优化 Visual Basic 查询的方法

姚卫新　（中国纺织大学无锡校区　214063）

摘要：Visual Basic 允许用户使用结构化语言(SQL)从 Jet databases (MDB 文件)中取出数据，本文提出了一些优化查询操作的方法。

关键词：查询　SQL　优化

1. 引言

在 Visual Basic 程序设计中不可避免地要用到查询，如何提高查询的速度和效率是每一位软件人员不断探索的问题。各种方法在书籍和文章中零星的有所报道，但不够全面。本人在总结了多年的使用心得后全面介绍了查询优化的方法。

2. 优化方法

·表设计

在设计表中字段时，尽量选择长度较短的数据类型，这样可以增加每页中存放的记录个数。连接(JOIN)使用的字段必须有相同或兼容的字段。

·压缩数据库

压缩数据库有两点好处：

(1)随着数据库中数据量的增加，原有的优化方案可能不再有效，压缩数据库可以最大限度地克服这个缺点。

(2)随着数据库中数据量的增加，它就变得比较“零碎”，压缩数据库可以把一个表的数据集中到硬盘的连续页中，提高顺序扫描的执行速度。压缩数据库的例子如下(它将保留一个备份)：

```
DBEngine.CompactDatabase"C:/VB/BIBLIO.MDB","C:/VB/BIBLIO2.MDB"
Kill"C:/VB/BIBLIO.BAK"
Name" C:/VB/BIBLIO > MDB" As" C:/VB? BIBLIO > BAK"
Name" C:/VB/BIBLIO2. MDB" As " C:/VB/BIBLIO.MDB"
```

如果数据库被频繁更新，最好考虑经常压缩数据库。

·查询输出中尽量避免表达式

某个带有表达式的查询输出作为另一个查询的输入时，可能会产生问题。如下例，查询 1 是第二个 SELECT 语句的输入：

```
Dim DB As Database
Dim RS As RecordSet
Set DB = DBEngine.Workspaces(0).Opendatabase("Biblio.MDB")
DB.CreateQueryDef("Query1",SELECT IIF(Au_ID=1,'Hello','Goodbye')AS X FROM Authors")
Set RS = DB.OpenRecordSet("SELECT * FROM Query1 WHERE X="Hello'")
```

查询 1 中的 IIF()表达式不能被优化，所以第二个 SELECT 语句中的 WHERE 条件也不能被优化，所以整个查询就不能优化。可以合并为一个嵌套的 SQL 语句：

```
Set RS = DB.OpenRecordSet("SELECT * FROM Authors WHERE Au_ID=1")
```

对于复杂的嵌套查询，把字段名直接写在表达式中：

```
DB.CreateQueryDef("Query1","SELECT IIF(Au_ID=1,'Hello','Goodbye') AS X,Au_ID,FROM Authors")
Set RS=DB.OpenRecordSet("SELECT * FROM Query1 WHERE Au_ID=1")
```

如果在查询输出中无法避免计算，把计算放在上层查询中，不要放在下层查询中。

·仅输出需要的字段

建立查询时，不需要返回值的字段不要包含在 SELECT 语句中。

·分组、连接和累计

连接两个表时，如果分组和累计字段名相同，确保分组字段累计(Sum，Count 等)字段来自同一个表。

例：下面的查询效率较低，因为 SUM 的字段在 Ord 表中，而 GROUP BY 的字段在 Cust 表中。

```
SELECT Cust.CustID,
  FIRST(Cust.CustName) AS CustName,
  SUM(Ord.Price) AS Total
FROM Cust INNER JOIN Ord ON Cust.CustID = Ord.CustID
GROUP BY Cust.CustID
```

例：下面的查询效率较高，因为 SUM 和 GROUP BY 的字段均为 Ord.CustID，它们来自同一个表。

```
SELECT Ord.CustID,
  FIRST(Cust.CustName) AS CustName,
  SUM(Ord.Price) AS Total
FROM Cust INNER JOIN Ord ON Cust.CustID = Ord.CustID
GROUP BY Ord.CustID
```

·分组字段越少越好

GROUP BY 中的字段越多,查询花费的时间越长,使用 FIRST()累计函数有助于减少 GROUP BY 语句中字段的个数。

效率低的例子:

```
SELECT CUst.CustID,
    Cust.CustName,
    Cust.Phone,
    SUM(Ord.Price) AS Total
FROM Cust INNER JOIN Ord ON Cust.CustID = Ord.CustID
GROUP BY Cust.CustID,Cust.CustName,Cust.Phone
```

效率高的例子:

```
SELECT Ord.CustID,
  FIRST(Cust.CustName) AS CustName,
  FIRST(CUst.PHone) AS Phone,
  SUM(Ord.Price) AS Total
FROM Cust INNER JOIN Ord ON Cust.CustID = Ord.CustID
GROUP BY Ord.CustID
```

·在连接前嵌套 GROUP BY 语句

如果连接两个表并且用其中一个表中的字段进行分组,把该 SELECT 语句分解为两个查询语句更为高效。把带有 GROUP BY 子句的 SELECT 语句作为一个嵌套查询加入到上层查询的非分组表中。低效例子:

```
SELECT Ord.CustID,
  FIRST (Cust.CustName) AS CustName,
  FIRST(Cust.Phone) AS Phone,
  SUM(Ord.Price) AS Total
FROM Cust INNER JOIN Ord ON Cust.CustID = Ord.CustID
GROUP BY Ord.CustID
```

高效例子:

查询 1:

```
SELECT CustID,SUM(Price) AS Total
FROM Ord
GROUP BY CustID
```

查询 2:

```
SELECT Query1.CustID,CUst.CustName,Cust.Phone,查询 1.Total
FROM Cust INNER JOIN Ord ON Cust.CustID = Ord.CustID
```

·索引 JOIN 子句的字段

用 JOIN 连接表时,索引 JOIN 子句中用到的字段后,查询优化器将使用比较高级的内部连接策略来优化查询语句。

如果有一个表相对比较小(占 1-2 页面),就不必要对该表建索引。这样读到内存中的页比较少,所以是否对表建索引要看情况而定。

·使用可优化的表达式

尽量建立可以使用 Rushmore 技术进行优化的查询,Rushmore 并不会自动地加速所有的查询。

·用 COUNT(*)代替 COUNT([字段名])

微软的数据库引擎对 COUNT(*)特别关照,使得 COUNT(*)执行时比 COUNT([字段名])快得多。另外这两个操作略有不同:COUNT(*)计算返回的所有行,COUNT([字段名])仅计算该字段名不为 NULL 的行。

·参数中避免 LIKE

由于查询在编译时参数的值尚未确定,索引不起作用,最好在 SQL 语句中写明参数的值。

·避免 LIKE 中的通配符在开头

如果使用 LIKE 操作符来寻找匹配的行,通配符不能在开头位置,否则索引不起作用。下面的例子索引起作用:

Like "Smith",Like "Sm*"

下面的例子索引不起作用:

Like"*sen"

Like"*sen*"

·使用中间结果表(临时表)

使用语句建立临时表,尤其是当查询结果要被其他的查询使用时,尽量使用临时表,其效果非常显著。

·避免 NOT IN 与子查询一起使用

NOT IN 与子查询一起使用效果非常差,最好将其转化成嵌套查询或使用 OUTER JOIN。

下面的例子是查找没有订过货的客户:

低效查询:

```
SELECT Customers.*
    FROM Customers
    WHERE Customers.[Customer ID]
      NOT IN (SELECT [Customer ID] FROM Orders);
```

高效查询:

```
SELECT Customers.*
FROM Customers LEFT JOIN Orders
  ON Customers.[Customer ID]=Orders.[Customer ID]
WHERE ((Orders.[Customer ID]Is Null));
```

参考文献

[1] P. McManus J. 使用 Visual Basic 5.0 编程,电子工业出版社,19998,110-121

[2] Brierley E,Prince A,Rinald D. Visual Basic 6.0 开发人员指南,机械工业出版社,1999,445-464

[3] Siler B,Spotts J. Visual Basic6.0 开发使用手册,机械工业出版社,1999,389-394

(来稿时间:1999 年 8 月)