Classified Designs

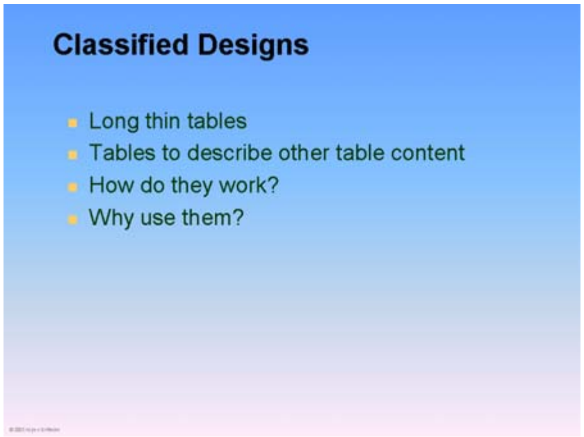
Neil Smith & Suzy Smith
Aspen Software

Summary

The need to respond to change within business never ceases to diminish. As system developers we can all struggle to cope with the incessant demand for changes from our customers.

In "Classified Designs" we focus on a combination of generally applicable, classic design themes that can give the flexibility and performance to help meet these demands. Drawing on mission critical systems that our clients have been using for over a decade, you will see how we have used R:BASE, Oterro and ActiveX technologies to face up to some of these challenges.

Classic Design



Classified Designs

- Long thin tables
- Tables to describe other table content
- How do they work?
- Why use them?

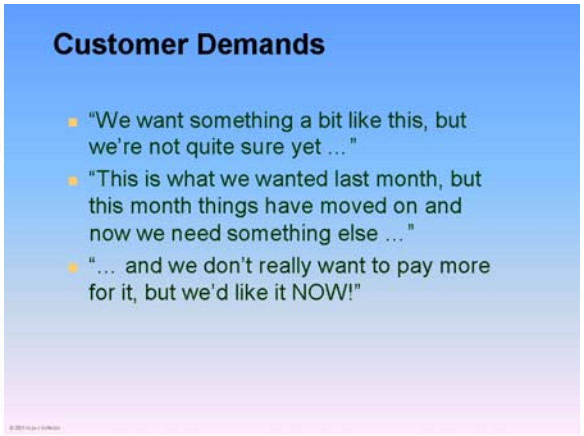
“Long thin tables” is a classic design mantra - we all know that it is a good design philosophy and we should always try to make our tables long and thin instead of short and fat.

A classic example of using long thin tables is when you are storing contact details for people. You could have a column in the contact table for email address, main contact number and cell phone number, but what happens when you need to store the person’s home number as well, or when you’ve been given a number to return someone’s call and you need to search the database to see who it belongs to – you have to search for the value in at least two columns, maybe more. A much more flexible approach is to store the contact numbers in a separate table with a description for each number (eg home, office etc) and link the table to the main contact details table.

Using “tables to describe other tables” extends the concept and provides an immense amount of flexibility - within R:BASE the SYS_TABLES table and the other system tables are examples of this. The content of the SYS_TABLES and SYS_COLUMNS tables holds the data which “describes!” the structure of the database.

We will describe the practicalities of using these techniques with some real life examples – but, good database design practise aside - why are we doing this anyway?

Customer Demands



Customer Demands

- “We want something a bit like this, but we’re not quite sure yet ... ”
- “This is what we wanted last month, but this month things have moved on and now we need something else ... ”
- “... and we don’t really want to pay more for it, but we’d like it NOW!”

We all know what customers can be like ... Some of these demands will probably be very familiar to you.

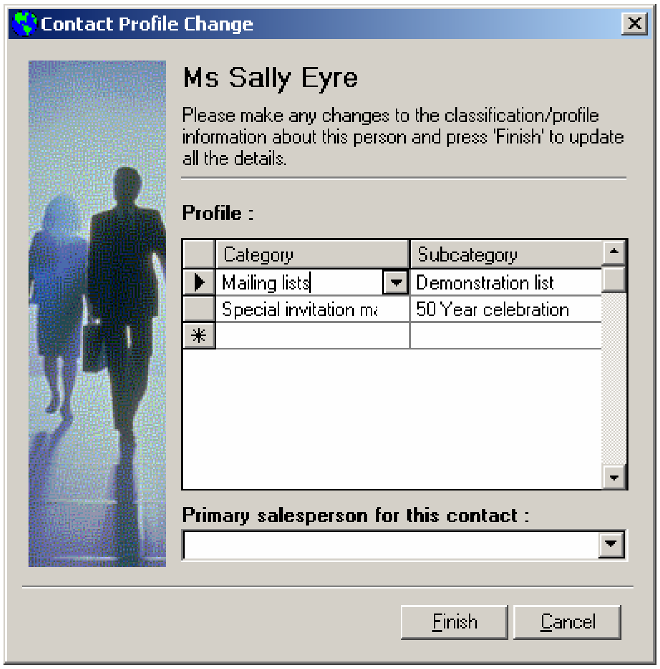
Our customers (the people who use the systems we develop – whether within our organisation or as fee paying clients) have to be able to respond to rapidly changing business demands and so do we. It’s our job, as developers to help them. We have found that by adopting this approach to database design and writing reusable tools which enhance the design, we can be more responsive to our customer’s needs. This enables us to deliver flexible, robust solutions which the customers can adapt themselves without being forced to come back to us for programming changes.

A cynic might think that we should make designs inflexible and “lock customers in” so they have to come back to us for changes. However, what we have found is that using this extensible “open” approach makes our customers happy and they do still come back, but asking for new features which gives us more interesting development projects to work on.

Let us look at two real-life, operational examples of this, which use the same underlying table structures.

Classification & Profiling

In this example, our customer wanted to be able to record pieces of information about his business contacts and sales leads and to be able to add to the types of information over time. What's more he wasn't sure what sort of information he might be interested in. So the classic way of providing columns in the contact table for the information was not going to work. Instead we decided to keep track of the information in another linked table. To ensure the robustness of the data, the user selects data from drop down lists which are driven by the values in the supporting tables. When the customer decides that another category of information is needed, they can just add new values to the supporting tables which drive the drop down lists. We don't just use this to profile contacts either. The system is a complete CRM with information about leads, accounts, proposals etc and we use the same approach to enable the user to store profiles for all these different items. The information can be held in the same table, regardless of the type of item it is related to.



Ms Sally Eyre

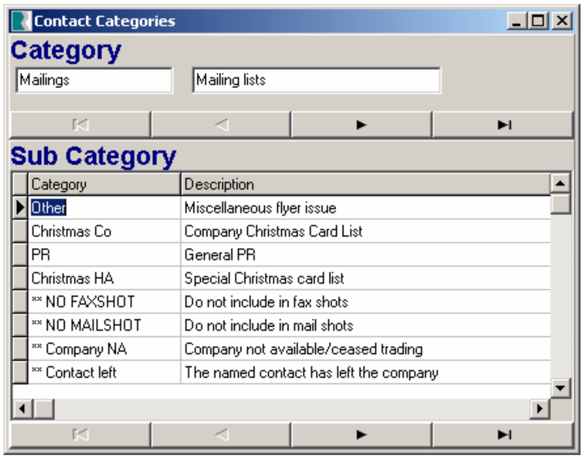
Please make any changes to the classification/profile information about this person and press 'Finish' to update all the details.

Profile :

Category	Subcategory
Mailing lists	Demonstration list
Special invitation ma	50 Year celebration

Primary salesperson for this contact :

The underlying data used to drive profiling is in two tables – The first table has a simple three column structure – type of object, category and description. The second table contains the sub categories and is linked to the first by the object type and the main category column.

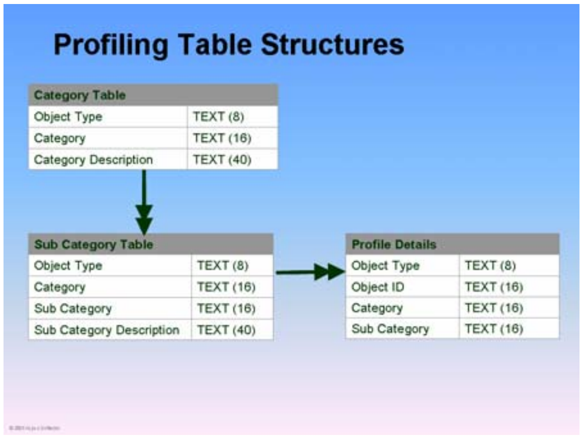


Category

Mailings | Mailing lists

Sub Category

Category	Description
Other	Miscellaneous flyer issue
Christmas Co	Company Christmas Card List
PR	General PR
Christmas HA	Special Christmas card list
** NO FAXSHOT	Do not include in fax shots
** NO MAILSHOT	Do not include in mail shots
** Company NA	Company not available/ceased trading
** Contact left	The named contact has left the company



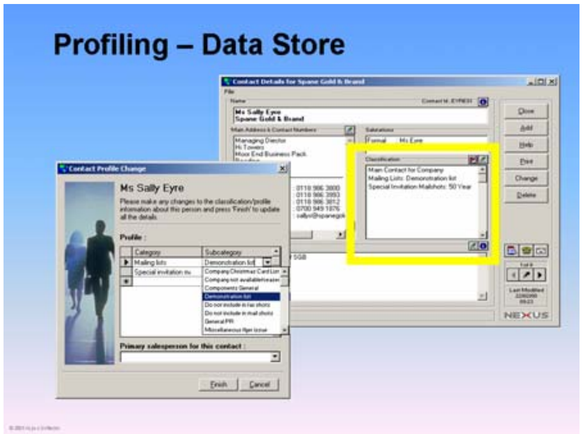
Profiling Table Structures

Category Table	
Object Type	TEXT (8)
Category	TEXT (16)
Category Description	TEXT (40)

Sub Category Table	
Object Type	TEXT (8)
Category	TEXT (16)
Sub Category	TEXT (16)
Sub Category Description	TEXT (40)

Profile Details	
Object Type	TEXT (8)
Object ID	TEXT (16)
Category	TEXT (16)
Sub Category	TEXT (16)

The example illustrated shows an R:BASE form displaying the "Mailing List" category for the "Contact" object type and the sub categories are the mailing lists which are available to choose from. Another category, for example, could be preferred method of contact and the sub categories would be email, telephone, SMS etc.



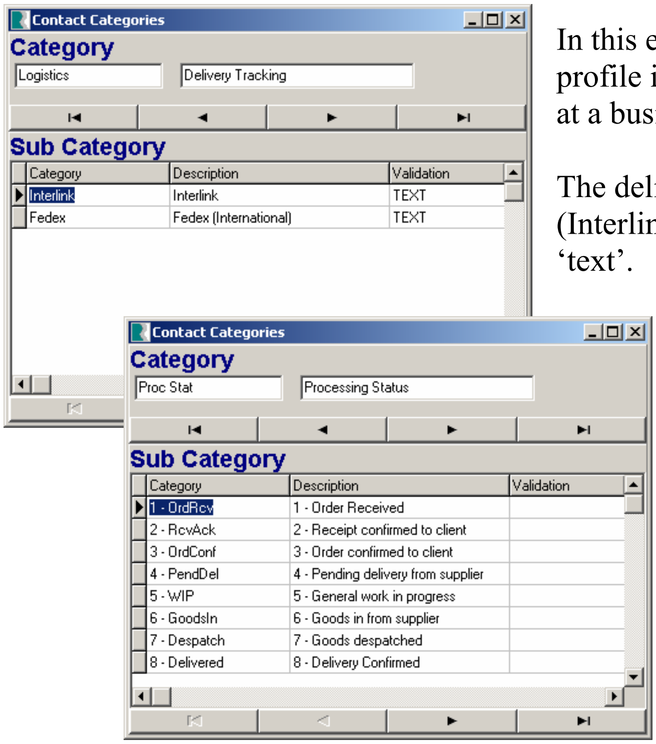
Profiling – Data Store

As shown in the Visual Basic and Oterro application - the area inside the yellow box is the display of the contact profile information from the profile details table. To change the information the user clicks on the green pencil and a data entry form is displayed as shown on the left. In the grid the user can choose the category and a corresponding sub category. This example has the drop down box showing the mailing list subcategories from which the user can choose.

Extending the Design

The customer used the application for some years, adding new categories and sub categories as needed and using them to drive the profiler. However, over time the nature of some of the information he wanted to record changed and we found that we needed to extend our original design because drop down lists were too restrictive. The new information was better suited to a free format data entry.

We expanded the Sub Category table to include a validation type which is used to indicate whether to display the information as a “Yes/No” radio button, as a drop down list box or as a free format data entry field. When the data entry form is displayed, the data in this new column is used to determine how to display the subcategory information.



Contact Categories

Category

Logistics | Delivery Tracking

Sub Category

Category	Description	Validation
Interlink	Interlink	TEXT
Fedex	Fedex (International)	TEXT

Contact Categories

Category

Proc Stat | Processing Status

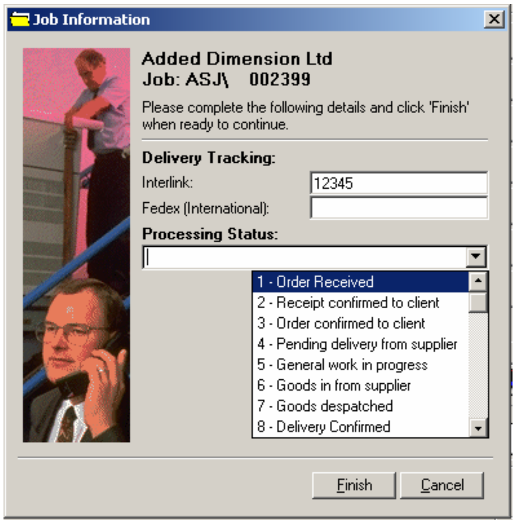
Sub Category

Category	Description	Validation
1 - OrdRcv	1 - Order Received	
2 - RcvAck	2 - Receipt confirmed to client	
3 - OrdConf	3 - Order confirmed to client	
4 - PendDel	4 - Pending delivery from supplier	
5 - WIP	5 - General work in progress	
6 - GoodsIn	6 - Goods in from supplier	
7 - Despatch	7 - Goods despatched	
8 - Delivered	8 - Delivery Confirmed	

In this example we are looking at the classification profile information for a “job” (before we were looking at a business contact).

The delivery tracking category and its two subcategories (Interlink and FedEx) have the validation type set to ‘text’.

For the processing status sub categories the validation type is blank.



Job Information

Added Dimension Ltd
Job: ASJ\ 002399

Please complete the following details and click 'Finish' when ready to continue.

Delivery Tracking:

Interlink:	12345
Fedex (International):	

Processing Status:

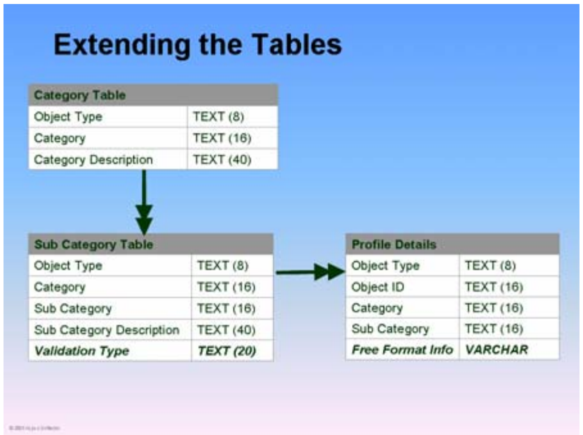
1 - Order Received
2 - Receipt confirmed to client
3 - Order confirmed to client
4 - Pending delivery from supplier
5 - General work in progress
6 - Goods in from supplier
7 - Goods despatched
8 - Delivery Confirmed

When the profiler form is displayed, it generates a drop down list, as before to allow the user to choose the processing status from, but, for the delivery tracking categories instead of a drop down list, a free format text box is displayed, allowing the user to type in a tracking number.

Of course, we also had to modify the table in which the data is stored as well. We had to add a variable length text column to hold the free format information where required.

These are the new data structures - the Sub Category table now includes the validation type column and the Profile Details table has a new column to hold the free format information.

Extending the design in this way made it even more flexible.



Extending the Tables

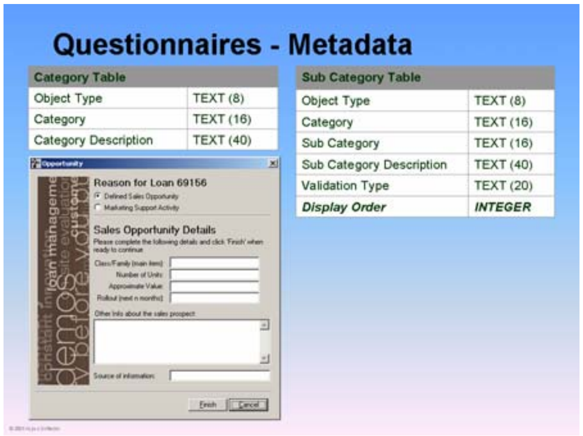
Category Table	
Object Type	TEXT (8)
Category	TEXT (16)
Category Description	TEXT (40)

Sub Category Table	
Object Type	TEXT (8)
Category	TEXT (16)
Sub Category	TEXT (16)
Sub Category Description	TEXT (40)
Validation Type	***TEXT (20)***

Profile Details	
Object Type	TEXT (8)
Object ID	TEXT (16)
Category	TEXT (16)
Sub Category	TEXT (16)
Free Format Info	***VARCHAR***

Classification & Questionnaires!

Some time later, another customer wanted to get information from their clients in a structured manner, but it wasn't clear what the questions were going to be and the chances were they would change over time. Hard coding data entry forms for this was not going to give them the flexibility they needed. What they wanted was a facility to generate on-the-fly questionnaires which could be easily modified over time.



Questionnaires - Metadata

Category Table	
Object Type	TEXT (8)
Category	TEXT (16)
Category Description	TEXT (40)

Sub Category Table	
Object Type	TEXT (8)
Category	TEXT (16)
Sub Category	TEXT (16)
Sub Category Description	TEXT (40)
Validation Type	TEXT (20)
Display Order	***INTEGER***

Opportunity

Reason for Loan 69156
● Defined Sales Opportunity
○ Marketing Support Activity

Sales Opportunity Details
Please complete the following details and click 'Finish' when ready to continue

Class/Family (main item):
Number of Units:
Approximate Value:
Rollout (next n months):
Other Info about the sales prospect:
Source of information:

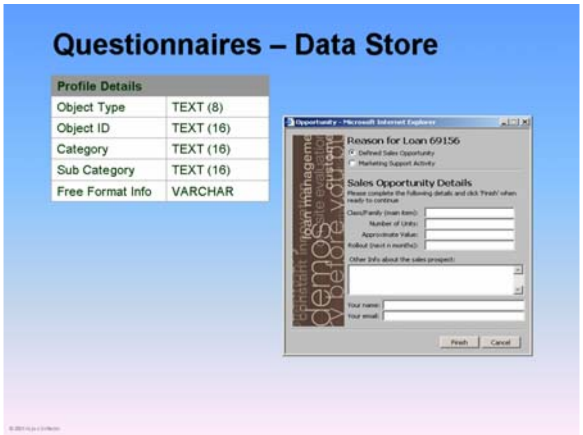
We already had the profiling tool and we decided that we could solve our customer's problem using the existing tool with one slight enhancement.

The sub category table already had the validation type column which allowed us to specify free format text, radio button or drop down display. We added another validation type – multi line – which indicates that the data entry form should show a multi line text box.

We also expanded the Sub Category table again to include a display order. The display order is used to control the sequence of the prompts on the form. This gave the customer even more control over how the form is displayed and was important for the questionnaire situation where one question might logically follow another.

In this example you can see there are radio buttons at the top and the rest of the questionnaire is free format text, with a multi line text box towards the bottom of the page.

The "Profile Details" table which stores the questionnaire responses did not require any structure changes at all.

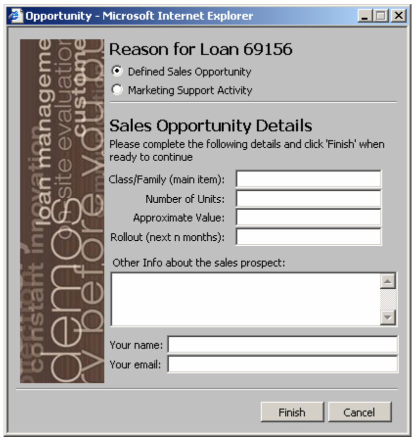


Questionnaires – Data Store

Profile Details	
Object Type	TEXT (8)
Object ID	TEXT (16)
Category	TEXT (16)
Sub Category	TEXT (16)
Free Format Info	VARCHAR

In Practise

So far we have focussed on the table structures, but not really touched on how that data is read and displayed.



Opportunity - Microsoft Internet Explorer

Reason for Loan 69156

● Defined Sales Opportunity
○ Marketing Support Activity

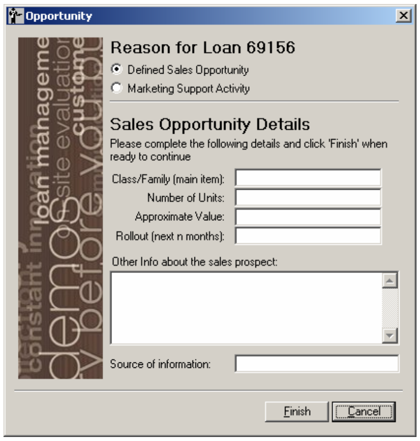
Sales Opportunity Details

Please complete the following details and click 'Finish' when ready to continue

Class/Family (main item):
Number of Units:
Approximate Value:
Rollout (next n months):

Other Info about the sales prospect:

Your name:
Your email:



Opportunity

Reason for Loan 69156

● Defined Sales Opportunity
○ Marketing Support Activity

Sales Opportunity Details

Please complete the following details and click 'Finish' when ready to continue

Class/Family (main item):
Number of Units:
Approximate Value:
Rollout (next n months):

Other Info about the sales prospect:

Source of information:

In these cases we used Visual Basic to write Activex DLLs which read the data from the database and display it in the format we want on the data entry form.

Usng a tiered approach to the ActiveX DLL design also enabled us to reuse the DLL's to provide the same user experience whether from a Windows application or from a web aplication. The pictures above show the same questionnaire being displayed as a web page and as a Windows form. The design allows us to use the same table structures to display the data in Windows or Web format.

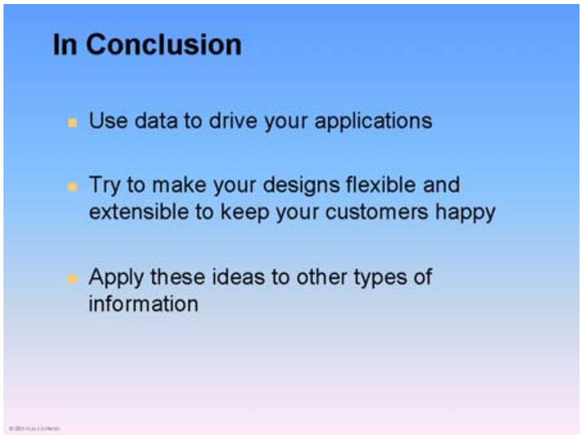
Of course, within a Windows application, you can now do something similar with R:BASE forms as well, particularly now that more and more object properties are being exposed within the forms environment.

Conclusions

What can we conclude from our “classified designs”?

Using the data in the database to drive your application, as well as being a repository for the information to store, makes the application easy to modify and extend.

By a simple change or addition to the data, the customer can change the way the application runs without coming back to you, the developer.



In Conclusion

- Use data to drive your applications
- Try to make your designs flexible and extensible to keep your customers happy
- Apply these ideas to other types of information

There are many more examples where we use data within the database to drive our applications. Over the years, this has helped us to produce generic tools which we can use in different guises for different customers saving many hours of development time.

We’re sure that you have lots of your own ideas as well!

How do I find out more?

To find out more about the themes we have covered you may want try the following resources:

Oterro & Visual Basic

- ✓ You can find out more from www.oterro.com
- ✓ Additional references can also be found from the paper available at www.aspensoftware.co.uk/RBASE/otrecycle.htm

Other Resources

- ✓ www.asb2.com for information about the Aspen Software generic tools that have been used in the production of the examples featured in this paper.
- ✓ info@aspensoftware.co.uk if you have any questions about this subject.

References & Acknowledgements

- ✓ Niall Young, Oracle Consulting – “SQL & Generic Databases” presented at the RUG UK 4th International Conference 1994.
- ✓ Questionnaire “images” produced courtesy of Added Dimension Ltd www.added-dimension.co.uk.