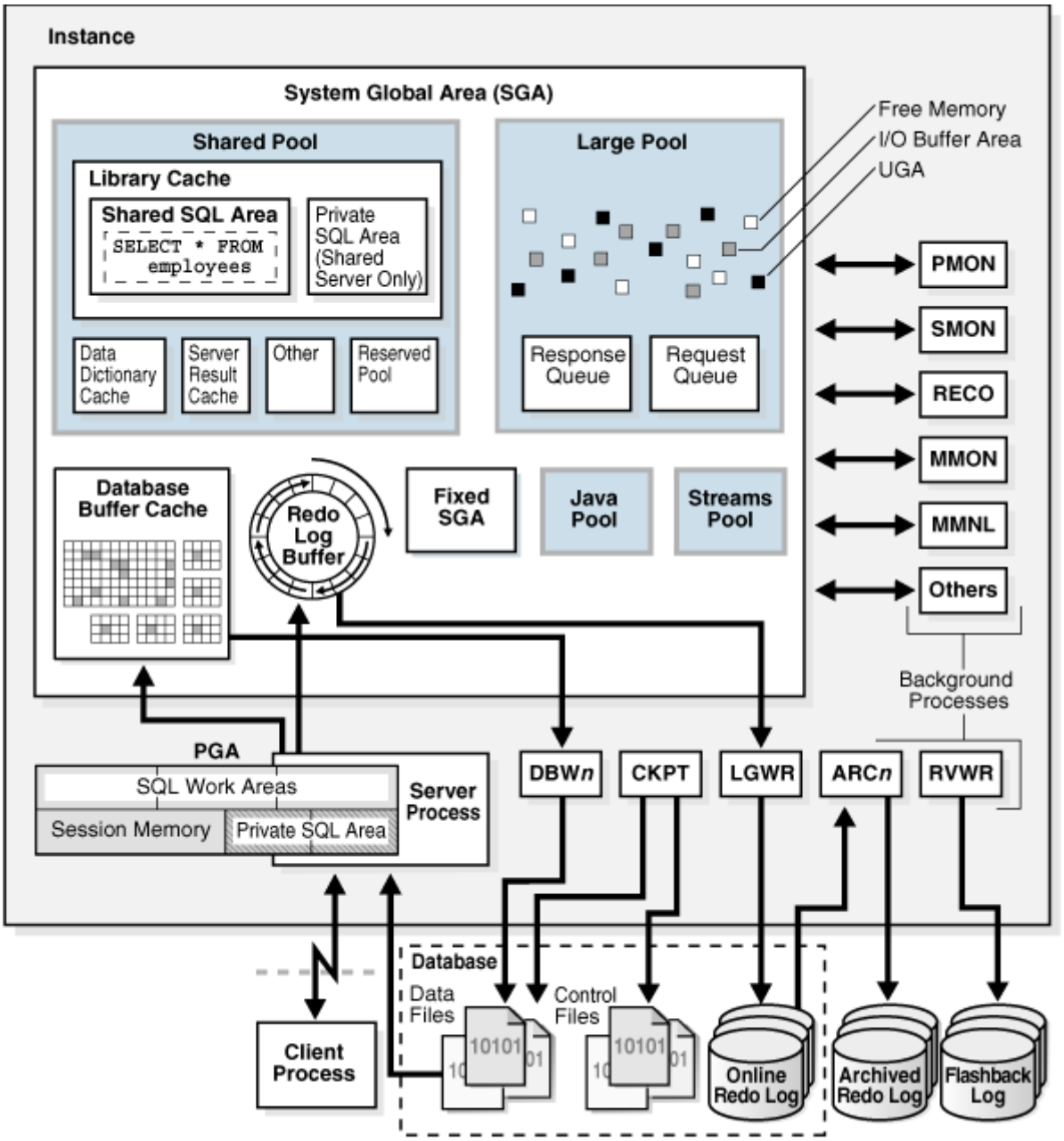
数据库的一个基本任务是用于存储数据，Oracle数据库的存储结构可以分为物理存储结构和逻辑存储结构。

其中，物理存储结构主要包括数据文件、重做日志文件、控制文件、参数文件、归档日志文件、跟踪文件和告警文件等文件；逻辑存储结构主要包括数据块、区、段和表空间这些数据库逻辑组件。本篇就Oracle数据库的数据库结构进行讲解。

1 物理存储结构

Oracle数据库的物理存储结构在操作系统级别是可见的，即物理存储结构主要是操作系统级别的文件。

如图，展示了数据库实例和数据文件：



1.1 数据文件

在操作系统级别，Oracle数据库将数据库数据存储在称为数据文件的结构中，每个Oracle数据库必须至少有一个数据文件。

1.1.1 数据文件用途

Oracle数据库物理的将表空间的数据存储在数据文件中。

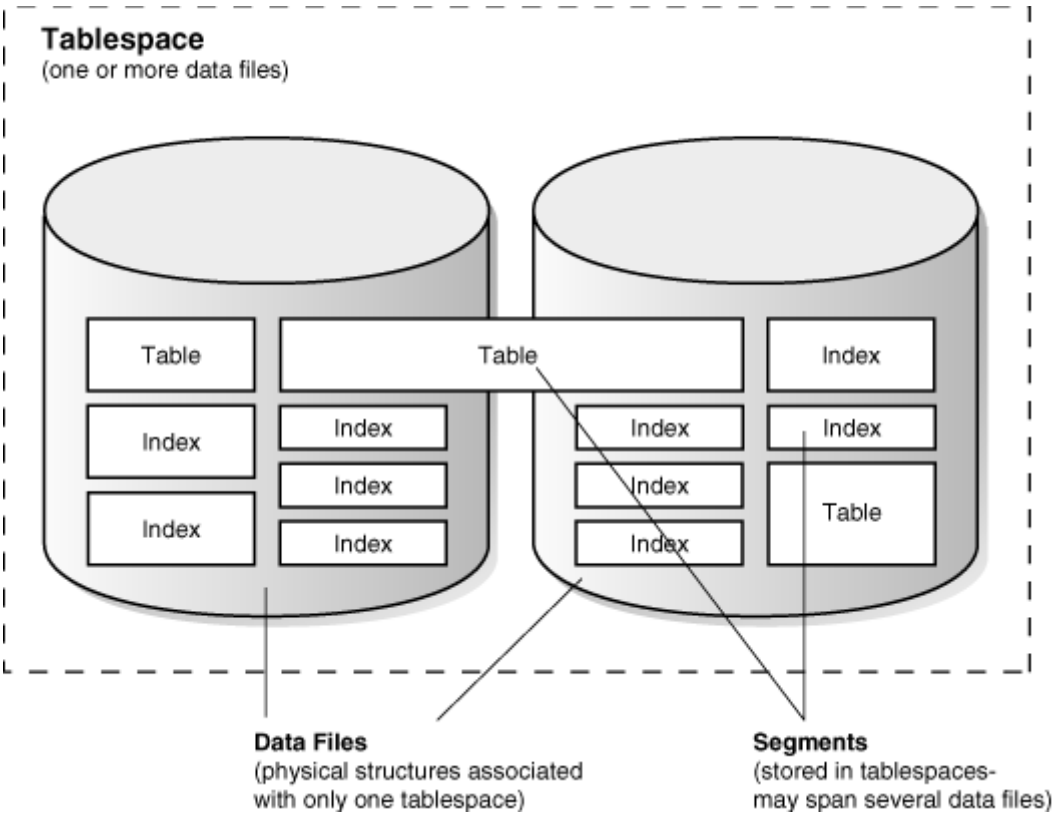
每个非分区的模式对象和对象的每个分区都存储在它自己的段中，该段只属于一个表空间。例如，非分区表的数据存储在单个段中，而这些段又存储在一个表空间中。表空间和数据文

件紧密相连，但又有重要的区别：

- 每个表空间由一个或多个数据文件组成；
- 数据库的数据集中存储在每个表空间的数据文件中；
- 段可以跨一个或多个数据文件，但不能跨多个表空间；
- 数据库必须有SYSTEM和SYSAUX表空间；

  系统表空间包含数据字典，数据字典是一组包含数据库元数据的表，通常，数据库还有一个undo表空间和一个临时表空间（通常命名为TEMP）。

如图，展示了表空间、数据文件和段的关系：



1.1.2 永久和临时数据文件

永久表空间包含持久的模式对象，永久表空间中的对象存储在数据文件中。

临时表空间只在会话期间包含模式对象，本地管理的临时表空间有临时文件，这些临时文件专门用来存储哈希、排序和其他操作的数据，当内存中空间不足时，临时文件也会存储结果集数据。

临时文件和永久数据文件类似，但有以下例外：

- 永久数据库对象（如表）永远不会存储在临时文件中；

- 临时文件总是设置为NOLOGGING模式，这意味着它们不会产生redo，介质恢复也不需要临时文件；
- 不能将临时文件设置为read-only模式；
- 不能使用Alter Database语句创建临时文件；
- 当创建或调整临时文件大小时，并不总保证为指定的文件大小分配磁盘空间。在Linux和Unix文件系统中，临时文件被创建为稀疏文件，在这种情况下，磁盘块不是在文件创建或调整大小时分配的，而是在第一次访问这些块时分配的。
- 临时文件信息存储在DBA_TEMP_FILES和V$TEMPFILE中，而不是在DBA_DATA_FILES或V$DATAFILE视图中。

1.1.3 在线和离线数据文件

数据文件有两种状态，即在线和离线。可以通过脱机或将其联机来更改单个数据文件或临时文件的可用性，数据库不能访问离线的文件，直到它们联机。

出于多种原因，可以将数据文件离线，包括执行脱机备份或块损坏。如果数据库不能写入数据文件，数据库将数据文件自动脱机。

与数据文件一样，表空间也有在线和离线两种状态，当将一个在线的表空间的一个数据文件离线时，表空间仍然处于在线状态，可以通过将表空间离线的方式来让表空间的所有数据文件暂时不可用。

从Oracle 12c开始，可以使用ALTER DATABSE MOVE DATFILE语句在数据库打开并访问文件时将在线数据文件从一个物理文件移到到另一个物理文件，可以使用此技术实现下面目标：

- 将表空间从一种存储空间移到另一种存储空间；
- 将不经常访问的数据文件移到更低成本的存储上；
- 将表空间置为read-only状态，并将其数据文件移到一次写存储上，例如一次写多次读驱动器；
- 将数据库移到Oracle ASM上；

1.1.4 数据文件结构

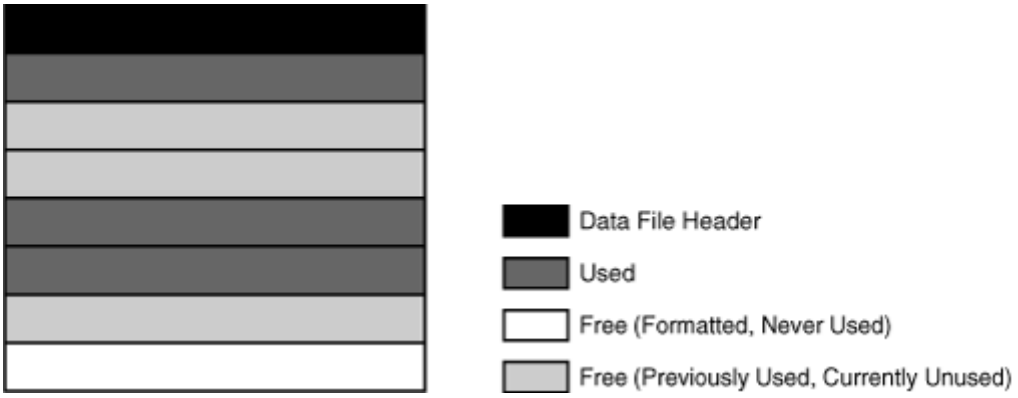
Oracle数据库通过分配指定的磁盘空间和数据文件头的开销为表空间创建数据文件，而Oracle数据库运行的操作系统负责清除文件中的旧信息以及授权，然后将其分配给数据

库。

数据文件头包含关于数据文件的元数据，比如数据文件的大小和检查点SCN，每个数据文件头包含一个绝对文件号（用于唯一的标识数据库中的数据文件）和一个相对文件号（用于唯一的标识一个表空间中的数据文件）。

当Oracle数据库首次创建数据文件时，已分配的磁盘空间被格式化，但不包含用户数据。但是，数据库会保留这些空间，用于供以后使用。随着表空间中数据的增长，Oracle数据库使用数据文件中的空闲空间给段分配区。

如图，展示了数据文件中不同类型的空间，区要么被使用（意味着它们包含段数据），要么空闲（意味着它们可以被重用）。随着时间推移，表空间中对象的更新和删除会产生很多的空闲空间，这些空闲空间不够大，不能用来存放新的数据，对于这种类型的空闲空间将其称之为碎片空闲空间。



1.2 控制文件

数据库的控制文件是一个小的二进制文件，每个数据库都有一个唯一的控制文件，但是它会维护多个完全相同的副本。

1.2.1 控制文件用途

控制文件在数据库创建的时候创建，用来记录数据库的物理结构、定位数据库文件以及管理数据库的状态。

控制文件包含如下信息：

- 数据库名称和数据唯一识别符DBID
- 数据文件、重做日志文件和归档重做日志文件的信息

- 数据库创建的时间戳
- 当前日志序列号
- 检查点信息
- 表空间信息
- RMAN备份

控制文件的作用如下：

- 它包含有关数据文件、在线重组日志文件等打开数据库所需的信息；
  控制文件跟踪数据库结构的变化，例如，当增加、重命名或删除数据文件或在线重做日志文件时，数据库会更新控制文件，以反映这些变化；
- 它包含数据库未打开时必须被可以访问的元数据；
  例如，控制文件包含恢复数据库所需的信息，包括检查点。检查点指示在重做流中实例恢复需要开始的SCN。通过检查点SCN，在其之前每个提交的更改被保证存放到磁盘上的数据文件中。至少每隔三秒钟检查点进程会在控制文件记录在线重做日志文件中检查点的位置。

在数据库使用期间，Oracle数据库不断地对控制文件进行读写操作，并且在数据库打开时必须能够进行写入操作。例如，恢复数据库需要从控制文件读取数据库中包含的所有数据文件的名称，其他操作（如增加数据文件）则会更新存储在控制文件中的信息。

1.2.2 多路控制文件

Oracle数据库允许同时打开多个相同的控制文件，并将其写入同一个数据库。通过在不同的磁盘上多路复用一个控制文件，数据库可以实现冗余，从而避免单点故障。

> Oracle建议在不同的磁盘上维护多个控制文件副本。

如果一个控制文件不可用，那么当数据库试图访问它时，数据库实例会失败，当存在其它当前控制文件副本时，那么可以重新挂载数据库，然后打开它，而不需要介质恢复。如果数据库的所有控制文件都丢失了，那么数据库实例会失败，并且需要进行介质恢复。

1.2.3 控制文件结构

有关数据库的信息存储在控制文件的不同部分，每一部分都是关于数据库的某一方面的一组记录。例如，控制文件中的一个部分追踪数据文件并包含一组记录，每个数据文件一个记

录。每个部分都存储在多个逻辑控制文件块中，记录可以跨块。

控制文件包含如下类型的记录：

- 循环重用记录
  循环重用的记录包含非关键信息，如果需要可以覆盖写入这些信息。当所有可用的记录槽都已满时，数据库可以扩展控制文件为新纪录开辟空间，或者覆盖写入那些最旧的记录。示例包括关于归档重做日志文件和RMAN备份的记录；
- 非循环重用记录
  非循环重用记录包含关键信息，这些信息不经常更改，并且不能被覆盖。信息示例包括表空间、数据文件、在线重做日志文件和重做线程。除非从表空间删除相应的对象，否则Oracle数据库不会重新使用这些记录；

可以通过查询动态性能视图，即V$视图查看存储在控制文件中的信息。例如，可以查询V$DATABASE获得数据库的名称和DBID。但是，只有数据库可以修改控制文件中的信息。

控制文件块的读取和写入与数据块的读取和写入不同，对于控制文件，Oracle数据库直接从磁盘读写到PGA，每个进程为控制文件块分配一定数量的PGA内存。

1.3 在线重做日志文件

对于数据库恢复最重要的结构是在线重做日志，它由两个或多个预先分配的、存储数据库变化的文件组成，在线重做日志将数据库的变化记录在数据文件中。

1.3.1 在线重做日志用途

Oracle数据库维护在线重做日志文件以防止数据丢失，具体来说，实例故障后，数据库能够使用在线重做日志文件恢复尚未写入到数据文件的已提交的数据。

服务器进程将每个事务同步地写入重做日志缓冲区，LGWR进程随后将其写入在线重做日志。在线重做日志的内容包括未提交的事务、模式和对象管理语句。

当数据库对undo段进行更改时，数据库也会将这些更改写入在线重做日志。因此，在线重做日志总是包含永久对象的undo数据。可以配置数据库将临时对象的所有undo数据存储在临时undo段中，这样可以节省空间并提高性能，或者允许数据库将永久和临时undo数据都存储在在线重做日志中。

Oracle数据库使用在线重做日志仅仅用于恢复，但是，管理员可以通过Oracle LogMiner工具中的SQL接口查询在线重做日志文件，重做日志文件是有关数据库活动的历史信息的有用来源。

1.3.2 如何写入在线重做日志

对数据库实例而言，在线重做日志被称为重做线程。

在单实例配置中，只有一个实例访问数据库，因此只有一个重做线程，然而，在RAC配置中，多个实例同时访问数据库，每个实例都有自己的重做线程。对于每个实例，分开的重做线程避免了对在线重做日志文件的争用。

在线重做日志由两个或更多在线重做日志文件组成，Oracle数据库需要至少两个文件，以确保在清除或归档另一个文件时，一个文件始终可以用来写入。

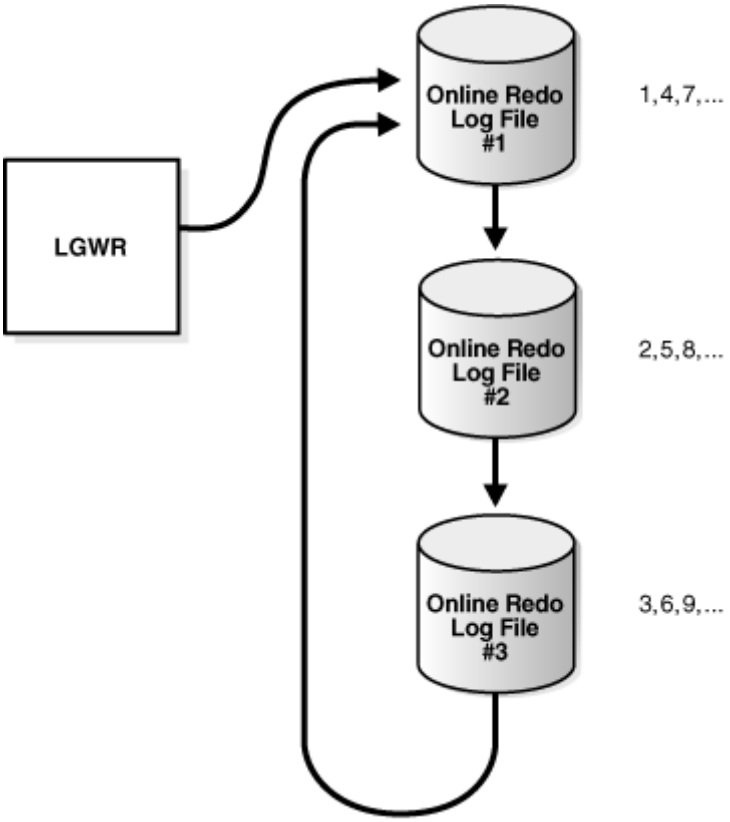
1.3.2.1 在线重做日志切换

Oracle数据库每次只使用一个在线重做日志文件用于存储从重做日志缓冲区写入的记录。LGWR进程正在写入的在线重做日志文件称为当前在线重做日志文件（current online redo log file）。

当数据库停止对一个在线重做日志文件进行写入并开始对另一个日志文件进行写入时，将发生日志切换，通常，当当前重做日志文件已满且必须继续写入时，就会发生日志切换。然而，可以配置日志定期进行切换而不用管当前重做日志文件是否已填满，也可以手动强制日志切换。

日志写入器以循环的方式写入在线重做日志文件，当日志写入器填满最后可用的在线重做日志文件时，进程将写入第一个日志文件，重新启动循环。

如图，展示了重做日志的循环写入：

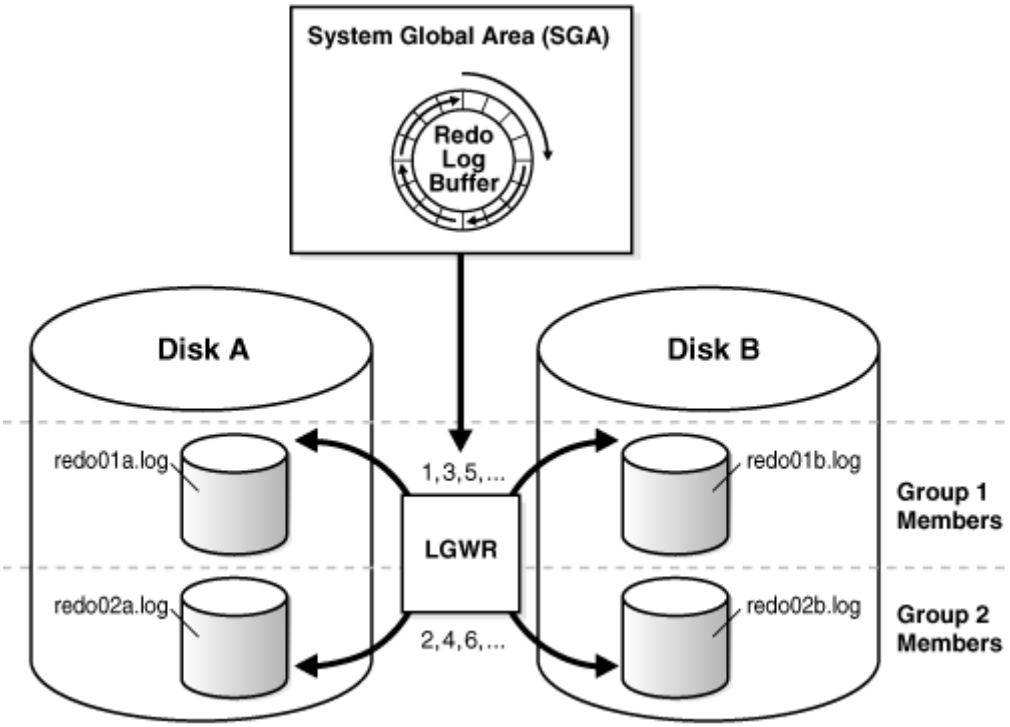


1.3.2.2 在线重做日志多路复用

Oracle数据库可以在不同的位置自动维护在线重做日志的两个或多个相同的副本。在线重做日志组由在线重做日志文件及其冗余副本组成，每个相同的副本是在线重做日志组的成员，日志组通过数字进行定义，例如组1、组2等。

维护在线重做日志组的多个成员可以防止重做日志丢失，理想情况下，日志成员的位置应该放在不同的磁盘上，这样磁盘故障不会导致整个在线重做日志的丢失。

如图，展示了在线重做日志文件的多路复用：



1.3.2.3 归档重做日志文件

Oracle数据库允许将已填满的重做日志文件组保存到一个或多个目的地，这些离线保存的日志统称为归档重做日志。将重做日志文件转换成归档的重做日志文件的过程称为归档，该过程仅运行在数据库处于归档模式下，可以自动归档，也可以手动归档。

当数据库处于归档模式时，在重做日志组归档之前，日志写进程LGWR不能重用和覆盖该日志组；当启动自动归档时，后台进程ARCn自动执行归档操作，数据库会根据需要启动多个归档进程，以确保已填满的重做日志的归档不会落后。

使用归档重做日志，可以进行如下操作：

- 恢复数据库；
- 更新standby数据库；
- 利用LogMiner工具获得数据库的历史信息；

1.3.3 在线重做日志结构

在线重做日志文件包含重做记录，重做记录由一组更改向量组成，每个更改向量描述对数据块的更改。例如，employees表中工资的更新会产生一个重做记录，它描述了对表数据段块、undo段数据块以及undo段的事务表的更改。

重做记录包含所有更改的相关的元数据，包括以下内容：

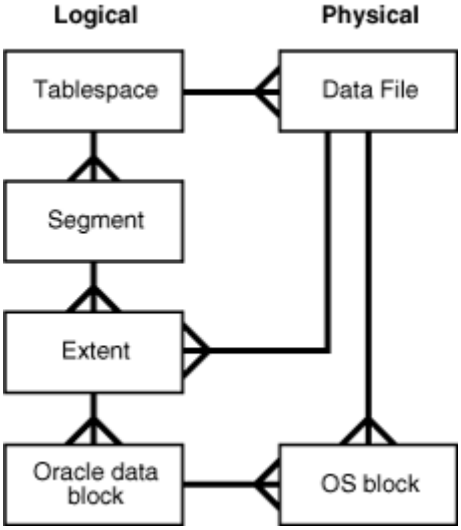
- 更改的SCN和时间戳；
- 生成更改的事务的事务ID；
- 事务提交时的SCN和时间戳（如果提交）；
- 做出更改的操作的类型；
- 被修改的数据段的名称和类型；

2 逻辑存储结构

2.1 逻辑存储结构介绍

Oracle数据库给数据库中所有的数据分配逻辑空间，数据库空间分配的逻辑单元是数据块、区、段以及表空间。在物理层面，数据存储在磁盘上的数据文件中，数据文件的数据存储在操作系统块中。

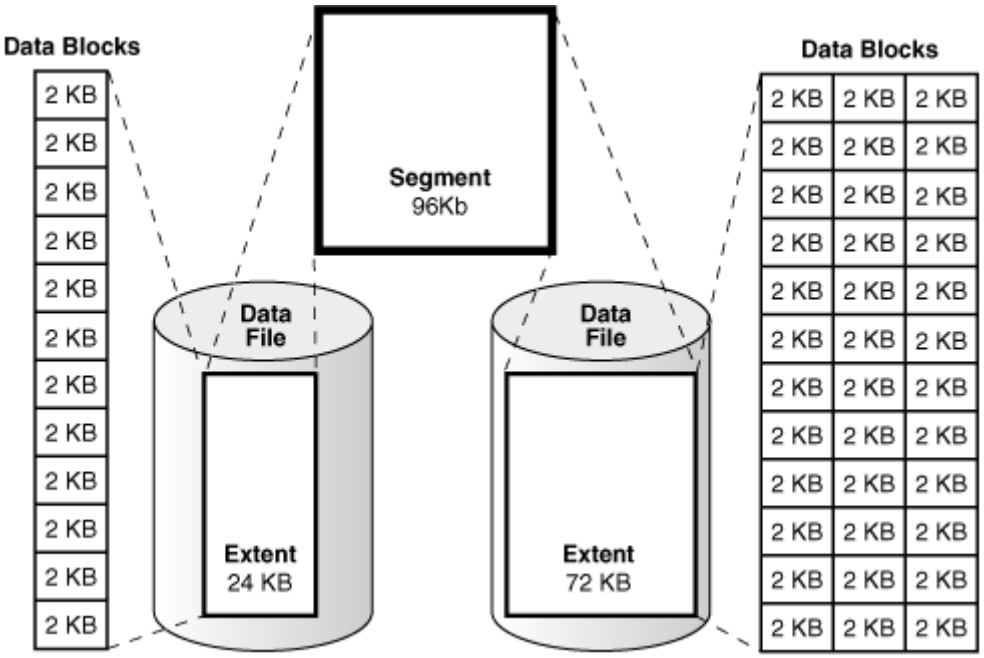
如图，展示了物理和逻辑存储的实体关系：



2.1.1 逻辑存储层次

段包含一个或多个区，每一个区包含多个数据块。

如图，展示了一个表空间中数据块、区、段的关系：



从最低到最高的粒度，Oracle数据库存储数据：

- 数据块是数据库中数据存储的最小逻辑单元，一个数据块对应物理磁盘空间中特定字节数；
- 区是特定数量的相邻Oracle数据块（通过一次分配获得），用于存储特定类型的信息；
- 段是为特定数据库对象(如表)分配的一组区；
- 表空间是包含一个或多个段的数据库存储单元；每个段属于一个并且只有一个表空间，因此，一个段的所有区存储在相同的表空间中，在表空间中，一个段能够包含来自于多个数据文件的区，单个区不能跨越数据文件。

2.1.2 逻辑空间管理

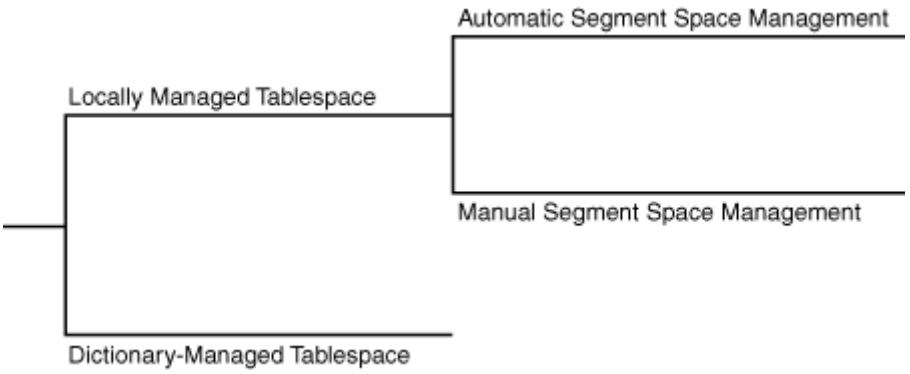
Oracle数据库必须使用逻辑空间管理来追踪和分配表空间中的区，Oracle数据库基于创建的表空间的类型来管理空间，可以创建下面某一类型的表空间：

- 本地管理的表空间

  默认的管理方式，该种方式下，数据库使用位图来管理表空间中的区，因此，本地管理的表空间中有一部分空间用于存放位图信息。在表空间内部，数据库可以使用自动段空间管理（ASSM）或手动段空间管理（MSSM）来管理段。
- 字典管理的表空间

  数据库使用数据字典来管理区。

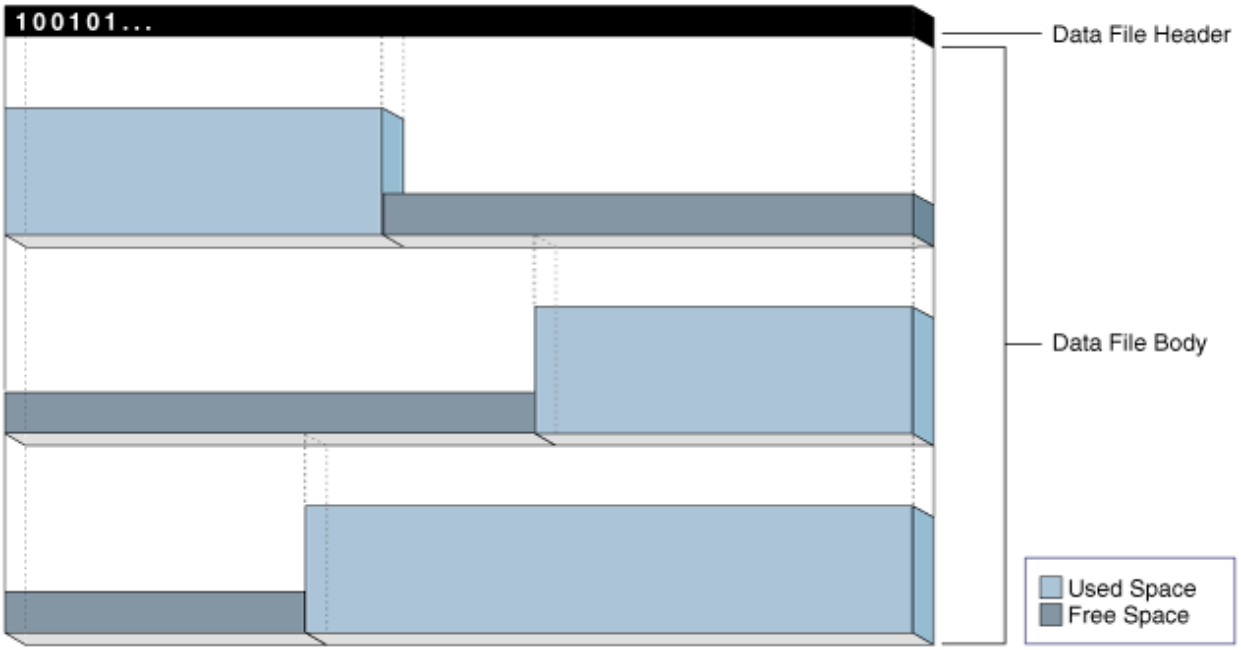
如图，展示了表空间中逻辑空间管理的方式：



2.1.2.1 本地管理的表空间

本地管理的表空间在数据文件的头部维护一个位图信息用于追踪数据文件体中空闲的空间和已使用的空间，每个位图对应一组数据块，当空间分配或释放时，Oracle数据库更改位图值以反映数据块的新状态。

如图，展示了位图管理存储的概念，1表示已使用的空间，0表示空闲空间：



本地管理的表空间有以下优点：

- 避免使用数据字典来管理区；
- 自动追踪相邻的空闲空间；
- 自动确定本地管理的区的大小；

> Oracle强烈建议使用自动段空间管理的本地管理的表空间。

2.1.2.2 字典管理的表空间

字典管理的表空间使用数据字典来管理区。每当分配或释放一个区以供重用时，Oracle数据库都会更新数据字典中的表。例如，当表需要区时，数据库查询数据字典表并搜索空闲区，如果数据库找到了空间，那么它将修改一个数据字典表并将一行插入到另一个数据字典表。通过这种方式，数据库通过修改和移动数据来管理空间。

Oracle数据库通过在后台运行递归SQL来获得数据库对象的空间，而频繁的使用递归SQL会对性能产生影响，因为对数据字典的更新必须以串行化的方式更新。而本地管理的表空间，是默认的表空间管理方式，避免了这种性能问题。

2.2 数据块

2.2.1 数据块介绍

数据块是Oracle数据库执行I/O的最小单元，从最细的层面来讲，Oracle数据库的数据存储在数据块中，一个数据块与磁盘中特定的字节数的物理空间相对应，每个表空间的数据块大小是在创建表空间时指定的，数据库以Oracle数据块为单位使用和分配空闲数据库空间。

数据库块大小通过DB_BLOCK_SIZE初始化参数指定，默认为8K。

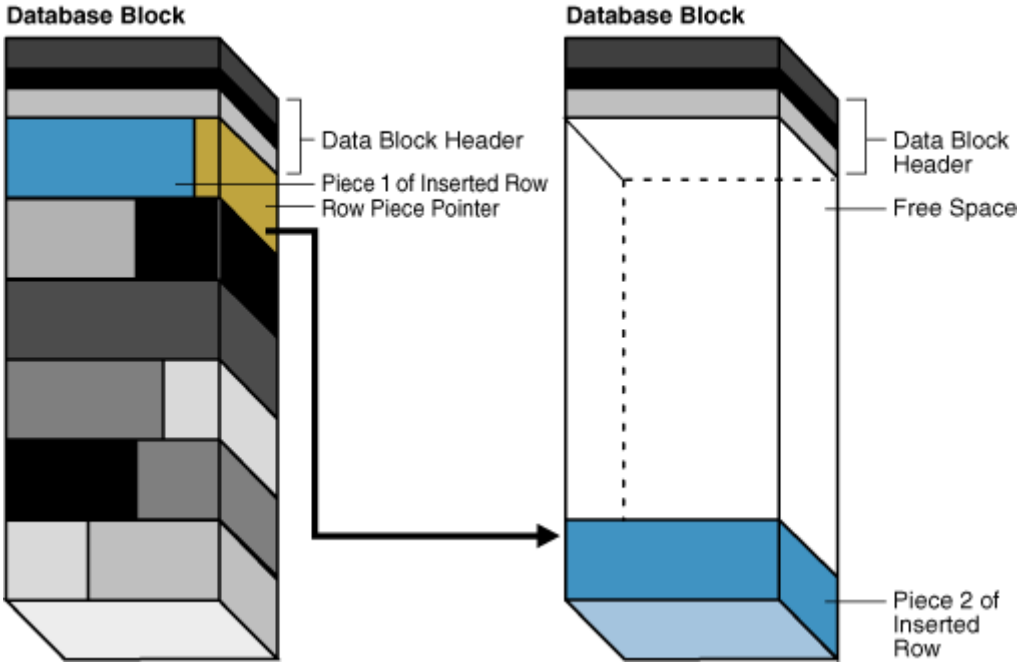
2.2.2 行迁移行链接

Oracle数据库使用行链接和行迁移来管理太大而不能装入单个块的数据行。

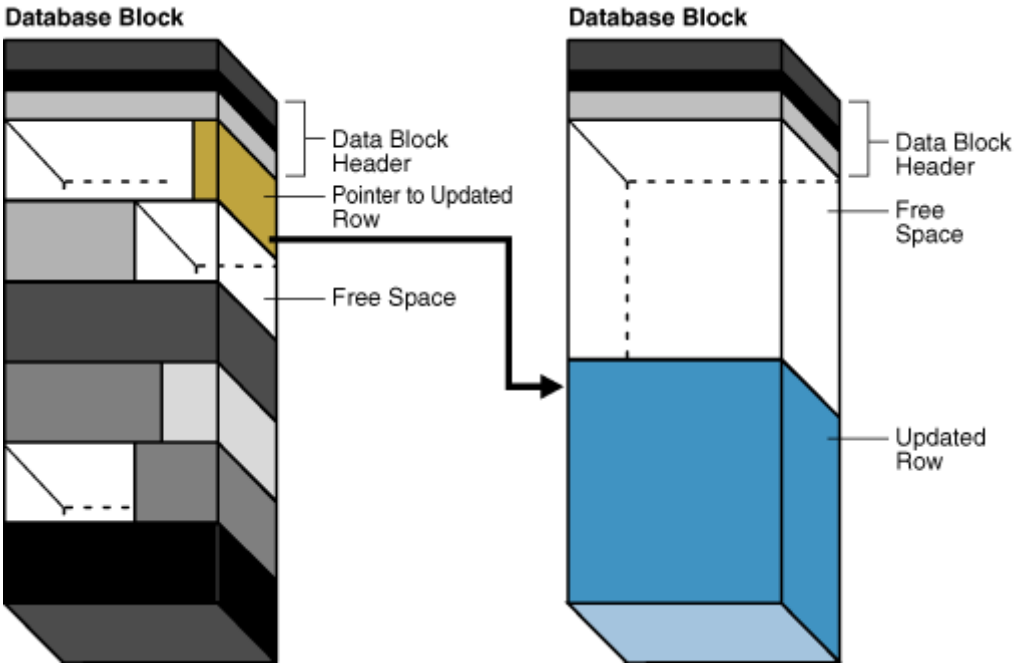
以下两种情况下，表中某行的数据可能太大，无法包含在单个数据块中：

- 行链接：在第一次插入行时，行太大，无法包含在一个数据块中。在这种情况下，Oracle数据库服务器将在为该段保留的一系统数据块（一个或多个）中存储该行的数据，较大的行通常需要进行行链接，例如包含数据类型为Long或Long Raw的列的行，在这些情况下，行链接是无法避免的；
- 行迁移：最初可以包含在一个数据块中的行在更新后，整个行的长度增加了，而块的空闲空间已经完全填满。在这种情况下，Oracle数据块服务器假定整个行可以包含在新数据块中，从而将整个行的数据迁移到新数据块中，数据库将保留迁移行的原始行片段，以便指向包含迁移行的新块，迁移行的Rowid不变。

如图，描述了在数据块中插入大行的情况，该行对于左边的块来说太大，因此数据库通过将第一个行片放入左边的块，第二个行片放入右边的块来链接该行（**行链接**）：



如图，左边的块包含一个更新的行，结果该行对于块来说就太大了。数据库将整个行移到右边的块，并在左边的块中留下一个指向已迁移行的指针（**行迁移**）：



当行链接或迁移后，与此行关联的输入/输出性能会降低，因为Oracle数据库必须扫描多个数据块来检索该行的信息。例如，如果数据库执行一个I/O来读取索引，一个I/O来读取未迁移的表行，那么需要一个额外的I/O来获取迁移行的数据。

2.3 区

逻辑数据库空间的下一级是区，区是特定数量的相邻Oracle数据块（通过一次分配获得），用于存储特定类型的信息，一个区中的Oracle数据块在逻辑上是相邻的，但在物理上可以分布在磁盘上的不同位置（RAID条带化和文件系统实施会导致此现象）。

默认情况下，当段创建时，数据库为段分配一个初始化区，而区总是包含在一个数据文件中。

2.4 段

2.4.1 段介绍

区的上一级逻辑数据库存储称为段，段是为某个逻辑结构分配的一组区，段可以分为：

- 数据段
  每一个以非集群、非索引方式组织的表都有一个数据段，但外部表、全局临时表和分区表除外，它们每一个都有一个或多个段。表中的所有数据都存储在相应数据段的区中。对于分区表，每个分区都有一个数据段，每个集群也都有一个数据段，集群中每个表的数据都存储在集群的数据段中；
- 索引段
  每个索引都有一个索引段，存储其所有数据，对于分区索引，每个分区都有一个索引段；
- 还原段
  针对每个数据库实例，都会创建一个UNDO表空间，该表空间包含大量用于临时存储还原信息的还原段。还原段中的信息用于生成读一致性数据库信息，并且在数据库恢复过程中，用于为用户回退未提交的事务处理；
- 临时段
  临时段是在需要临时工作区来执行SQL语句时有Oracle数据库创建的，语句完成执行后，临时段的区将返回到实例以备将来使用，可以为每个用户指定一个默认临时表空间，也可以指定一个在数据库范围内使用的默认临时表空间。

> 另外还有一些上面未列出的其他类型的段，也有一些方案对象（如视图、程序包、触发器等等），虽然是数据库对象，但不被视为段，段拥有单独的磁盘空间分配，其余对象存储为系统元数据段中的行。

Oracle数据库对空间进行动态分配，如果段中的现有区都已满，将会再增加一些区，因为区是根据需要分配的，因此段中的区在磁盘中可能是相邻的，也可能是不相邻的，这些区可以来自属于同一个表空间的不同数据文件。

2.4.2 段空间与高水位线

为了实现对空间的管理，Oracle数据库追踪段中块的状态。高水位标记（High water mark HWM）可以理解为段中一个点，在该点之上，数据块没有被格式化，并且从未被使用过。

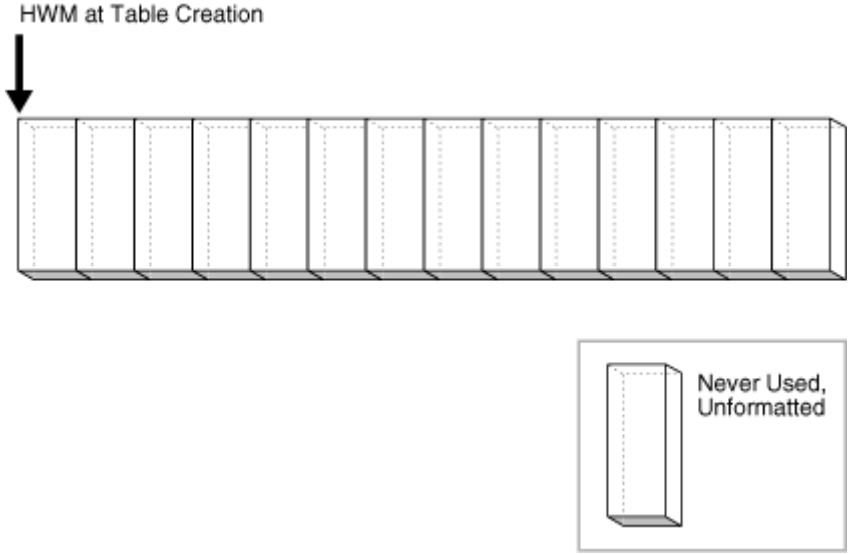
MSSM使用free list来管理段空间，在表创建时，段中的任何块都不会被格式化，当会话首次向表中插入行时，数据库搜索free list来查找可用的数据块，如果数据库没有找到可用的数据块，那么它将预先格式化一组块，将它们放进free list中，并开始将数据插入到数据块。在MSSM中，全表扫描会读取HWM下所有的数据块。

ASSM不使用free list，因此必须以不同的方式管理空间。当会话首次向表中插入数据时，数据库格式化一个位图块，而不是像MSSM那样预先格式化一组块，它使用位图而不是free list来追踪段中块的状态。数据库使用位图来查找空闲块，然后在填充数据之前对每个块进行格式化。为了避免并发问题，ASSM在块之间展开插入。

ASSM段中的每个数据块都处于下列一种状态：

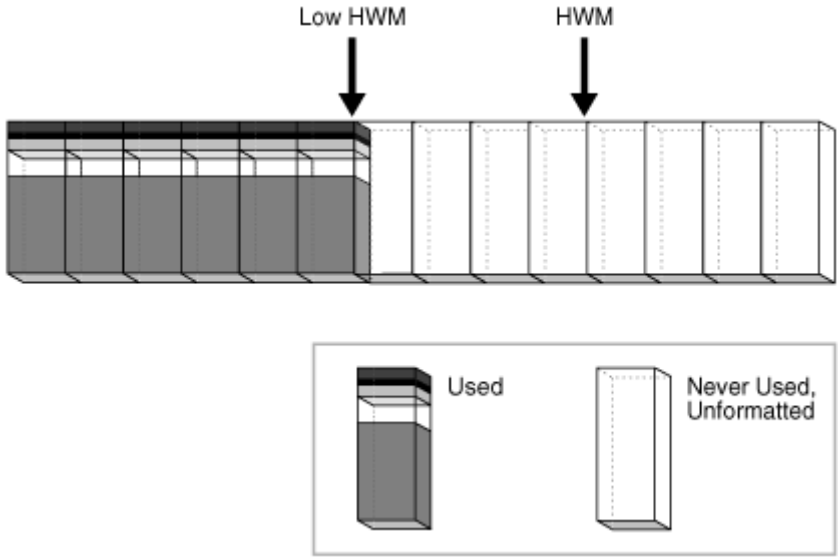
- 处于HWM之上：处于HWM上面的数据块未被格式化，并且从未被使用过；
- 处于HWM之下：处于HWM下面的数据块处于以下状态之一：
    - 已分配，但目前未格式化和未使用；
    - 已格式化并包含数据；
    - 已格式化，但是空闲状态，因为数据被删除了。

如图，在表创建时，高水位线位于左侧的段的开头，因为还没有插入数据，所以段中所有的块都是未格式化的，并且从未使用过：

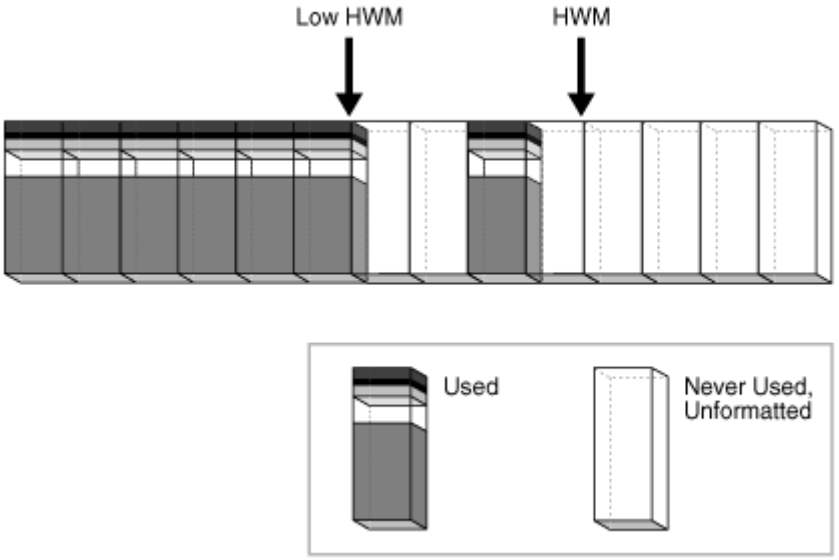


假设一个事务向该段插入数据行，数据库必须分配一组块来保存这些行，分配的块落在HWM下面，数据库在该组中格式化一个位图块来保存元数据，但不会预先格式化组中剩余的块。

如图，HWM下的块被分配，而HWM上的块既未被分配也未被格式化。当插入发生时，数据库可以写入任何具有可用空间的块，低HWM标识一个点，在其下所有的块都是被格式化的，因为它们要么当前包含数据，要么之前包含数据：

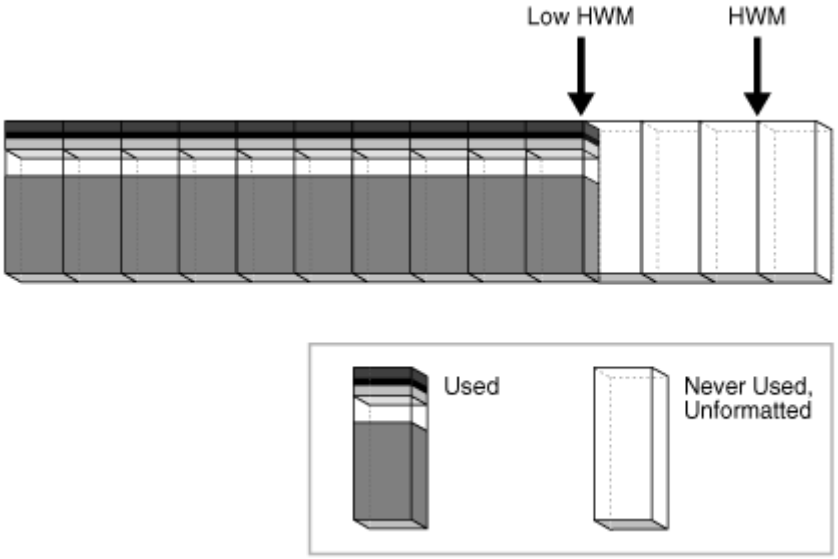


如图，数据库在HWM和低HWM直接选择一个块，并写入它。数据库可以很容易地在HWM和低的HWM之间选择任何数据块，或者在具有可用空间的低的HWM下的任何数据块，下图展示了新填充块两边的数据块都是未格式化的：



低HWM在全表扫描中很重要，由于HWM下面的块只有在使用时被格式化，一些块可能未被格式化。正如上图所示，基于这个原因，数据库读取位图块以获得低HWM的位置，数据库读取低HWM下所有的块，因为它们已被格式化，然后读取低HWM和HWM之间仅格式化的块。

假设一个新的事务向表中插入行，但是位图显示HWM下空闲空间不足，如图，数据库向右推进HWM，分配了一组新的未被格式化的块：



当HWM与低HWM之间的块被填满时，HWM向右移动，低HWM移动到旧的HWM的位置，随着数据库插入数据，HWM继续向右移动，同时低HWM总是尾随其后，除非手动重建、截取或收缩（rebuild、truncate、shrink）该对象，否则HWM不会下移。

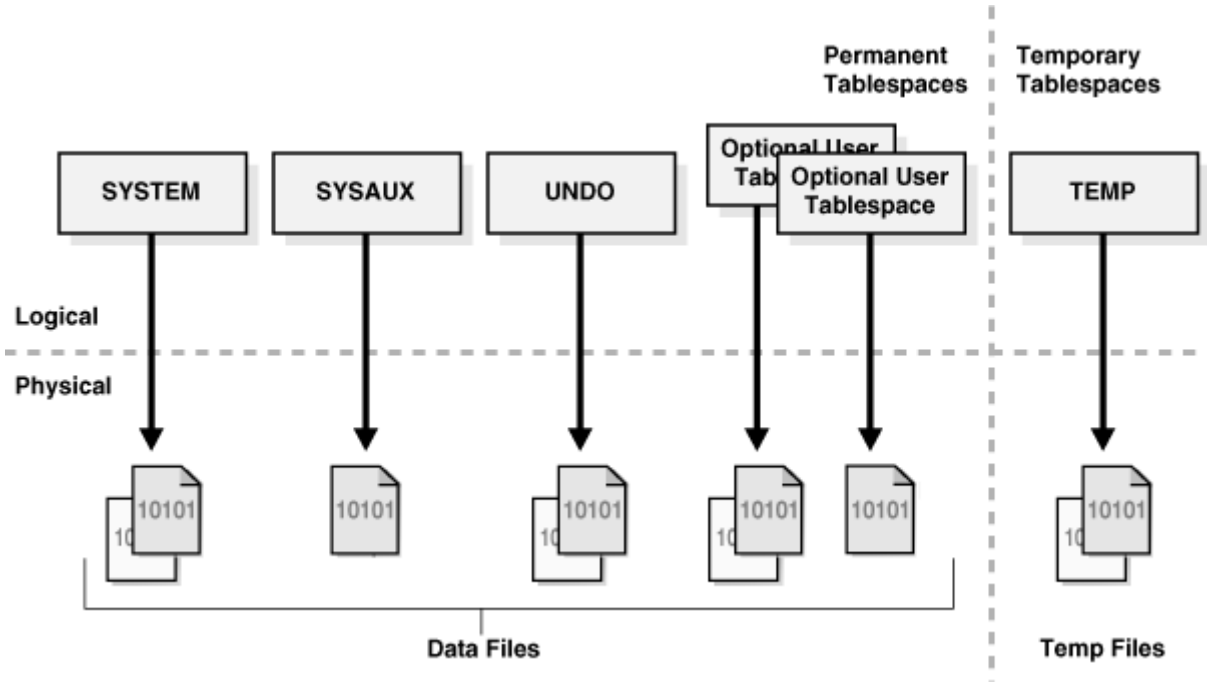
> 参照实验演示高水位线。

2.5 表空间

表空间是数据库的逻辑存储容器，用于存放段，段是消耗存储空间的数据库对象，例如表和索引。在物理层面，表空间将数据存储在一个或多个数据文件或临时文件中。

数据库必须具有SYSTEM和SYSAUX表空间。

如图，展示了典型数据库中的表空间以及表空间类型：



2.5.1 永久表空间

永久表空间对持久性模式对象进行分组，表空间中对象的段物理地存放在数据文件中。

每个数据库用户都会分配一个默认的永久表空间，对于很小型的数据库可能只需要默认的SYSTEM和SYSAUX表空间，但是，Oracle建议至少创建一个表空间用于存放用户和应用数据，使用表空间可以完成以下目标：

- 控制分配给数据库数据的磁盘空间；
- 为数据库用户进行空间限额；
- 在线或离线单个表空间，不至于影响整个数据库的可用性；
- 执行单个表空间的备份和恢复操作；
- 使用数据泵技术导入或导出应用数据；
- 创建可传输的表空间；

2.5.1.1 系统表空间

系统表空间SYSTEM是数据库创建时必要的管理表空间，Oracle数据库使用SYSTEM进行数据库的管理。

系统表空间由SYS用户拥有，包含下面的信息：

- 数据字典；
- 包含关于数据库管理信息的表和试图；
- 已编译的存储对象，例如触发器、过程和包；

> Oracle强烈建议使用DBCA创建数据库，所有的表空间，包括SYSTEM表空间，默认都是本地管理的表空间。

2.5.1.2 辅助表空间

SYSAUX是SYSTEM表空间的辅助表空间，包含一些特性和产品的数据信息，以减轻SYSTEM表空间的负担。

创建或更新数据库会自动创建SYSAUX表空间，在正常数据库操作期间，SYSAUX表空间不允许被删除或重命名。如果SYSAUX表空间不可用，核心数据库功能仍然可用，使用SYSAUX表空间的数据库特性将会不可用，或者功能部分可用。

2.5.1.3 撤销表空间

UNDO表空间是为系统管理的撤销数据保留的本地管理的表空间。

撤销表空间用于回滚事务，以及提供与DML语句同时运行在相同的表或表集上的select语句的读一致性，并支持大量Oracle闪回特性，例如闪回查询(Flashback Query)。

2.5.2 临时表空间

相对于其他表空间而言，临时表空间(temp tablespace)主要用于存储Oracle数据库运行期间所产生的临时数据。数据库可以建立多个临时表空间。当数据库关闭后，临时表空间中所有数据将全部被清除。除临时表空间外，其他表空间都属于永久性表空间。

2.5.3 表空间模式

表空间的模式决定了表空间的可访问性：

- 读写/只读表空间
    - 读写模式，即read/write模式；
    - 只读模式，即read-only模式；
- 在线/离线表空间
    - 在线模式，即online模式；
    - 离线模式，即offline模式；

2.5.4 表空间的大小

表空间有两种，分别为大文件表空间（bigfile tablespace）和小文件表空间（smallfile tablespace），默认是小文件表空间。

下面列出了大文件表空间和小文件表空间的区别：

- 小文件表空间可以包含多个多数据文件或临时文件，但是文件不能像大文件表空间中的文件那样大，小文件表空间是默认的表空间类型；
- 大文件表空间包含一个非常大的数据文件或临时文件，该种类型的表空间可以做到以下几点：
    - 增加数据库的存储容量；
    - 减少管理许多数据文件和临时文件的负担；
    - 在表空间上而不是在单独的文件上执行操作；