Calendar Conundrum Outline

DevCon 2006

Created by John Mark Osborne - www.databasepros.com

1. Relational Structure

- 1.1. Tables
  - 1.1.1. Days
  - 1.1.2. Events
  - 1.1.3. View
- 1.2. Fields
  - 1.2.1. Days
    - 1.2.1.1. Date (date)
  - 1.2.2. Events
    - 1.2.2.1. Date (date)
    - 1.2.2.2. TimeStart (time)
  - 1.2.3. View
    - 1.2.3.1. No fields at this point
  - 1.2.4. Keep fields to a minimum to reduce size
    - 1.2.4.1. Especially important for Event table which will carry a lot of records
- 1.3. Relationships
  - 1.3.1. Days to Events using Date field as key field
    - 1.3.1.1. Equals operator
    - 1.3.1.2. Allow deletion of related records
  - 1.3.2. New Table Occurrences
    - 1.3.2.1. DAYS_Month (DAYS)
    - 1.3.2.2. EVENTS_Month (EVENTS)
  - 1.3.3. More relationships later, including VIEW table
- 1.4. Days Script
  - 1.4.1. Allow User Abort
  - 1.4.2. Test for maximum records with Go to Record/Request/Page [Last]
    - 1.4.2.1. No need for additional relationship or slow calculation
  - 1.4.3. Looping script
    - 1.4.3.1. Use variable instead of global field - faster
  - 1.4.4. Script shown in glossary - “Generate [Days - Event Loop]”
- 1.5. Events Script
  - 1.5.1. Script to create 96 - 15 minute increment records per day (35,040 event records per year)
  - 1.5.2. Reduce number of event records by excluding night hours
  - 1.5.3. Complete script in glossary - “Generate [Events - Loop]”
- 1.6. Speed Up Scripts
  - 1.6.1. Create EVENTTEMP table
  - 1.6.2. Import 96 time records from Events to EventTemp
  - 1.6.3. No need to import Days records
    - 1.6.3.1. Far fewer records so import will not impact speed
  - 1.6.4. Show how to test speed difference

2. Month View

- 2.1. Global Navigation Fields
  - 2.1.1. Create Fields in View table
    - 2.1.1.1. xMonth (global text)
    - 2.1.1.2. xYear (global number)
  - 2.1.2. Add Value Lists
    - 2.1.2.1. Static Month Names (custom)
    - 2.1.2.2. Field Based (Years from Days table)
      - 2.1.2.2.1. Requires calculation field - Year(Date)
        - 2.1.2.2.1.1. If Date was in Events table, I would use custom value list to keep size down
    - 2.1.2.3. Popup Menus or Drop-Down Lists?

2.1.2.3.1. Popup menus not keyboard driven
2.1.2.3.2. Popup menus make shadow, no control over interface
2.1.2.3.3. Can replicate popup button functionality in drop-down list using built-in feature or button

2.2. Convert Month to Number
2.2.1. Calculation to convert month names to numbers
2.2.1.1. `Position("**JanFebMarAprMayJunJulAugSepOctNovDec"; Left(xMonth; 3); 1; 1) / 3`
2.2.2. No need for custom function because will use once and custom functions tend to be a little slower

2.3. Custom Functions
2.3.1. First Day Custom Function
2.3.1.1. `Date(MonthNum; 1; YearNum)`
2.3.2. Last Day Custom Function
2.3.2.1. `Date(MonthNum + 1; 0; YearNum)`
2.3.3. Will be using in various places in the solution so a Custom Function is a good idea in order to reuse code and have one place to update the formula

2.4. Filter the Relationship
2.4.1. cMonthYearLow calculation
2.4.1.1. `FirstDay(cMonth; xYear) - DayOfWeek(FirstDay(cMonth; xYear)) + 1`
2.4.2. cMonthYearHigh calculation
2.4.2.1. `LastDay(cMonth; xYear) + 14`
2.4.3. Rename Table Occurrence
2.4.3.1. VIEW = VIEW_Month
2.4.4. Draw relationship lines
2.4.4.1. VIEW_Month::cMonthYearLow <= DAYS_Month::Date
2.4.4.2. VIEW_Month::cMonthYearHigh >= DAYS_Month::Date

2.5. Plethora of Portals
2.5.1. Create the 42 portals representing calendar
2.5.1.1. Displays related records in month view with a single relationship
2.5.2. Create one portal and then duplicate, changing the initial row
2.5.3. Add Date field
2.5.3.1. Will format date later
2.5.4. Overlap portal edges so looks better
2.5.5. Do not allow entry into Date field via Field Behavior

2.6. Highlight Colors
2.6.1. Highlight current month days and current day
2.6.2. Container fields in View table
2.6.2.1. HighlightMonth
2.6.2.2. HighlightDay
2.6.2.3. Can't be global fields
2.6.3. Calculations for month highlight in View table
2.6.3.1. cMonthYearLowHighlight
2.6.3.1.1. `FirstDay(cMonth; xYear)`
2.6.3.2. cMonthYearHighHighlight
2.6.3.2.1. `LastDay(cMonth; xYear)`
2.6.3.3. Auto-Enter calculation cause must be stored for relationship
2.6.3.4. Uncheck the "do not replace existing value" auto-enter option
2.6.3.5. Changing of globals updates auto-enter calculations
2.6.4. Calculations for day highlight in View table
2.6.4.1. Create field = Date_Current (date)
2.6.4.1.1. Auto-enter calculation
2.6.4.1.1.1. `Evaluate(Quote(Get(CurrentDate)); [xMonth; xYear])`
2.6.4.1.1.2. Second parameter in Evaluate function triggers calculation when xMonth or xYear is modified
2.6.4.1.1.3. Add additional fields to Evaluate function when create Weekly

and Daily views if want current day highlighted

2.6.5. Relationships
2.6.5.1. New TO called VIEW_Month_Hilite
2.6.5.1.1. DAYS_Month::Date >= VIEW_Month_Hilite::cMonthYearLowHilite
2.6.5.1.2. DAYS_Month::Date <= VIEW_Month_Hilite::cMonthYearHighHilite
2.6.5.2. New TO called VIEW_Day_Hilite_Month
2.6.5.2.1. DAYS_Month::Date = VIEW_Day_Hilite_Month::Date_Current
2.6.6. Place VIEW_Month_Hilite::HighlightMonth and VIEW_Day_Hilite_Month::Highlight_Day fields in every portal
2.6.6.1. Will display correct highlight color based on perspective of the current portal
2.6.6.2. Send field to back so don't obscure Date field
2.6.6.3. Fill highlight fields with transparent pattern so don't obscure each other
2.6.6.4. Do not allow entry into highlight fields via Field Behavior
2.6.7. Changing month or year popup, doesn't update until commit records (fixed later)

2.7. Colored Day Numbers
2.7.1. Format Date field to display as just day number
2.7.2. Show how to make fields gray (previous or next month) versus black (current month)
2.7.2.1. Calculation in Days table to change colors

2.8. Navigation Buttons
2.8.1. Buttons to navigate back and forth through months
2.8.2. Basic scripts (see glossary)
2.8.2.1. Next Month [Basic]
2.8.2.2. Previous Month [Basic]
2.8.3. Make dynamic for next and previous month (see glossary)
2.8.3.1. Next/Previous Month [Basic]
2.8.4. What about changing of year?
2.8.4.1. Need to change month and year when January or December
2.8.4.2. See glossary for Next/Previous Month [Final]

2.9. Interface
2.9.1. Borrow from finished calendar to save time
2.9.1.1. Tabs
2.9.1.2. Buttons
2.9.1.3. Shadow background
2.9.1.4. Title
2.9.2. Cover briefly
2.9.2.1. Leggos - easier to modify for different screen sizes
2.9.2.2. Graphics stored once
2.9.2.3. Faster display and smaller footprint
2.9.3. Remove popup menus and replace with xMonth and xYear merge fields
2.9.4. Add day names to top of columns with text tool

3. Week View

3.1. Squids and Spiders
3.1.1. Spider = all TOs connected by relationships
3.1.2. Squid = groupings of table occurrences based on functionality
3.1.3. Squid method fits perfectly with different views of calendar (Month, Week and Day)

3.2. Add Relationships
3.2.1. Add Fields
3.2.1.1. xWeekStart (global date)
3.2.1.2. xWeekEnd (global date)
3.2.2. Add TO's
3.2.2.1. View_Week
3.2.2.2. Days_Week
3.2.2.3. Events_Week
3.2.3. Draw Relationship Lines
3.2.3.1. VIEWS to DAYS table
3.2.3.1.1. VIEW_Week::xWeekStart <= DAYS_Week::Date
3.2.3.1.2. VIEW_Week::xWeekEnd >= DAYS_Week::Date
3.2.3.2. DAYS to EVENTS table

3.2.3.2.1. DAYS_Week::Date = EVENTS_Week::Date
3.3. Create Week Layout
3.3.1. Add portal for times showing first 96 rows
3.3.1.1. Display TimeStart field and prevent entry into field using Field Behavior
3.3.2. Add Title field (text) to EVENTS table
3.3.3. Add 7 Portals showing 96 rows (each day of the week)
3.3.3.1. Display Title field
3.3.3.2. Each portal starts at different initial row (e.g. second portal starts at 97)
3.3.4. Copy and Paste interface from finished calendar
3.4. Adding Appointments
3.4.1. Create Add Events layout showing records from EVENTS Table Occurrence
3.4.1.1. Add Fields to Events table
3.4.1.1.1. xTitle (global text)
3.4.1.1.2. xTimeStart (global time)
3.4.1.1.3. xTimeEnd (global time)
3.4.1.1.4. EventGroup (time)
3.4.1.1.5. xEventGroup (global time)
3.4.2. Create relationship (self-join)
3.4.2.1. EVENTS_Time::TimeStart = EVENTS::xTimeStart
3.4.2.2. EVENTS_Time::Date = EVENTS::Date
3.4.3. Create Add Event script
3.4.3.1. See glossary for Add Event [Basic] script
3.4.4. Labels table
3.4.4.1. “Add” button disappears on appointments by hiding with color
3.4.4.2. Fields
3.4.4.2.1. Label (Text)
3.4.4.2.2. Color (Container)
3.4.4.3. Add colors to table
3.4.4.4. Add Label field to Events table
3.4.4.5. Add TO for Labels table called LABELS_Week
3.4.4.6. Draw relationship
3.4.4.6.1. EVENTS_Week::Label = LABELS_Week::Label
3.4.4.7. Display Label choices
3.4.4.7.1. Add TO called LABELS_Select
3.4.4.7.1.1. LABELS_Select::Label x EVENTS::Label
3.4.4.7.1.2. Cartesian product shows all related records regardless of key fields
3.4.5. Script to make label selection
3.4.5.1. Add xLabel global field
3.4.5.2. Script uses Set Field to place LABELS_Select::Label calculation result into EVENTS::xLabel target field
3.4.5.3. Add Set Field step to Add Event [Basic] script to set Label to xLabel as loops through records
3.4.6. Add LABELS_Week::Color field to portals
3.4.6.1. Do not allow entry via Field Behavior
3.4.6.2. Format with Graphic... dialog to enlarge or reduce while NOT maintaining proportions
3.4.6.3. Overlap color field on edges of portal first row so color completely fills the row
3.4.6.4. Place button behind Color field but Title field above
3.4.6.4.1. When color added, hides button but not title
3.4.6.5. Add conditional to Add Event [Basic] script to prevent clicking of Add Event when an event already exists (hidden by color but not inactive)
3.4.7. Add “Add Event” button to Month view
3.4.7.1. Modify script to add Set Error Capture and additional GTRR
3.4.7.1.1. Go to Related Record from: Events_Month
3.4.7.1.2. Set Error Capture prevents error message from GTRR step
3.5. Deleting Appointments
3.5.1. Add TO for VIEW_TrashCan

3.5.1.1. EVENTS_Week::EventsGroup <> VIEW_TrashCan::TrashCan
3.5.1.2. Works because EventsGroup is empty on all records except the group of records that have an event assigned. When EventGroup is empty, the relationship is invalid. When the EventGroup is filled, the relationship is valid but the EventGroup does not equal the current date.
3.5.2. Create script
3.5.2.1. See glossary for Delete Event [Basic] script
3.5.3. Add TrashCan field to portals
3.5.3.1. Delete icon appears using TO
3.5.3.2. Set to Crop through Graphic... dialog
3.5.3.3. Set to not allow entry via Field Behavior
3.5.3.4. Turn on Show >Sample Data so can resize field easily
3.6. Revising Appointments
3.6.1. Combination of Delete and Add Appointment scripts
3.6.1.1. Add script that calls Delete Event and then Add Event script
3.6.1.1.1. Add optional script parameter to Perform Script with text “Change”
3.6.1.2. Make a few changes to Add Event script to preserve end time, title and label
3.6.1.2.1. Test for script parameter “Change”
3.6.1.3. See glossary for script
3.6.2. Change add event button to allow for revision
3.6.2.1. Must make it similar to how trash can disappears
3.6.2.2. Add Field to View table
3.6.2.2.1. AddEvent (Container)
3.6.2.2.2. Add graphic to field
3.6.2.3. Add TO for VIEW_AddEvent
3.6.2.3.1. EVENTS_Week::Title <> VIEW_AddEvent::Date_Current
3.6.2.3.2. Works because Title is empty on all records except first of group. When Title is empty, the relationship is invalid. When the title is filled, the relationship is valid but the title does not equal the current date.
3.7. Navigating Weeks
3.7.1. Previous/Next Week Set Field steps
3.7.1.1. Target = xWeekStart; Formula = `VIEW_Week::xWeekStart + Case(Get(ScriptParameter) = "Next"; 7; -7)`
3.7.1.2. Target = xWeekEnd; Formula = `VIEW_Week::xWeekEnd + Case(Get(ScriptParameter) = "Next"; 7; -7)`
3.8. Complete Interface
3.8.1. Color work hours with rectangle tool
3.8.2. Year in upper left corner
3.8.2.1. EVENTS_Week::Date formatted to only show year
3.8.3. Day/Date above each column
3.8.3.1. Use 7 one row portals from DAYS_Week TO
3.8.3.2. Start portal at different initial row (1 through 7)
3.8.3.3. Format date field to show Day Name, Month and Day Number
3.8.4. Highlight Current Day
3.8.4.1. Requires TO for Day titled VIEW_Day_Hilite_Month
3.8.4.1.1. DAYS_Week::Date = VIEW_Day_Hilite_Month::Date_Current
3.8.4.2. Update Current_Date field to include xWeekStart and xWeekEnd
3.8.4.3. Place Highlight_Day field from VIEW_Day_Hilite_Month TO in each portal
3.8.5. Highlight Work Hours
3.8.5.1. Work hours can be highlighted with rectangle tool
3.8.6. Continue and Cancel buttons on Add Event layout
3.8.6.1. Continue is just the Resume step
3.8.6.2. Cancel closes the current window and halts the current script

4. More...

4.1. There is certainly more that can be covered but we are limited by time
4.1.1. Calculation to display events in Month view using List function in example
4.1.2. Bonus scripts in example file that navigate based on selected portal row

5. Glossary

5.1. Generate [Days - Event Loop] script

5.1.1. Complete formula for Show Custom Dialog step from script at right

```
"Are you sure you want to
create dates for the " &
Case(Get(TotalRecordCount) =
0; Year(Get(CurrentDate));
Year(DAYS_Month::Date) + 1)
& " calendar year?"
```

```
Perform Script ["Allow User Abort"]
Go to Layout ["DAYS_Month" (DAYS_Month)]
Show All Records
Go to Record/Request/Page [Last]
Show Custom Dialog ["Create Dates"; "Are you sure you want to create dates for the " & Case(Get(TotalRecordCount) = 0; Year(Get(
Set Variable [$ElapseTime; Value:Get(CurrentTime)]
If [Get(LastMessageChoice) = 1]
    Freeze Window
    Set Variable [$Date; Value:Date( 1; 1; Case(Get(TotalRecordCount) = 0; Year(Get(CurrentDate)); Year(DAYS_Month::Date) + 1) )]
    Loop
        New Record/Request
        Set Field [DAYS_Month::Date; $Date]
        Set Variable [$Date; Value:$Date + 1]
        Perform Script ["Generate [Events - Loop]"; Parameter: DAYS_Month::Date]
        Exit Loop If [Year(DAYS_Month::Date) <> Year($Date)]
    End Loop
End If
Set Variable [$ElapseTime; Value:Get(CurrentTime) - $ElapseTime]
Show Custom Dialog ["Elapsed Time"; "Elapsed Time: " & GetAsNumber($ElapseTime) & " seconds."]
```

5.2. Generate [Events - Loop] script

- Go to Layout ["EVENTS_Month" (EVENTS_Month)]
- Set Variable [$Time; Value:Time(0; 0; 0)]
- Loop
- New Record/Request
- Set Field [EVENTS_Month::Date; Get(ScriptParameter)]
- Set Field [EVENTS_Month::TimeStart; $Time]
- Exit Loop If [EVENTS_Month::TimeStart = Time(23; 45; 0)]
- Set Variable [$Time; Value:$Time + 900]
- End Loop
- Go to Layout [original layout]

5.3. Generate [Events - Import] script

- Go to Layout ["EVENTS_Month" (EVENTS_Month)]
- Set Field [EVENTSTEMP::xDate; Get(ScriptParameter)]
- Import Records [No dialog; "CAL.fp7"; Add; Mac Roman]
- Go to Layout [original layout]

5.4. Next Month [Basic]

- Set Field [VIEW_Month::xMonth; Let(ValueList = ValueListItems(Get(FileName);
- Commit Records/Requests [Skip data entry validation; No dialog]

5.4.1. Complete Set Field calculation

```
Let(

ValueList = ValueListItems(Get(FileName); "Months");

LeftWords(

MiddleValues(
ValueList;
ValueCount(Left(ValueList; Position(ValueList; VIEW_Month::xMonth; 1;
     1))) + 1; //Make - 1 for previous month
1);

999999999)

)
```

5.5. Previous Month [Basic]

5.5.1. Only difference from above calculation is + 1 versus - 1 (bold and italics in calculation above)

5.6. Next/Previous Month [Basic]

5.6.1. Just replace + 1 or - 1 with the following formula

5.6.2. Make sure to attach script parameters to next and previous buttons

```
+ Case(Get(ScriptParameter) = "Next"; 1; -1)
```

5.6.3. Next/Previous Month [Final]

```
#Combines Next and Previous buttons into a single script.
#IF tests if first or last month of year so can go to the beginning or end of the y
If [Case(  Get(ScriptParameter) = "Next"; // Check if button clicked was Next or Previc
  Set Field [VIEW_Month::xMonth; Case(  Get(ScriptParameter) = "Next"; // If Next or I
  Set Field [VIEW_Month::xYear; Case(  Get(ScriptParameter) = "Next"; // Check if Nex
  #Else runs when any month other than January or December is currently selec
Else
  Set Field [VIEW_Month::xMonth; Let(  ValueList = ValueListItems(Get(FileName); "Mo
End If
Commit Records/Requests [Skip data entry validation; No dialog]
```

5.6.3.1. Formula from IF step:

```
Case(

Get(ScriptParameter) = "Next"; // Check if button clicked was Next or
Previous

VIEW_Month::xMonth = "December"; // If Next, result in true if last month
of year or false if any other month

VIEW_Month::xMonth = "January" // If Previous, result in true if first
month of year or false if any other month

)
```

5.6.3.2. Formula from first Set Field step:

```
Case(

Get(ScriptParameter) = "Next"; // If Next or Previous button is
clicked

"January"; // If Next button clicked

"December" // If Previous button clicked

)
```

5.6.3.3. Formula from second Set Field step:

```
Case(

Get(ScriptParameter) = "Next"; // Check if Next or Previous button clicke

VIEW_Month::xYear + 1; // If Next, add one to the year

VIEW_Month::xYear - 1 // If Previous, subtract one from the year

)
```

5.6.3.4. Formula from third Set Field step is the same as Next/Previous Month [Basic]

5.7. Add Event [Basic] script

```
# Go to Related Record [From table: "EVENTS_Week"; Using layout: "EVENTS_Add" (EVENTS); New window]
# Set Field [EVENTS::xTimeStart; EVENTS::TimeStart]
# Set Field [EVENTS::xTimeEnd; ""]
# Set Field [EVENTS::xTitle; ""]
# Pause/Resume Script [Indefinitely]
# Go to Related Record [From table: "EVENTS_Time"; Using layout: "EVENTS_Add" (EVENTS)]
# Set Field [EVENTS::Title; EVENTS::xTitle]
# Loop
#    Set Field [EVENTS::EventGroup; EVENTS::xTimeStart]
#    Exit Loop If [EVENTS::xTimeEnd = EVENTS::TimeStart]
#    Go to Record/Request/Page [Next; Exit after last]
# End Loop
# Close Window [Current Window]
```

```
Perform Script ["Allow User Abort"]
If [not IsEmpty(EVENTS_Week::EventGroup)]
   Go to Related Record [From table: "EVENTS_Week"; Using layout: "EVENTS_Add" (EVENTS); New window]
   Set Variable [$EventGroup; Value:EVENTS::EventGroup]
   Set Field [EVENTS::xTitle; Case(Get(ScriptParameter) = "Change"; EVENTS::Title)]
   Set Field [EVENTS::xLabel; Case(Get(ScriptParameter) = "Change"; EVENTS::Label)]
   Loop
      Set Field [EVENTS::EventGroup; ""]
      Set Field [EVENTS::Label; ""]
      Set Field [EVENTS::Title; ""]
      Go to Record/Request/Page [Next; Exit after last]
      Exit Loop If [$EventGroup ≠ EVENTS::EventGroup]
   End Loop
   Set Field [EVENTS::xTimeEnd; Case(Get(ScriptParameter) = "Change"; EVENTS::TimeStart - 900)]
   Close Window [Current Window]
End If
```

5.9. Revise Event (Modified Add Event script)

```
Perform Script ["Allow User Abort"]
Perform Script ["Add Event [Basic]"; Parameter: "Change"]
```

```
Perform Script ["Allow User Abort"]
Perform Script ["Set Error Capture"]
If [IsEmpty(EVENTS_Week::EventGroup) or Get(ScriptParameter) = "Change"]
    Go to Related Record [From table: "EVENTS_Week"; Using layout: "EVENTS_Add" (EVENTS); New window]
    Go to Related Record [From table: "EVENTS_Month"; Using layout: "EVENTS_Add" (EVENTS); New window]
    Set Field [EVENTS::xTimeStart; EVENTS::TimeStart]
    Set Field [EVENTS::xTimeEnd; Case(Get(ScriptParameter) = "Change"; Max(EVENTS_Groups::TimeStart))]
    Set Field [EVENTS::xTitle; Case(Get(ScriptParameter) = "Change"; EVENTS::Title)]
    Set Field [EVENTS::xLabel; Case(Get(ScriptParameter) = "Change"; EVENTS::Label)]
    Pause/Resume Script [Indefinitely]
    If [Get(ScriptParameter) = "Change"]
        Perform Script ["Delete Event [Basic]"]
    End If
    Go to Related Record [From table: "EVENTS_Time"; Using layout: "EVENTS_Add" (EVENTS)]
    Set Field [EVENTS::Title; EVENTS::xTitle]
    Loop
        Set Field [EVENTS::EventGroup; EVENTS::xTimeStart]
        Set Field [EVENTS::Label; EVENTS::xLabel]
        Exit Loop If [EVENTS::xTimeEnd = EVENTS::TimeStart]
        Go to Record/Request/Page [Next; Exit after last]
    End Loop
    Close Window [Current Window]
    Refresh Window [Flush cached join results]
End If
```