Introduction

This paper presents five challenges for your data management tools. Can you quickly and easily solve these problems using one or more of the tools in your data management toolkit? Solutions to each of the problems is given using V4®, a revolutionary new database tool, developed and supported by MKS Inc.

All of the problems presented are based on relational tables taken from actual applications. Only the necessary columns required for each particular challenge are listed. It is assumed that the data is in relational form. If an OLAP product is to be used, then your “solution” should include the effort required to map from flat tables to the multi-dimensional cube.

V4 uses new concepts. Its syntax may appear foreign to you. The point here is not to explain the basics of V4 or the particulars of a solution. What we want to show is how easily and quickly one can create solutions using V4. Note that each challenge presented has been implemented in V4 for a real client. The results shown here are based on real data, but some numbers and descriptions have been altered due to the proprietary nature of the data. The time to implement ranged from about 30 minutes for Challenge II to 8 hours for Challenge V. You may not want to actually implement a solution to all of these challenges, but at a minimum, estimate the level of effort that would be required.

These challenges are about manipulating data, not about providing a pretty GUI front-end or generating multicolor reports. Most products, including V4, directly or indirectly provide these ease-of-use facilities. The example reports shown below are images taken directly from Microsoft Excel using the V4 Add-In.

Challenge I – Ranking Of Sales Reps By Month and Sales Manager

This challenge takes information from two tables. The first, Sale, is a record of each sale. The columns in the table are Date (date of the sale), SaleAmount (dollar amount of the sale), and Emp (the sales representative responsible for the sale). The second table, Emp, contains a record for each employee. Employees include sales representatives and their managers. The columns are Id (identifier of the employee), Name (the employee’s name), and Mgr (the employee who is the manager of the sales rep).

The required output shows months across and managers down. For each manager and month, the top five sales representatives are shown as well as their corresponding sales for that month (i.e. two columns for each month). The top sales rep for each month is highlighted in red. Note that any range and/or list of months may be specified, and that a manager may not have five sales reps reporting to him/her.

The V4 Challenge for SQL and OLAP

SalesRep Ranking by Manager

Evaluating [RepRank Month:Jul-99..Dec-99] would result in output looking like-
The V4 Challenge for SQL and OLAP

Challenge II – Extract Lists of Customers Based on Keywords in Name

Given a list of keywords, scan through the customer table, first extracting all customers whose name contains the first keyword, then customers whose name contains the second (but not the first), and so forth for all keywords. The number of keywords is variable with a maximum of 100 keywords. The single table required for this analysis is:

Cus(Id Name City State)

The single V4 rule required to perform this is shown below:

```
Bind [CusGroups List..]
  Do(Project(Dim:Cus Dim:Alpha Area:=1 Unique:=0) Project(Dim:OK Dim:Cus Point:=OK:True)
  Enum(Cus.. @Enum(Str(Cus.Name ListOf:="" Trim?) @Project(Cus Alpha)))
  Enum(List @Enum(Project(Dim:Cus Str(NId:=1))) If::OK:=Cus @Do(Project(OK:False Cus) EchoT(NId Cus.Id .Name .City .State)))))
```

To run this, extracting all customers with “Band”, then “Orchestra”, then “Music” would be done by simply evaluating “[CusGroups (Band Orchestra Music)]”. The output would be a tab delimited file with columns for the keyword, the customer Id, the customer name, city and state.

Challenge III – Variable Bucket Accounts Receivable Aging

The challenge is to generate what is called an Aging Report. It looks at all invoices with balances due, aggregates by the sales representative responsible for the sale, and “ages” the totals. Aging is done by determining how old the invoice is, or how long the balance has been due. For example, an invoice dated 1-Feb-00 would be considered “over 30 days old” as of 1-Oct-00. Two tables are required. The Emp table gives the name and identifier of each sales rep. The Inv(oice) table contains the current balance due, the employee (sales rep), and invoice date.

The analysis should handle variable number of aging buckets.

```
Emp(Id Name)
Inv(BalanceAmt Emp InvDate)
```

The V4 rules to perform this analysis are shown below. Evaluating “[CusAgeByRep Periods:30,45,60,90,120 Date:1-Jun-00]” generated the report immediately following the V4 rules.

```
Bind [AgeTallyArgs Periods.. Date..]
  MakeL(Sum::Inv.BalanceAmt By::Inv.Emp MakeP(Tag:Bind MakeI(AgeAmt Periods*))
  MakeP(Tag::IF Once MakeI(IntMod:LE @[Inv.InvDate] {Date - Periods*})]) Sort(Periods* Reverse::Periods*)
Bind [Func:CusAgeByRep Periods.. Date..]
  Do(Tally(Inv.. (Sum::Inv.BalanceAmt By::Inv.Emp Bind::[ARBalance]) ByList::[RepList])
    [AgeTallyArgs] (Sum::Inv.BalanceAmt By::Inv.Emp Bind::[AgeCur])
    EchoST("SlsRep":Name "Balance":Current "Over":AgeCur End::(0 EchoST("Grand Total": RCV(3) RCV() RCV()/Periods*)
EchoST(End::Str("Customer Aging By SlsRep As of " Date*)))
```

```
      SlsRep   Name                  Balance   Current Over 30 Over 45 Over 60 Over 90 Over 120
 1005 John Smith             17,426.73  -98.30  0    17,128.05 394.30  0    714.69
 1010 Harry Jones            7,904.91  810.00  0    2,962.18   959.44  0    226.56  945.73
 1015 William Johnson        39,993.85  4,799.75 7,956.33 21,722.51  5,392.26  0    104.00
 1020 Melanie Wukasy        3,795.15  108.30  0    1,815.10  0.00  0    1,871.75
 1025 Peppe Andropov         6,960.51  2,940.51 2,520.00  0.00  0.00  0.00  0.00
 1040 Fred Harrison          5,100.51  1,586.50 1,013.22  2,021.22  92.03  295.65
 1045 Sally Bodine           27,686.94  2,110.71 131.29  7,130.86 11,622.90  6,228.53  454.55
 1050 Mary Contrary          193.27  0.00  0.00  0.00  193.27
 1080 Ralph Simms            60,800.07 15,695.01 7,730.45 30,461.13  6,695.46  0.00  0.00
  Grand Total                148,679.54 27,391.48 19,360.32 83,245.04 26,484.39 6,470.69 3,968.53
```

Challenge IV – Number of Unique Customers Purchasing Items

This challenge scans purchases or sales for two separate time periods. The number of unique customers purchasing each item in the two time periods is calculated and output. The output is sorted by the vendor supplying the inventory and then by item. It is often the case that the most elegant solution is not the most efficient. This challenge is computationally expensive due to the requirement for unique customers. The example solution given below was run off of 3.9 million sales records in less than 7.5 CPU minutes on a 400 megahertz PC. Over eight thousand lines of output were generated in the actual report. Only a few are shown here as an example.

The three tables required are shown below. The Ven(dor) table contains the name and identifier for each vendor. The IM (inventory) table has the identifier, name, and vendor of each item. The Sale(s) table contains the customer, inventory item, and time period of each sale.
**The V4 Challenge for SQL and OLAP**

Ven(Id Name)  
IM(Id Name Ven)  
Sale(Cus IM Period)

Evaluating "[NumCusVen UP1:9901..9903 UP2:9907..9909]" results in the report shown following the V4 rules.

```
Bind [NumCusVen UP1.. UP2..]
Do(Tally(Sale.. (By::Sale.IM ByList::[NIMList])
(UCount::Sale.Cus By::Sale.IM Bind::[CusCount1] If::{Sale.Period == UP1*})
(UCount::Sale.Cus By::Sale.IM Bind::[CusCount2] If::{Sale.Period == UP2*}))
EchoS(T:"Id" ;"Description" TR:"Period 1" ;"Period 2" Echo::0)
Enum(Sort([NIMList] By::IM.Ven By::IM*) @EchoS(IM* IM.Desc [CusCount1],0 [CusCount2],0)
After:(IM.Ven EchoS("" Str("Vendor " IM.Ven.Id " - " IM.Ven.Name"))
EchoS(End::Str("Number NABCA Licensees Purchasing Product (Period " UP1* versus " UP2*)") })
```

Evaluating "[NumCusVen UP1:9901..9903 UP2:9907..9909]" results in the report shown following the V4 rules.

```
Bind [NumCusVen UP1.. UP2..]
Do(Tally(Sale.. (By::Sale.IM ByList::[NIMList])
(UCount::Sale.Cus By::Sale.IM Bind::[CusCount1] If::{Sale.Period == UP1*})
(UCount::Sale.Cus By::Sale.IM Bind::[CusCount2] If::{Sale.Period == UP2*}))
EchoS(T:"Id" ;"Description" TR:"Period 1" ;"Period 2" Echo::0)
Enum(Sort([NIMList] By::IM.Ven By::IM*) @EchoS(IM* IM.Desc [CusCount1],0 [CusCount2],0)
After:(IM.Ven EchoS("" Str("Vendor " IM.Ven.Id " - " IM.Ven.Name"))
EchoS(End::Str("Number NABCA Licensees Purchasing Product (Period " UP1* versus " UP2*)") })
```

Number NABCA Licensees Purchasing Product
Period 1999/01..1999/03 versus 1999/07..1999/09

<table>
<thead>
<tr>
<th>Id Description</th>
<th>Period 1</th>
<th>Period 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ven 1506 - CLICQUOT INC</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>Ven 1610 - CLOS DU VAL WN</td>
<td>19</td>
<td>21</td>
</tr>
<tr>
<td>Ven 1618 - CLOVER HILL VIN</td>
<td>74</td>
<td>70</td>
</tr>
<tr>
<td>Ven 1641 - CODERA WINE GRP</td>
<td>43</td>
<td>50</td>
</tr>
<tr>
<td>Ven 1643 - CODORNIU USA</td>
<td>20</td>
<td>19</td>
</tr>
<tr>
<td>8586 CLOVER HILL SPACED A</td>
<td>25</td>
<td>34</td>
</tr>
<tr>
<td>7306 CLOVER HILL CORSIC</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>7307 CLOVER HILL PINK CAT</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>8255 BLACKSTONE MERLOT</td>
<td>43</td>
<td>50</td>
</tr>
</tbody>
</table>

Challenge V – Raw Materials Requirements Based on Sales Projection

This is the last and most complex challenge. The problem is to take weekly sales projections of finished goods, explode them through bills-of-material and generate a consolidated list of raw materials needed to satisfy the projection. Additionally, the onhand quantities of the raw materials and the projected receipts (total quantity and detail) from open purchase orders should be shown. Raw material shortages should also be shown by week.

Seven tables necessary for this analysis. The IM table contains entries for each inventory item (identifier and description). The IML table contains entries giving the on-hand quantity (uOHQty) for each item at each warehouse location (Loc). The PO and POD tables describe open purchase orders with detail for each outstanding inventory item- the quantity remaining to be received (uRemainQty) and the Expected due date. The bill-of-material (BOM) table has an entry for each finished good and the BOMD has entries for each raw material component needed to make the finished product. The Yield is the amount/quantity of the finished product. The BOMIM is the component raw material, and BOMQty is the quantity/amount of the raw material. The SPW table has the weekly projects sales for the finished goods. Note that in the “real world” these quantities may be given in different units. The recipe in the bill-of-material may specify a quantity in grams, but the warehouse quantity (in the IML table) may be in kilograms or pounds.

IM(Id Desc)  
IML(IM Loc uOHQty)  
PO(Id Loc)  
POD(PO IM uRemainQty Expected)  
BOM(IM Yield)  
BOMD(BOM ParentIM BOMIM BOMQty)  
SPW(Id UWeek uQty IM)

The rules to perform this analysis are shown below. To run the project for four weeks beginning with 29-May-00, for warehouse location Main on finished goods projection Test one would evaluate “[ProjRMSPW Week:29-May-00..19-Jun-00 Loc:Main SPWId:Test]”.

Rev 000628  Page 3 of 4
The V4 Challenge for SQL and OLAP

Bind [Explode IM.. Week.. uIMQty:]
DefQ(IM.BOMList
  @Enum(IM.BOMList @Explode BOMD.BOMIM uIMQty:={BOMD.BOMQty * (uIMQty* / BOMD.BOM.Yield)})
  @Do(Context(MakeP(Dim:Exp)) BindQE(Exp.IM IM*) BindQE(Exp.uIMQty uIMQty*) BindQE(Exp.Week Week*))
  @Do(Tally(Ref: IM.. Loc.. SPWId..)
      SUM(Exp.uOHQty By::SPW.Id)
      )

Bind [CRMQty IM.. Week..]
  SUM(List::Week* To::Week*)

Bind [ProjRMSPW Week.. Loc.. SPWId..]
  Do(Tally(Ref: IML.. If::{IML.Loc == Loc*} (SUM::IML.uOHQty By::IML.IId)
     Tally(POD.. Parent::POD.PO If::{PO.Loc == Loc*} If::{POD.Expected == Week*}
          (ListOf::POD* By::POD.IM Bind::[RcvList]) (SUM::POD.uRemainQty By::POD.IM Bind::[uOO])
     Tally(BOMD.. Id::BOMD (List::BOMD* By::BOMD.ParentIM Bind::[BOMList]))
     Tally(Exp.. (SUM::Exp.uIMQty By::Exp.IM Bind::[ExpList]))
     Tally(SPW.. If::{SPW.Id == SPW.Id} Bind::[SPW.List])
    )

Enum(IMList)
  Do(DefQ(IMPrjQty) @Explode IM* Week*
     EchoS("Id":"Description" "OnOrder" "OnHand" "Week" "Needed" "Short")
     EchoS("End":"Raw Materials Requirements Projection Schedule SPWId")

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>OnDate</th>
<th>OnHand</th>
<th>Needed</th>
<th>6/6/00</th>
<th>6/7/00</th>
<th>6/8/00</th>
<th>6/9/00</th>
<th>6/10/00</th>
</tr>
</thead>
<tbody>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/5</td>
<td>0</td>
<td>0</td>
<td>9000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/6</td>
<td>20040</td>
<td>15040</td>
<td>22444</td>
<td>16549</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/7</td>
<td>0</td>
<td>0</td>
<td>9000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/8</td>
<td>20040</td>
<td>15040</td>
<td>22444</td>
<td>16549</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/9</td>
<td>0</td>
<td>0</td>
<td>9000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/10</td>
<td>20040</td>
<td>15040</td>
<td>22444</td>
<td>16549</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200065</td>
<td>PEANUTS</td>
<td>6/11</td>
<td>0</td>
<td>0</td>
<td>9000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/1</td>
<td>371</td>
<td>371</td>
<td>371</td>
<td>371</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/2</td>
<td>444</td>
<td>232</td>
<td>232</td>
<td>232</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/4</td>
<td>444</td>
<td>232</td>
<td>232</td>
<td>232</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/6</td>
<td>371</td>
<td>371</td>
<td>371</td>
<td>371</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/21</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/22</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/26</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/27</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/28</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/29</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200066</td>
<td>TRAY CANTY COOKY</td>
<td>6/30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rev 000628 Page 4 of 4