SeQueL 9 - Nested SubQueries

by Clark Anderson

A SubQuery is a query nested within a query. I first encountered them in Oracle SQL.

The SubQuery must be inside parentheses. Most examples of SubQueries, I have seen, have been in the WHERE clause. e.g.:

```
SELECT MenuDescription, Price
 FROM tblMenu
 WHERE Price >
  (SELECT Price FROM tblMenu WHERE MenuDescription = 'Baked Ziti' )


SELECT MenuNo, MenuDescription, Price, Discontinued
 FROM tblMenu
 WHERE NOT EXISTS
  (SELECT * FROM tblOrderDetails WHERE MenuNo = tblMenu.MenuNo )

SELECT * FROM tblImport
 WHERE OrderNo IN
  (SELECT OrderNo FROM tblImport GROUP BY OrderNo HAVING Count(*) > 1 )
 ORDER BY OrderNo
```

For years I have built queries that that worked with the data from other stored queries. e.g.:

```
SELECT Year, Sum(Gasoline) AS SumOfGasoline
 FROM qExpTransp
 GROUP BY Year
```

qExpTransp was defined as:

```
SELECT Year,  Pymt AS Gasoline
 FROM [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary
 WHERE LEFT(CatDescr,9) IN ('AUTO-GAS ')
```

I discovered that I could make it all one query by incorporating the referenced as a SubQuery in the FROM clause e.g.:

```
SELECT Year, SUM(Gasoline) AS SumOfGasoline
 FROM
  (SELECT Year, Pymt  AS Gasoline
   FROM  [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary
   WHERE  LEFT(CatDescr,9) IN ('AUTO-GAS ') )
 GROUP BY Year
```

This can be extended to Statistics:

```
SELECT MIN(SumOfGasoline) AS MinAnnualGasExp,   FORMAT(AVG(SumOfGasoline),'#000.00') AS
MeanAnnualGasExp,   MAX(SumOfGasoline) AS MaxAnnualGasExp
FROM
 (SELECT Year, SUM(Gasoline) AS SumOfGasoline
  FROM
   (SELECT Year, Pymt  AS Gasoline
    FROM  [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary
    WHERE  LEFT(CatDescr,9) IN ('AUTO-GAS ') )
  GROUP BY Year )
```

Next we will nest subqueries in the SELECT clause to bring together a value from the current record and the corresponding value from the previous record:

```
 (SELECT CatDescr, PaidToDescr, PaidDate AS LastDate ,
```

The FROM tables are given aliases: Exp1 in the inner subquery and Exp0 in the outer subquery. The inner subquery compares values of PaidDate from the inner Exp1 table and the outer Exp0 table:

```
  (SELECT TOP 1 PaidDate
   FROM  [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary AS Exp1
   WHERE (CatDescr & PaidToDescr LIKE '*haircut*elite*')
    AND Exp1.PaidDate < Exp0.PaidDate
   ORDER BY PaidDate DESC )
```

The single value returned by the inner subquery is given an alias:

```
  AS PrevDate,
```

The outer subquery SELECT clause continues with an additional value, which is calculated from a value in the outer Exp0 subquery table and the value returned from the inner Exp1 subquery table. The CSng() function is used to convert the formatted values, WeeksBetw, into Single Floating Point Values.:

```
  CSng(FORMAT(DateDiff ('d', PrevDate, LastDate )/7,'#0.0')) AS WeeksBetw
```

The remainder of the outer subquery is:

```
 FROM  [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary AS Exp0
 WHERE (CatDescr & PaidToDescr LIKE '*haircut*elite*')
  AND PaidDate > #7/6/1996#
 ORDER BY PaidDate DESC )
```

Statistics are derived from wrapping these subqueries in an outer aggregating query. :

```
SELECT CatDescr, PaidToDescr, MIN(WeeksBetw) AS MINWeeksBetw, FORMAT(AVG(WeeksBetw),'#0.0') AS
MEANWeeksBetw,
MAX(WeeksBetw) AS MAXWeeksBetw FROM
  :
  :
GROUP BY CatDescr, PaidToDescr
```

This is the final query:

```
SELECT CatDescr, PaidToDescr, MIN(WeeksBetw) AS MINWeeksBetw, FORMAT(AVG(WeeksBetw),'#0.0') AS
MEANWeeksBetw,
MAX(WeeksBetw) AS MAXWeeksBetw FROM
 (SELECT CatDescr, PaidToDescr, PaidDate AS LastDate ,
  (SELECT TOP 1 PaidDate
   FROM  [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary AS Exp1
   WHERE (CatDescr & PaidToDescr LIKE '*haircut*elite*')
    AND Exp1.PaidDate < Exp0.PaidDate
   ORDER BY PaidDate DESC ) AS PrevDate,
  CSng(FORMAT(DateDiff ('d', PrevDate, LastDate )/7,'#0.0')) AS WeeksBetw
 FROM  [D:\HseHold\Fin\ExpSmry.mdb].ExpenseSummary AS Exp0
 WHERE (CatDescr & PaidToDescr LIKE '*haircut*elite*')
  AND PaidDate > #7/6/1996#
 ORDER BY PaidDate   DESC )
GROUP BY CatDescr, PaidToDescr
```

What kind of fun can you have nesting your subqueries to solve your problems?