讀取資料

* 檢查Buffer cache(緩衝區快取)是否有所有需的資料
* 無，將硬碟檔(db01.mdf)的資料寫到Buffer Cache(Physical Read) 實體讀取
* 讀取Buffer Cache的資料(Logical Read)-邏輯讀取

RAM很重要，暫存的概念。

寫入資料（紅色是寫入的階段）

* 檢查要異動額資料是否被鎖定(locked)

是，等待鎖定被解放(lock released)

* 鎖定要異動的資料範圍(Lock acquired)
* 將異動寫入交易記錄檔(db01\_log.ldf)
* 異動Buffer Cache上的資料，表示為dirty
* 釋放鎖定
* 將dirty pages寫入資料儅（db01.mdf）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | DataType | Filegroup | Size | Growth |
| db01  db01\_log | Rows data  Log | PRIMARY  Not available | 8MB  8MB | 4MB  4MB |
| db01\_before  db01\_2006  db01\_2007  db01\_2008 | Rows Data  Rows Data  Rows Data  Rows Data | TR\_BEFORE  TR\_2006  TR\_2007  TR\_2008 | 8MB  8MB  8MB  8MB | 4MB  4MB  4MB  4MB |
|  |  |  |  |  |

Shrink Database: 將資料庫未使用的空間釋放給作業

Compress Database: 改變資料資料儲存的讀算邏輯，減少載資料庫空間的需求

Fragmentation(資料破碎)

**查詢的時候有兩種行爲：**

1. Scan （掃描）sum, group by 對於壓縮效率較大
2. Seck （搜尋）對於壓縮效率較小

Binary Tree 二元樹

Node 節點

Parent Node 父節點

Child Node 子節點 （最多2個，最少0）

Intermediate Node 中間節點

Root Node 根節點

Leaf Node 葉節點

Perfect Binary Tree: 除最不面的葉節點，沒有子節點所有節點都有2個子節點

Complete Binary Tree 最不面2層外所有節點有2個子節點，最不面一層往左爹

SQL 🡪 DB Engine (SQL Parser（器）(1)檢查SQL語法是否正確，（2）檢查SQL語法中的物件名稱，欄位名稱是否正確 及SQL Optimizer優化處理器(1)利用資料庫中的統計物件(statistics)優化SQL羽凡 (2)產生執行計劃(Execution plan) 🡪SQL OS（1）根據執行計劃編譯為可執行檔（2）執行程式，存取資料

|  |  |
| --- | --- |
| Rebuild Index (重建索引) | Reorganize（重整索引） |
| 重新排列整個個B+ Tree結構 | 只重新排列Leaf-Pages |
| 企業版支援on-line執行，其他版本必須off-line執行 | 所有版本都是online執行 |
| 執行中斷必須Rollback回未執行之前 | 執行中斷不必rollback |
| 會自動更動統計物件(statistics) | 不會更新統計物件 |
| 可以變更index參數（如 Fillfactor | 不能變更Index參數 |
| Fragmentation >=30% | Fragmentation 10%~30% |
| 有停機作業,小型db | 7天24小時，大型db |