ORACLE 9i 資料庫管理實務講座〔三〕

(原文刊登於 2002 年 01 月 Linuxer 雜誌)

圖文： 何致億 Rich Ho

Oracle 9i Instance 組成與運作原理

本文內容索引

前言

前兩期的專欄內容已分別為大家介紹了 Oracle9i for Linux 的安裝方式，以及Oracle9i 企業版概觀。但是如何組態與管理 Oracle9i 資料庫才是我們要討論的重點。本文將深入探討 Oracle9i Instance的組成要素與其運作原理，為您揭開 Oracle9i 的神秘面紗。

本文將涵蓋以下主題：

- Oracle9i 伺服器架構簡介。
- 何謂 Oracle9i Instance
- 系統全域區
- 背景處理程序

在開始任何 Oracle9i 資料庫管理工作之前，您必須先熟悉 Oracle9i 伺服器之整體架構及其運作原理。所以我先為大家簡介 Oracle9i 伺服器的基本組成架構。

Oracle9i 伺服器架構

以 Oracle9i 資料庫系統本身而言，大致上可區分為兩個主要部分：

- Oracle9i 執行個體(Oracle9i Instance)
- Oracle9i 資料庫檔案(Database files)。

簡言之，Oracle9i Instance 是指資料庫伺服器的記憶體與相關處理程序。您可以想像它就是Oracle9i的心臟。資料庫實體則由作業系統內的各式檔案組成。如果您想深入瞭解Oracle9i 系統運作或從事進階的效

能調校，那麼一定要搞清楚這兩部分彼此的互動關係才行！Oracle9i伺服器基本架構如圖 1 所示：

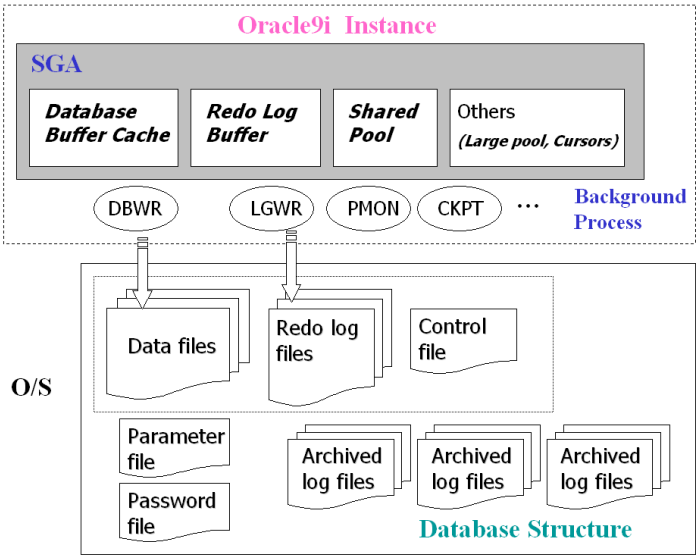

圖 1：Oracle9i 伺服器基本架構。

圖 1 上半部為記憶體內 Oracle9i Instance，下半部則是位於作業系統的各種資料庫檔案(有關這些檔案的細部資訊將在下一期說明)。彼此之間是藉由各個背景處理程序互相溝通。

接下來我們先討論 Oracle9i Instance 的組成要素。

何謂 Oracle9i Instance?

第一次接觸 Oracle 資料庫的使用者經常對「Oracle Instance」這個名詞感到混淆，因為Instance這個單字從字典上查到的意義跟資料庫一點關係也沒有！某些中文書譯者喜歡將 Oracle Instance 譯為「Oracle 實例」，但是我認為這個譯名十分不恰當！我個人比較傾向稱它為「Oracle執行個體」；俗稱「Oracle資料庫引擎」。既然是資料庫引擎，就表示Oracle資料庫內大大小小的事都跟它有關係，當然這也是我們一開始就先討論的主要原因囉。

Oracle9i 執行個體主要是由以下兩項要素組成：

- 系統全域區(System Global Area)
- 背景處理程序(background processes)

來看看什麼是「系統全域區」？

系統全域區

當您啓動 Oracle9i 資料庫時，系統會先在記憶體內規劃一個固定區域，用來儲存每位使用者所需存取的資料，以及Oracle9i運作時必備的系統資訊。我們稱此區域為系統全域區(System Global Area)，俗稱SGA（註 1）。

註 1：在多人使用的環境下，SGA 的資料可分享給所有同時上線的連線階段使用，所以 SGA 有時也稱為 Shared Global Area。

在 Oracle8i 時，SGA 的大小是由起始參數檔(initialization parameter file)內的某些參數所設定(註 2)。但最麻煩的是每次調整參數之後必須等重新啓動資料庫才生效(感覺就像在Windows系統修改了一些設定就要重新開機)。光是資料庫的關閉與啓動就花去不少時間(難怪大家羨幕 Oracle Consultant 很好賺.....，just kidding!!)，何況是執行關鍵性任務的資料庫哪能一天到晚開來關去!? 從 Oracle9i 開始，DBA 可以動態配置記憶體的大小；這樣的技術我們稱為「dynamic SGA」。有了 dynamic SGA ，SGA 的各組成區域都可以動態地進行規劃與調整，而不需先關閉資料庫。

一般來說，我們還是應該在啓動 Oracle9i 之前就妥善規劃好適當的記憶體空間。例如，起始參數檔的 SGA_MAX_SIZE(註 3)可設定 SGA 所佔用的最大記憶體空間。如果考慮 Oracle9i 的執行效能，理應將此參數盡可能設到最大！但有一點需要注意的是：SGA_MAX_SIZE 盡量不要超過實體記憶體大小，否則將會使用到硬碟上的虛擬記憶體，反而導致效能低落。

SGA 的大小可由起始參數檔之特定參數所控制，表 1 整理出與 SGA 相關的參數名稱及其意義：

參數名稱	用途
SGA_MAX_SIZE	設定 SGA 總大小
DB_CACHE_SIZE	設定由標準區塊組成的 Database Buffer Cache 大小
LOG_BUFFER	設定 Log Buffer 之大小
SHARED_POOL_SIZE	設定 Shared pool 之大小
LARGE_POOL_SIZE	設定 large pool 大小；預設為 0

如圖 1 所示，SGA 又包含數個重要區域，分別是：

- Database Buffer Cache（資料快取緩衝區）
- Redo Log Buffer（重置日誌緩衝區）
- Shared Pool（共享區）
- 其他，如 Large pool

以下是每個區域之意義、用途，以及相關設定方式。

資料快取緩衝區

為SGA 的主要成員，用來存放讀取自資料檔案的資料區塊複本，或是使用者曾經處理過的資料。其用途在於有效減少存取資料時造成的磁碟讀寫動作，進而提昇資料存取之效能。所有同時上線的使用者都可以共用此緩衝區的資料。

整個資料快取緩衝區包含兩種緩衝區串列，分別是 write list 與 LRU list：

- Write list 存放dirty buffers(註4)之複本，會在適當時機寫入磁碟。
- LRU list 包含：free buffers，dirty buffers 與pinned buffers。其中free buffer 為空白的緩衝區，隨時可存放資料；pinned buffer 則是目前使用中的緩衝區。

(註 2)
起始參數檔(Initialization Parameter file)之意義
如上所述，Oracle Instance被啓動時，系統必須藉由某些參數值來配置適當大小的記憶體空間。換言之，我們可以在啓動Instance之前就先規劃這些參數的設定值，並儲存在作業系統下的某個檔案裡。往後只要利用此檔案就可開啓相對應的 Oracle Instance — 我們就將這個檔案稱之為起始參數檔。

(註 3)
如果 SGA_MAX_SIZE 之設定值小於其他 SGA 相關參數設定值的總和；或是小於各相關參數預設值的總和，則 SGA_MAX_SIZE 之設定值無效。

(註 4)
dirty buffer 是存放"已修改，但尚未寫入磁碟的資料"之緩衝區。

資料快取緩衝區運作原理

當使用者第一次向Oracle9i 送出資料查詢請求時，Oracle9i 會先在資料快取緩衝區內尋找該資料。如果欲查詢的資料恰好已存在於緩衝區內(這樣的情況我們稱之為 cache hit)，就直接從記憶體讀出資料。

反之，如果緩衝區內並沒有使用者欲查詢的資料(此情況稱為 cache miss)，Oracle9i就會先從磁碟上資料檔案讀出適當的資料區塊，放入緩衝區之後，使用者才從緩衝區讀取資料。您可以想像一下：在“cache hit” 的情況下查詢資料的速度是不是比在 ”cache miss” 的情況還快很多呢？事實上，這就是資料快取緩衝區的主要用途所在。

讓我們進行更深入的討論！

當資料區塊從磁碟讀出，準備放入緩衝區時，系統必須先確定資料快取緩衝區內有free buffers。這時候Oracle9i 會開始掃描 LRU list，掃描的原則為：

- 從 LRU 端掃到 MRU 端
- 當掃描到 free buffer；或是已掃描的緩衝區數目超過臨界值時，就會停止掃描動作
- 掃描 LRU list 時如果發現了dirty buffer ，就將它移到 write list ，然後繼續掃描

如果掃描過程順利在LRU list 內找到 free buffer ，那麼Oracle9i就會把從磁碟讀出的資料區塊放入此 free buffer中，然後再把它移到 LRU list 的 MRU 端。

但是，如果 LRU list 真的都沒有free buffer 怎麼辦呢？那麼Oracle9i 就會停止掃描動作，然後通知資料庫寫入器(database writer)背景處理程序將部分 dirty buffers 先寫入磁碟，接著從 LRU list 的LRU 端開始清除緩衝區。如此一來就可以空出新的 free buffer 了。

Tips:

LRU list 與 LRU 演算法

所謂 LRU (least recently used)演算法之基本概念為：當記憶體內剩餘可利用的空間不足時，緩衝區盡可能先保留使用者最常使用的資料；換言之，優先清除”較不常使用的資料”，並釋放其空間。我以圖示方式為大家說明：

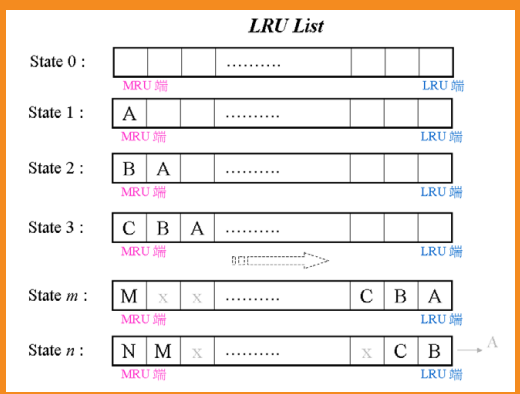

圖 2：LRU list 第一種使用情況。

我們將 LRU list 想像成是長條型一連串的緩衝區集合，兩端點分別為「MRU 端」以及「LRU 端」。所謂的「MRU」為「Most Recently Used」之縮寫，我將之譯為"最常使用的"(或是"最近使用的")。所以，愈靠近 MRU 端的緩衝區，代表其為被使用者最近查詢過的資料。同理，我們將「LRU」端視為"最不常被查詢的資料(Least Recently Used)"，或是"很久都沒被再次查詢的資料"。如上圖2中的 State 0 所示：(假設目前 LRU list 沒有任何資料存放)

圖2的State1、State2、State3模擬某個使用者欲從Oracle9i資料庫查詢出三筆資料(A資料、B資料、C資料)。這三筆資料從磁碟讀出後，依序放入LRU list的free buffers。以State3為例：因為C資料是"最近剛剛使用"的資料，所以最靠近MRU端；相較之下，A資料是"有一段時間都沒用的資料"，所以比較靠近LRU端。依此類推，如果使用者持續地查詢資料或進行相關資料處理動作，則LRU list內的緩衝區都會被填滿，如圖2的State m所示：M資料填滿最後一個free buffer。
如果下一次的查詢動作準備再從磁碟讀出N資料時存入緩衝區時，LRU list 已經沒有 free buffer 可以使用，系統只好清空 LRU 端存放 A 資料的緩衝區，接著再放入 N 資料，結果如圖 2State n 所示。這就是 LRU 演算法的基本運作方式。

讓我們討論另一種情況：
假設在State3之後，恰好有某些使用者持續地查詢A資料—這會導致 A 資料一直存放在靠近 MRU 端的緩衝區。如果 M 資料被放入 LRU list 最後一個緩衝區，結果將如圖 3 的 State m' 所示：您會發現圖 3 的 State m' 與圖 2 的 State m 有點不同(緩衝區的存放資料完全相同，但存放位置不盡相同)。這時候LRU list內也沒有 free buffer了，所以再放入N資料時，系統會清除LRU端存放B資料的緩衝區。不同於第一種情況的是：因為"A 資料查詢率高於 B 資料"，所以 LRU 演算法讓A資料在緩衝區存放較長的時間，而先移除"較不常使用"的B資料。

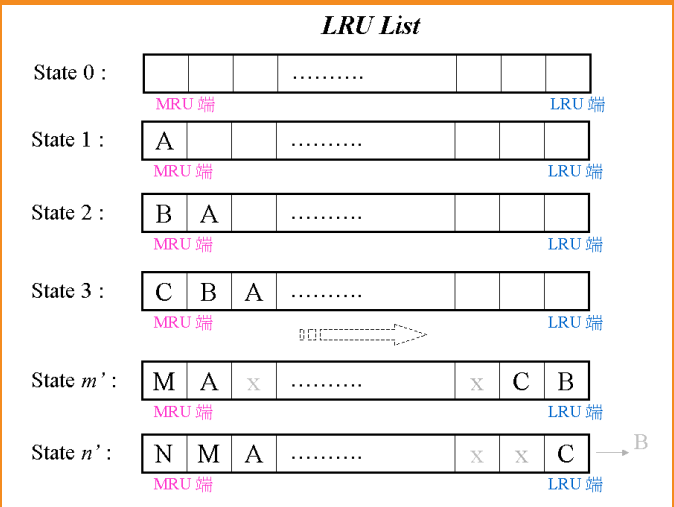

圖 3：LRU list 第二種使用情況。

設定資料快取緩衝區之大小

在 Oracle9i 設定資料快取緩衝區的方式有別於 Oracle8i 資料庫。Oracle8i 的資料快取緩衝區容量由下列公式所決定：

Oracle8i 資料快取緩衝區大小 =
DB_BLOCK_SIZE * DB_BLOCK_BUFFERS

DB_BLOCK_SIZE:
資料區塊(data block)之單位大小。

DB_BLOCK_BUFFERS:
緩衝區數目；每個緩衝區大小相當於一個資料區塊之大小。

需注意的是Oracle8i內DB_BLOCK_SIZE在資料庫建立之後就不能更改。

然而Oracle9i資料庫支援多重區塊大小 — 除了預設的DB_BLOCK_SIZE之外，DBA也可以另外設定其他大小的資料區塊。因此在Oracle9i資料庫中由DB_BLOCK_SIZE所設定的資料區塊，我們稱為「標準資料區塊(standard block)」，其合理的大小可設定在2k – 32k 之間。Oracle9i的起始參數DB_CACHE_SIZE就是設定以"標準區塊"所構成的資料快取緩衝區之容量。

重置日誌緩衝區

紀錄Oracle資料庫內所有資料異動的詳細資訊(例如資料異動前後的新舊資料)，我們將這些資訊的儲存地點稱為redo entries。系統也會在適當時機將redo entries內的資訊寫入磁碟內的檔案(註5)，以便資料庫毀損時可進行必要的復原(Recovery)動作。

(註5)
由LGWR背景處理程序負責將redo entries寫入重置日誌檔(Redo log files)。

設定重置日誌緩衝區的大小

起始參數檔之LOG_BUFFER參數可用來設定此緩衝區的容量，單位為bytes。一般來說，如果重置日誌緩衝區的容量較大，可減少日誌檔讀寫動作。重置日誌緩衝區的預設值等於作業系統資料區塊的四倍。

共享區

當使用者將SQL指令送至Oracle資料庫後，系統將會先解析(parse)語法是否正確。解析時所需要的系統資訊，以及解析後的結果(parse tree與execution plan)將放置在共享區內。如果不同的使用者執行了相同的SQL指令，就可以共享已解析好的parse tree與execution plan，加速SQL指令的執行速度。

共享區之組成

共享區內包含數種不同用途的快取緩衝區，主要可分為兩類：

- 函式快取緩衝區(Library cache)

包含：共享 SQL 區(Shared SQL Area)，私有 SQL 區(Private SQL Area)，以及PL/SQL程式單元區。解析完成的parse tree與execution plan就是放在共享 SQL 區內。

■ 資料字典緩衝區(Dictionary cache)。
Oracle9i 在解析 SQL 敘述句時所需要的系統資訊都是放在此緩衝區內，可能包含：資料表(視觀表)的名稱、欄位名稱與資料型態；資料庫使用者之相關系統管理權限，或是物件存取權限佈央 C

設定共享區之大小

起始參數檔內的SHARED_POOL_SIZE可設定共享區之大小，其預設值大小為 8M(註 6)。

(註 6)
在 32-bit 作業系統下，共享區預設大小為 8M；如果在 64-bit 作業系統，共享區預設大小為 64M。

背景處理程序

除了 S G A 之外，系統也會自動啓動數個特定的背景處理程序(Background Processes)，主要的背景處理程序為：

■ DBWn (Database Writer)
負責將資料快取緩衝區內異動過的資料區塊回寫至硬碟內的資料檔案(這個動作又稱為 checkpoint)。Oracle 系統預設只會啓動一個 DBWn(DBW0)處理程序。但在一般的大型線上交易(OLTP)系統下，資料庫異動情況可能十分頻繁，您可依實際需求額外配置其他的Database Writer 處理程序(DBW1-DBW9)，可以有效地提昇 Oracle9i 寫入資料檔案之效率！

有一點需要注意的是：在單一處理器的伺服器系統，配置額外的 DBWn 處理程序並無實質幫助。

■ LGWR (Log Writer)
在交易被確認時，LGWR 會遵循「先期寫入協定」，負責將重置日誌緩衝區內的資料異動紀錄循序寫入重置日誌檔。

LGWR 之動作時機為：
1. 當使用者確認(commit)某交易時，LGWR會寫入一筆確認紀錄。
2. 下列幾種情況：
   ■自動週期性地動作，間隔時間為 3 秒
   ■重置日誌緩衝區之剩餘空間不到 2/3
   ■當 DBWn 回寫資料檔時，必要的時候 LGWR 也會動作

Tips:
何謂「先期寫入協定(Write-ahead protocol)」？
在DBWn將dirty buffer回寫至資料檔之前，重置日誌緩衝區內相關的都必須完成寫入動作。如果DBWn發現某些重置紀錄尚未寫入重置日誌檔，它也會通知LGWR前來處理。等到LGWR將重置日誌緩衝區的紀錄寫入完畢時，DBWn才會開始寫入資料檔—此即為「先期寫入協定」。

■ SMON (System Monitor)
如果是因為停電或是其他因素導致Oracle資料庫不正常被關閉，下

一次啓動資料庫時將由 SMON 進行必要的資料庫修復動作。其主要工作有：

- 資料庫重新啓動後的必要修復動作
- 回收不用的暫時性區段(temporary segments)
- 收集詴r典管理資料表空間茯榻ree extents
- 修復交易不正常結束造成的問題

PMON (Process Monitor)

當某個使用者處理程序異常終止時，PMON 會執行「程序修復(process recovery)」動作 — 清除資料快取緩衝區內不再使用的空間，並釋放該程序之前使用的系統資源。舉例來說，PMON 會重新設定transaction table的狀態，釋放原交易被鎖定的資料，然後從程序清單中移除已終止之程序代號(process ID)。

PMON也會定期檢查各伺服器處理程序以及分配器之狀態，如果某個處理程序因故停擺，也是由 PMON 負責將它重新啓動。

CKPT (Checkpoint)

CKPT 會在適當時候產生一個 checkpoint 事件，其意義為：

1. 確保緩衝區內經常被異動的資料也要定期被寫入資料檔。

   對於這點，也許大家會覺得非常疑惑。Oracle9i的LRU演算法不是會盡量將使用者"最常使用的資料"保留在緩衝區內，以提昇資料存取的效率嗎？沒錯，但請大家思考一個問題：如果完全遵循 LRU 演算法的話，DBWn 只會將 " 最不常使用 " 的資料回寫至資料檔，這些"經常被使用"的資料反而沒機會存回硬碟！要是資料庫當機或毀損，這些資料只能從重置日誌檔的紀錄才能還原回來，無形中造成系統的額外負擔。所以由 CKPT 掌控 checkpoint 時機，以確保這些資料照樣可被妥善儲存。

2. 在checkpoint之後，因為所有更新過的資料已經回寫至磁碟資料檔案，萬一 Oracle9i 需要進行 instance recovery 時，就不再需要checkpoint之前的重置紀錄，可縮短資料庫重新啓動的時間。

   此外，checkpoint 發生後，CKPT 會先通知DBWn將資料快取緩衝區的dirty buffers回寫到資料檔案，然後更新資料檔案與控制檔之 checkpoint 資訊。

RECO(Recover)

在Oracle9i分散式資料庫環境中，RECO處理程序會自動處理分散式交易失敗時產生的問題。何謂分散式交易呢？簡單的說，就是在同一個交易內針對多個資料庫同時進行資料處理動作。與傳統資料交易相較之下，分散式交易的要求將嚴苛許多！因為這些資料庫可能分散在網路上，所以分散式交易的成功與否其中一個關鍵因素就在於網路傳輸速度與品質。試想，某個分散式交易需要將資料同時送進5個Oracle9i資料庫，此時可能因為網路問題(例如斷線或是回應等待時間過長)，導致其中一個資料庫無法更新這筆交易資料。那麼此交易就會發生問題，無法做最後的commit動作；我們將此種交易稱為in-doubt transaction。這時候RECO處理程序會自動重新登入包含在分散式交易之各資料庫，並嘗試進行 in-doubt transaction 之修復動作。（註 7）

（註 7）

如果起始參數檔 DISTRIBUTED_TRANSACTIONS 參數值設定為 0 ，則啓動 Oracle Instance 時不會執行 RECO 處理程序。

ARCn(Archiver)

Oracle9i 資料庫設定為 ARCHIVELOG mode(封存日誌模式)時，ARCn 處理程序會在 "log switch" 發生時自動將重置日誌檔複製一份到指定的目錄下；重置日誌檔之副本稱為 "Archived logs"(封存日誌檔)。每個 Oracle9i instance 最多可啓動 10 個 ARCn 處理程序(ARC0 to ARC9)。

起始參數檔之 LOG_ARCHIVE_MAX_PROCESSES 允許您組態 ARCn 處理程序之個數；或是執行 ALTER SYSTEM 指令動態地調整。該參數值之預設值為 1 。事實上，您並不需要設定此參數，因為在封存模式下 Oracle9i 會自動判斷所需的 ARCn 個數。一但目前的 ARCn 處理程序

Tips

何謂 Log switch?

當目前的重置日誌檔被填滿時，LGWR 會切換到下一個可使用的重置日誌檔，這個日誌檔的切換動作就稱為 「log switch」。預設情況下，log switch 會自動發生。

無法負荷時，LGWR 會自動啓動新的 ARCn 處理程序。

除了以上七個重要的背景處理程序之外，Oracle9i資料庫運作時還有其他的背景處理程序互相搭配及運作，例如 Job Queue 、LMS 、QMNn 等處理程序。如果您想獲取進一步的資訊，可參閱 OTN 網站的《Oracle9i Database Concepts》。

結語

本期內容介紹了Oracle9i的記憶體結構與其運作原理，下一期我將從資料庫與作業系統兩個角度深入討論Oracle9i資料庫之實體結構，與常用的資料庫物件。

作者簡介

何致億

恆逸資訊系統開發部技術顧問。專長為 SQL Server 、Oracle 等關聯式資料庫系統管理，資料倉儲規劃建置，以及資料庫應用程式系統開發。擁有 MCSD 、MCDBA ，Oracle OCP ，RHCE ，SCJP ，Borland JBuilder Product Certified 等十餘項國際認證。目前正致力於 Oracle9i 應用系統開發，並負責 Oracle9i 系列書籍中文化及 Oracle Press 技術校稿工作。

曾任台灣微軟E-Developer、TechEd 2000 、Enterprise Server 2000 上市博覽會等大型研討會講師，Windows 2000雜誌、Linuxer雜誌專欄作者，SQL Magazine國際中文版編輯顧問。他同時也是美商甲骨文公司、昇陽公司等原廠認證講師。您可以透過rich_ho@uuu.com.tw與他聯繫。