**[Oracle Select SQL Tuning 查詢效能調整](http://tomkuo139.blogspot.tw/2009/08/oracle-select-sql-tuning.html)**

前提: 以下的幾種檢查方式，都是我的經驗談，但不管哪一種檢查方式，都是為了 "減少資料筆數的讀取，以加快查詢速度"。

檢查 1: **建立 Index Key**

大多數的查詢速度慢，利用新增 Index Key，效率上大都會有一定地提升。  
  
曾經有查詢 1 個多小時，在 Order By 所需的欄位上建立 Index，之後查詢不到 10 秒鐘 (差很大)。  
  
若建立的 Index 為 ( key1, key2, key3 )，而 where 條件是 (key1)，則此 Index 會有作用。  
  
若建立的 Index 為 ( key1, key2, key3 )，而 where 條件是 (key2, key3)，則 10g 以後此 Index 會作用；但 10g 以前此 Index 不會作用，需另建 (key2, key3) Index 才可。

檢查 2: **Where 條件中，是否用函數**

Where 條件的欄位若使用函數，會讓建立在欄位的 Index Key 無法使用，如: nvl, decode, trunc, to\_char, ... 等，儘量避免在欄位上使用，如 nvl( col, 'Y' ) = 'Y' 可改用 (col = 'Y' or col is null)。

檢查 3: **Where 條件用 計算公式，會使建立在欄位的 Index Key 無法使用**

較慢範例: where (qty\*10) > 1200  
  
較快範例: where qty > (1200 / 10)

檢查 4: **Where 條件中，是否使用 In / Not In / Exists**

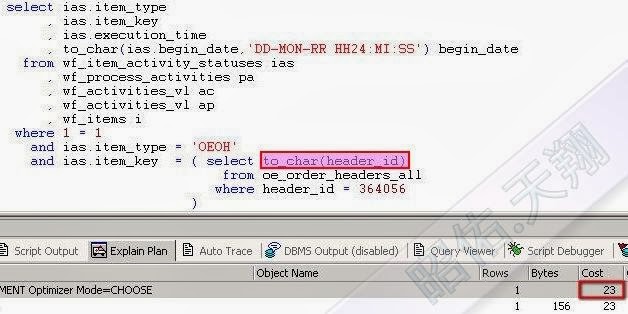
使用 In / Not In / Exists，有可能會使查詢所需的 Cost 增加，若真要使用，這三種方式可用 explain 進行偵測，看目前 SQL 在哪一種方式較好。

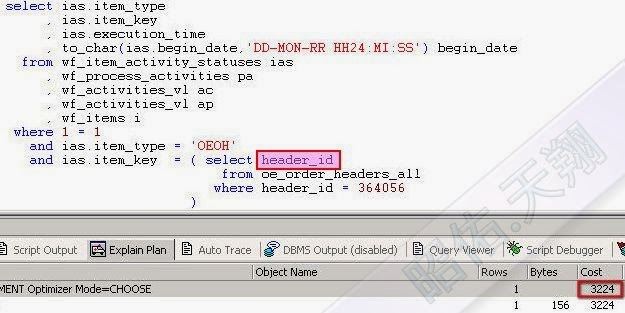
檢查 5: **Where 條件中，是否使用 Outer Join**

使用 Outer Join，會導致查詢所需的 Cost 增加，解決方式: 分成兩個 SELECT，一個 SELECT 用 Join，一個 SELECT 用 IS NOT NULL，然後用 Union or Union All 加起來。  
(曾經有查詢 3 分多鐘，在改成 Union All 之後，查詢不到 10 秒鐘)  
  
另一種方式，就是將 Outer Join 的 Table，做為 Subquery 當作 Columns 使用。  
(這方式有時效能反而更差，需與 Outer Join 方式，測試比較看看)

檢查 6: **Where 條件中，是否比較不同的資料型態**

這個有可能在速度上會差異很多，如: charCol = numCol 改用 charCol = to\_char(numCol)

[](http://3.bp.blogspot.com/-OlGQMLi0R3w/UwroG6xMJwI/AAAAAAAAuto/SgANrOOKnj4/s1600/001.jpg)

[](http://3.bp.blogspot.com/-hBeDFQgi5H4/UwroGmYl9rI/AAAAAAAAuts/sI9xOqTtawY/s1600/002.jpg)

檢查 7: **Where 條件中，若使用變數，可能讓建立在欄位的 Index Key 無法使用**

一般來說，是不會發生這種情形，但萬一遇到了，解決方式為: 接收到實際值後，將變數轉換成實際值，然後再執行之。  
  
詳細資訊，請見 Oracle Metalink <http://metalink.oracle.com/metalink/plsql/tar_main.this_tar?tar_num=5363227.993&p_ctryCode=840>.

檢查 8: **是否使用 Distinct / Order By**

使用 Distinct / Order By 會使查詢所需的 Cost 增加，若可以不用，盡量少用之。

檢查 9: **是否使用 View 有些情況可能會影響速度**

如 View 關聯了 7 個 Table，但實際需求只要關聯 2 個 Table 的欄位資料，多餘的 5 個 Table 關聯，只會影響查詢速度。

檢查 10: **一個 Select 的效能 > 多個 Select 的效能**

解決方式: 將多個 Select 改用 Union or Union All or Subquery。  
(曾經有查詢 5 分多鐘，在改成 Subquery 之後，查詢不到 10 秒鐘)

檢查 11: **多個 TABLE 同時關聯**

多個 TABLE 同時關聯，造成查詢速度太慢，解決方式: 可以先將一個過濾資料 Table 作成 Subquery，然後再將 Subquery 與其他 Table 關聯。  
(曾經有查詢 6 分多鐘，在改成 Subquery 之後，查詢不到 1 分鐘)

檢查 12: **查詢條件的增加 or 取回資料的範圍縮小**

用意當然在於減少取回的資料量，減少 Database Buffer 的使用量。

檢查 13: **資料量的增加，會使所需的 Cost 增加**

這個我想，每個人都知道的，資料量一直不斷地增加，導致查詢速度越來越慢。  
  
解決方式: 減少資料量的取回 or 將資料搬到歷史檔 or 擴充硬體設備。

檢查 14: **將常查詢但很費時的資料，建立 Materialized View or Summary Table**

利用晚上較沒人使用的時間，將查詢要很久時間的資料，收集起來，就可快速查詢。

檢查 15: **查詢來源是否有用到 DB Link**

資料量不多，有時查詢很快 (不用 1 秒鐘)，有時查詢很久 (幾十分鐘)，看看同一個 Select SQL 是否有用到 DB Link，  
  
將這個 DB Link 資料來源，在一開始先寫入到同一個 DB 的 Temp Table，再由這個 Temp Table 來查詢。

檢查 16: **將不常用的 Inactive Data，放在其他 Tablespace or Database 中**

這種做法，目的就是減少空間，增加存取效能。

檢查 17: **建立 Partition Table**

這種做法，就是區隔資料，把它想像成 Index 就容易理解了。

檢查 18: **若 Select 是採用 Rule Base，考慮 From Table 順序 / Where 條件順序**

Oracle 查詢方式有兩種：

* **Rule Base**：擺放在 from、where 的位置，**會**影響查詢速度。
* **Choose Base**：擺放在 from、where 的位置，**不會**影響查詢速度。

若 Select 是採用 Rule Base，則擺放在 from / where 的位置，就會影響查詢速度。

檢查 19: **在 Select 之後 /\*+ ... \*/**

有時 Oracle 分析 Select 語句後，判斷有誤而採用錯誤的方式，如該用 Index 卻沒使用，這時就自力自強吧, 而常用的有 :  
/\*+ INDEX (tableName indexName) \*/  
/\*+ RULE \*/  
/\*+ CHOOSE \*/  
/\*+ FIRST\_ROWS \*/  
/\*+ ALL\_ROWS \*/  
  
如: Select /\*+ RULE \*/ colName1, colName2, ... from ...  
  
其中  
1) /\* 與 + 之間無空格，否則只能視為備註，而非提高效能方式。  
2) 若 Table 有用 Alias Name，則也請用 Alias Name，否則無法強制使用 Index。

檢查 20: **Explain Plan**

參考 Oracle 提供的 Explain Plan 分析 SQL 的執行路徑，看哪種 SQL 寫法 Cost 時間較少。  
  
注意: 有時 Cost 少不代表實際查詢速度快喔。

檢查 21: **資料被 Locked**

不用懷疑，有時資料的 Locked，可能造成查詢速度變慢。

檢查 22: **資料量不多，但卻查詢要很久的時間**

曾經遇過查詢 6 筆卻要查詢 3 分鐘之久，原因很簡單，此 Table 常做新增 / 刪除動作，導致雖然資料量不多，但資料分散太大。  
  
解決方式: 重建 Table。

檢查 23: **找 DBA 協助**

若上述方式都用過了，查詢速度仍然很慢，找 DBA 吧 !!

檢查 24: **改變程式邏輯或系統架構**

最後，連找 DBA 也沒辦法的話，若就請思考其他方式，更改程式邏輯或修改系統架構吧。