系统监控进程)

当 Oracle 系统由于某种原因出现故障,如断电,SGA 中已经提交但还未被写入数据文件中的数据将丢失。当数据库重启时,系统监视器进程 SMON 将自动执行 Oracle 实例的恢复工作,过程如下。

(1) 执行前滚,将已提交到重做日志文件中但还未写到数据文件中的数据写到数据文件中。

(2) 前滚完成后立即打开数据库,这时数据文件中可能还有一些没有提交的数据。

(3) 回滚未提交的事务。

(4) 执行一些磁盘空间的维护工作。

4. PMON(进程监控进程)

进程监控进程 PMON 在用户进程出现故障时执行进程恢复,负责清理内存储区和释放该进程所使用的资源。例如,重置活动事务表的状态,释放封锁,将该故障的进程的 ID 从活动进程表中移去。PMON 还周期性地检查调度进程(DISPATCHER)和服务器进程的状态,如果已停止工作,则重新启动。

PMON 有规律地被唤醒,检查是否需要,或者其他进程发现需要时可以被调用。当某个用户进程崩溃时(如未正常退出),PMON 将负责清理工作,具体过程如下。

(1) 回滚用户当前的事务。

(2) 释放用户所加的所有表一级和行一级的锁。

(3) 释放用户所有的其他资源。

5. CKPT(检查点进程)

Oracle 为了提高系统效率和保证数据库的一致性,引入检查点事件。DBWR 将 SGA 中所有已改变了的数据库缓冲区高速缓存中的数据(包括已提交的和未提交的)写入数据文件中时,将产生检查点事件。检查点进程保证了所有到检查点为止的变化了的数据都已经写入数据文件中。在实例恢复时,检查点之前的重做日志记录已经不再需要,从而加快了实例的恢复速度。检查点事件发生时,Oracle 要将检查点号写入数据文件头中,还要将检查点号、重做日志序列号、归档日志名称和 SCN 号都写入控制文件中。

过于频繁地检查点会使联机操作受到冲击,因此需要在实例的恢复速度和联机操作之间折中(大多在 20min 以上)。

检查点进程在检查点出现时,对全部数据文件的标题进行修改,指示该检查点。在通常的情况下,该任务由 LGWR 执行。然而,当检查点明显地降低系统性能时,可使 CKPT 进程运行,将原来由 LGWR 进程执行的检查点的工作分离出来,由 CKPT 进程实现。对于许多应用情况,CKPT 进程是不必要的。只有当数据库有许多数据文件,LGWR 在检查点时

明显地降低性能时才使 CKPT 进程运行。CKPT 进程不将块写入磁盘,该工作是由 DBWR 完成的。

初始化参数 CHECKPOINT_PROCESS 控制 CKPT 进程的使能或不使能。默认为 FALSE,即为不使能。

检查点发生的时机如下。

(1) 重做日志文件的切换。

(2) LOG_CHECKPOINT_TIMEOUT 这个延迟参数的到达。

(3) 相应字节(LOG_CHECKPOINT_INTERVAL * size of IO OS blocks)被写入当前的重做日志文件。

(4) 执行 ALTER SYSTEM SWITCH LOGFILE 命令。

(5) 执行 ALTER SYSTEM CHECKPOINT 命令。

查看数据库的检查点号,可使用如下语句:

```
SELECT checkpoint_change#
FROM v$database;
```

查看数据库当前的 SCN 号,可使用如下语句:

```
SELECT current_scn
FROM v$database;
```

6. RECO(恢复进程)

恢复进程是在具有分布式选项时所使用的一个进程,自动解决在分布式事务中的故障。一个节点 RECO 后台进程自动连接到包含有悬而未决的分布式事务的其他数据库中,RECO 自动解决所有悬而未决的事务。任何相应于已处理的悬而未决的事务的行将从每个数据库的悬挂事务表中删去。

当一个数据库服务器的 RECO 后台进程试图建立同一远程服务器的通信,当远程服务器是不可用或者网络连接不能建立时,RECO 自动在一个时间间隔之后再次连接。

7. ARCH(归档进程)

当数据库运行在归档日志模式下时,ARCH/ARCn 进程将把日志切换后的联机重做日志文件中的数据复制到归档日志文件中,保证不会因联机日志文件组的循环切换而导致日志数据丢失,从而保证数据库的可完全恢复。归档日志文件是脱机的。Oracle 确保在一组重做日志的归档操作完成之前不会重新使用该组重做日志。

8. LCKn(锁进程)

锁进程在具有并行服务器选项环境下使用,可多至 10 个进程(LCK0,LCK1,…,LCK9),用于实例间的封锁。

9. Dnnn(调度进程)

调度进程允许用户进程共享有限的服务器进程(SERVER PROCESS)。没有调度进程时,每个用户进程需要一个专用服务进程(DEDICATEDSERVER PROCESS)。对于多线索服务器(MULTI-THREADED SERVER)可支持多个用户进程。如果在系统中具有大量用户,则多线索服务器可支持大量用户,尤其在客户/服务器环境中。

3.4 Oracle 12c 多租户架构

Oracle 12c 引入了 CDB 与 PDB 的新特性，在 Oracle 12c 数据库引入的多租户环境(Multitenancy Environment)中，允许一个数据库容器(Container Database，CDB)承载多个可插拔数据库(Pluggable Database，PDB)。这个特性允许在 CDB 容器数据库中创建并且维护多个数据库，在 CDB 中创建的数据库被称为 PDB，每个 PDB 在 CDB 中是相互独立存在的，在单独使用 PDB 时，与普通数据库无任何区别。

容器数据库可以由多个位于不同地理位置的同构或异构的数据库构成，由 Oracle 12c 将这些数据库整合在一起进行管理。将这些数据库统一到一个数据库中，就如同将物品放置到一个容器里一样。Oracle 12c 的多租户环境主要由 CDB 根容器(Root Container)、PDB 种子(PDB Seed)和 PDBs 组成，如图 3-7 所示。

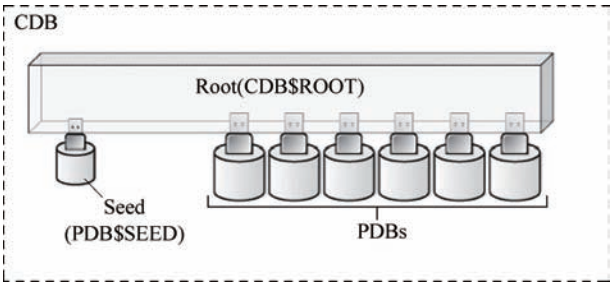

图 3-7　多租户结构示例

(1) CDB 根容器。CDB 根容器数据库是 CDB 环境中的根数据库，在根数据库中含有主数据字典视图，其中包含了与 Root 容器有关的元数据和 CDB 中所包含的所有的 PDB 信息。在 CDB 环境中被标识为 CDB$ROOT，每个 CDB 环境中只能有一个 Root 容器数据库。

(2) PDB 种子。PDB 种子为 PDB 的种子，其中提供了数据文件，在 PDB 环境中被标识为 PDB$SEED，是创建新的 PDB 的模板，可以连接 PDB$SEED，但是不能执行任何事物，因为 PDB$SEED 是只读的，不可进行修改。

(3) PDBs。PDB 数据库，在 CDB 环境中每个 PDB 都是独立存在的，与传统的 Oracle 数据库基本无差别，每个 PDB 拥有自己的数据文件和 objects。唯一的区别在于 PDB 可以插入 CDB 中，也可以在 CDB 中拔出，并且在任何一个时间点之上 PDB 必须拔出或者插入一个 CDB 中，当用户连接 PDB 时不会感觉到根容器和其他 PDB 的存在。

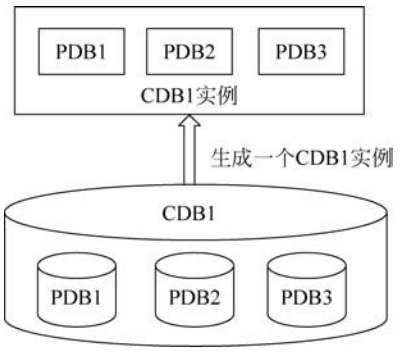

图 3-8　CDB 实例

在 Oracle 12c 之前的版本中，实例与数据库是一对一或多对一的关系。即一个实例只能与一个数据库相关联，或者多个实例加载到一个数据库中，实例与数据库不能是一对多的关系。对于 Oracle 12c，实例与数据库可以是一对多的关系。如图 3-8 所示，整个容器数据库生成一个实例 CDB1，其对应着 3 个不同的

PDB 数据库。每个 PDB 都有其全局唯一标识 ID,用于区别其他的 PDB。

图 3-9 展示了一个多租户容器数据库的体系结构。其中有 4 个容器：根和 3 个可插拔的数据库。每个可插拔数据库有自己的专用应用程序,由自己的 DBA 管理或者由容器管理员管理。可插拔数据库是一组数据库模式,它们在逻辑上对用户和应用程序作为一个独立的数据库。但在物理层,多租户容器数据库有一个数据库实例和数据库文件,就像非 CDB 那样。

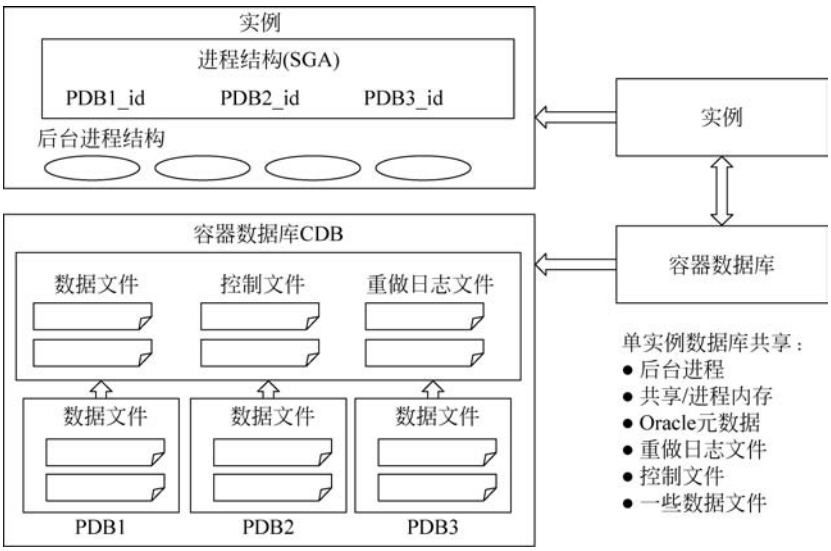

图 3-9　Oracle 12c 多租户架构体系结构

一个 CDB 对多个应用程序进行分组,最后以一个实例结束,从而产生一组后台进程、一个 SGA 分配,以及一个系统数据字典在根容器中。对于所有 PDBs 来说,每个 PDB 都维护自己的应用程序数据字典。当应用程序需要修补或升级时,将执行维护操作在 CDB 上只有一次,因此所有的应用程序都在同一时间更新。

3.5　本 章 小 结

本章详细介绍了 Oracle 的体系结构。Oracle 数据库服务分为 Oracle 数据库和 Oracle 实例。根据不同层面的划分,体系结构有着不同的类型,包括物理存储结构、逻辑存储结构、内存结构和进程结构。其中,Oracle 的物理存储结构主要是构成数据库的文件；Oracle 的逻辑结构可分为表空间、段、区、块以及逻辑对象。Oracle 实例分为内存结构和进程结构。内存结构分为系统全局区 SGA 和进程全局区 PGA；进程结构主要描述了 Oracle 实例与 Oracle 数据库进行交互的常用后台进程。

习　题　3

1. 简述 Oracle 数据库的物理存储结构。
2. 简述 Oracle 数据库的逻辑存储结构。
3. 简述 Oracle 实例的构成、SGA 的作用。
4. 简述 Oracle 实例主要的后台进行及其作用。