Web Application Maker 2.14

Manual (Preliminary Draft)

16-October-2009, WAM 2.13

Mário Araújo

Miguel Calejo

Sónia Mota

With contributions by Alexandra Baldaque,Carlos Éloi Barros, Luis Carvalho and others

TABLE OF CONTENTS

1 Introduction 5
1.1 What is it 5
1.2 The WAM multilevel spaghetti problem 5
1.3 History and credits 7
1.4 Application Portfolio 7

2 WAM user interface: an application tour 11
2.1 Northwind application 11
2.2 Entry page 11
2.3 Browsing around: opening lists and rows, and following lookups 13
2.4 Personalizing a list 17
2.5 Finder fields, simple list searches, and list filters 20
2.5.1 Finder Fields 20
2.5.2 Simple list searching 21
2.5.3 List filters: searches to remember 25
2.6 Changing and editing data 28
2.7 User interface shortcuts 32
2.8 Philosophy 32
2.9 GUI fragments/parts 34
2.10 Foreign key graph 37
2.11 Database objects and how they impact GUI generation: GUI patterns 38

3 WAM development tour 40
3.1 Webifying the 'pubs' database 40
3.1.1 Step 1: Get a database 40
3.1.2 Step 2: Generate default WAMmodel 41
3.1.3 Step 3: Demo 43
3.2 Introducing the user interface for WAM applications 44
3.2.1 Personalizing the information displayed in a list 44
3.2.1.1 Adding/removing columns 45
3.2.1.2 Display search filter 46
3.2.2 Navigation between tables 47
3.2.2.1 Detail to Master Navigation 48
3.2.2.2 Master to Detail Navigation 48
3.3 WAMmodel (explicit) improvements 48
3.3.1 Adding labels/captions 49
3.3.2 Model-based row layout tuning 51
3.3.3 Adapt a lookup for a row 52
3.3.4 Add a (deep) detail list 55
3.3.5 Make a detail list embedded in a row 56
3.4 Improving the database 57
3.4.1 Adding tables 57
3.4.2 Add a check constraint to a table 58
3.4.3 Add fields to a table 59
3.4.4 Add a trigger with parameterized error message 62
3.5 ASP scripting 63
3.5.1 Improve the entry page 63
3.5.2 Customize a row page 65
3.5.3 *** Enhancing appearance with CSS 67
3.5.4 Add a browser client script 67
3.5.5 Add an external ASP 69
3.5.6 Tour end - architecture overview 70

4 Database server layer: meta information and WAMmodel 72
4.1 WAMmodel: the WAM tables 72
4.1.1 WAM_PRESENTATION 73
4.1.1.1 Multi-lingual support 74
4.1.2 WAM_ROW_COLUMN 74
4.1.3 WAM_LOOKUP_COLUMN 75
4.1.4 WAM_LIST 75
4.1.5 WAM_LIST_COLUMN 75
4.1.6 WAM_CRITERION 76
4.1.7 WAM_PROCEDURE_CALL 76
4.1.8 WAM_ROW_GROUP 77
4.1.9 WAM_TABLE_GROUP 77
4.1.10 WAM_TREE 78
4.1.11 WAM_CONSTRAINT_VIEW_USAGE 78
4.1.12 WAM_HELP 78
4.1.13 WAM_PREFERENCES 79
4.1.14 WAM_FINDER 79

4.2 Treating VIEWS (almost) as tables 79
4.2.1 VIEW updating....80
4.3 SQL and system permissions and how they are used....80
4.4 Predefined SQL user data types....80
4.4.1 wamdate, wamtime and wammoment....80
4.4.2 wamhierarchical1 and wamhierarchical2....81
4.4.3 wamphone 81
4.4.4 wamemail, wamfax and wamurl....81
4.4.5 wamimage 81
4.4.6 wamexternalimage....81
4.4.7 wamexternalfile 81
4.4.8 wamcustomresource (for customized file upload and download) 81
4.4.9 wamcolor (for color pickers) 82
4.4.10 wamrange 82
4.4.11 wamglatlng (for Google Maps)....82
4.4.12 wamclass (for entity subclassing support) 82
4.5 *** Support for free text search....82
4.6 Support for object-oriented modeling....83
4.7 Initial WAMmodels to start development....83
4.7.1 Empty WAMmodel 83
4.7.2 Default WAMmodel....83
4.8 Better delete/update cascading for SQL Server....84
4.8.1 The two additional WAMmodel tables....84
4.8.1.1 WAM_DELETE_RULES.... 84
4.8.1.2 WAM_UPDATE_RULES 84
4.8.2 Data manipulation procedures 85
4.8.2.1 WAM_Delete.... 85
4.8.2.2 WAM_Update.... 85
4.9 About SQL engine built-in meta information 86
4.9.1 SQL-92's INFORMATION_SCHEMA 86
4.9.2 Additional VIEWs and tables 86
4.9.2.1 WAM_DETAIL_LISTS 86
4.9.2.2 WAM_REFERENTIAL_CONSTRAINTS.... 86
4.9.2.3 WAM_STORED_PROCEDURES 87
4.9.2.4 WAM_AUDIT_LOG.... 87
4.10 WAMmodel SQL utilities....87
4.10.1 Copying list preferences between users 88
4.10.2 Copying WAMmodel definitions for a table to a view 88
4.10.3 Editing foreign keys in a WAMmodel....88

5 Web server and browser layers: the WAMLibrary objects....89
5.1 Dispensing with ASP programming: the standard pages 89
5.1.1 A standard entry page: default.asp....89
5.1.1.1 Customizing the entry page.... 89
5.1.2 All other pages: standard.asp....90
5.2 The default interface navigational structure....90
5.3 About custom pages and WAM directories 92
5.3.1 File directory structure for a WAM application....92
5.3.2 WAM internals: WAMLibrary directory structure 92
5.3.2.1 Files in root directory.... 92
5.3.2.2 WAMLibrary subdirectories 93
5.4 Programming ASPs with WAM objects 95
5.4.1 Including the WAMLibrary....95
5.4.2 Using the WAMStandardGUI object....95
5.4.2.1 Implementing a page for a row, without customization.... 95
5.4.2.2 A row with some customization.... 95
5.4.2.3 A row with embedded list in the middle.... 95
5.4.2.4 A page for a standalone list, without customization 96
5.4.2.5 A page for a standalone list, with some customization 96
5.4.2.6 A page for a standalone detail list, without customization.... 96
5.4.2.7 A page for a standalone detail list, with user customization 96
5.4.3 Client Javascripting 97
5.4.3.1 Predefined variables in a (customized) row.... 97
5.4.3.2 Predefined variables in a (customized) list.... 97
5.4.3.3 WAM high-level events 97
5.4.4 WAM events for server Javascript....98
5.4.5 Row objects....98
5.4.5.1 WAMRow 98
5.4.5.2 WAMRowColumn.... 98
5.4.5.3 WAMRowLookup 99
5.4.5.4 How to execute lazy lookups.... 101
5.4.6 Operations: WAMCaller....102
5.4.6.1 WAM caller user arguments: WAMA.... 102
5.4.6.2 How to control the enabled state of the WAM callers' buttons (enabled/disabled).... 103
5.4.6.3 How to execute WAMCallers with more arguments, beyond primary key columns.... 104
5.4.6.4 How to obtain results from a WAMCaller, and to open a window depending on them.... 105
5.4.7 Lists: WAMList....105
5.4.7.1 How to create a customized List.... 107
5.4.7.2 How to build lists referring tables from external DBS.... 107

5.4.7.3 Allowing multiple selection ... 108
5.4.7.4 Invoking row callers from a list ... 108
5.4.8 Other list objects ... 109
5.4.8.1 WAMListColumn ... 109
5.4.8.2 WAMListCriterion (search filter) ... 110
5.4.8.3 WAMEmbeddedDetailList ... 110
5.4.8.4 WAMStandaloneDetailList ... 111
5.4.9 Other objects ... 111
5.4.9.1 WAMTableGroup ... 111
5.4.9.2 WAMListGroup ... 111
5.4.9.3 WAMFinder ... 111
5.4.9.4 WAMFinderGroup ... 111
5.4.9.5 WAMTree ... 111
5.4.9.6 WAMDialog ... 112
5.4.9.7 The WAMAPI object ... 113
5.4.10 How to customize the default entry page ... 113
5.5 Global WAM variables: the global.asa file ... 114
5.6 Global WAM styles and GIFs ... 115
5.6.1 GIF icons ... 115
5.6.2 CSS styles ... 116
5.7 WAM cache and its freshness vs. the database ... 116
5.8 Custom user authentication ... 117
5.9 WAP support – the WAM Mobile interface ... 118

6 Error and warning Handling ... 119
6.1 Raising and handling errors in the SQL layer ... 119
6.1.1 Raising and handling warnings in the SQL layer ... 119
6.2 Raising and handling errors in the ASP (web server) layer ... 119
6.3 Raising and handling errors in the web client layer ... 120

7 WAMAdmin ... 121
7.1 Browser-based WAMmodel editing ... 121
7.2 WAM API Reference ... 124
7.3 Developer Documentation ... 124
7.4 Database structure Information ... 124
7.5 WAMmodel consistency with database structure ... 124
7.6 Database Compare (dbCompare) ... 126
7.7 Available databases (dbInspector) ... 126
7.8 *** Other WAMAdmin features ... 127

8 WAM installation and application setup ... 128
8.1 Using the installer ... 128
8.2 Manual installation ... 128
8.2.1 WAMmodel setup ... 128
8.2.2 Web application setup ... 128
8.2.3 Removing WAM ... 128
8.3 Browser limitations: Firefox and Safari ... 129

9 Misc ... 130
9.1 Hands-on workshop template ... 130
9.1.1 Basics ... 130
9.1.2 Installing WAM and simple WAMmodel tweaking ... 130
9.1.3 FK path wonders ... 130
9.1.4 Better databases ... 131
9.1.5 Finally, some web application code! ... 131
9.2 WAM model-customization continuum and project methodology ... 131
9.3 Comparison with other tools ... 133
9.4 ***Performance considerations ... 133
9.5 Help system ... 133
9.5.1 User perspective ... 133
9.5.2 Developer perspective: adding help content ... 135
9.5.2.1 WAM_HELP Description tags ... 137
9.6 Charts for lists ... 138
9.7 References ... 139

1 Introduction

1.1 What is it

Declarativa's Web Application Maker (WAM) is a software development tool to build and maintain web interface front-ends to relational database back-ends, using a *model-based* approach. It does *not* generate source code to be worked upon: it is a runtime model-driven system.

Whereas existing model-based user interface development environments are based on models invented or adapted specifically for the interface development process[1], WAM is pragmatically based on a mainstream application model: the *relational database schema* itself. One may get a flavor by reading "WAM development tour".

WAM takes a database with its built-in meta-data, and automatically defines a reasonable multi-user profile browser GUI for data entry and browsing, ready for easy customization and extension. Further iterative application development is based on pursuing two vectors of change:

- *Enriching a simple meta-model*, WAMmodel, a set of database tables complementing the built-in meta-data of the database engine; and enhancing the database structure and stored procedures
- Higher-level and more robust ASP coding, as compared to current industry practice. The ASP development experience is significantly enhanced by the WAM system because it *exposes the GUI database model-driven generation capabilities at runtime*, through the WAMLibrary API.

WAM's innovation consists in the articulated combination of:

- Use of WAMmodel, a small database resident declarative model, complemented with the dynamic introspection of the standard database engine's built-in meta-information; this minimizes conflicts between database schema changes and application changes
- GUI personalization and improvement through the end application itself, which can incorporate WAMmodel edition capabilities
- Just-in-time generation of reactive user interface fragments (already including browser and server data type validation) from concise declarative specifications
- Making the GUI generating capabilities available to the application programmer at runtime for his/her customized code, through a high level JavaScript API

This last aspect encourages a programming style that minimizes the traditional conflict between model-driven development and customized "handcrafted" code. Because in addition to encapsulating data editing/browsing functionality, as in other GUI components for database access, WAM runtime objects accept declarative (database) model references as method arguments, much less verbose and minimizing code dependencies on database schema changes.

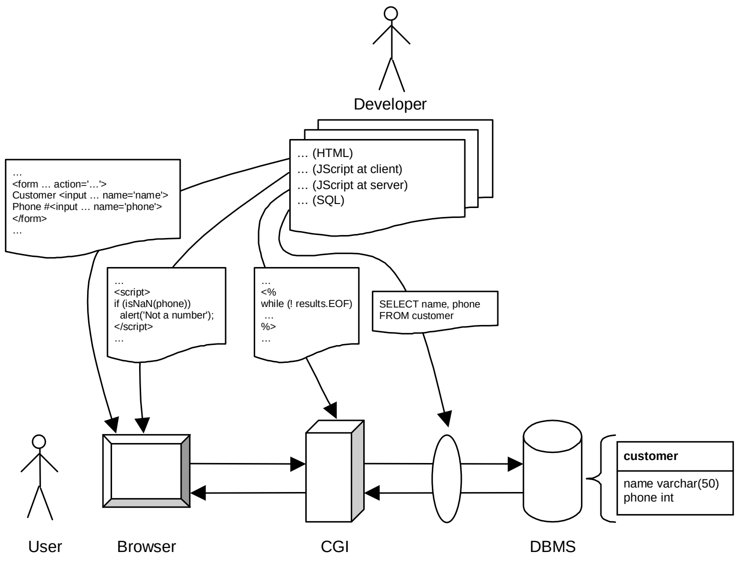
As illustrated in the next section, it's easier and more robust to obtain the bulk of a form at runtime by writing `new WAMRow(SomeDBTable)` than having to specify a SELECT statement, field titles and types, follow some data binding convention, etc.

1.2 The WAM multilevel spaghetti problem

Web development, and in particular web database development, has made Meta programming a mainstream practice, as developers routinely produce single code fragments amalgamating diverse levels, referring database objects, web server objects and web client objects, such as in a PHP or an Active Server Page (ASP) file [Microsoft 2001a]. But, no matter

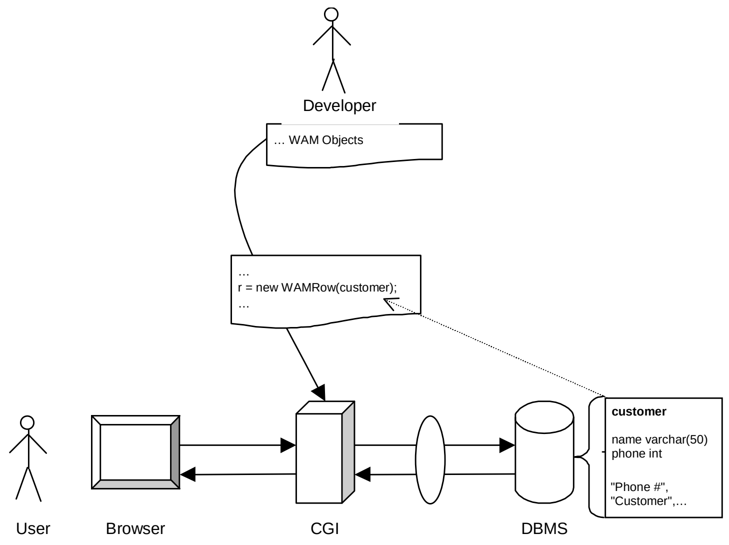
[1] See [Griffiths et al 1998] [Silva 2000] for surveys

how much syntactic coloring and sugaring IDE editors provide, the result is an organized mess and a maintenance nightmare, as can be depicted for the Classic ASP platform:



Although declarative object models are present in the browser and CGI layers, there's no use of the intrinsic DBMS model. The programmer fabricates SQL strings and gets back structured table/view results, but he/she must maintain the bindings with the other layers.

WAM simplifies the situation by introducing an extra Meta level whose domain of discourse is made of DBMS concepts, and by encapsulating code generation functionality into runtime reusable objects based on the application's data model:



WAM does not include business domain knowledge; it's intended instead to complement other tools, by adding its database-oriented web interface development capabilities. It supports Microsoft SQL Server 2000/2005, Oracle and

Informix as database engines[2], Active Server Pages at the CGI layer, and Internet Explorer, Firefox and Safari browsers; it also offers preliminary support for WAP phones.

***** *To include later: comparison, in terms of focus and implementation, with ASP.net and Java Server Faces (they focus on GUI abstractions, we focus on data layer abstractions, etc.). See 9.3 for more details.***

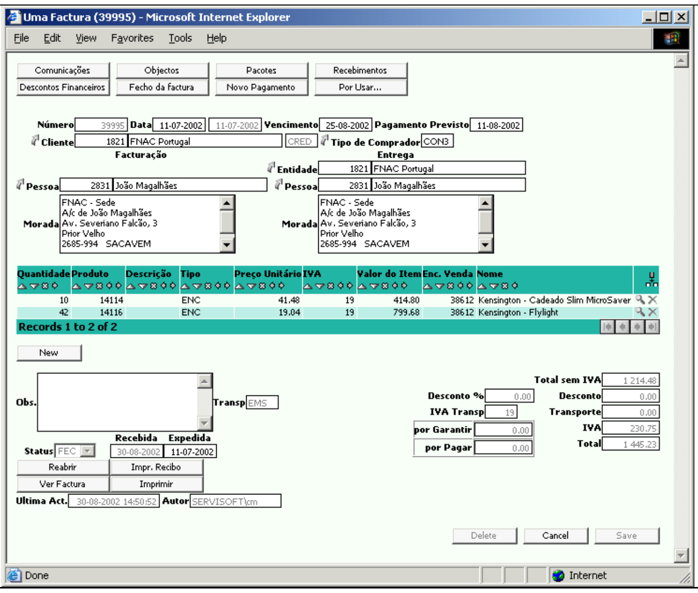
1.3 History and credits

WAM emerged from Declarativa custom web application projects, and it now underlies all of them. At this stage WAM development incorporates approximately 6 person-years of dedicated development over 2000-2007, reflecting 20 + person-years of professional experience developing web applications plus another 15 on other platforms.

In addition to the authors other people worked on parts of the WAM implementation: Nuno Soares (WAMmodel triggers, web showroom) and Rui Marante (WAP front-end). Thanks also to contributions to this manual by Alexandra Baldaque, Luis Sousa, Hernâni Fernandes and Carlos Elói Barros.

1.4 Application Portfolio

WAM has been used to generate whole or parts of most web pages in the following projects, among others:

Customer / project	*Screen shots*
Servisoft, Porto, a hardware/software distributor: extension and re-implementation over intranet/extranet of its Information System, a custom-made "ERP light" system; to cover all major activities from web click-stream monitoring to invoicing, supply shipment tracking etc. About 53 database tables, 4 distinct user profiles, hundreds of SQL stored procedures with business logic. Used by 7 people in-house, 300 dealers over extranet, a few thousand "anonymous" users through public site.	

Número 39995 Data 11-07-2002 11-07-2002 Vencimento 25-08-2002 Pagamento Previsto 11-08-2002
Cliente 1821 FNAC Portugal CRED Tipo de Comprador CON3

Facturação
Pessoa 2831 João Magalhães
Morada FNAC - Sede
A/c de João Magalhães
Av. Severiano Falcão, 3
Prior Velho
2685-994 SACAVEM

Entrega
Entidade 1821 FNAC Portugal
Pessoa 2831 João Magalhães
Morada FNAC - Sede
A/c de João Magalhães
Av. Severiano Falcão, 3
Prior Velho
2685-994 SACAVEM

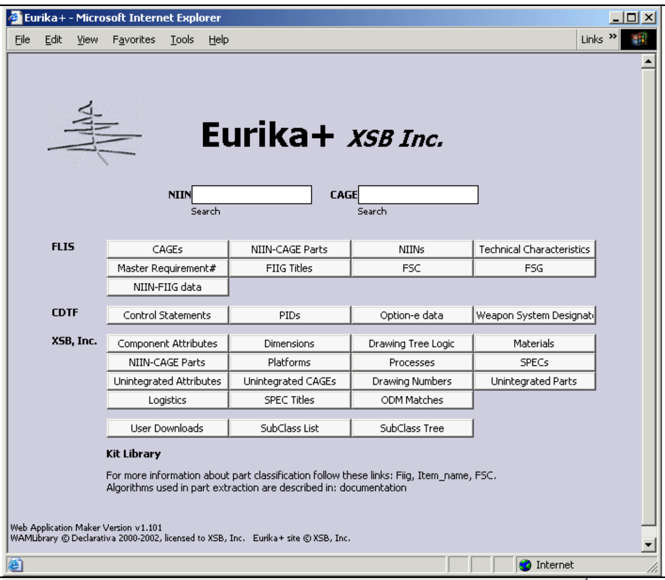
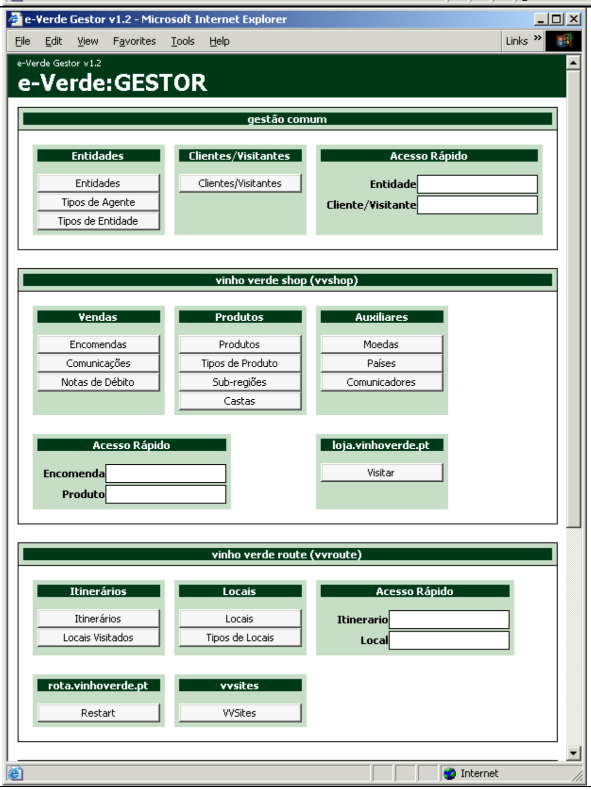
Quantidade	Produto	Descrição	Tipo	Preço Unitário	IVA	Valor do Item	Enc. Venda	Nome
10	14114		ENC	41.48	19	414.80	38612	Kensington - Cadeado Slim MicroSaver
42	14116		ENC	19.04	19	799.68	38612	Kensington - Flylight

Records 1 to 2 of 2

Obs.
Transp EMS
Status FEC
Recebida 30-08-2002 Expedida 11-07-2002
Ultima Act. 30-08-2002 14:50:52 Autor SERVISOFT\cm

Desconto % 0.00
IVA Transp 19
por Garantir 0.00
por Pagar 0.00
Total sem IVA 1 214.48
Desconto 0.00
Transporte 0.00
IVA 230.75
Total 1 445.23

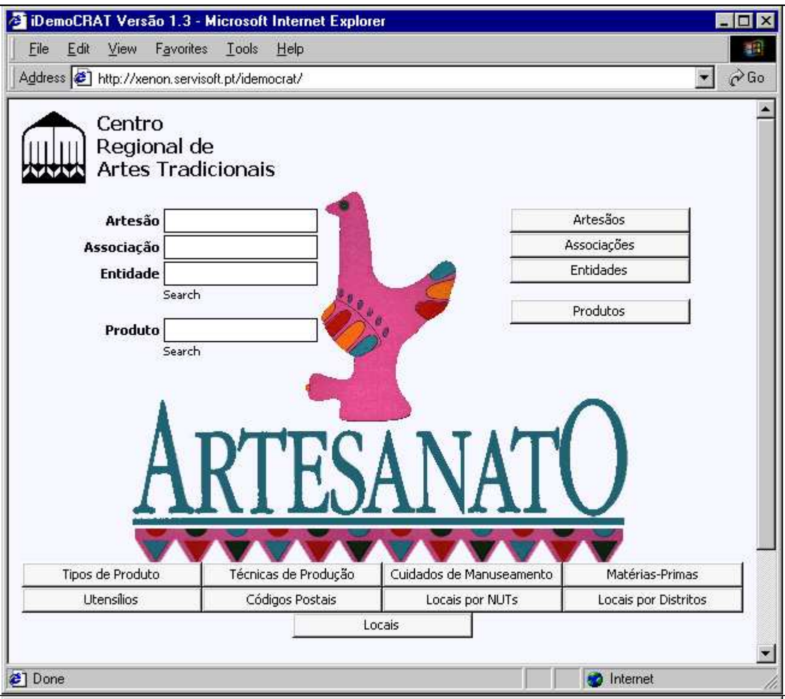
[2] The present document focuses on the SQL Server version. Oracle 9 support will be available summer 2007; the Informix IDS9 is slightly behind and may be discontinued in the future

XSB, Inc., Stony Brook, NY, USA: front-end to SQL Server database, project for DoD's Defense Logistics Agency. About 30 tables, 2 distinct user profiles, many millions of records.	
Vinho Verde Commission, Porto: regional portal for the largest wine region in Europe, where WAM is being used in all backoffice modules; several extranet modules to remotely fill forms for wine production declaration and other processes.	

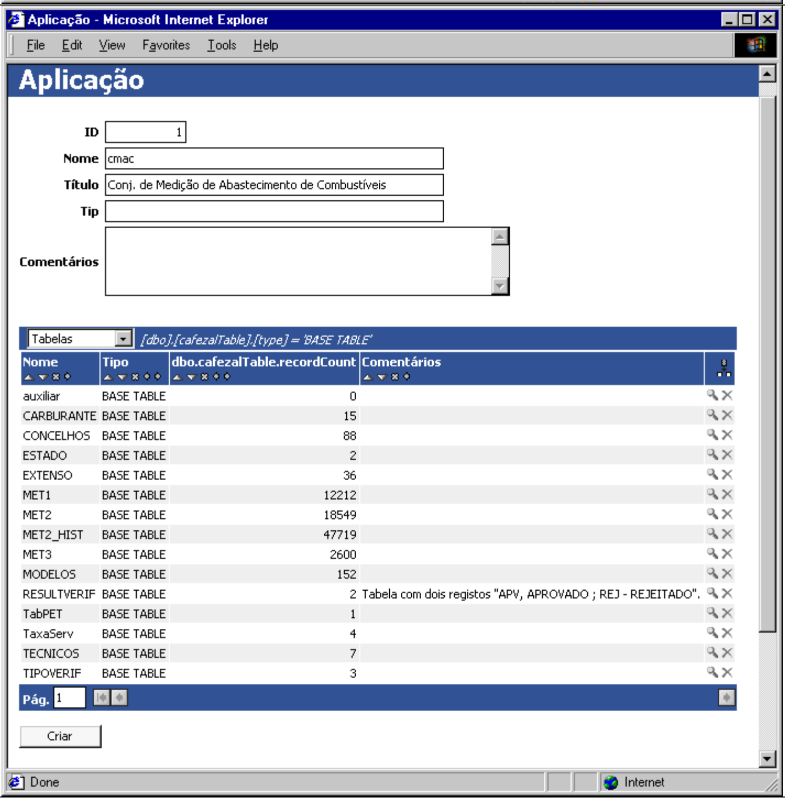
For more information about part classification follow these links: Fiig, Item_name, FSC.
Algorithms used in part extraction are described in: documentation

Web Application Maker Version v1.101
WAMLibrary © Declarativa 2000-2002, licensed to XSB, Inc. Eurika+ site © XSB, Inc.

Regional Center for Craftsmanship (**CRAT**), Porto: small intranet app to maintain their database (places, products, artisans) with images etc., including a synchronized public online version. About 12 tables, 3 user profiles. Entry page for internal users shown here.



Portuguese Ministry of Economy, North Delegation, Porto:
Institutional meta-information repository documenting all existing software applications, shown here.

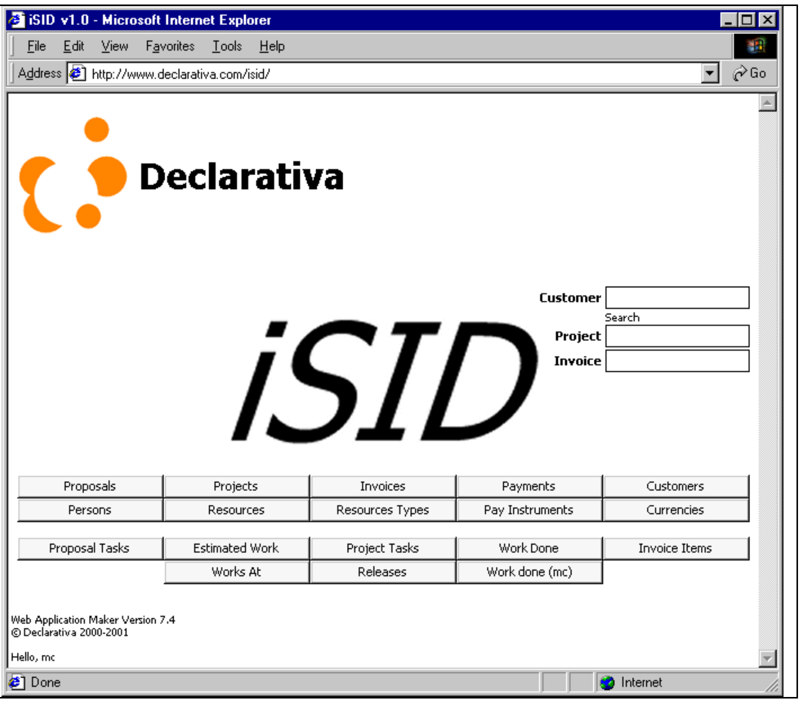

Aplicação - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Aplicação
Criar
Done
Internet

ID: 1
Nome: cmac
Título: Conj. de Medição de Abastecimento de Combustíveis
Tip:
Comentários:

Tabelas [dbo].[cafezalTable].[type] = 'BASE TABLE'

Nome	Tipo	dbo.cafezalTable.recordCount	Comentários
auxiliar	BASE TABLE	0	
CARBURANTE	BASE TABLE	15	
CONCELHOS	BASE TABLE	88	
ESTADO	BASE TABLE	2	
EXTENSO	BASE TABLE	36	
MET1	BASE TABLE	12212	
MET2	BASE TABLE	18549	
MET2_HIST	BASE TABLE	47719	
MET3	BASE TABLE	2600	
MODELOS	BASE TABLE	152	
RESULTVERIF	BASE TABLE	2	Tabela com dois registos "APV, APROVADO ; REJ - REJEITADO".
TabPET	BASE TABLE	1	
TaxaServ	BASE TABLE	4	
TECNICOS	BASE TABLE	7	
TIPOVERIF	BASE TABLE	3	

Pág. 1

Declarativa, Porto: application for invoicing and project control, includes interface for customers	

iSID v1.0 - Microsoft Internet Explorer

Address http://www.declarativa.com/isid/

Customer

Search

Project

Invoice

Proposals	Projects	Invoices	Payments	Customers
Persons	Resources	Resources Types	Pay Instruments	Currencies
Proposal Tasks	Estimated Work	Project Tasks	Work Done	Invoice Items
	Works At	Releases	Work done (mc)	

Web Application Maker Version 7.4
© Declarativa 2000-2001

Hello, mc

Done

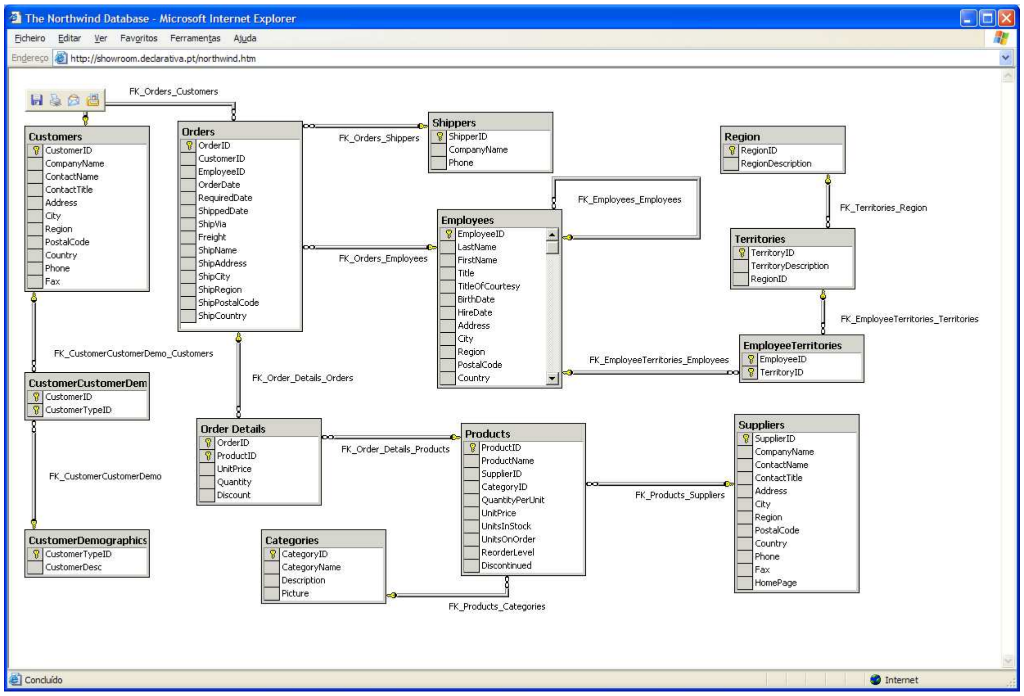
2 WAM user interface: an application tour

We'll show the look and feel of a WAM application for a sample database, the well-known Northwind database[3]. The images below were taken from an unfinished prototype running at http://showroom.declarativa.pt , a web site where some WAM application prototypes can be tried and modified.

The WAM show room provides several versions of each application, with more or less elaboration by the developer. We'll use "NorthwindB", an **unfinished version** totally dependent only on its WAMmodel: it has **zero lines of code** beyond the original database structure.

2.1 Northwind application

The Northwind database application manages sales data for a fictitious company "Northwind Traders", which imports and exports specialty foods from around the world. The underlying database tables cover costumers, products, orders and some related concepts:

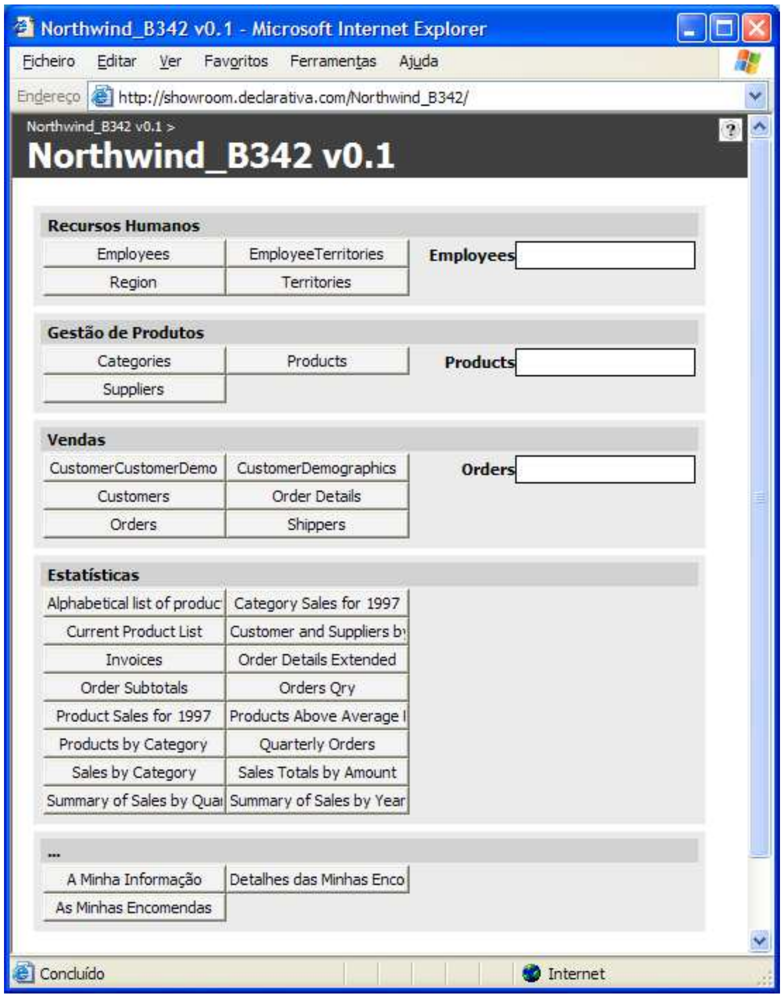


Customers place orders, with the assistance of an employee, using a particular shipper. Each order line (detail) refers a single product, which has a supplier; products are typified in categories. Employees may have a supervisor, and they cover a sales territory; territories are aggregated in regions. Etc.

2.2 Entry page

This is the main page, from where data browsing and editing is available for the whole database:

[3] A Microsoft sample database obtainable at http://www.microsoft.com/downloads/details.aspx?familyid=06616212-0356-46a0-8da2-eebc53a68034&displaylang=en



There we have four[4] main groups (*** English captions missing for WAM_TABLE_GROUPs!) organized like this:

- The first group, **Human Resources** has four tables:
  - Employees – which has all employees' personal details.
  - Employees by territory – where we can find which territories each employee works with.
  - Territories – where we can find the territories where this enterprise operate.
  - Regions – the main territory regions.
- **Product Management** has three tables:
  - Categories – where we can see each category of products for sale.
  - Products - where we can see all products details.
  - Suppliers - where we can see all suppliers details.
- **Sales** has six tables:
  - Demographic characteristics

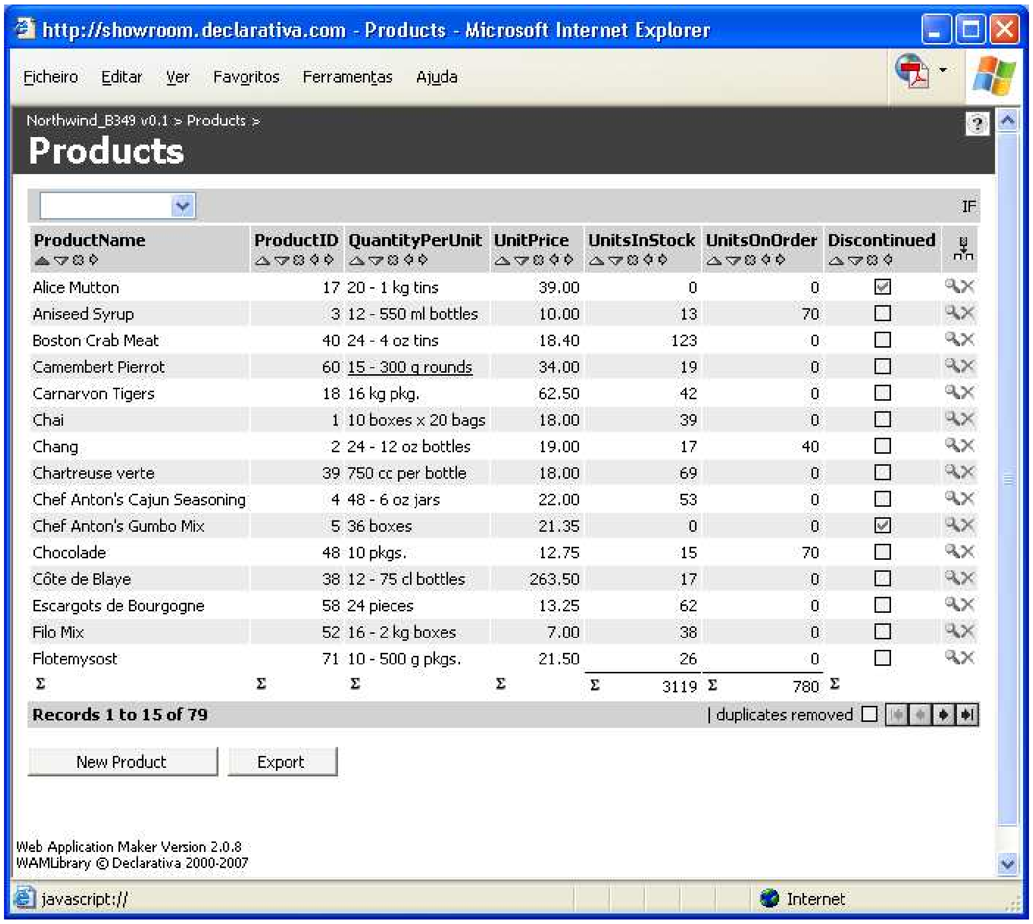
[4] The remaining (fifth) group refers tables whose information was not explicitly grouped by the developer

  - o Customer Demographics
  - o Costumers - where we can see all costumers details.
  - o Orders - where we can see all orders main information like Order date, Shipped date, etc.
  - o Orders details - where we can see all orders lines irrespective of which order they belong to
  - o Shippers - where we can see shipping company details.
- The fourth group named Statistics contains several options to showing lists of data ordered by a certain criteria.

In each of first three groups we have a “finder field”, through we can quickly find specific orders given a (table primary key) value.

2.3 Browsing around: opening lists and rows, and following lookups

To see data about a topic, we simply click the appropriate button; clicking the Products button in the **Product Management** group another new web page opens up, showing Product Name, ID, etc.:

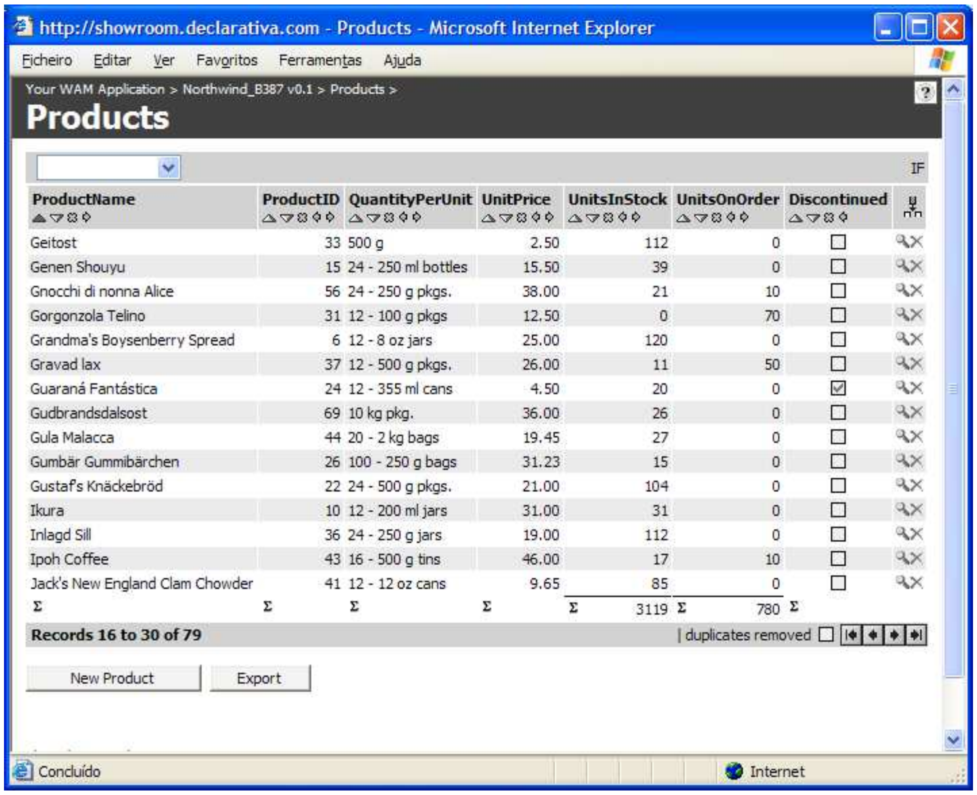

http://showroom.declarativa.com - Products - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_B349 v0.1 > Products >
Products
IF

ProductName	ProductID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOnOrder	Discontinued
Alice Mutton	17	20 - 1 kg tins	39.00	0	0	☑
Aniseed Syrup	3	12 - 550 ml bottles	10.00	13	70	☐
Boston Crab Meat	40	24 - 4 oz tins	18.40	123	0	☐
Camembert Pierrot	60	15 - 300 g rounds	34.00	19	0	☐
Carnarvon Tigers	18	16 kg pkg.	62.50	42	0	☐
Chai	1	10 boxes x 20 bags	18.00	39	0	☐
Chang	2	24 - 12 oz bottles	19.00	17	40	☐
Chartreuse verte	39	750 cc per bottle	18.00	69	0	☐
Chef Anton's Cajun Seasoning	4	48 - 6 oz jars	22.00	53	0	☐
Chef Anton's Gumbo Mix	5	36 boxes	21.35	0	0	☑
Chocolade	48	10 pkgs.	12.75	15	70	☐
Côte de Blaye	38	12 - 75 cl bottles	263.50	17	0	☐
Escargots de Bourgogne	58	24 pieces	13.25	62	0	☐
Filo Mix	52	16 - 2 kg boxes	7.00	38	0	☐
Flotemysost	71	10 - 500 g pkgs.	21.50	26	0	☐
Σ	Σ	Σ	Σ	Σ 3119	Σ 780	Σ

Records 1 to 15 of 79 | duplicates removed ☐

New Product
Export
Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007
javascript://
Internet

The bottom data line shows the total of units in stock. Below that there’s a navigation bar, allowing navigation in the data set the list is displaying. Clicking the ▶ button:


http://showroom.declarativa.com - Products - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Products >
Products
IF
New Product
Export
Concluído
Internet

ProductName	ProductID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOnOrder	Discontinued
Geitost	33	500 g	2.50	112	0	☐
Genen Shouyu	15	24 - 250 ml bottles	15.50	39	0	☐
Gnocchi di nonna Alice	56	24 - 250 g pkgs.	38.00	21	10	☐
Gorgonzola Telino	31	12 - 100 g pkgs	12.50	0	70	☐
Grandma's Boysenberry Spread	6	12 - 8 oz jars	25.00	120	0	☐
Gravad lax	37	12 - 500 g pkgs.	26.00	11	50	☐
Guaraná Fantástica	24	12 - 355 ml cans	4.50	20	0	☑
Gudbrandsdalsost	69	10 kg pkg.	36.00	26	0	☐
Gula Malacca	44	20 - 2 kg bags	19.45	27	0	☐
Gumbär Gummibärchen	26	100 - 250 g bags	31.23	15	0	☐
Gustaf's Knäckebröd	22	24 - 500 g pkgs.	21.00	104	0	☐
Ikura	10	12 - 200 ml jars	31.00	31	0	☐
Inlagd Sill	36	24 - 250 g jars	19.00	112	0	☐
Ipoh Coffee	43	16 - 500 g tins	46.00	17	10	☐
Jack's New England Clam Chowder	41	12 - 12 oz cans	9.65	85	0	☐
Σ	Σ	Σ	Σ	Σ 3119	Σ 780	Σ

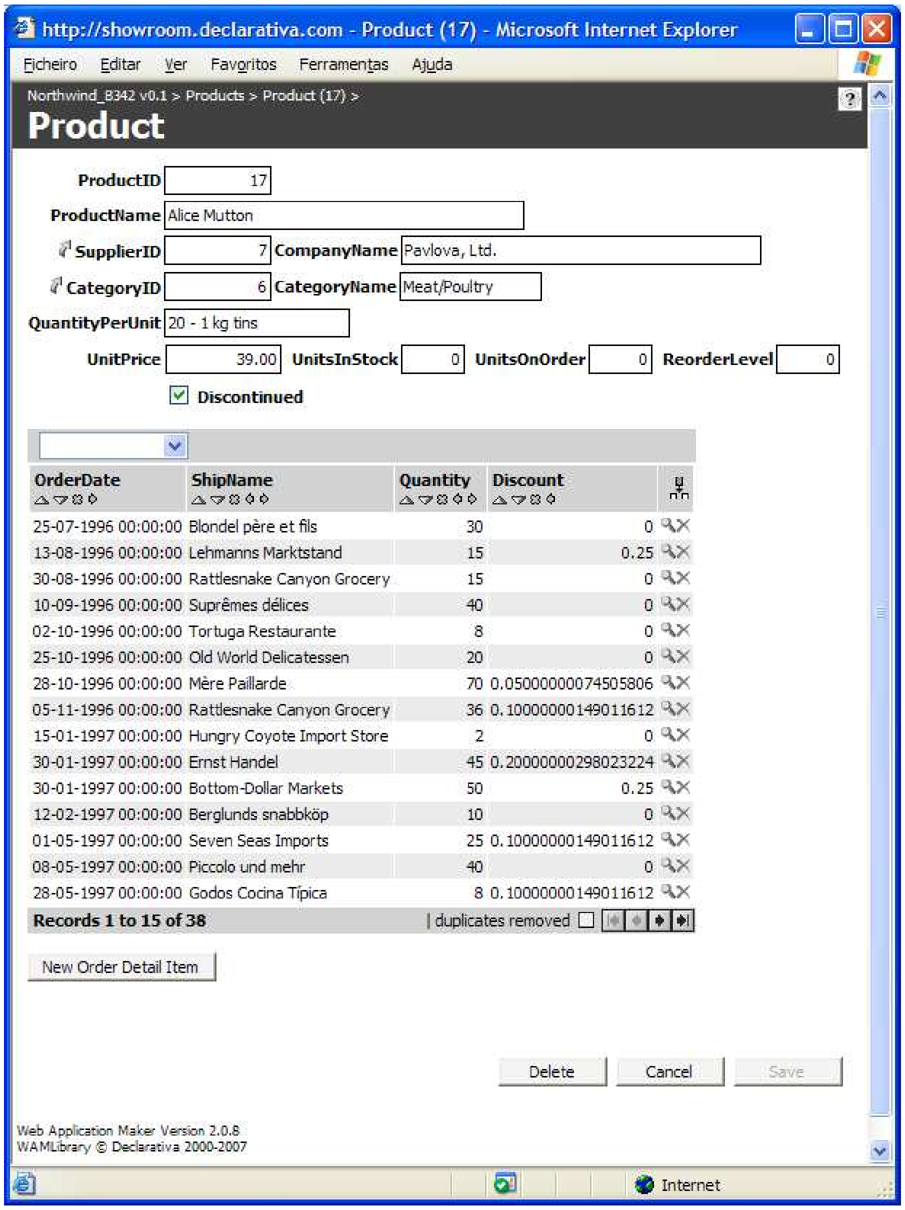
Records 16 to 30 of 79 | duplicates removed ☐

The other navigation buttons provide the obvious (if not, just hover with the mouse over them…) functions; navigates to the first data page, no matter what part of the data set is being show.

The Export button exports all list data (for the visible columns) into a HTML (but Excel-ready) file to download.

To view in detail (or to edit) a record, we need to click on the "magnifying glass" icon to the right of its data row. For example, clicking that icon for the first line in the list opens up the following new web, with the full data of the chosen product; at its bottom the product orders are shown[5]:

[5] Actually, order (line) details.


http://showroom.declarativa.com - Product (17) - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_8342 v0.1 > Products > Product (17) >

Product

ProductID	17						
ProductName	Alice Mutton						
SupplierID	7	CompanyName	Pavlova, Ltd.				
CategoryID	6	CategoryName	Meat/Poultry				
QuantityPerUnit	20 - 1 kg tins						
UnitPrice	39.00	UnitsInStock	0	UnitsOnOrder	0	ReorderLevel	0

☑ Discontinued

OrderDate	ShipName	Quantity	Discount
25-07-1996 00:00:00	Blondel père et fils	30	0
13-08-1996 00:00:00	Lehmanns Marktstand	15	0.25
30-08-1996 00:00:00	Rattlesnake Canyon Grocery	15	0
10-09-1996 00:00:00	Suprêmes délices	40	0
02-10-1996 00:00:00	Tortuga Restaurante	8	0
25-10-1996 00:00:00	Old World Delicatessen	20	0
28-10-1996 00:00:00	Mère Paillarde	70	0.050000000745058060
05-11-1996 00:00:00	Rattlesnake Canyon Grocery	36	0.1000000014901161 2
15-01-1997 00:00:00	Hungry Coyote Import Store	2	0
30-01-1997 00:00:00	Ernst Handel	45	0.20000000298023224
30-01-1997 00:00:00	Bottom-Dollar Markets	50	0.25
12-02-1997 00:00:00	Berglunds snabbköp	10	0
01-05-1997 00:00:00	Seven Seas Imports	25	0.10000000149011612
08-05-1997 00:00:00	Piccolo und mehr	40	0
28-05-1997 00:00:00	Godos Cocina Típica	8	0.10000000149011612

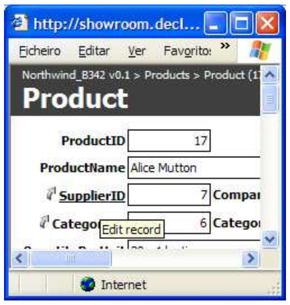
Records 1 to 15 of 38 | duplicates removed ☐

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

As you can see above, to the left of some field titles we have a “lookup button” ; this means that the field relates to information in other database tables[6], some of which may already be shown in the row page (for example, in the image above SupplierID has the supplier company name to the right; this information is in the Suppliers table).

When you hover with the mouse over the name of a field with a lookup, a link appears underneath the field title, together with a help tip “Edit record”; for example, for the SupplierID field:

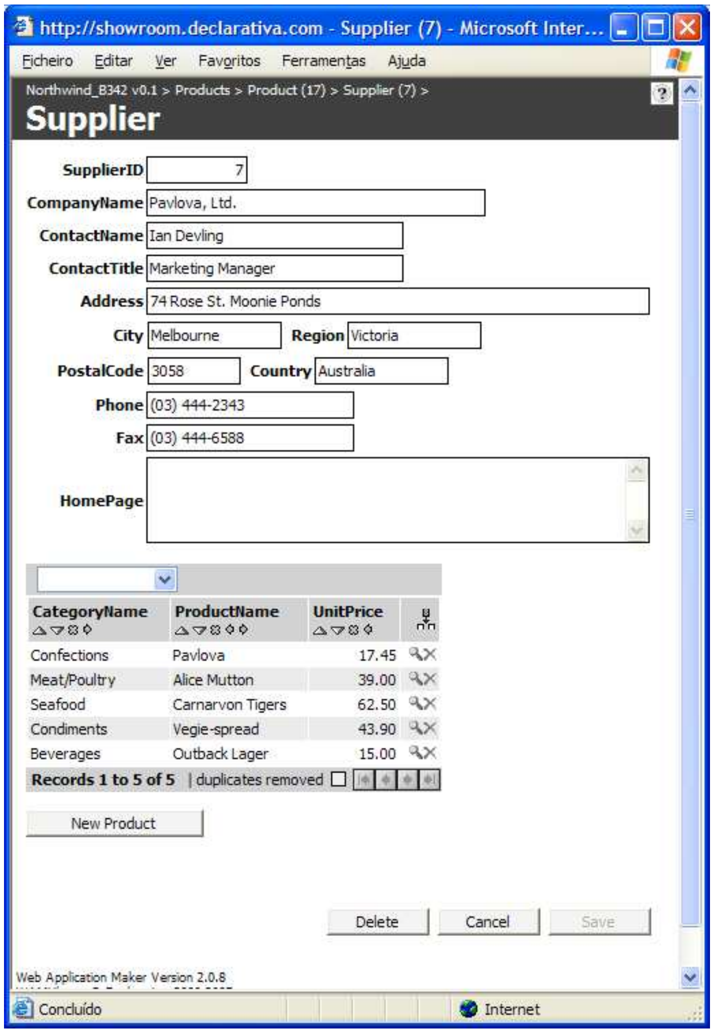
[6] Or that the field is a (or part of a) foreign key in the Products table.


http://showroom.decl...
Ficheiro Editar Ver Favoritos
Northwind_B342 v0.1 > Products > Product (1
Edit record
Internet

Product

ProductID	17	
ProductName	Alice Mutton	
SupplierID	7	Compa
Catego	6	Catego

Clicking the lookup field title link navigates to the related data in the Suppliers table, in a new web page:


http://showroom.declarativa.com - Supplier (7) - Microsoft Inter...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_B342 v0.1 > Products > Product (17) > Supplier (7) >
New Product
Delete
Cancel
Save
Web Application Maker Version 2.0.8
Concluído
Internet

Supplier

SupplierID	7
CompanyName	Pavlova, Ltd.
ContactName	Ian Devling
ContactTitle	Marketing Manager
Address	74 Rose St. Moonie Ponds
City	Melbourne
Region	Victoria
PostalCode	3058
Country	Australia
Phone	(03) 444-2343
Fax	(03) 444-6588
HomePage	

CategoryName	ProductName	UnitPrice
Confections	Pavlova	17.45
Meat/Poultry	Alice Mutton	39.00
Seafood	Carnarvon Tigers	62.50
Condiments	Vegie-spread	43.90
Beverages	Outback Lager	15.00

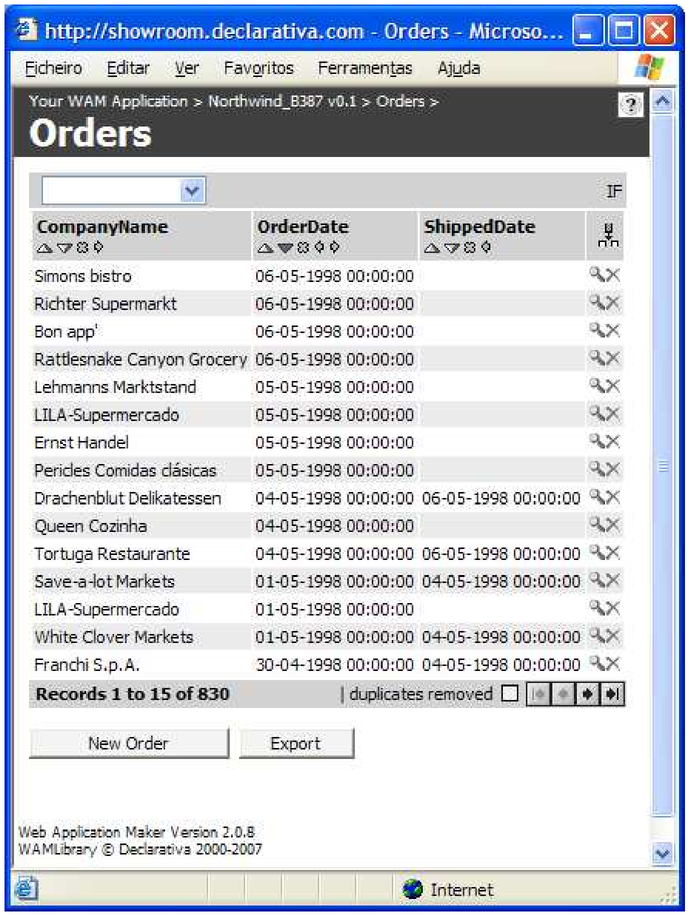
Records 1 to 5 of 5 | duplicates removed

In order to help the user navigate back to the other web pages, the application shows at the very top of the page a "breadcrumb path", with links to all web pages that lead directly to this one. Besides each page title there's its primary value if available; for example, the above link to the previous Product page has the number (17) because that's the Product ID there.

2.4 Personalizing a list

A list can[7] be *personalized* by the application user. The list can be ordered by one or more column fields; fields can be dragged left and right, or omitted from the list; and additional fields can be added from related database tables or views.

The next image shows the Customers list, ordered by OrderDate, showing more recent orders first:


http://showroom.declarativa.com - Orders - Microso...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
IF
New Order
Export
Internet

Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

CompanyName	OrderDate	ShippedDate
Simons bistro	06-05-1998 00:00:00	
Richter Supermarkt	06-05-1998 00:00:00	
Bon app'	06-05-1998 00:00:00	
Rattlesnake Canyon Grocery	06-05-1998 00:00:00	
Lehmanns Marktstand	05-05-1998 00:00:00	
LILA-Supermercado	05-05-1998 00:00:00	
Ernst Handel	05-05-1998 00:00:00	
Pericles Comidas clásicas	05-05-1998 00:00:00	
Drachenblut Delikatessen	04-05-1998 00:00:00	06-05-1998 00:00:00
Queen Cozinha	04-05-1998 00:00:00	
Tortuga Restaurante	04-05-1998 00:00:00	06-05-1998 00:00:00
Save-a-lot Markets	01-05-1998 00:00:00	04-05-1998 00:00:00
LILA-Supermercado	01-05-1998 00:00:00	
White Clover Markets	01-05-1998 00:00:00	04-05-1998 00:00:00
Franchi S.p.A.	30-04-1998 00:00:00	04-05-1998 00:00:00

Records 1 to 15 of 830 | duplicates removed

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Below each field column title there are 4 or 5 small buttons, depending on the column being or not in one of the extremities:

△ sorts the list in ascending order by that field; this button also shows the current ordering, by displaying in black (▲)

▽ sorts descending; the example list is thus

⊠ removes (hides) the column from the list

◇ move the column to the right

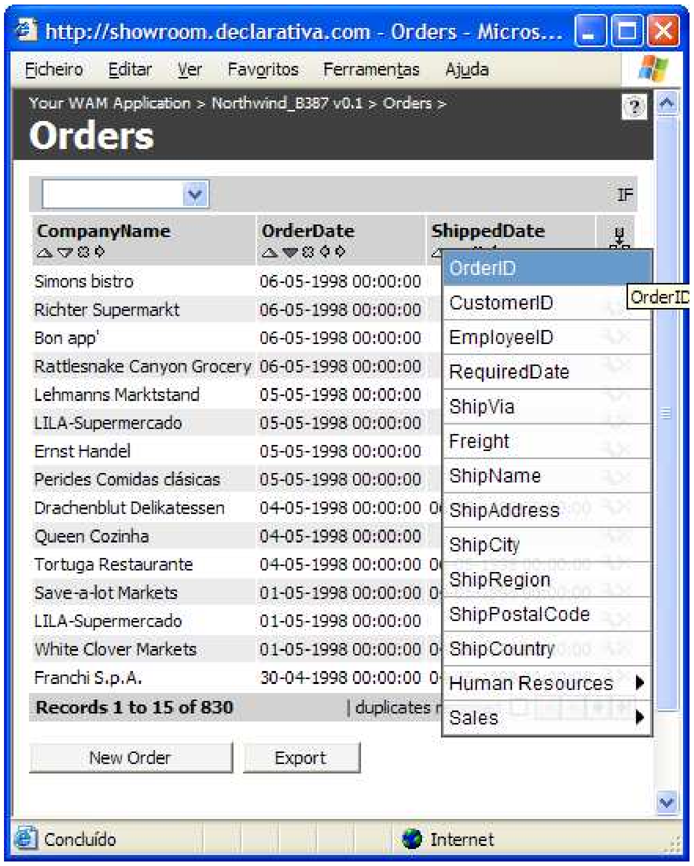
◇ move the column to the left

Instead of using the two last buttons to shift a field a column at a time, the user can simply drag the column title with the mouse.

[7] If SQL permissions in the WAMmodel allow it

A field column can also be resized with the mouse, by dragging the right boundary to the left or right. After resizing, the user can return to the automatic size with double-click on the right boundary.

To add a column to a list, click on the "Add columns to a list" icon, situated on the right at the top of the list; the remaining fields available for the list will be shown in a popup menu.



http://showroom.declarativa.com - Orders - Micros...

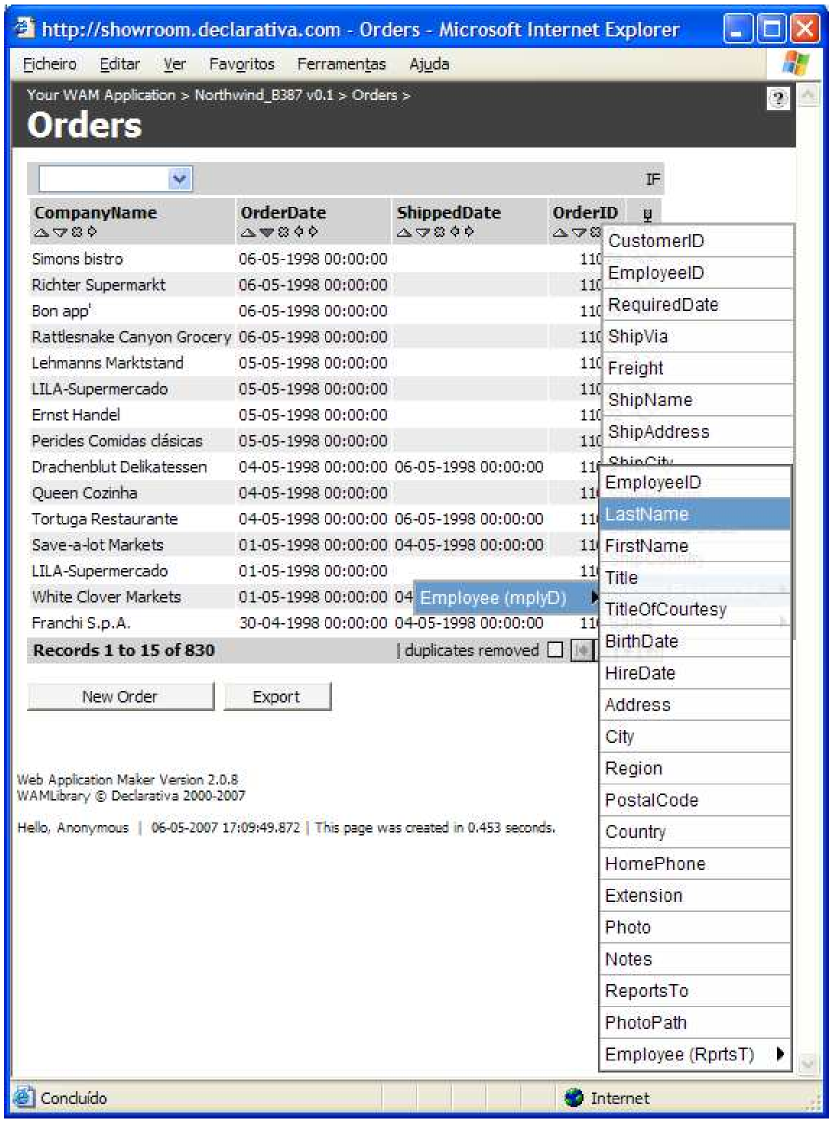
Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

CompanyName	OrderDate	ShippedDate
Simons bistro	06-05-1998 00:00:00	
Richter Supermarkt	06-05-1998 00:00:00	
Bon app'	06-05-1998 00:00:00	
Rattlesnake Canyon Grocery	06-05-1998 00:00:00	
Lehmanns Marktstand	05-05-1998 00:00:00	
LILA-Supermercado	05-05-1998 00:00:00	
Ernst Handel	05-05-1998 00:00:00	
Pericles Comidas clásicas	05-05-1998 00:00:00	
Drachenblut Delikatessen	04-05-1998 00:00:00	
Queen Cozinha	04-05-1998 00:00:00	
Tortuga Restaurante	04-05-1998 00:00:00	
Save-a-lot Markets	01-05-1998 00:00:00	
LILA-Supermercado	01-05-1998 00:00:00	
White Clover Markets	01-05-1998 00:00:00	
Franchi S.p.A.	30-04-1998 00:00:00	

Records 1 to 15 of 830 | duplicates

We'll use this to add the OrderID field. After the list page refreshes, we'll add the name of the employee who took care of the order; for this we'll navigate through the pop menu (Human Resources / Employee):


http://showroom.declarativa.com - Orders - Microsoft Internet Explorer

Ficheiro Editar Ver Favoritos Ferramentas Ajuda

Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

CompanyName	OrderDate	ShippedDate	OrderID
Simons bistro	06-05-1998 00:00:00		110
Richter Supermarkt	06-05-1998 00:00:00		110
Bon app'	06-05-1998 00:00:00		110
Rattlesnake Canyon Grocery	06-05-1998 00:00:00		110
Lehmanns Marktstand	05-05-1998 00:00:00		110
LILA-Supermercado	05-05-1998 00:00:00		110
Ernst Handel	05-05-1998 00:00:00		110
Pericles Comidas clásicas	05-05-1998 00:00:00		110
Drachenblut Delikatessen	04-05-1998 00:00:00	06-05-1998 00:00:00	110
Queen Cozinha	04-05-1998 00:00:00		110
Tortuga Restaurante	04-05-1998 00:00:00	06-05-1998 00:00:00	110
Save-a-lot Markets	01-05-1998 00:00:00	04-05-1998 00:00:00	110
LILA-Supermercado	01-05-1998 00:00:00		110
White Clover Markets	01-05-1998 00:00:00	04	
Franchi S.p.A.	30-04-1998 00:00:00	04-05-1998 00:00:00	110

Records 1 to 15 of 830 | duplicates removed

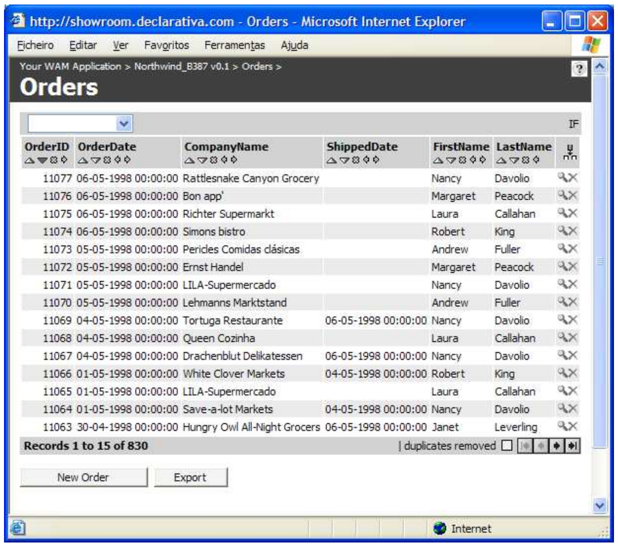
New Order Export

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Hello, Anonymous | 06-05-2007 17:09:49.872 | This page was created in 0.453 seconds.

Concluído Internet

The resulting list page now has 5 columns; after adding employee first name, and some more dragging and ordering of columns:


http://showroom.declarativa.com - Orders - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Orders >
Orders
New Order
Export
Internet

OrderID	OrderDate	CompanyName	ShippedDate	FirstName	LastName
11077	06-05-1998 00:00:00	Rattlesnake Canyon Grocery		Nancy	Davolio
11076	06-05-1998 00:00:00	Bon app'		Margaret	Peacock
11075	06-05-1998 00:00:00	Richter Supermarkt		Laura	Callahan
11074	06-05-1998 00:00:00	Simons bistro		Robert	King
11073	05-05-1998 00:00:00	Pericles Comidas clásicas		Andrew	Fuller
11072	05-05-1998 00:00:00	Ernst Handel		Margaret	Peacock
11071	05-05-1998 00:00:00	LILA-Supermercado		Nancy	Davolio
11070	05-05-1998 00:00:00	Lehmanns Marktstand		Andrew	Fuller
11069	04-05-1998 00:00:00	Tortuga Restaurante	06-05-1998 00:00:00	Nancy	Davolio
11068	04-05-1998 00:00:00	Queen Cozinha		Laura	Callahan
11067	04-05-1998 00:00:00	Drachenblut Delikatessen	06-05-1998 00:00:00	Nancy	Davolio
11066	01-05-1998 00:00:00	White Clover Markets	04-05-1998 00:00:00	Robert	King
11065	01-05-1998 00:00:00	LILA-Supermercado		Laura	Callahan
11064	01-05-1998 00:00:00	Save-a-lot Markets	04-05-1998 00:00:00	Nancy	Davolio
11063	30-04-1998 00:00:00	Hungry Owl All-Night Grocers	06-05-1998 00:00:00	Janet	Leverling

Records 1 to 15 of 830 | duplicates removed

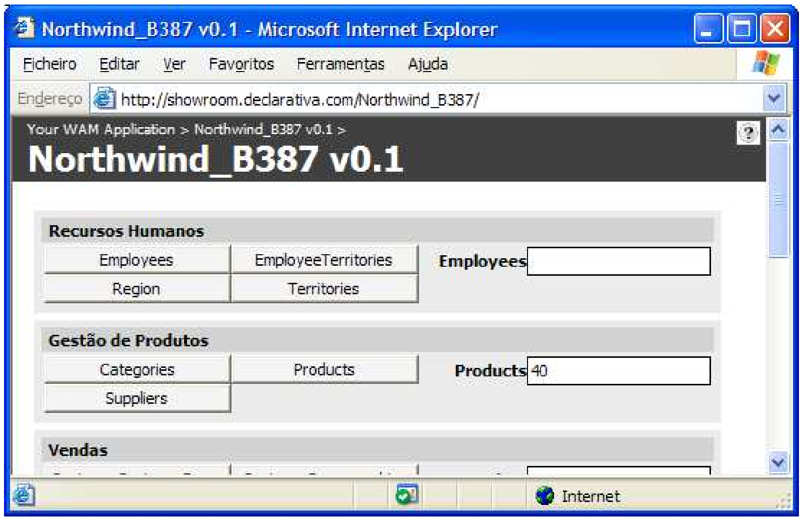
Each user has therefore his *personalized lists*, which persist between sessions and browsers; the next time the user accesses the above list, from the same or another computer, it will appear with the same columns, ordering etc. In other words, list configurations are part of the user preferences.

2.5 Finder fields, simple list searches, and list filters

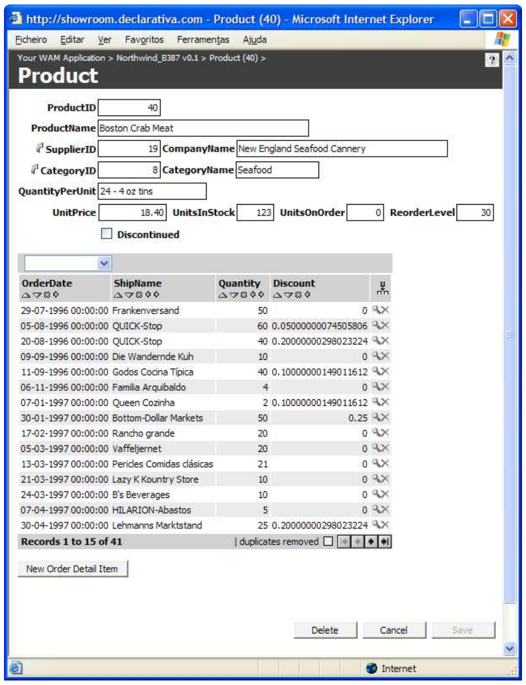
Browsing around huge data lists is not an option to search for information. A WAM application will typically provide at least 3 ways to search for something.

2.5.1 Finder Fields

The entry page has some "finder fields" next to list buttons, which can be used to type a primary key value and immediately navigate to a single data row page- or to a list with the data compatible with the (ambiguous or partial) data entered in the field. For example, typing 40 in the products finder field…:


Northwind_B387 v0.1 - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Endereço http://showroom.declarativa.com/Northwind_B387/
Your WAM Application > Northwind_B387 v0.1 >
Northwind_B387 v0.1
Recursos Humanos
Employees
EmployeeTerritories
Region
Territories
Employees
Gestão de Produtos
Categories
Products
Suppliers
Products 40
Vendas
Internet

…and hitting the Enter key opens up the Product with ID 40:


http://showroom.declarativa.com - Product (40) - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Product (40) >

Product

ProductID	40						
ProductName	Boston Crab Meat						
SupplierID	19	CompanyName	New England Seafood Cannery				
CategoryID	8	CategoryName	Seafood				
QuantityPerUnit	24 - 4 oz tins						
UnitPrice	18.40	UnitsInStock	123	UnitsOnOrder	0	ReorderLevel	30

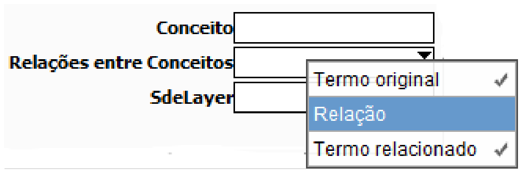
☐ Discontinued

OrderDate	ShipName	Quantity	Discount
29-07-1996 00:00:00	Frankenversand	50	0
05-08-1996 00:00:00	QUICK-Stop	60	0.0500000074505806
20-08-1996 00:00:00	QUICK-Stop	40	0.2000000298023224
09-09-1996 00:00:00	Die Wandernde Kuh	10	0
11-09-1996 00:00:00	Godos Cocina Típica	40	0.1000000149011612
06-11-1996 00:00:00	Familia Arquibaldo	4	0
07-01-1997 00:00:00	Queen Cozinha	2	0.1000000149011612
30-01-1997 00:00:00	Bottom-Dollar Markets	50	0.25
17-02-1997 00:00:00	Rancho grande	20	0
05-03-1997 00:00:00	Vaffeljernet	20	0
13-03-1997 00:00:00	Pericles Comidas clásicas	21	0
21-03-1997 00:00:00	Lazy K Kountry Store	10	0
24-03-1997 00:00:00	B's Beverages	10	0
07-04-1997 00:00:00	HILARION-Abastos	5	0
30-04-1997 00:00:00	Lehmanns Marktstand	25	0.2000000298023224

Records 1 to 15 of 41 | duplicates removed ☐

Real WAM applications will typically use finder fields accepting data to be searched simultaneously against *several* table fields (one at a time, until compatible data is found).

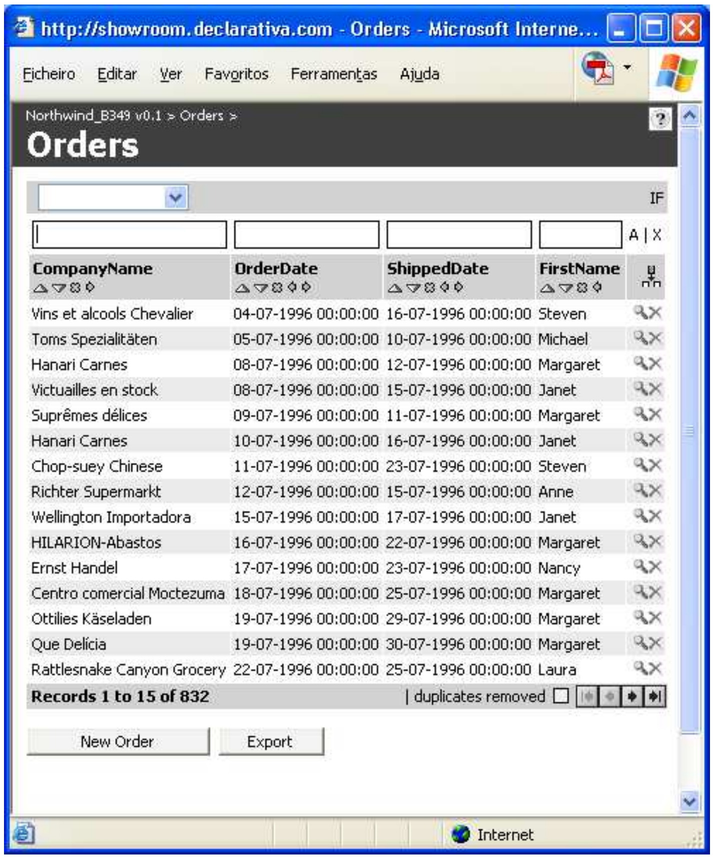
When a finder has several table fields, the user can restrict the search by clicking on the arrow and unselect the fields that shouldn't be tried to match:



2.5.2 Simple list searching

Data lists have the ability to let the user search for values - not just within the visible data in the page, but over the remaining list data as well. Clicking the IF button[8] at the top right of a list displays a search field over each column title, immediately and without refreshing the web page:

[8] IF: Inline Filter; waiting for a better icon!

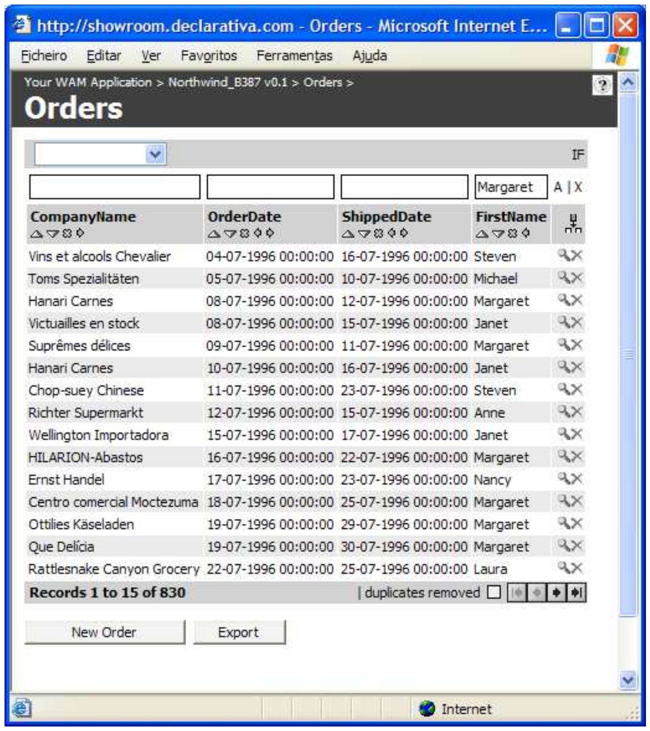

http://showroom.declarativa.com - Orders - Microsoft Interne...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_B349 v0.1 > Orders >
IF
A | X
New Order
Export
Internet

Orders

CompanyName	OrderDate	ShippedDate	FirstName
Vins et alcools Chevalier	04-07-1996 00:00:00	16-07-1996 00:00:00	Steven
Toms Spezialitäten	05-07-1996 00:00:00	10-07-1996 00:00:00	Michael
Hanari Carnes	08-07-1996 00:00:00	12-07-1996 00:00:00	Margaret
Victuailles en stock	08-07-1996 00:00:00	15-07-1996 00:00:00	Janet
Suprêmes délices	09-07-1996 00:00:00	11-07-1996 00:00:00	Margaret
Hanari Carnes	10-07-1996 00:00:00	16-07-1996 00:00:00	Janet
Chop-suey Chinese	11-07-1996 00:00:00	23-07-1996 00:00:00	Steven
Richter Supermarkt	12-07-1996 00:00:00	15-07-1996 00:00:00	Anne
Wellington Importadora	15-07-1996 00:00:00	17-07-1996 00:00:00	Janet
HILARION-Abastos	16-07-1996 00:00:00	22-07-1996 00:00:00	Margaret
Ernst Handel	17-07-1996 00:00:00	23-07-1996 00:00:00	Nancy
Centro comercial Moctezuma	18-07-1996 00:00:00	25-07-1996 00:00:00	Margaret
Ottilies Käseladen	19-07-1996 00:00:00	29-07-1996 00:00:00	Margaret
Que Delícia	19-07-1996 00:00:00	30-07-1996 00:00:00	Margaret
Rattlesnake Canyon Grocery	22-07-1996 00:00:00	25-07-1996 00:00:00	Laura

Records 1 to 15 of 832 | duplicates removed ☐

Then we can define a search condition by typing the specific criteria in each textboxes above the fields. For instance, let’s search orders list by choosing only the orders created by the employee “Margaret”, typing this name of column FirstName:


http://showroom.declarativa.com - Orders - Microsoft Internet E...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
IF
A | X
New Order
Export
Internet

Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

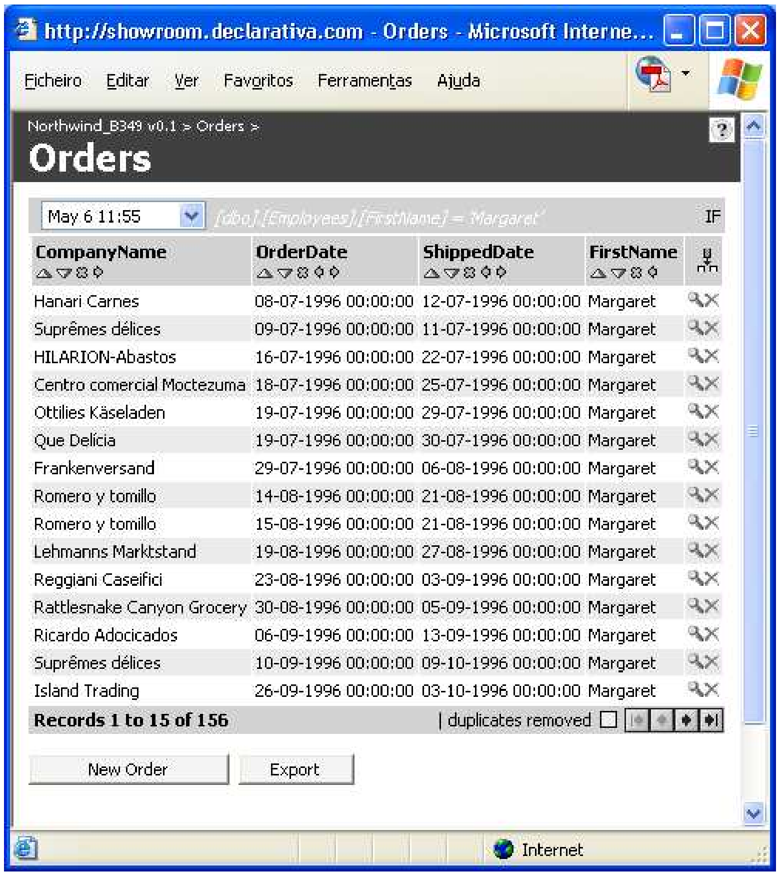
Margaret

CompanyName	OrderDate	ShippedDate	FirstName
Vins et alcools Chevalier	04-07-1996 00:00:00	16-07-1996 00:00:00	Steven
Toms Spezialitäten	05-07-1996 00:00:00	10-07-1996 00:00:00	Michael
Hanari Carnes	08-07-1996 00:00:00	12-07-1996 00:00:00	Margaret
Victuailles en stock	08-07-1996 00:00:00	15-07-1996 00:00:00	Janet
Suprêmes délices	09-07-1996 00:00:00	11-07-1996 00:00:00	Margaret
Hanari Carnes	10-07-1996 00:00:00	16-07-1996 00:00:00	Janet
Chop-suey Chinese	11-07-1996 00:00:00	23-07-1996 00:00:00	Steven
Richter Supermarkt	12-07-1996 00:00:00	15-07-1996 00:00:00	Anne
Wellington Importadora	15-07-1996 00:00:00	17-07-1996 00:00:00	Janet
HILARION-Abastos	16-07-1996 00:00:00	22-07-1996 00:00:00	Margaret
Ernst Handel	17-07-1996 00:00:00	23-07-1996 00:00:00	Nancy
Centro comercial Moctezuma	18-07-1996 00:00:00	25-07-1996 00:00:00	Margaret
Ottilies Käseladen	19-07-1996 00:00:00	29-07-1996 00:00:00	Margaret
Que Delícia	19-07-1996 00:00:00	30-07-1996 00:00:00	Margaret
Rattlesnake Canyon Grocery	22-07-1996 00:00:00	25-07-1996 00:00:00	Laura

Records 1 to 15 of 830 | duplicates removed

After hitting the Enter key, or clicking the "A" button on the right, we get a list displaying only orders by Margaret[9], with a brief description, at the top, of the (SQL) search condition for the whole list:

[9] Or to be more precise: all orders placed by an employee whose first name is Margaret (could be more than one employee)



http://showroom.declarativa.com - Orders - Microsoft Interne...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_B349 v0.1 > Orders >
May 6 11:55
IF
New Order
Export
Internet

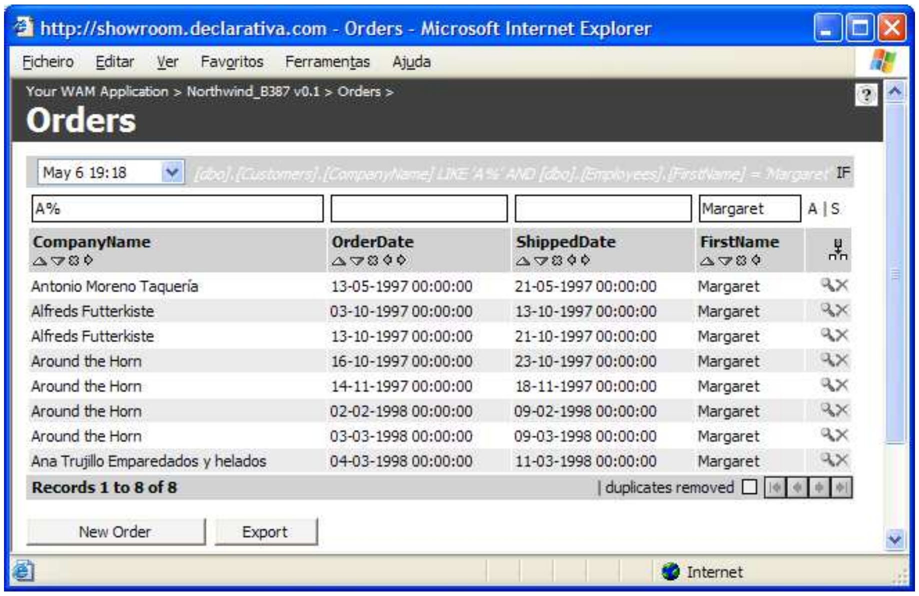
Orders

[dbo].[Employees].[FirstName] = 'Margaret'

CompanyName	OrderDate	ShippedDate	FirstName
Hanari Carnes	08-07-1996 00:00:00	12-07-1996 00:00:00	Margaret
Suprêmes délices	09-07-1996 00:00:00	11-07-1996 00:00:00	Margaret
HILARION-Abastos	16-07-1996 00:00:00	22-07-1996 00:00:00	Margaret
Centro comercial Moctezuma	18-07-1996 00:00:00	25-07-1996 00:00:00	Margaret
Ottilies Käseladen	19-07-1996 00:00:00	29-07-1996 00:00:00	Margaret
Que Delícia	19-07-1996 00:00:00	30-07-1996 00:00:00	Margaret
Frankenversand	29-07-1996 00:00:00	06-08-1996 00:00:00	Margaret
Romero y tomillo	14-08-1996 00:00:00	21-08-1996 00:00:00	Margaret
Romero y tomillo	15-08-1996 00:00:00	21-08-1996 00:00:00	Margaret
Lehmanns Marktstand	19-08-1996 00:00:00	27-08-1996 00:00:00	Margaret
Reggiani Caseifici	23-08-1996 00:00:00	03-09-1996 00:00:00	Margaret
Rattlesnake Canyon Grocery	30-08-1996 00:00:00	05-09-1996 00:00:00	Margaret
Ricardo Adocicados	06-09-1996 00:00:00	13-09-1996 00:00:00	Margaret
Suprêmes délices	10-09-1996 00:00:00	09-10-1996 00:00:00	Margaret
Island Trading	26-09-1996 00:00:00	03-10-1996 00:00:00	Margaret

Records 1 to 15 of 156 | duplicates removed ☐

The search condition can involve more than one field, and include partial conditions:



http://showroom.declarativa.com - Orders - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Orders >
May 6 19:18
IF
A | S
New Order
Export
Internet

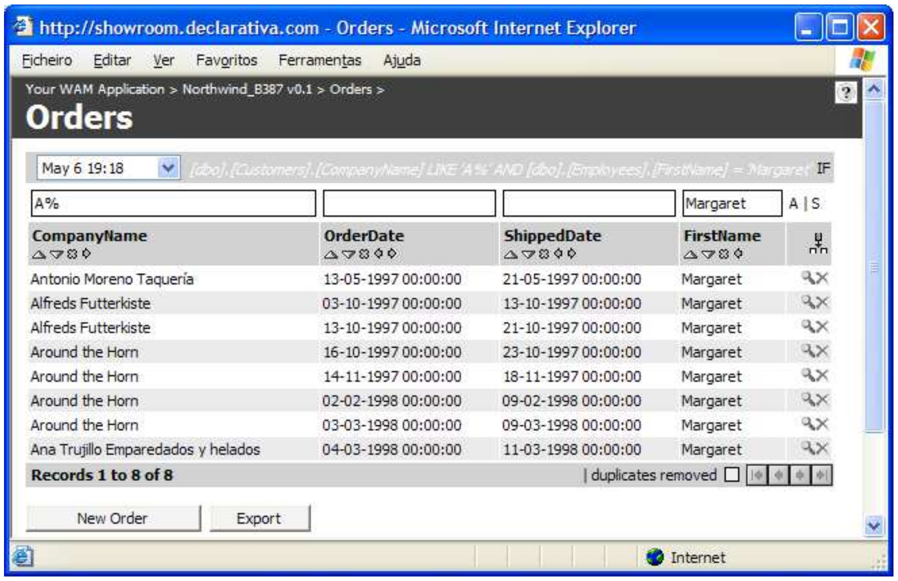
Orders

[dbo].[Customers].[CompanyName] LIKE 'A%' AND [dbo].[Employees].[FirstName] = 'Margaret'

CompanyName	OrderDate	ShippedDate	FirstName
A%			Margaret
Antonio Moreno Taquería	13-05-1997 00:00:00	21-05-1997 00:00:00	Margaret
Alfreds Futterkiste	03-10-1997 00:00:00	13-10-1997 00:00:00	Margaret
Alfreds Futterkiste	13-10-1997 00:00:00	21-10-1997 00:00:00	Margaret
Around the Horn	16-10-1997 00:00:00	23-10-1997 00:00:00	Margaret
Around the Horn	14-11-1997 00:00:00	18-11-1997 00:00:00	Margaret
Around the Horn	02-02-1998 00:00:00	09-02-1998 00:00:00	Margaret
Around the Horn	03-03-1998 00:00:00	09-03-1998 00:00:00	Margaret
Ana Trujillo Emparedados y helados	04-03-1998 00:00:00	11-03-1998 00:00:00	Margaret

Records 1 to 8 of 8 | duplicates removed ☐

After applying the new search condition (the *conjunction*[10] of all individual column conditions), the resulting list shows all orders for a customer whose company name starts with A, placed by an employee with first name Margaret:


http://showroom.declarativa.com - Orders - Microsoft Internet Explorer

Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

May 6 19:18 [dbo].[Customers].[CompanyName] LIKE 'A%' AND [dbo].[Employees].[FirstName] = 'Margaret' IF

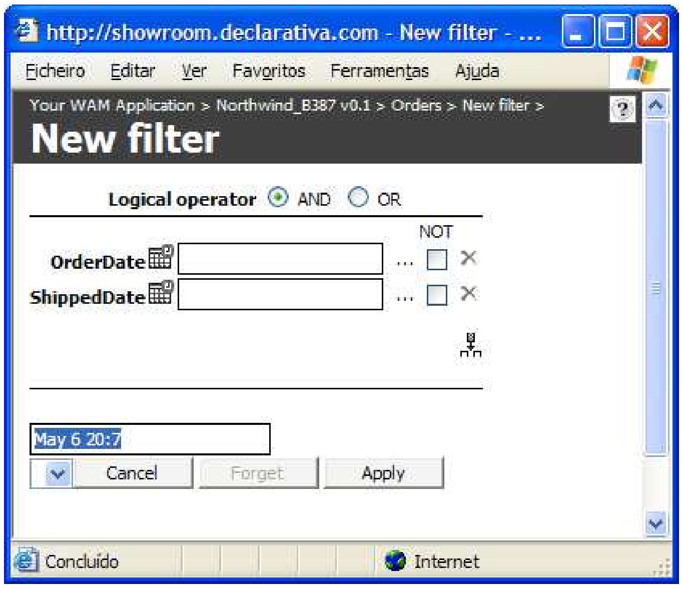
CompanyName	OrderDate	ShippedDate	FirstName
A%			Margaret
Antonio Moreno Taquería	13-05-1997 00:00:00	21-05-1997 00:00:00	Margaret
Alfreds Futterkiste	03-10-1997 00:00:00	13-10-1997 00:00:00	Margaret
Alfreds Futterkiste	13-10-1997 00:00:00	21-10-1997 00:00:00	Margaret
Around the Horn	16-10-1997 00:00:00	23-10-1997 00:00:00	Margaret
Around the Horn	14-11-1997 00:00:00	18-11-1997 00:00:00	Margaret
Around the Horn	02-02-1998 00:00:00	09-02-1998 00:00:00	Margaret
Around the Horn	03-03-1998 00:00:00	09-03-1998 00:00:00	Margaret
Ana Trujillo Emparedados y helados	04-03-1998 00:00:00	11-03-1998 00:00:00	Margaret

Records 1 to 8 of 8 | duplicates removed

"%" is a wildcard character usable in search fields; there are more, stay tuned for the next section.

2.5.3 List filters: searches to remember

A search condition can be stored as part of its list configuration, as a user preference; in that cases it is called a **search filter**. To create a search filter click the popup menu at the top left in the previous page, and choose the New Filter option:


http://showroom.declarativa.com - New filter - ...

Your WAM Application > Northwind_B387 v0.1 > Orders > New filter >

New filter

Logical operator [✓ AND] OR

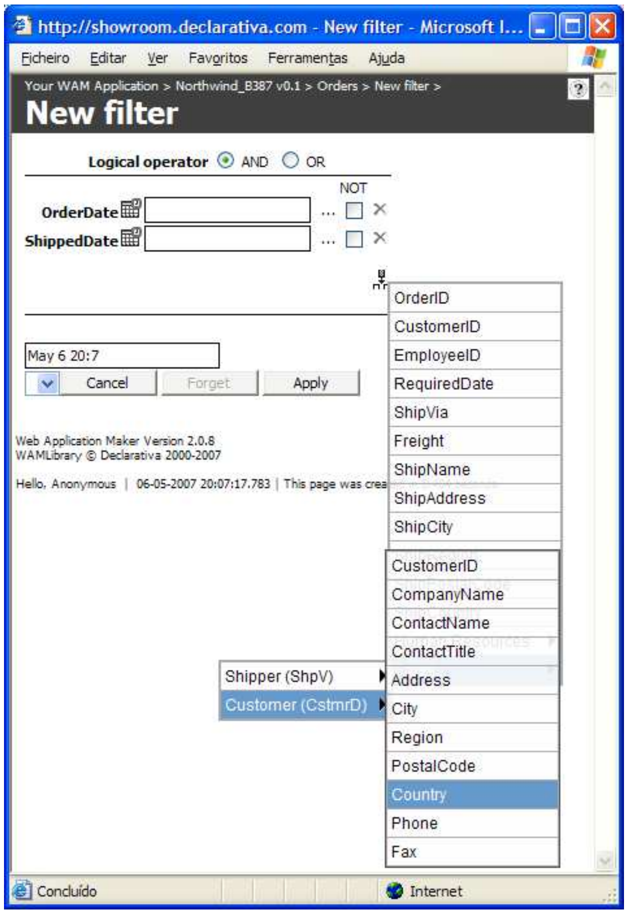
OrderDate

ShippedDate

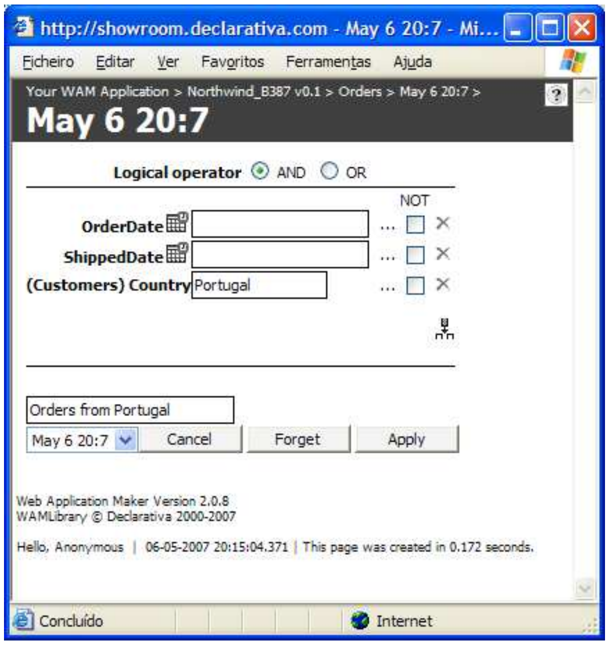
May 6 20:7

[10] See 2.5.3for more flexible query formula.

Initially only the fields from the list base table appear, Orders in this case; the other two columns in the example list, CompanyName and FirstName, belong to other tables. As in the lists themselves, it is possible to add column fields to a filter definition, by clicking the button (it is convenient to augment the filter window size first):


http://showroom.declarativa.com - New filter - Microsoft I...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Orders > New filter >
New filter
Logical operator AND OR
NOT
OrderDate
ShippedDate
May 6 20:7
Cancel Forget Apply
Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007
Hello, Anonymous | 06-05-2007 20:07:17.783 | This page was crea
OrderID
CustomerID
EmployeeID
RequiredDate
ShipVia
Freight
ShipName
ShipAddress
ShipCity
Shipper (ShpV)
Customer (CstmrD)
CustomerID
CompanyName
ContactName
ContactTitle
Address
City
Region
PostalCode
Country
Phone
Fax
Concluído
Internet

The following filter will filter list data to include only orders from customers in Portugal:



http://showroom.declarativa.com - May 6 20:7 - Mi...

Ficheiro Editar Ver Favoritos Ferramentas Ajuda

Your WAM Application > Northwind_B387 v0.1 > Orders > May 6 20:7 >

May 6 20:7

Logical operator [✓ AND] OR

		NOT
OrderDate		☐
ShippedDate		☐
(Customers) Country	Portugal	☐

Orders from Portugal

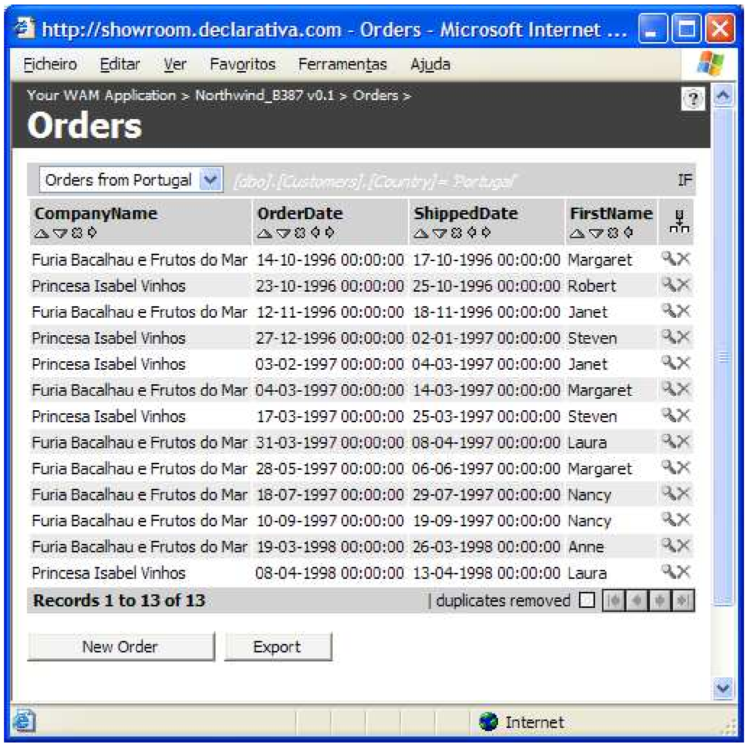
May 6 20:7 | Cancel | Forget | Apply

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Hello, Anonymous | 06-05-2007 20:15:04.371 | This page was created in 0.172 seconds.

Concluído | Internet

The field at the bottom lets the filter definition have a user friendly name, “Orders from Portugal”. Clicking the Apply button closes de filter page and refreshes the list, applying the new filter:



http://showroom.declarativa.com - Orders - Microsoft Internet ...

Ficheiro Editar Ver Favoritos Ferramentas Ajuda

Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

Orders from Portugal [dbo].[Customers].[Country]= Portugal IF

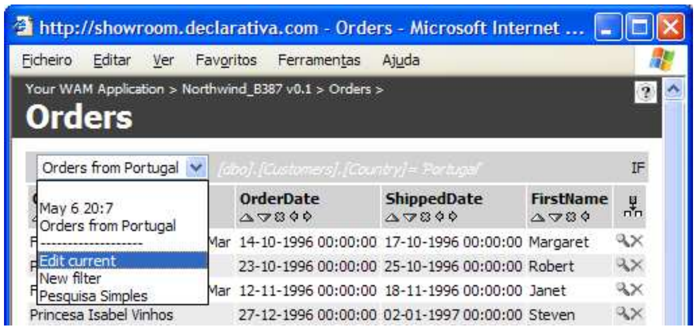
CompanyName	OrderDate	ShippedDate	FirstName
Furia Bacalhau e Frutos do Mar	14-10-1996 00:00:00	17-10-1996 00:00:00	Margaret
Princesa Isabel Vinhos	23-10-1996 00:00:00	25-10-1996 00:00:00	Robert
Furia Bacalhau e Frutos do Mar	12-11-1996 00:00:00	18-11-1996 00:00:00	Janet
Princesa Isabel Vinhos	27-12-1996 00:00:00	02-01-1997 00:00:00	Steven
Princesa Isabel Vinhos	03-02-1997 00:00:00	04-03-1997 00:00:00	Janet
Furia Bacalhau e Frutos do Mar	04-03-1997 00:00:00	14-03-1997 00:00:00	Margaret
Princesa Isabel Vinhos	17-03-1997 00:00:00	25-03-1997 00:00:00	Steven
Furia Bacalhau e Frutos do Mar	31-03-1997 00:00:00	08-04-1997 00:00:00	Laura
Furia Bacalhau e Frutos do Mar	28-05-1997 00:00:00	06-06-1997 00:00:00	Margaret
Furia Bacalhau e Frutos do Mar	18-07-1997 00:00:00	29-07-1997 00:00:00	Nancy
Furia Bacalhau e Frutos do Mar	10-09-1997 00:00:00	19-09-1997 00:00:00	Nancy
Furia Bacalhau e Frutos do Mar	19-03-1998 00:00:00	26-03-1998 00:00:00	Anne
Princesa Isabel Vinhos	08-04-1998 00:00:00	13-04-1998 00:00:00	Laura

Records 1 to 13 of 13 | duplicates removed ☐

New Order | Export

Internet

Later on, in another session and even from another computer, the user can apply this or other filters (one at a time) by using the top left popup menu. The menu also has an empty item option, which removes the filter, as well as “Edit current”, to edit the currently applied filter:


http://showroom.declarativa.com - Orders - Microsoft Internet ...

Ficheiro Editar Ver Favoritos Ferramentas Ajuda

Your WAM Application > Northwind_B387 v0.1 > Orders >

Orders

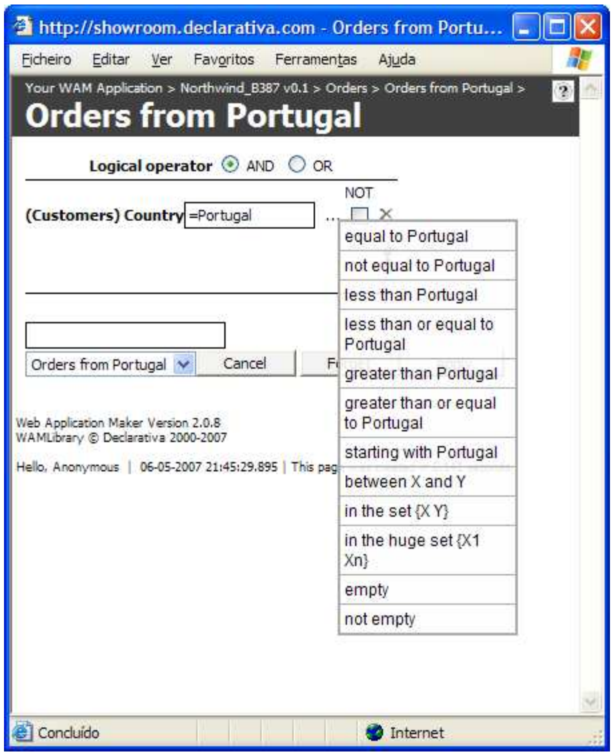
Orders from Portugal [dbo].[Customers].[Country]= 'Portugal' IF

May 6 20:7
Orders from Portugal

Edit current
New filter
Pesquisa Simples

		OrderDate	ShippedDate	FirstName
F	Mar	14-10-1996 00:00:00	17-10-1996 00:00:00	Margaret
F		23-10-1996 00:00:00	25-10-1996 00:00:00	Robert
F	Mar	12-11-1996 00:00:00	18-11-1996 00:00:00	Janet
Princesa Isabel Vinhos		27-12-1996 00:00:00	02-01-1997 00:00:00	Steven

By default the filter page assumes a (AND) conjunction of column conditions; a radio button at the top is available for disjunction (OR) instead. Each column condition as a NOT check box to negate it, and a ✕ button to remove the condition (empty field or removed field can have a different meaning, for example for bit columns). The … button pops up a help menu with the available search operators:


http://showroom.declarativa.com - Orders from Portu...

Ficheiro Editar Ver Favoritos Ferramentas Ajuda

Your WAM Application > Northwind_B387 v0.1 > Orders > Orders from Portugal >

Orders from Portugal

Logical operator [✓ AND] OR

NOT

(Customers) Country =Portugal ...

equal to Portugal
not equal to Portugal
less than Portugal
less than or equal to Portugal
greater than Portugal
greater than or equal to Portugal
starting with Portugal
between X and Y
in the set {X Y}
in the huge set {X1 Xn}
empty
not empty

Orders from Portugal | Cancel | F

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Hello, Anonymous | 06-05-2007 21:45:29.895 | This pag

Concluído | Internet

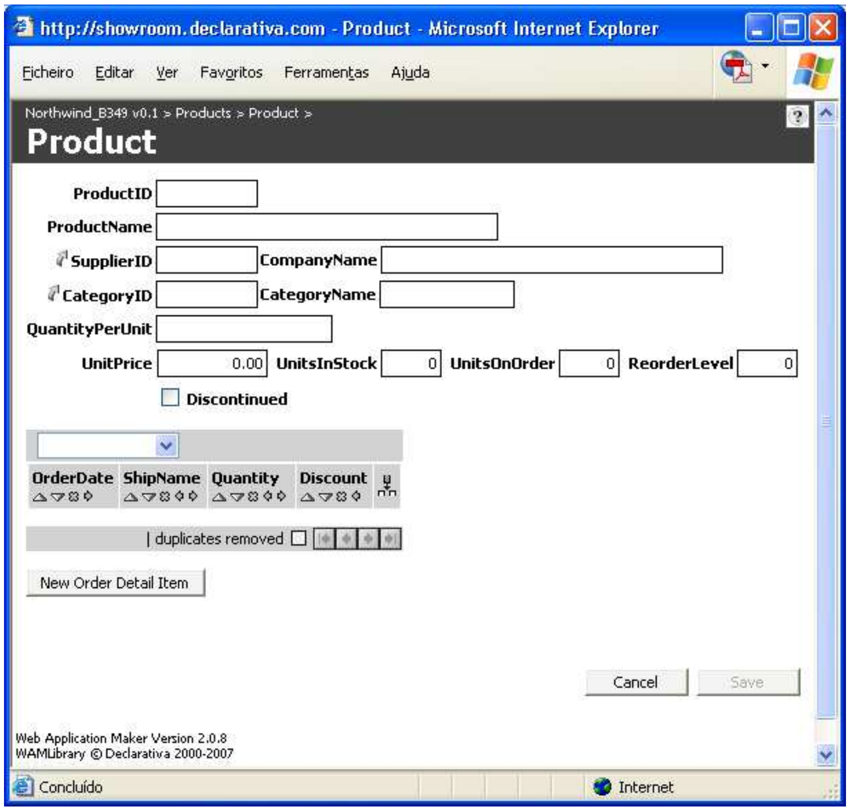
“In the huge set” supports large value sets, specified in a separate window.

List filters can be shared[11], by copying the list URL to the clipboard using the share item in the filter menu.

2.6 Changing and editing data

To add a new Product, click the. New Product button in the list page; as a shortcut, ctrl-click the Products button in the entry page. Afterwards a Product row page opens up, with all fields empty except the default[12] values:

[11] If so allowed in global.asa, see 5.5



http://showroom.declarativa.com - Product - Microsoft Internet Explorer

Northwind_B349 v0.1 > Products > Product >

Product

ProductID							
ProductName							
SupplierID		CompanyName					
CategoryID		CategoryName					
QuantityPerUnit							
UnitPrice	0.00	UnitsInStock	0	UnitsOnOrder	0	ReorderLevel	0

☐ Discontinued

OrderDate	ShipName	Quantity	Discount

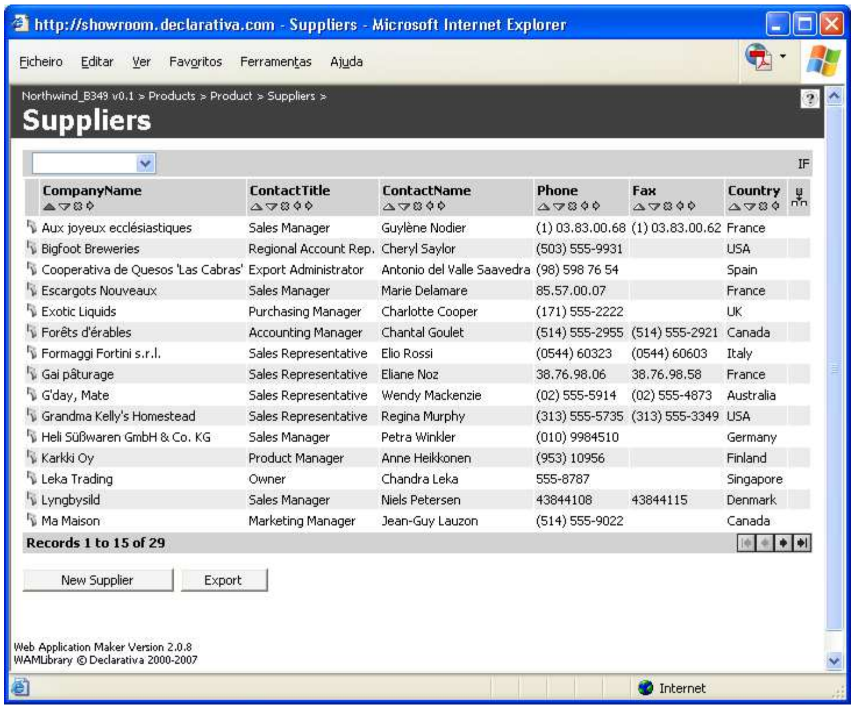
| duplicates removed ☐

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Each field can be filled at will; there will be error messages if the data type is not right, for example UnitPrice must have a number and not a date, etc. The ProductID field can be left blank, because it is a numeric key.

Let's look more closely at the **SupplierID** field, which has a lookup button. Clicking this lets the user choose one among the existing suppliers:

[12] In the example, determined simply by SQL defaults.


http://showroom.declarativa.com - Suppliers - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
New Supplier
Export
Internet

Northwind_B349 v0.1 > Products > Product > Suppliers >

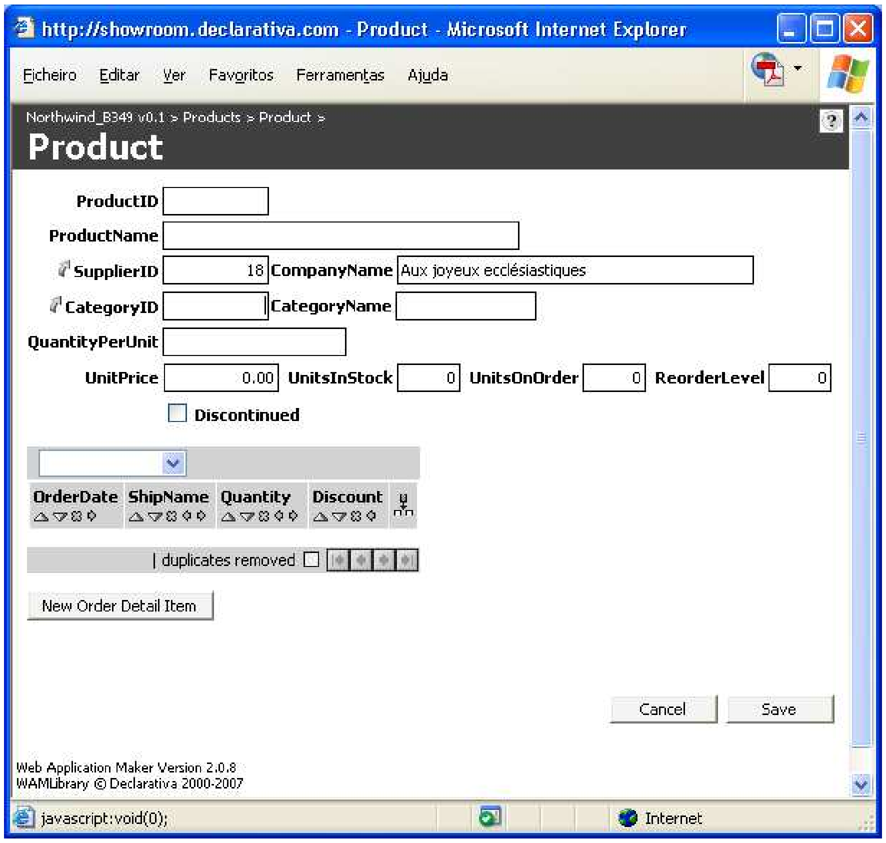
Suppliers

CompanyName	ContactTitle	ContactName	Phone	Fax	Country
Aux joyeux ecclésiastiques	Sales Manager	Guylène Nodier	(1) 03.83.00.68	(1) 03.83.00.62	France
Bigfoot Breweries	Regional Account Rep.	Cheryl Saylor	(503) 555-9931		USA
Cooperativa de Quesos 'Las Cabras'	Export Administrator	Antonio del Valle Saavedra	(98) 598 76 54		Spain
Escargots Nouveaux	Sales Manager	Marie Delamare	85.57.00.07		France
Exotic Liquids	Purchasing Manager	Charlotte Cooper	(171) 555-2222		UK
Forêts d'érables	Accounting Manager	Chantal Goulet	(514) 555-2955	(514) 555-2921	Canada
Formaggi Fortini s.r.l.	Sales Representative	Elio Rossi	(0544) 60323	(0544) 60603	Italy
Gai pâturage	Sales Representative	Eliane Noz	38.76.98.06	38.76.98.58	France
G'day, Mate	Sales Representative	Wendy Mackenzie	(02) 555-5914	(02) 555-4873	Australia
Grandma Kelly's Homestead	Sales Representative	Regina Murphy	(313) 555-5735	(313) 555-3349	USA
Heli Süßwaren GmbH & Co. KG	Sales Manager	Petra Winkler	(010) 9984510		Germany
Karkki Oy	Product Manager	Anne Heikkonen	(953) 10956		Finland
Leka Trading	Owner	Chandra Leka	555-8787		Singapore
Lyngbysild	Sales Manager	Niels Petersen	43844108	43844115	Denmark
Ma Maison	Marketing Manager	Jean-Guy Lauzon	(514) 555-9022		Canada

Records 1 to 15 of 29

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

After clicking the return button to the left of a record, the **SupplierID** and the **Company Name** textboxes on the **New Product** window will be automatically fulfilled as shown bellow:


http://showroom.declarativa.com - Product - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_B349 v0.1 > Products > Product >

Product

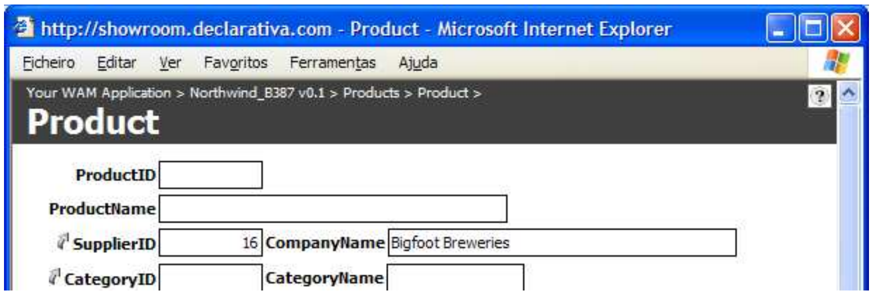
ProductID							
ProductName							
SupplierID	18	CompanyName	Aux joyeux ecclésiastiques				
CategoryID		CategoryName					
QuantityPerUnit							
UnitPrice	0.00	UnitsInStock	0	UnitsOnOrder	0	ReorderLevel	0

☐ Discontinued

OrderDate	ShipName	Quantity	Discount

| duplicates removed
New Order Detail Item
Cancel
Save
Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007
javascript:void(0);
Internet

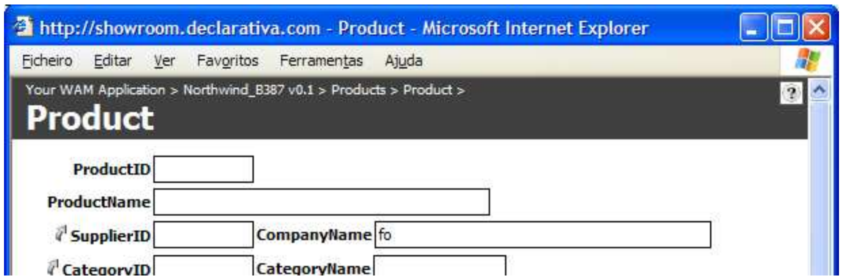
Alternatively, if the SupplierID was known, it could be typed directly, without the need to open up a new page. For example, after typing 16:


http://showroom.declarativa.com - Product - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Products > Product >

Product

ProductID			
ProductName			
SupplierID	16	CompanyName	Bigfoot Breweries
CategoryID		CategoryName	

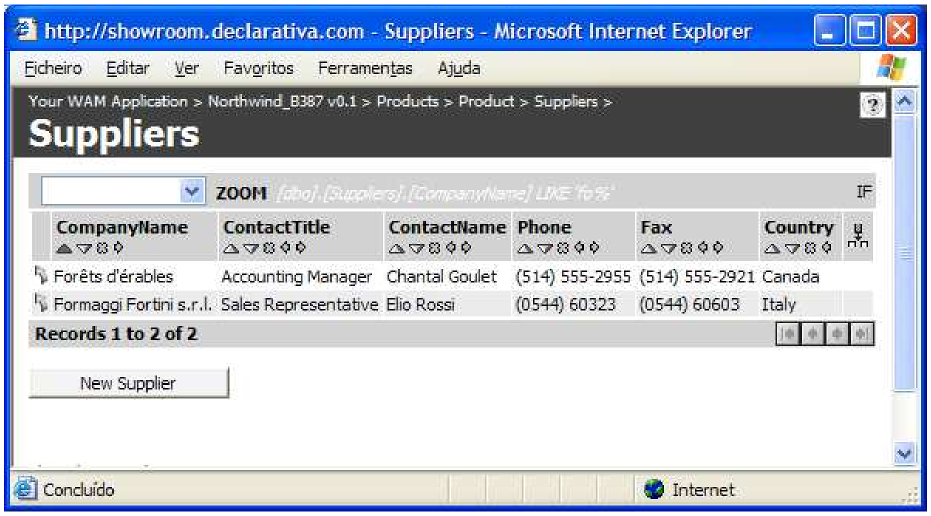
This is called a **direct lookup**. Yet another (and more likely) alternative: the user does not know the supplier ID, but he knows the first two characters, so he'll attempt an **inverse lookup**:


http://showroom.declarativa.com - Product - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Products > Product >

Product

ProductID			
ProductName			
SupplierID		CompanyName	fo
CategoryID		CategoryName	

After entering tab, and because there is more than one supplier starting with “Fo”, a new list page appears[13]:


http://showroom.declarativa.com - Suppliers - Microsoft Internet Explorer
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Your WAM Application > Northwind_B387 v0.1 > Products > Product > Suppliers >
Suppliers
ZOOM [dbo].[Suppliers].[CompanyName] LIKE 'fo%'
IF
New Supplier
Concluído
Internet

CompanyName	ContactTitle	ContactName	Phone	Fax	Country
Forêts d'érables	Accounting Manager	Chantal Goulet	(514) 555-2955	(514) 555-2921	Canada
Formaggi Fortini s.r.l.	Sales Representative	Elio Rossi	(0544) 60323	(0544) 60603	Italy

Records 1 to 2 of 2

And, like before, the user should pick the supplier by clicking one of the buttons. Finally, the user can click the Save button in the Product page.

To avoid waiting for a lookup match, WAM has a type-ahead feature which allows the user to keep filling the form while the system is still searching for a match (*** IMAGE HERE ***).

When a direct lookup or an inverse lookup doesn’t produce any results, the user has the option to choose between: creating a new one, choosing one from the list or cancelling the operation. However, if the user previously knows that the Supplier doesn’t exist, the user can create it directly by pressing Ctrl-click on the “lookup button” . After completing the creation, the SupplierID will be automatically fulfilled.

To **delete** a Product (or any other database record), either open the record from a list and hit the Delete button, or simply click one of the ✕ buttons in the list. In both cases a confirmation dialog will be displayed, and error messages will be shown if the deletion violates database integrity.

2.7 User interface shortcuts

There are several shortcuts available on all WAM generated pages:

- Hitting ALT-S in a row saves the row
- Ctrl-click in a save button (in a row) saves and opens up a "clone" of the saved row
- Ctrl-click in a list button (e.g. in a ListGroup) opens up an empty row for its base table, typically to create a new record
- Ctrl-click in a lookup button opens an empty row to create a new record, and fulfill the lookup field.
- To copy the current URL to clipboard, we can use the link “This page was created in X seconds.” present in the footer.

2.8 Philosophy

This section contains a preliminary description of a “WAM theory for GUI generation”, for applications such as referred in 1.4 and detailed in “WAM user interface: an application tour”. WAM and the enclosing web server can be

[13] If only one such supplier existed, it would be copied to the Product fields as in the direct lookup scenario

seen as an interpreter for a theory on *how to generate web pages from a WAMmodel instance*, in answer to HTTP requests.

The current WAM version implements the principles below; future R&D may focus on better generation from the same or related database server information.

Consider a human interface for a database, running as a web browser (intranet or extranet) custom-made application with many inter-linked pages.

How much of it could be *determined* just by the database-server SQL layer alone (data structure, triggers, stored procedures), avoiding the pain of developing a web Graphical User Interface? *Would it be possible to "infer" the missing code needed to construct the GUI* that must run in the web server/CGI and browser layers?

WAM is a tool for a partial "yes" to the above challenge. WAM's idealistic vision is to be a GUI function deriving the interface from the application database alone, through the following formula[14]:

Web interface = wam(Database Meta Information)

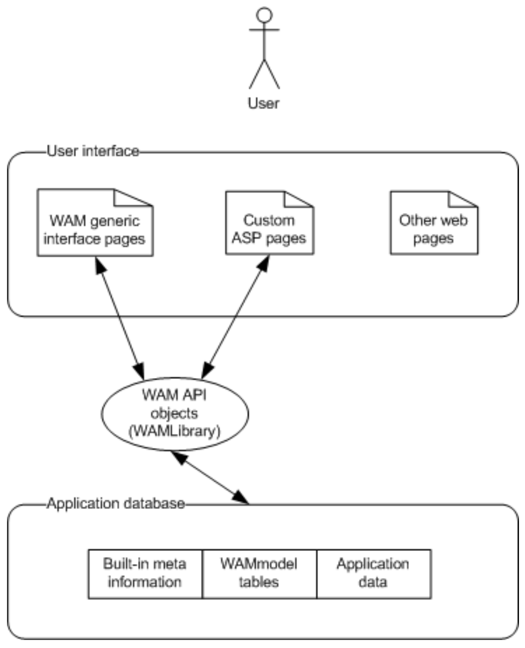
There is however a limit to the meta information included in database engines; for example, linguistic information is not there. Therefore WAM uses a few extra database tables, where additional (but not redundant) meta information can be placed by the developer, the WAMmodel:

Web interface = wam(Database built-in Meta Information + WAMmodel)

Reality makes this still too radical for most real world projects, so WAM effectively supports a more general and realistic vision:

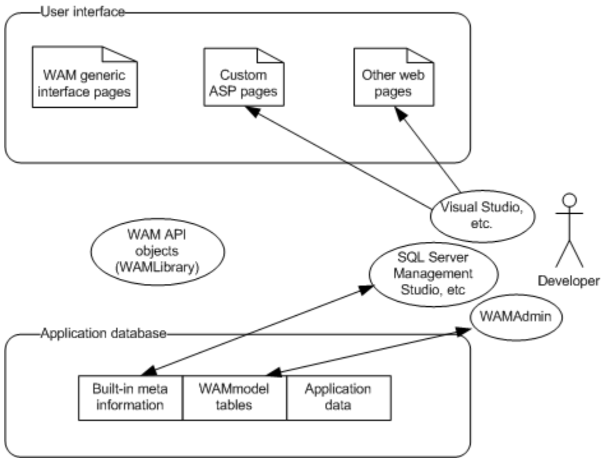
Web interface = wam(DB Meta Information + WAMmodel) + Custom pages using wam objects + Other pages

The next picture shows the overall architecture of a WAM application, with the user interface boundary showing the software components involved in the above equation:



[14] If the reader feels inclined towards the formal, he may regard the right side of the equation as "set of URLs that the web application responds to, satisfactorily"

Development of a WAM application can be depicted as follows; traditional database design and web programming are complemented with WAMmodel editing:



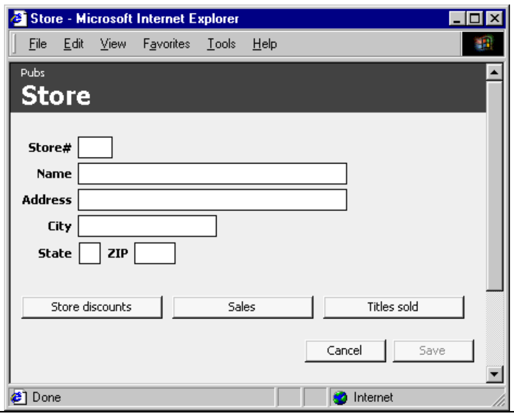
For the *developer of custom ASP pages using WAM object*, the WAMLibrary API implements a higher "semantic" level over the Microsoft Active Data Objects that support it. Rather than dealing with RecordSets, data field validations, SQL and HTML generation and many other concerns, the developer can delegate GUI generation, database access orchestration and navigation issues to the WAM objects, in whole or in part depending on application requirements. This is accomplished by using object constructors and methods referring concise database concepts - such as table or view, foreign key path, etc. - saving the developer a lot of details - data types, joins, table structure, etc.

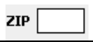
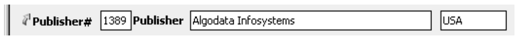
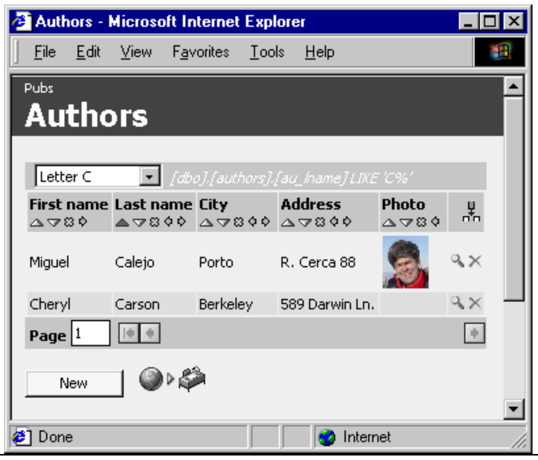
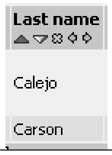
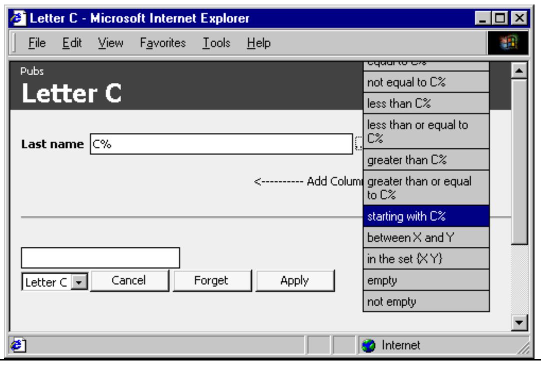
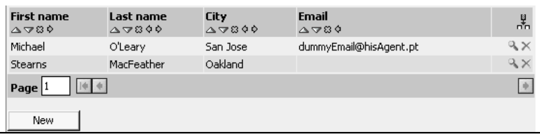
Therefore WAM encourages the developer to store more application knowledge into declarative models and *database server logic*, and less in CGI or browser-side procedural code, or in redundant models. This approach pursues fast and incremental development, as well as easier maintenance (mostly done in the database layer), in comparison with other methods requiring heavy CGI or client-side coding.

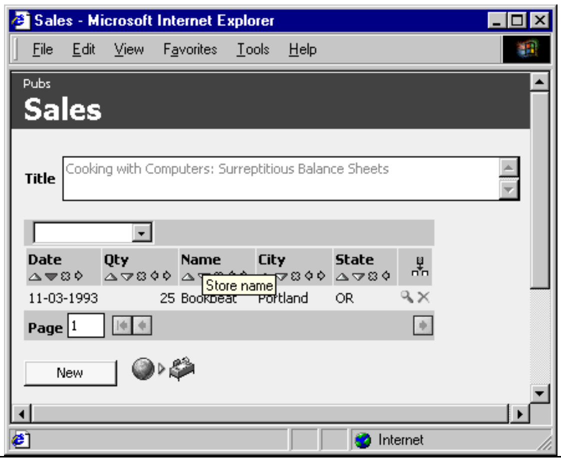
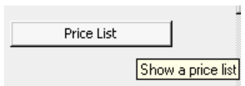
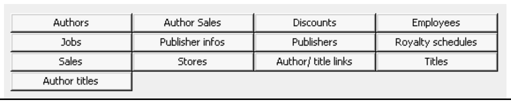
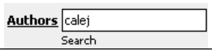
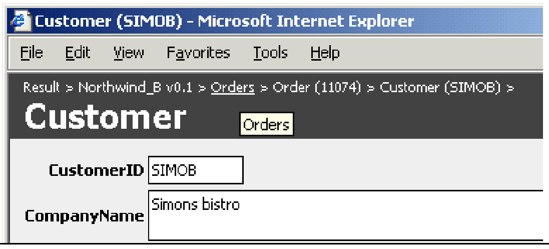
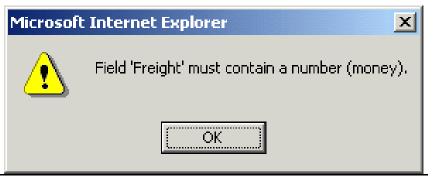
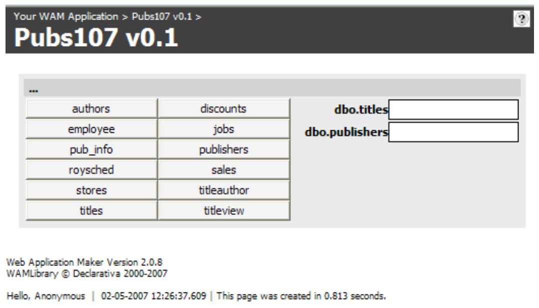
2.9 GUI fragments/parts

WAM is essentially an application front-end fragment factory, which works based on minimal declarative specifications closely tied to the underlying (database) application. It serves a particular class of GUI front-ends, those interfacing database applications.

Due to the fact that "all database GUIs are similar", at least to a certain extent, it is possible to identify recurrent fragments, each comprising GUI widgets, scripting, database access methods, error handling etc.:

GUI fragment	Description	Example
Row	A web page to display/edit a record or view tuple	 Store - Microsoft Internet Explorer File Edit View Favorites Tools Help Pubs Store Store# Name Address City State ZIP

Row field	A row element for a table column, possibly with a title	 ZIP
Lookup field	Displays a field in a related table; typically one or more lookup field appears next to the relating foreign key	 Publisher# 1389 Publisher Algodata Infosystems USA
List	A web page displaying a subset of a table, view or user-defined join, including user-defined sorting and navigation over the record set, possibly filtered with a search filter, data exporting button, etc.	 Authors - Microsoft Internet Explorer Pubs Authors Letter C [dbo].[authors].[au_lname] LIKE 'C%' First name, Last name, City, Address, Photo Miguel, Calejo, Porto, R. Cerca 88 Cheryl, Carson, Berkeley, 589 Darwin Ln. Page 1 New
List column	Displays all values of a database column in a list	 Last name Calejo Carson
Filter	A web page, closely linked to a list, to specify list filter criteria over columns in the list table and related tables	 Letter C - Microsoft Internet Explorer Pubs Letter C Last name C% <---------- Add Colum not equal to C%; less than C%; less than or equal to C%; greater than C%; greater than or equal to C%; starting with C%; between X and Y; in the set {X Y}; empty; not empty Letter C Cancel Forget Apply
Embedded (detail) list	Like a list, but restricted to the context of a related table; appears inside the master row	 First name, Last name, City, Email Michael, O'Leary, San Jose, dummyEmail@hisAgent.pt Stearns, MacFeather, Oakland Page 1 New

Standalone detail list	Like Embedded list, but appears in a standalone web page, and includes a description of the related table context, using lookup fields	 Sales - Microsoft Internet Explorer Pubs Sales Title: Cooking with Computers: Surreptitious Balance Sheets Date, Qty, Name, City, State 11-03-1993, 25, Bookbeat, Portland, OR Store name Page 1 New
Procedure invoking button	A button which invokes either a database stored procedure or an external (ASP) web page	 Price List Show a price list
List set	A set of buttons to navigate to lists in the application	 Authors, Author Sales, Discounts, Employees, Jobs, Publisher infos, Publishers, Royalty schedules, Sales, Stores, Author/ title links, Titles, Author titles
Finder	Similar to a row field, but providing quick search and navigation to a single tuple row or to a list	 Authors: calej Search
Path	A "breadcrumb" style hierarchical navigation style, appearing at the top of WAM generated pages and providing direct navigation to previous application pages	 Customer (SIMOB) - Microsoft Internet Explorer Result > Northwind_B v0.1 > Orders > Order (11074) > Customer (SIMOB) > Customer Orders CustomerID: SIMOB CompanyName: Simons bistro
Error Report	Transact-SQL or JavaScript originated multi-lingual error alerts, followed by field focusing if appropriate	 Microsoft Internet Explorer Field 'Freight' must contain a number (money). OK
Application main (entry) page	A default web page with buttons accessing all lists, and some finders	 Your WAM Application > Pubs107 v0.1 > Pubs107 v0.1 authors, discounts, employee, jobs, pub_info, publishers, roysched, sales, stores, titleauthor, titles, titleview dbo.titles dbo.publishers Web Application Maker Version 2.0.8 WAMLibrary © Declarativa 2000-2007 Hello, Anonymous | 02-05-2007 12:26:37.609 | This page was created in 0.813 seconds.

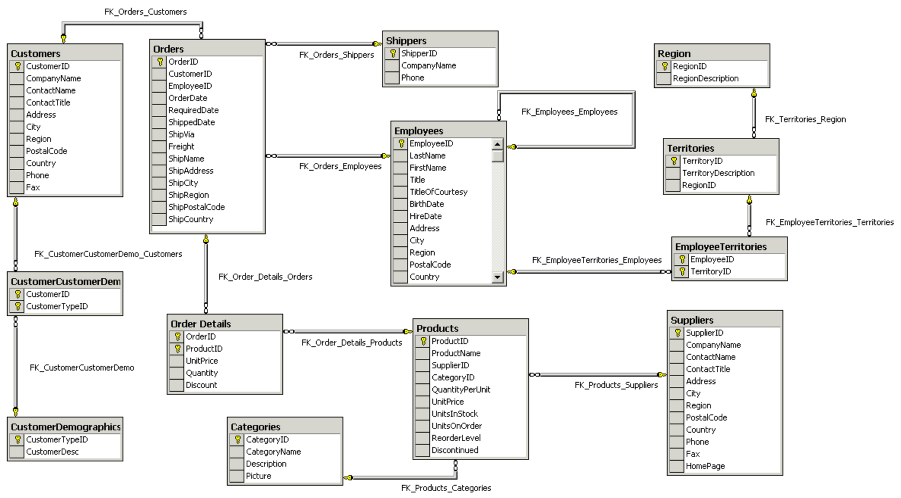
In the next sections we'll see how WAM is guided to produce the above fragments. These fragments can also be used by the developer in his custom pages, typically to obtain more control over appearance or functionality of lists and rows.

2.10 Foreign key graph

Relations between database tables are in important ingredient to GUI structure. In order to better use it, it's convenient to define its *foreign key graph (*abbreviated *FK graph)*; this is **implicit in the built-in database schema** information, and is defined as follows:

- Each table or view in the database defines a graph node
- Each relationship FK/PK between a table Detail with foreign key FK and table Master with primary key PK originates a directed edge from the Detail node to the Master node

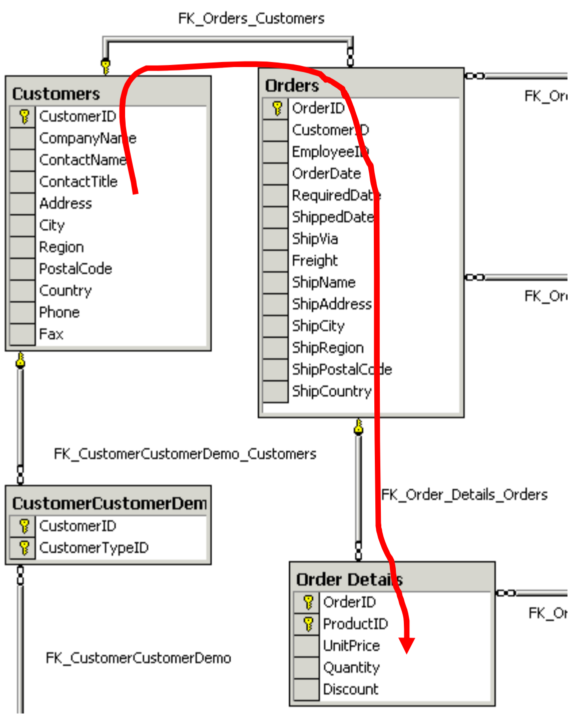
This definition is nothing more than the formalization of the database diagrams shown by (for example) SQL Server Management Studio, where the foreign keys can be seen in the edges:



In addition, in order to support VIEWs as "real tables", we'll define some additional meta information, WAM_CONSTRAINT_VIEW_USAGE , to simulate (definition of) foreign keys between VIEWs too. This meta information defines additional FK graph edges:

- Each tuple in table WAM_CONSTRAINT_VIEW_USAGE originates a directed edge from its Detail_table (or view) node to its Master_table (or view) node; this WAM table allows the use of relationships involving views

The foreign key graph is an important piece of WAM's conceptual background, because **foreign key paths** (or FK paths) are used as names (hence generation specifications) of GUI fragments. For example, consider the following path in the same database diagram:



This path can be represented (for WAM) as `FK_Orders_Customers,dbo.FK_Order_Details_Orders` (a sequence of schema.tableName; the first schema is omitted. Notice that this concise representation is able to abstract from some lower level details, such as which are the primary key columns of each table, which are the foreign key columns, what SQL expression should be used to implement the joins, etc.

We'll now introduce the WAM interface generation principles.

2.11 Database objects and how they impact GUI generation: GUI patterns

The previously defined GUI fragments can be used to solve database web interface design problems. In the next table we overview the "recipes" for solving these problems:

Database concept or its facet	**Consequence on GUI**
Table or View	A table record can be visualized and edited in a row. Table records can be visualized and filtered in a list. A finder can provide simple front-end navigation.
Column	A row field, a list column, a looked-up column
Data type	Row field and list column validation, formatting
User-defined SQL data type	Special formatting, validation, user navigation (e.g. email, URL, image)
Primary key	Determines default auto-numbering policy, avoiding the need to fill the key fields
Check Constraint	Restrict edition at the GUI, introspect into admissible values
Permission	Hide or disable GUI objects and navigation links for a particular user or group

Database objects have programmer names	Use a presentation function: WAM_PRESENTATION
Foreign key path (possibly with one FK only)	Minimal specification of lookups, detail lists, user-driven joins in lists
Foreign key graph	Default navigational structure
Table tuples representing persisted objects	For a particular object persistency strategy only, see section 4.5

Each line in the table can be seen as a "pattern", in the sense of [Patterns 2001], and so the whole can be seen as "WAM's GUI pattern catalog". We find this somehow related in spirit to efforts like [Coram and Lee 2001], although in our case with a more restricted scope - database applications.

External presentation strings are absent from database engine meta-models, hence the central role WAM_PRESENTATION assumes in the WAMmodel.

Navigational web link structure will reflect the relational data structure; WAM provides an automatic structure (see section 5.3.1).

3 WAM development tour

To overview WAM let's take the venerable 'pubs' database example[15], build a web server front-end for it with a single click, and demo it. Afterwards we will enhance it.

NOTE: you can visit http://showroom.declarativa.com , to try most of the following steps online;

Database application enhancement with new requirements can be a nightmare, as hidden dependencies manifest, causing the application to have to be updated in more than one place, such as data dictionary, form objects, GUI scripts… To minimize this problem, WAM promotes a GUI development style that is mostly (database) model-driven. The next sections illustrate development styles with increasing power and decreasing maintainability:

- "Single click" - interface obtained automatically from the initial database
- Improving the WAMmodel
- Improving the database
- ASP scripting

The following sections will focus on changing the application, not using it; for that perspective refer instead to “WAM user interface: an application tour”.

Although the following steps currently take more than a single click, they're generic red tape similar for any application, and can be automated - no step in this section is specific to the 'pubs' application. The WAM Installer provides a wizard doing it.

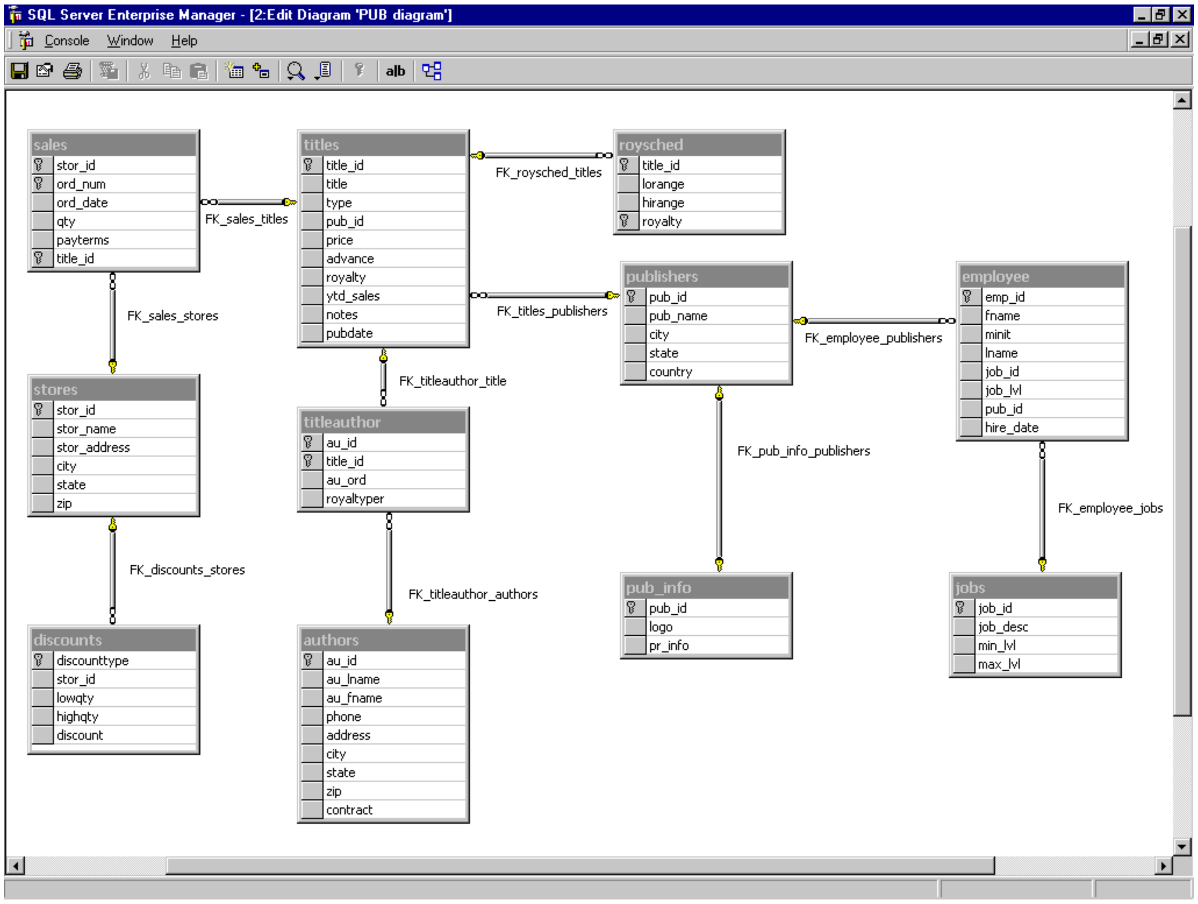
3.1 Webifying the 'pubs' database

We’ll start with getting the WAM up and running on a machine configured with SQL Server and IIS. We’ll use the ‘pubs’ database to illustrate this installation and the following examples as it comes with SQL Server and this way we don’t require users to have their own database right from the start.

3.1.1 Step 1: Get a database

Here's a diagram for the venerable 'pubs' example database, as produced by Microsoft SQL Server’s Enterprise Manager tool, depicting both each table structure and their foreign key master-detail relationships:

[15] Available as a Microsoft sample database at http://www.microsoft.com/downloads/details.aspx?familyid=06616212-0356-46a0-8da2-eebc53a68034&displaylang=en

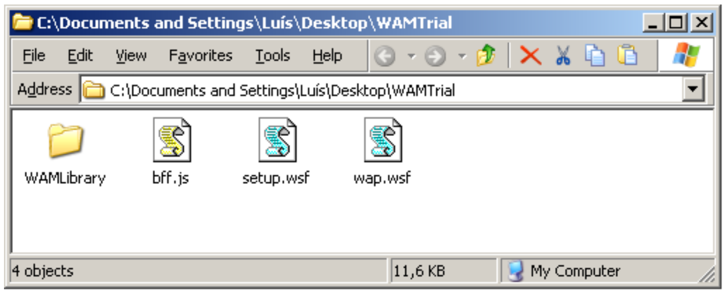

SQL Server Enterprise Manager - [2:Edit Diagram 'PUB diagram']
Console Window Help
sales: stor_id, ord_num, ord_date, qty, payterms, title_id
titles: title_id, title, type, pub_id, price, advance, royalty, ytd_sales, notes, pubdate
roysched: title_id, lorange, hirange, royalty
publishers: pub_id, pub_name, city, state, country
employee: emp_id, fname, minit, lname, job_id, job_lvl, pub_id, hire_date
stores: stor_id, stor_name, stor_address, city, state, zip
titleauthor: au_id, title_id, au_ord, royaltyper
pub_info: pub_id, logo, pr_info
jobs: job_id, job_desc, min_lvl, max_lvl
discounts: discounttype, stor_id, lowqty, highqty, discount
authors: au_id, au_lname, au_fname, phone, address, city, state, zip, contract
FK_sales_titles
FK_roysched_titles
FK_titles_publishers
FK_employee_publishers
FK_sales_stores
FK_titleauthor_title
FK_pub_info_publishers
FK_employee_jobs
FK_discounts_stores
FK_titleauthor_authors

WAM prefers a database having primary keys defined for all tables in the application, so we've declared primary keys in `discounts` and `roysched`. Otherwise the application would function, but the data editing capabilities would be nonoperational for those two tables. This is an important issue to take into account when webifying your database.

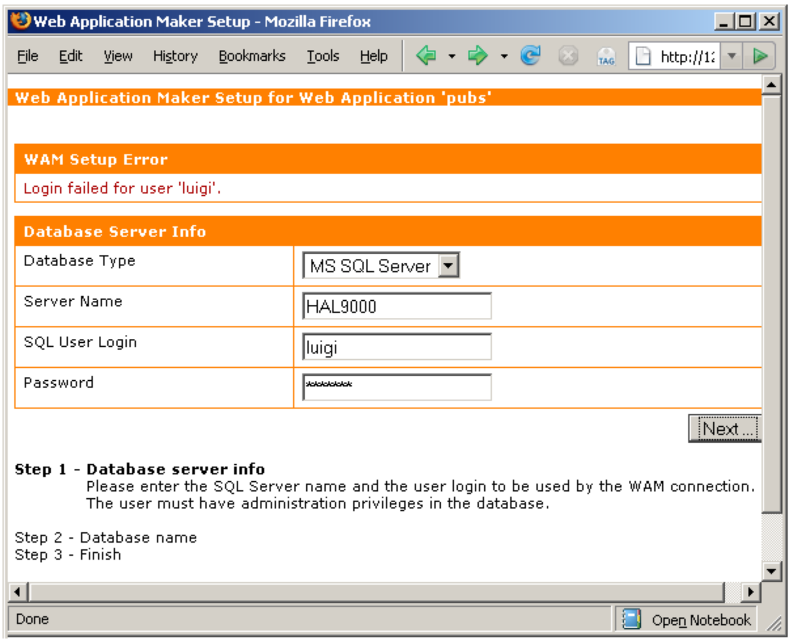
3.1.2 Step 2: Generate default WAMmodel

There is the possibility of running a single sequence of generic scripts that create the WAMmodel tables and generic stored procedures in database 'pubs', and that populate those with a default set of WAM meta-data, generated from the initial database structure. However, the WAM Installer utility does this automatically.

The WAM Installer can be executed by double-clicking on the `setup.wsf` file, from the WAM zip package.


C:\Documents and Settings\Luís\Desktop\WAMTrial
File Edit View Favorites Tools Help
Address C:\Documents and Settings\Luís\Desktop\WAMTrial
WAMLibrary
bff.js
setup.wsf
wap.wsf
4 objects
11,6 KB
My Computer

Note that it is assumed that the user has MS SQL Server installed on his machine and an IIS web site configured. Starting the WAM Installer will ask us where to configure our web application and which database to use. Then it will take us into the following screen on our web browser:


Web Application Maker Setup - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://1
Next ...
Done
Open Notebook

Web Application Maker Setup for Web Application 'pubs'

WAM Setup Error

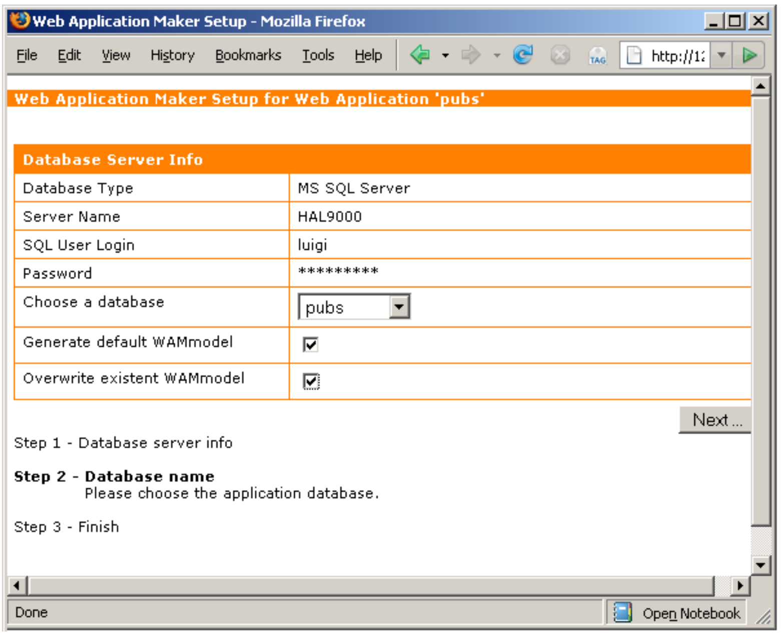
Login failed for user 'luigi'.

Database Server Info	
Database Type	MS SQL Server
Server Name	HAL9000
SQL User Login	luigi
Password	*******

Step 1 - Database server info
Please enter the SQL Server name and the user login to be used by the WAM connection.
The user must have administration privileges in the database.

Step 2 - Database name
Step 3 - Finish

Hitting next will take us too:


Web Application Maker Setup - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://1
Next ...
Done
Open Notebook

Web Application Maker Setup for Web Application 'pubs'

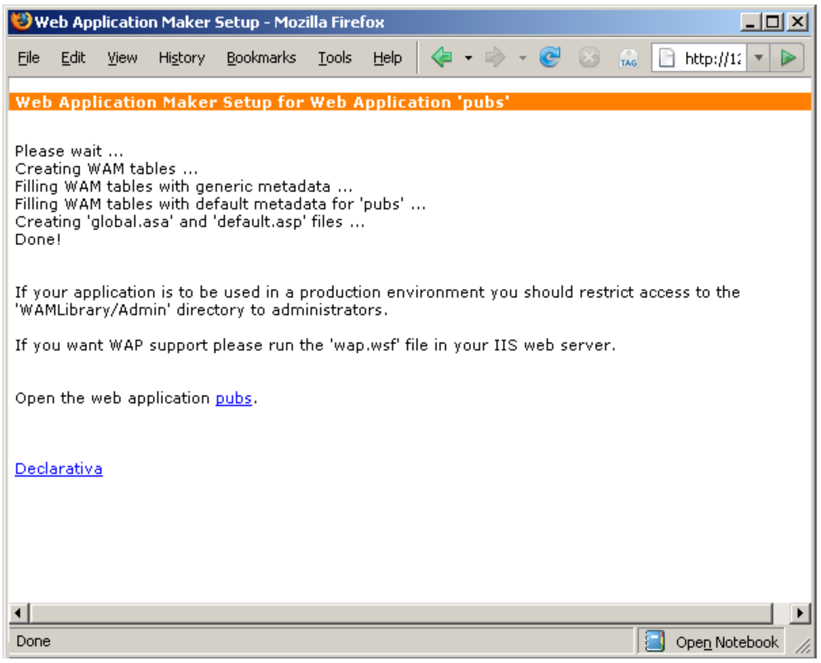
Database Server Info	
Database Type	MS SQL Server
Server Name	HAL9000
SQL User Login	luigi
Password	*********
Choose a database	pubs
Generate default WAMmodel	✓
Overwrite existent WAMmodel	✓

Step 1 - Database server info

Step 2 - Database name
Please choose the application database.

Step 3 - Finish

In our case we are overwriting the existing WAM Model, as we had already installed a version of WAM on the server. Hitting Next and then finish will complete the installation of WAM in our server and associate it with the 'pubs' database of MS SQL Server:



Web Application Maker Setup for Web Application 'pubs'

```
Please wait ...
Creating WAM tables ...
Filling WAM tables with generic metadata ...
Filling WAM tables with default metadata for 'pubs' ...
Creating 'global.asa' and 'default.asp' files ...
Done!
```

If your application is to be used in a production environment you should restrict access to the 'WAMLibrary/Admin' directory to administrators.

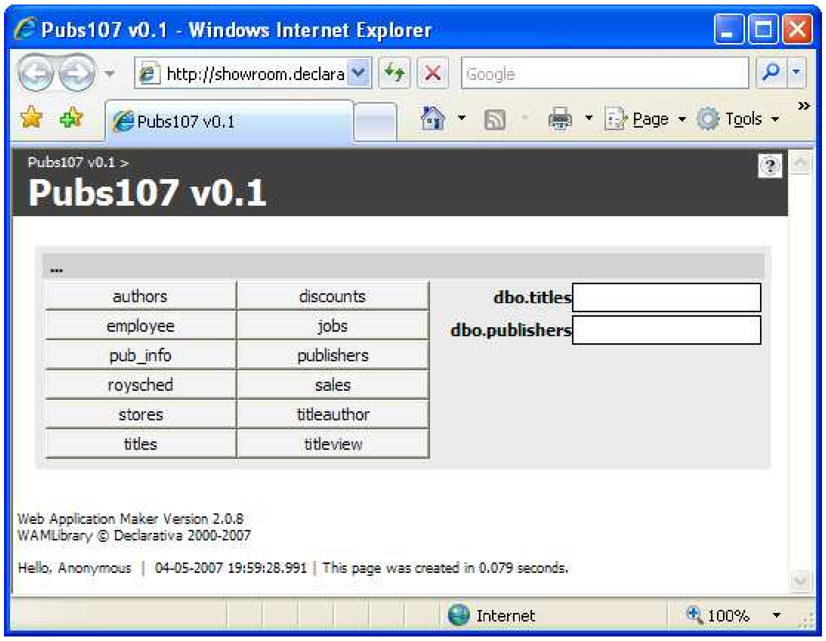
If you want WAP support please run the 'wap.wsf' file in your IIS web server.

Open the web application pubs.

Declarativa

3.1.3 Step 3: Demo

Open up the web application link in the previous screen (in the following screen shots the URL address will be different from yours). As nothing was stated yet regarding strings or external representations, initially database names are used as captions:



Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Hello, Anonymous | 04-05-2007 19:59:28.991 | This page was created in 0.079 seconds.

The above screen image shows the generic WAM application **entry page** for 'pubs', built from its WAMmodel:

- The 12 buttons provide navigation to lists for the 11 tables and 1 VIEW in the database.
- The caption/field pairs at the top right allow quick navigation to lists and records for the two most "interesting" tables in the application. Each caption/field pair constitutes a **Finder**.

Entering a value in a Finder field causes a simple search to be performed on its table, followed by navigation to a single record or to a list depending on how precise the value is. Clicking the field title navigates to the list, as suggested by the displayed ToolTip (near the invisible mouse cursor☺).

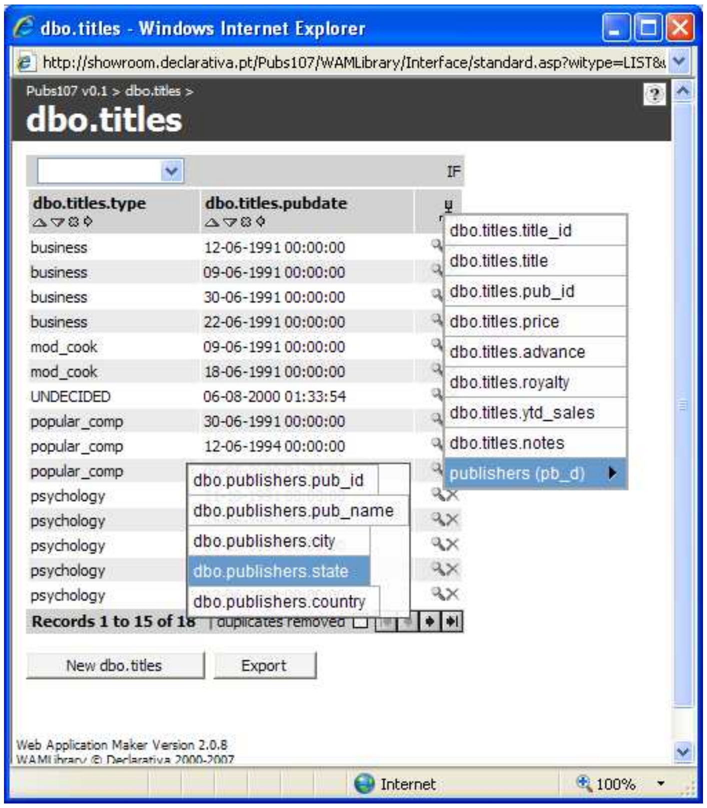
WAM uses a foreign key graph - based heuristic to automatically pick `titles` and `publisher` as "interesting tables" for pubs, and by default determines their primary key columns to be the columns to search. Some simple scripting can tune or expand this capability dramatically, adding multi-column and type sensitive searching, as will be seen later.

3.2 Introducing the user interface for WAM applications

This section provides an abridged description of the user interface functionality available to all WAM applications, with an eye on using its personalization aspects for development; for a more detailed and use-focused explanation see instead “WAM user interface: an application tour”.

3.2.1 Personalizing the information displayed in a list

Let's hit the 'dbo.titles' button and get the initial titles list, and hit the button at the top right:



dbo.titles.type	dbo.titles.pubdate
business	12-06-1991 00:00:00
business	09-06-1991 00:00:00
business	30-06-1991 00:00:00
business	22-06-1991 00:00:00
mod_cook	09-06-1991 00:00:00
mod_cook	18-06-1991 00:00:00
UNDECIDED	06-08-2000 01:33:54
popular_comp	30-06-1991 00:00:00
popular_comp	12-06-1994 00:00:00
popular_comp	
psychology	
psychology	
psychology	
psychology	
psychology	

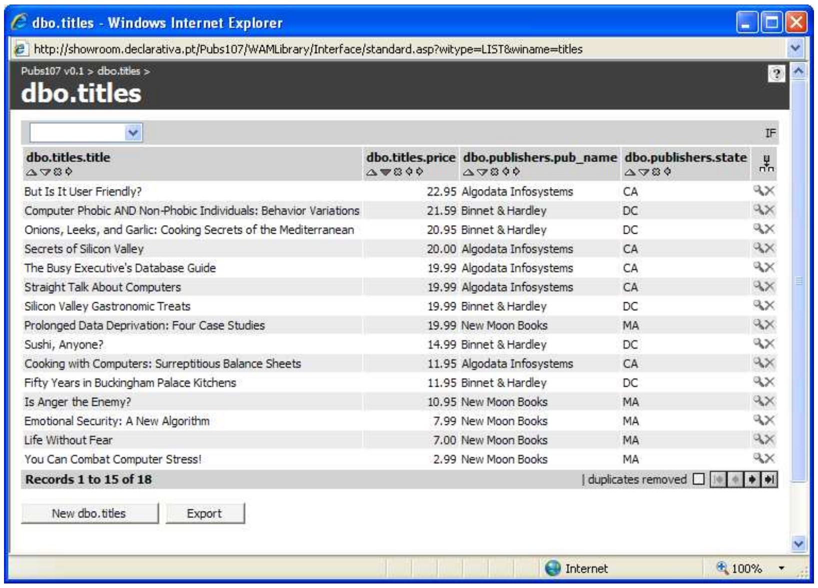
Records 1 to 15 of 18 | duplicates removed

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

On the absence of more information, WAM uses a generic data type-driven heuristic, initially configuring the **list** to display just the type and pubdate columns. However the user can add other columns, with the hierarchical popup menu shown; those can be columns either in the base table for the list, or in any *directly or indirectly related* table or view, as can be seen by the selected publishers table.

3.2.1.1 Adding/removing columns

Column adding/removal, sorting and search filter configure a *list instance personalized to each user*, persisting in the database **user preferences**. After a few (end-user) clicks he/she can get a more useful list, listing more expensive titles first and including publisher name and state. Additionally, the user may also rearrange the order of the several columns by dragging each column title to where it wants it to be placed:



dbo.titles - Windows Internet Explorer

http://showroom.declarativa.pt/Pubs107/WAMLibrary/Interface/standard.asp?witype=LIST&winame=titles

Pubs107 v0.1 > dbo.titles >

dbo.titles

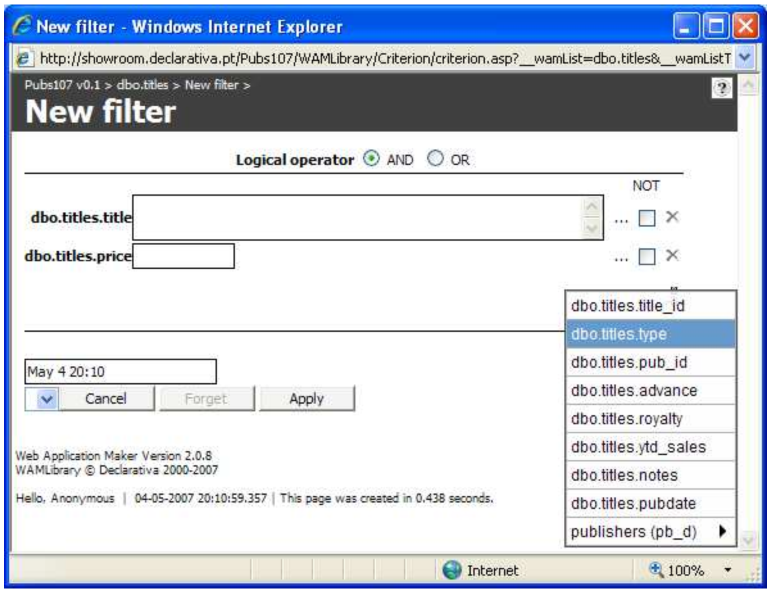
dbo.titles.title	dbo.titles.price	dbo.publishers.pub_name	dbo.publishers.state
But Is It User Friendly?	22.95	Algodata Infosystems	CA
Computer Phobic AND Non-Phobic Individuals: Behavior Variations	21.59	Binnet & Hardley	DC
Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean	20.95	Binnet & Hardley	DC
Secrets of Silicon Valley	20.00	Algodata Infosystems	CA
The Busy Executive's Database Guide	19.99	Algodata Infosystems	CA
Straight Talk About Computers	19.99	Algodata Infosystems	CA
Silicon Valley Gastronomic Treats	19.99	Binnet & Hardley	DC
Prolonged Data Deprivation: Four Case Studies	19.99	New Moon Books	MA
Sushi, Anyone?	14.99	Binnet & Hardley	DC
Cooking with Computers: Surreptitious Balance Sheets	11.95	Algodata Infosystems	CA
Fifty Years in Buckingham Palace Kitchens	11.95	Binnet & Hardley	DC
Is Anger the Enemy?	10.95	New Moon Books	MA
Emotional Security: A New Algorithm	7.99	New Moon Books	MA
Life Without Fear	7.00	New Moon Books	MA
You Can Combat Computer Stress!	2.99	New Moon Books	MA

Records 1 to 15 of 18 | duplicates removed

Notice that the URL for the page is a generic WAM page (actually the same as the previous screen; by default the page address is not displayed).

3.2.1.2 Display search filter

Lists can have a **search filter** applied. By using the popup menu at the left the following page opens up; using the button to add an additional data field to the filter definition…:



New filter - Windows Internet Explorer

http://showroom.declarativa.pt/Pubs107/WAMLibrary/Criterion/criterion.asp?__wamList=dbo.titles&__wamListT

Pubs107 v0.1 > dbo.titles > New filter >

New filter

Logical operator [✓ AND] OR

dbo.titles.title

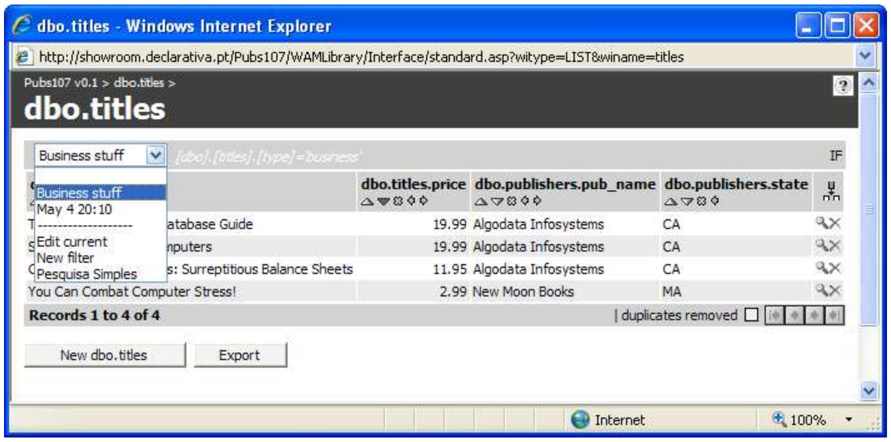
dbo.titles.price

May 4 20:10

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Hello, Anonymous | 04-05-2007 20:10:59.357 | This page was created in 0.438 seconds.

After typing "business" in the type field and "Business stuff" in the field at the bottom, and **Apply**ing the filter, the list appears with the search filter applied:


dbo.titles - Windows Internet Explorer

http://showroom.declarativa.pt/Pubs107/WAMLibrary/Interface/standard.asp?witype=LIST&winame=titles

Pubs107 v0.1 > dbo.titles >

dbo.titles

Business stuff [dbo].[titles].[type]='business' IF

Business stuff
May 4 20:10

Edit current
New filter
Pesquisa Simples

	dbo.titles.price	dbo.publishers.pub_name	dbo.publishers.state
T…atabase Guide	19.99	Algodata Infosystems	CA
S…nputers	19.99	Algodata Infosystems	CA
C…s: Surreptitious Balance Sheets	11.95	Algodata Infosystems	CA
You Can Combat Computer Stress!	2.99	New Moon Books	MA

Records 1 to 4 of 4 | duplicates removed

New dbo.titles | Export

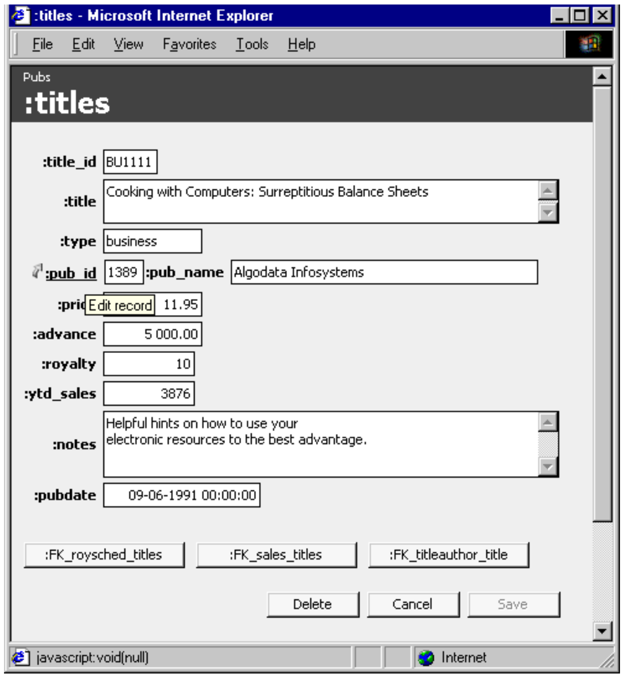
Internet 100%

The popup menu below shows the ability of WAM lists to have a set of filters defined and one (or none) picked as current Note that you can also add columns to your search filter and state what conditions will be applied on the chosen columns.

Search filters are also part of the list preferences, which reside in WAMmodel tables and can be copied, for example from expert to novice users. As they're server-based they'll hold consistent for the same authenticated user whenever he/she logs in from any browser.

3.2.2 Navigation between tables

Getting back to the list, let's click the magnifying glass to see a single record displayed in a WAM **row** page:


:titles - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Pubs

:titles

:title_id	BU1111
:title	Cooking with Computers: Surreptitious Balance Sheets
:type	business
:pub_id	1389 :pub_name Algodata Infosystems
:price	11.95
:advance	5 000.00
:royalty	10
:ytd_sales	3876
:notes	Helpful hints on how to use your electronic resources to the best advantage.
:pubdate	09-06-1991 00:00:00

Edit record

:FK_roysched_titles | :FK_sales_titles | :FK_titleauthor_title

Delete | Cancel | Save

javascript:void(null) Internet

The page displays all fields for the table, together with default (so far unspecified…) captions/titles, using a simple top-down layout policy; other policies are possible as will be seen below. Field database types are used to format and validate values.

The table’s relational role entails the potential for *detail to master* and *master to detail* navigation. As can be seen in the database diagram, 'titles' has one master table and 3 detail tables.

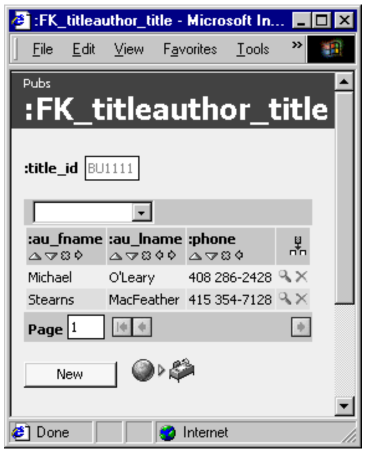
3.2.2.1 Detail to Master Navigation

First, let's examine *detail to master navigation.* `pub_id` is a foreign key into table publishers, and therefore WAM automatically creates a subjacent **lookup** object, providing the following abilities:

- Display values for a set of columns looked-up in the master table; a heuristic was used by WAM to initially pick `pub_name` as the single looked-up column; notice that this capability is akin to the automatic join performed in the titles list above.
- Automatic navigation to the related master record (the image was taken with the mouse hovering the "pub_id" title, hence the displayed ToolTip);
- Automatic navigation to a list of publishers from which to pick one, by clicking the upwards arrow

3.2.2.2 Master to Detail Navigation

Now for *master to detail navigation*, `roysched`, `titleauthor` and `sales` are detail tables, so initially there will be buttons in the row to navigate to standalone detail lists (as opposed to *row-embedded* detail lists, which we'll see later). Here are the **standalone detail lists** after we click into the titleauthor and sales buttons, and have a user adapt the lists to include related columns:

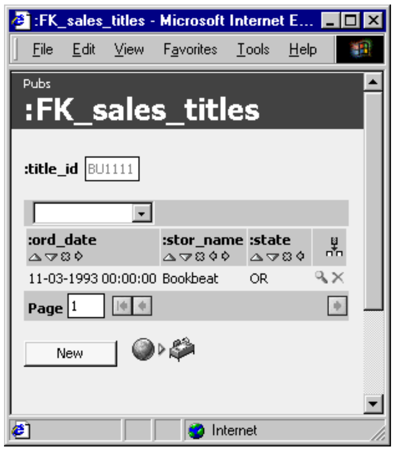

:FK_titleauthor_title - Microsoft In...
File Edit View Favorites Tools
Pubs
New
Done
Internet

:FK_titleauthor_title

:title_id BU1111

:au_fname	:au_lname	:phone
Michael	O'Leary	408 286-2428
Stearns	MacFeather	415 354-7128

Page 1


:FK_sales_titles - Microsoft Internet E...
File Edit View Favorites Tools Help
Pubs
New
Internet

:FK_sales_titles

:title_id BU1111

:ord_date	:stor_name	:state
11-03-1993 00:00:00	Bookbeat	OR

Page 1

Notice that each detail list includes a description of the (constraining) master record, initially (just) the primary key column. Simple configuring of the WAMmodel can refine this default policy, as will be seen later.

3.3   WAMmodel (explicit) improvements

So far we only saw (list) changes we can operate on the application’s output simply from its interface. In the following sections we will enhance the pubs application by filling or changing WAMmodel tables[16]. The WAMLibrary runtime system will automatically reflect the changes into the interface when the application pages are refreshed.

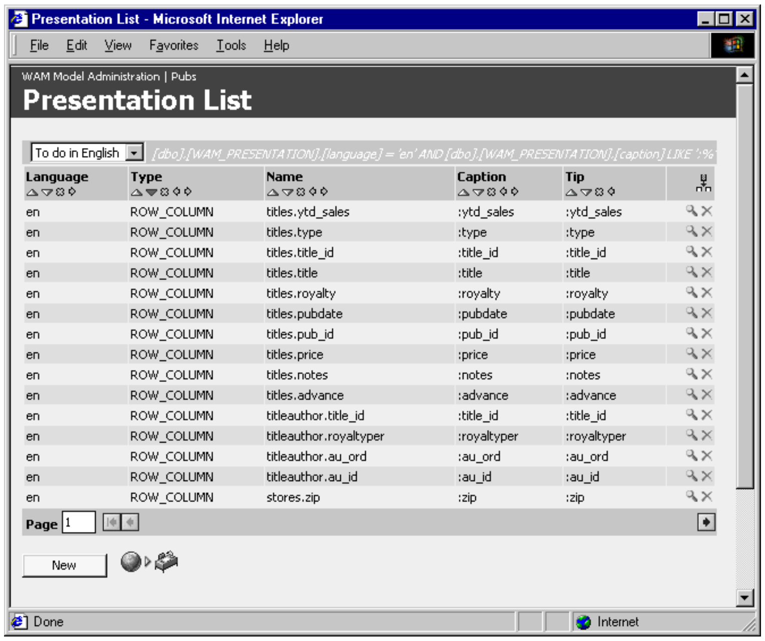
Most of what follows could also be performed through WAMAdmin (cf. section 7.1), instead of through a SQL script utility as shown

[16] Actually, the previous list changes already reflected into the WAMmodel as well, cf. section 4.1.5

3.3.1 Adding labels/captions

It's time to beautify the application, by making it use more meaningful captions to the interface rather than database object names. Opening up http://www.declarativa.com/pubs/WAMLibrary/Admin reveals a WAM administration facility to edit the WAMmodel tables:

Clicking the WAM Presentation button shows one of the WAMmodel tables:


Presentation List - Microsoft Internet Explorer
File Edit View Favorites Tools Help
WAM Model Administration | Pubs
Presentation List
To do in English
Page 1
New
Done
Internet

[dbo].[WAM_PRESENTATION].[language] = 'en' AND [dbo].[WAM_PRESENTATION].[caption] LIKE ':%

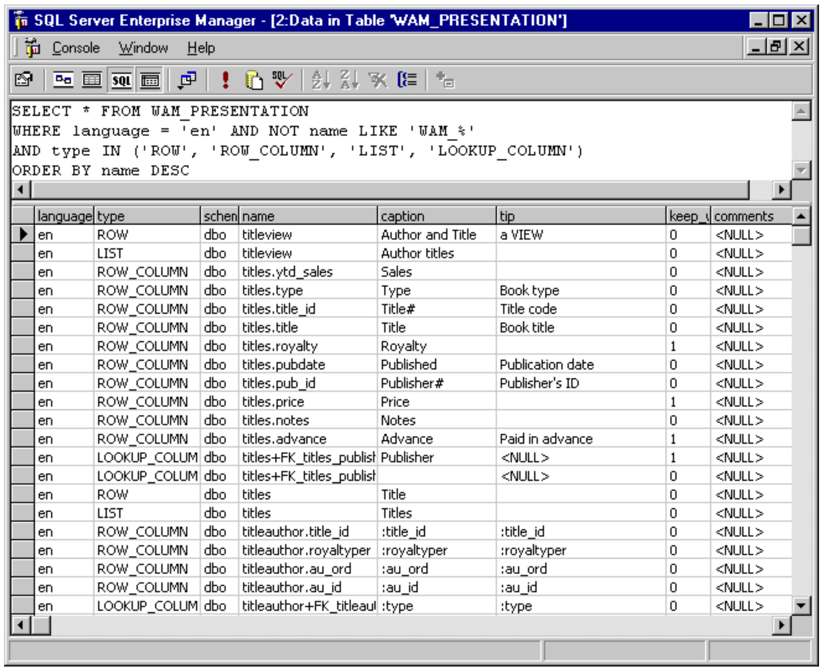
Language	Type	Name	Caption	Tip
en	ROW_COLUMN	titles.ytd_sales	:ytd_sales	:ytd_sales
en	ROW_COLUMN	titles.type	:type	:type
en	ROW_COLUMN	titles.title_id	:title_id	:title_id
en	ROW_COLUMN	titles.title	:title	:title
en	ROW_COLUMN	titles.royalty	:royalty	:royalty
en	ROW_COLUMN	titles.pubdate	:pubdate	:pubdate
en	ROW_COLUMN	titles.pub_id	:pub_id	:pub_id
en	ROW_COLUMN	titles.price	:price	:price
en	ROW_COLUMN	titles.notes	:notes	:notes
en	ROW_COLUMN	titles.advance	:advance	:advance
en	ROW_COLUMN	titleauthor.title_id	:title_id	:title_id
en	ROW_COLUMN	titleauthor.royaltyper	:royaltyper	:royaltyper
en	ROW_COLUMN	titleauthor.au_ord	:au_ord	:au_ord
en	ROW_COLUMN	titleauthor.au_id	:au_id	:au_id
en	ROW_COLUMN	stores.zip	:zip	:zip

The `WAM_PRESENTATION` table defines the external strings for the application in a particular language. Each database/WAM object can get a caption (to be interpreted as list column title, field label, page title etc.) and a ToolTip. The above list was already personalized to include only records for English which require revision - in the initial application setup step WAM names all captions with the database object name, prefixed with ":".

`WAM_PRESENTATION` can be filled either using the WAM Administration application above or by dumping data into the database from some other tool more convenient for the engineer, say a SQL Query tool or something closer to the users like Excel.

The pubs database has 11 tables totaling 64 column fields. Each field can appear in a row, list, lookup etc., and all can have unique captions defined, so for the `pubs` application several hundred captions would need to be inserted. However WAM uses several inheritance rules to "get away with" not having to define all captions, at least until the moment users demand them☺. For example, columns in lists and detail lists inherit row field captions, and views inherit captions from their base tables. These rules can be over-ridden simply by adding records to `WAM_PRESENTATION`.

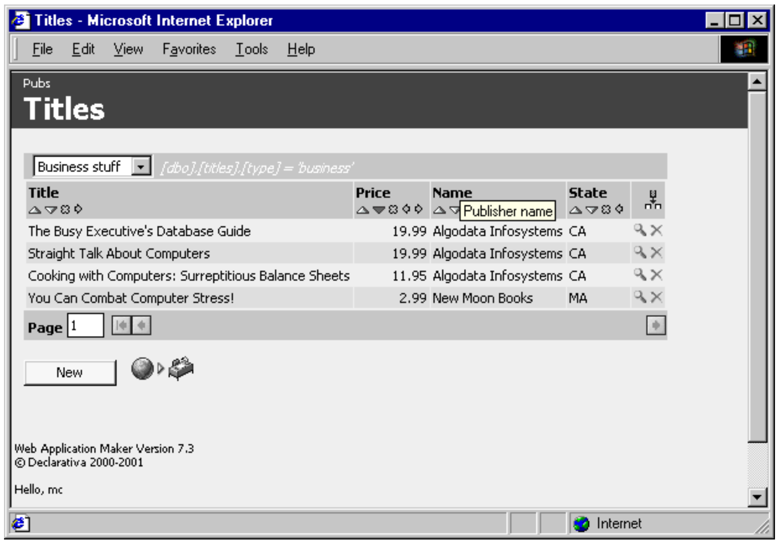
So we'll now define captions for most table fields, as they should appear in rows, plus another caption for each row title, list title and looked-up column. The next screen shows an SQL Server Enterprise Manager window with the first of the relevant records in `WAM_PRESENTATION`:


SQL Server Enterprise Manager - [2:Data in Table 'WAM_PRESENTATION']
Console Window Help

```
SELECT * FROM WAM_PRESENTATION
WHERE language = 'en' AND NOT name LIKE 'WAM_%'
AND type IN ('ROW', 'ROW_COLUMN', 'LIST', 'LOOKUP_COLUMN')
ORDER BY name DESC
```

language	type	schen	name	caption	tip	keep_	comments
en	ROW	dbo	titleview	Author and Title	a VIEW	0	<NULL>
en	LIST	dbo	titleview	Author titles		0	<NULL>
en	ROW_COLUMN	dbo	titles.ytd_sales	Sales		0	<NULL>
en	ROW_COLUMN	dbo	titles.type	Type	Book type	0	<NULL>
en	ROW_COLUMN	dbo	titles.title_id	Title#	Title code	0	<NULL>
en	ROW_COLUMN	dbo	titles.title	Title	Book title	0	<NULL>
en	ROW_COLUMN	dbo	titles.royalty	Royalty		1	<NULL>
en	ROW_COLUMN	dbo	titles.pubdate	Published	Publication date	0	<NULL>
en	ROW_COLUMN	dbo	titles.pub_id	Publisher#	Publisher's ID	0	<NULL>
en	ROW_COLUMN	dbo	titles.price	Price		1	<NULL>
en	ROW_COLUMN	dbo	titles.notes	Notes		0	<NULL>
en	ROW_COLUMN	dbo	titles.advance	Advance	Paid in advance	1	<NULL>
en	LOOKUP_COLUM	dbo	titles+FK_titles_publisl	Publisher	<NULL>	1	<NULL>
en	LOOKUP_COLUM	dbo	titles+FK_titles_publisl		<NULL>	0	<NULL>
en	ROW	dbo	titles	Title		0	<NULL>
en	LIST	dbo	titles	Titles		0	<NULL>
en	ROW_COLUMN	dbo	titleauthor.title_id	:title_id	:title_id	0	<NULL>
en	ROW_COLUMN	dbo	titleauthor.royaltyper	:royaltyper	:royaltyper	0	<NULL>
en	ROW_COLUMN	dbo	titleauthor.au_ord	:au_ord	:au_ord	0	<NULL>
en	ROW_COLUMN	dbo	titleauthor.au_id	:au_id	:au_id	0	<NULL>
en	LOOKUP_COLUM	dbo	titleauthor+FK_titleaul	:type	:type	0	<NULL>

Our first list will now look as follows:


Titles - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Titles
Business stuff
[dbo].[titles].[type] = 'business'
Publisher name
Page 1
New
Internet

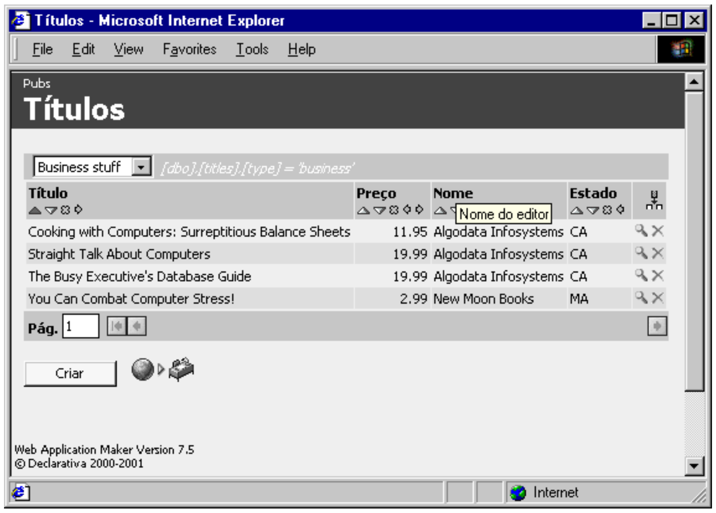
Title	Price	Name	State
The Busy Executive's Database Guide	19.99	Algodata Infosystems	CA
Straight Talk About Computers	19.99	Algodata Infosystems	CA
Cooking with Computers: Surreptitious Balance Sheets	11.95	Algodata Infosystems	CA
You Can Combat Computer Stress!	2.99	New Moon Books	MA

Web Application Maker Version 7.3
© Declarativa 2000-2001

Hello, mc

Notice that list columns are inheriting caption and ToolTip from the row field information of their base tables.

In order to show-off WAM's multilingual capability we also added a few Portuguese records to WAM_PRESENTATION. After the user reconfigures the browser to prefer Portuguese the previous page will appear refreshed instead as:


Títulos - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Business stuff
Nome do editor
Criar
Internet

Pubs

Títulos

[dbo].[titles].[type] = 'business'

Título	Preço	Nome	Estado
Cooking with Computers: Surreptitious Balance Sheets	11.95	Algodata Infosystems	CA
Straight Talk About Computers	19.99	Algodata Infosystems	CA
The Busy Executive's Database Guide	19.99	Algodata Infosystems	CA
You Can Combat Computer Stress!	2.99	New Moon Books	MA

Pág. 1

Web Application Maker Version 7.5
© Declarativa 2000-2001

Notice how all application strings now appears in Portuguese. Data of course continues in English, and the filter "Business stuff" remains untranslated because the user wrote it.

3.3.2 Model-based row layout tuning

Controlling layout is necessary, namely in rows, and ultimately may require ASP scripting as will be seen later. WAM performs a simple row layout based on database table column order and field type information, assuming that the order chosen by the programmer is, more often than not, pertinent to the GUI.

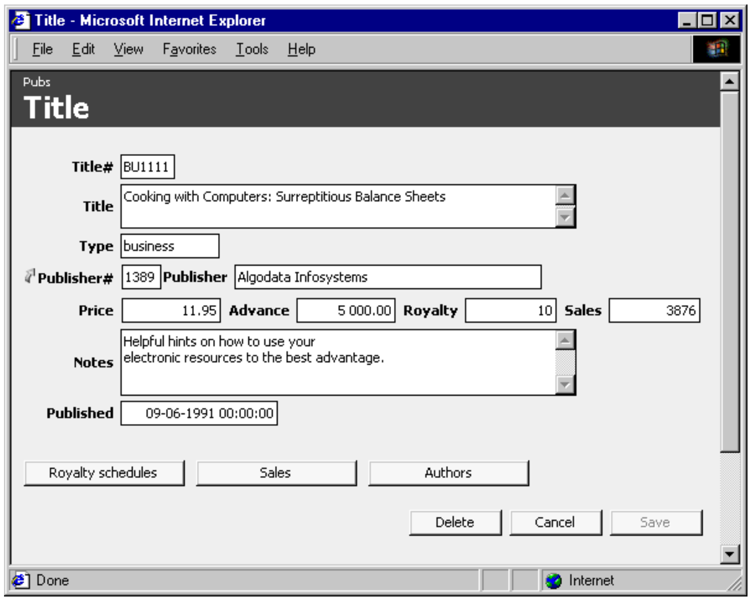
Were we model-driven fundamentalists we would have fattened the WAMmodel with layout information; we chose instead to keep the WAMmodel as lean as possible, and so column ordering is not there (except for lookup columns, which have no direct database concept counterpart), nor is field grouping. A *different field ordering can be specified with an extra database VIEW*, thereby defining a different row page, or with ASP scripting.

Nevertheless we wanted a minimal way to "beautify" forms with some form of "field grouping" in the WAMmodel. So we borrowed a Microsoft Word declarative layout mechanism it uses in paragraphs, a *keep with next* bit in the `WAM_PRESENTATION`, which conditions "line" breaks in a WAM row page.

For example, let's set to true those bits for Price, Advance and Royalty in the title row. This will now be shown as SQL instructions, to give a flavor of what's possible with direct WAMmodel manipulation:

```
UPDATE WAM_PRESENTATION SET keep_with_next=1
WHERE type='ROW_COLUMN' AND schema_name='dbo' AND
name IN ('titles.price', 'titles.advance','titles.royalty')
```

Now the "Cooking with Computers: Surreptitious Balance Sheets" record will appear with the sales-related numbers together:


Title - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Royalty schedules
Sales
Authors
Delete
Cancel
Save
Done
Internet

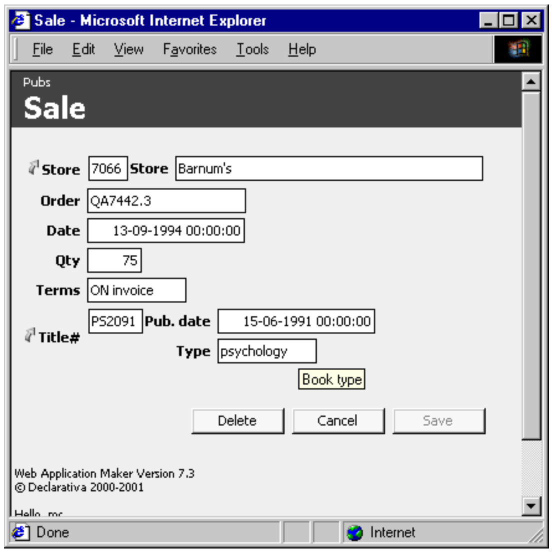
Title

Title#	BU1111
Title	Cooking with Computers: Surreptitious Balance Sheets
Type	business
Publisher#	1389
Publisher	Algodata Infosystems
Price	11.95
Advance	5 000.00
Royalty	10
Sales	3876
Notes	Helpful hints on how to use your electronic resources to the best advantage.
Published	09-06-1991 00:00:00

Note that these changes could have been performed from within the WAM Admin. As that is a straightforward process, we'll include the SQL versions here from now on.

3.3.3 Adapt a lookup for a row

Let's go back to the entry page, open up the Sales list and see one of its records:


Sale - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Book type
Delete
Cancel
Save
Web Application Maker Version 7.3
© Declarativa 2000-2001
Hello, mc
Done
Internet

Sale

Store	7066
Store	Barnum's
Order	QA7442.3
Date	13-09-1994 00:00:00
Qty	75
Terms	ON invoice
Title#	PS2091
Pub. date	15-06-1991 00:00:00
Type	psychology

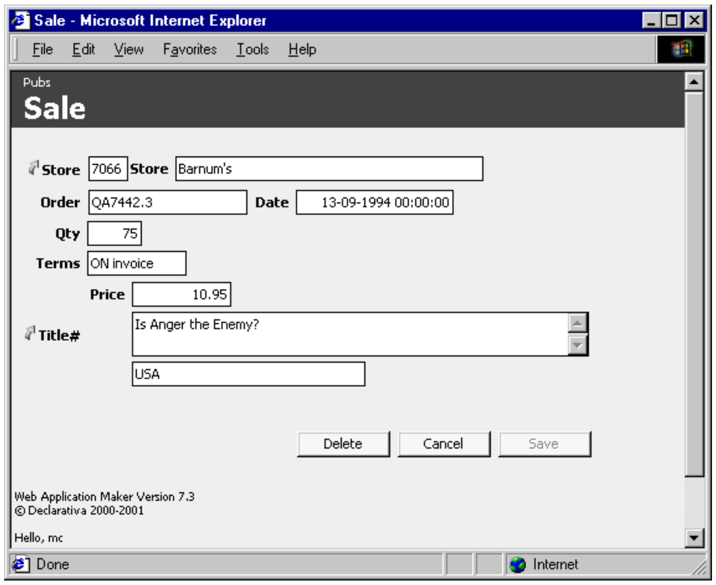
WAM did a reasonable guess for the looked-up column to use representing the related store, but probably not for the title.

Let's see now how to hide the Title# field, which users might dispense with, and show instead the price, title and the country of its publisher. This requires solely a few records changed in the WAM tables; notice how the lookup columns are named with the supporting foreign key (or in general, **foreign key path**) and are associated with the row they serve:

```
-- Remove the current (bad) lookup:
DELETE FROM WAM_LOOKUP_COLUMN
WHERE table_schema='dbo' and table_name='sales' and constraint_name='FK_sales_titles'
-- Add price and title:
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'sales', 'dbo', 'FK_sales_titles', 'price', '', 1, 1, 1 )
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'sales', 'dbo', 'FK_sales_titles', 'title', '', 2, 1, 1 )
-- Add country; since it is not in the master table a foreign key path must be given:
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'sales', 'dbo', 'FK_sales_titles', 'country',
'dbo.FK_sales_titles,dbo.FK_titles_publishers', 3, 1, 1 )
-- Add a label to the looked-up price:
INSERT INTO WAM_PRESENTATION
VALUES( 'en', 'LOOKUP_COLUMN', 'dbo', 'sales+FK_sales_titles+dbo.titles.price', 'Price', 'Book
price', 0, 'Optional developer comment here' )
-- Put the date closer to the order code:
UPDATE WAM_PRESENTATION SET keep_with_next=1
WHERE  language='en' AND type='ROW_COLUMN' AND schema='dbo' AND name='sales.ord_num')
```

WAM supports applications over several database schemas, so names must be qualified with the enclosing schema, `dbo` in the example.

Here's the result:


Sale - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Sale
Delete
Cancel
Save
Done
Internet

Store	7066	Store	Barnum's
Order	QA7442.3	Date	13-09-1994 00:00:00
Qty	75		
Terms	ON invoice		
Price	10.95		
Title#	Is Anger the Enemy?		
	USA		

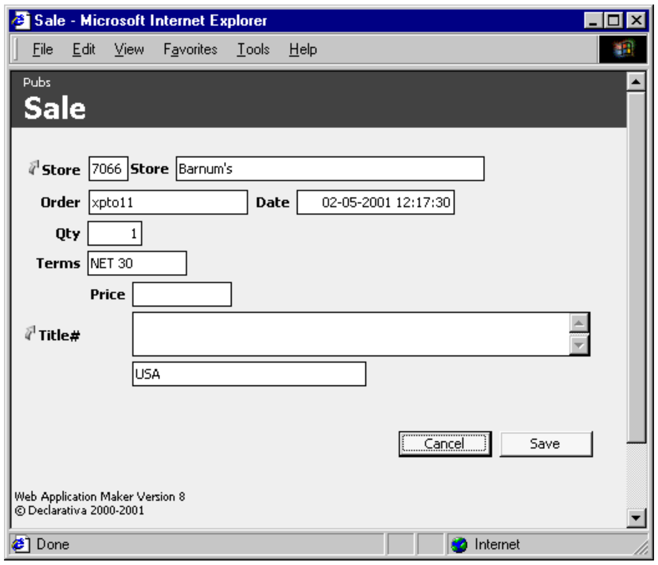
Web Application Maker Version 7.3
© Declarativa 2000-2001
Hello, mc

The last fields in the `WAM_LOOKUP_COLUMN` records specified an ordering among looked-up columns, and also that they are editable (see WAMmodel section); this means that **inverse lookups** will be possible: the user may type part of a string in a looked-up column and the system will either fill-out all the foreign key and looked-up columns, or it will open up a list window for a related record to be picked.

Let's go back to the Sales list, click "New" to add a Sales record, and:

- type 'Bar' in the (looked-up) store field,
- type an order code,
- type "2" in the date field,
- type quantity and terms
- type "USA" in the (looked-up) price field

Here's the resulting page, just before exiting the (looked-up) country field:


Sale - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Cancel
Save
Done
Internet

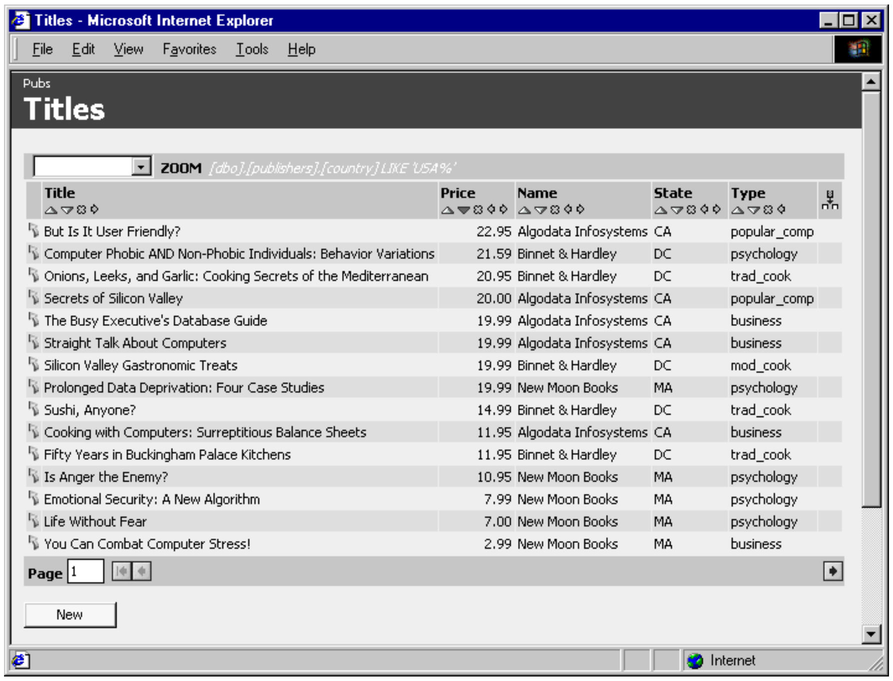
Pubs
Sale

Store	7066 Store Barnum's
Order	xpto11 Date 02-05-2001 12:17:30
Qty	1
Terms	NET 30
Price	
Title#	
	USA

Web Application Maker Version 8
© Declarativa 2000-2001

Notice that 'Bar' caused a successful **inverse lookup** in table `stores` (as only one record had store name beginning with Bar), and that the system expanded "2" to a full datetime value; were there database default values defined for the table, say for Qty, they would have been used.

On exiting the country field another inverse lookup is attempted; unlike the store lookup this one is ambiguous, so the following titles list pops up, in the special "zoom" context used to disambiguate lookups:


Titles - Microsoft Internet Explorer
File Edit View Favorites Tools Help
New
Internet

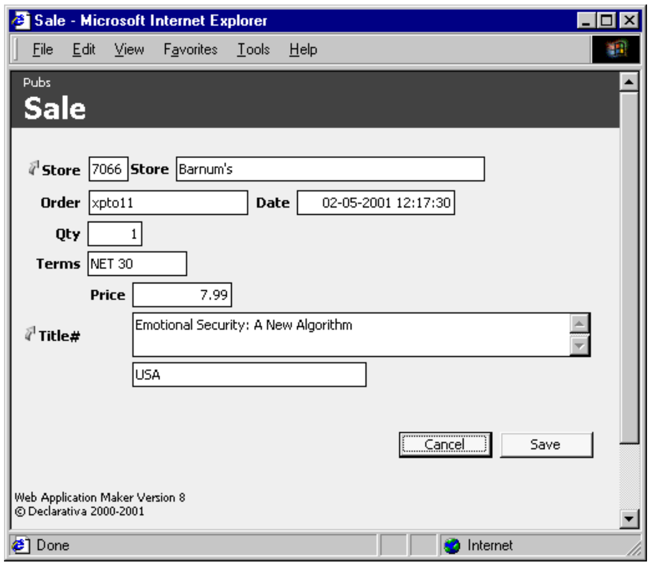
Pubs
Titles

ZOOM [dbo].[publishers].[country] LIKE 'USA%'

Title	Price	Name	State	Type
But Is It User Friendly?	22.95	Algodata Infosystems	CA	popular_comp
Computer Phobic AND Non-Phobic Individuals: Behavior Variations	21.59	Binnet & Hardley	DC	psychology
Onions, Leeks, and Garlic: Cooking Secrets of the Mediterranean	20.95	Binnet & Hardley	DC	trad_cook
Secrets of Silicon Valley	20.00	Algodata Infosystems	CA	popular_comp
The Busy Executive's Database Guide	19.99	Algodata Infosystems	CA	business
Straight Talk About Computers	19.99	Algodata Infosystems	CA	business
Silicon Valley Gastronomic Treats	19.99	Binnet & Hardley	DC	mod_cook
Prolonged Data Deprivation: Four Case Studies	19.99	New Moon Books	MA	psychology
Sushi, Anyone?	14.99	Binnet & Hardley	DC	trad_cook
Cooking with Computers: Surreptitious Balance Sheets	11.95	Algodata Infosystems	CA	business
Fifty Years in Buckingham Palace Kitchens	11.95	Binnet & Hardley	DC	trad_cook
Is Anger the Enemy?	10.95	New Moon Books	MA	psychology
Emotional Security: A New Algorithm	7.99	New Moon Books	MA	psychology
Life Without Fear	7.00	New Moon Books	MA	psychology
You Can Combat Computer Stress!	2.99	New Moon Books	MA	business

Page 1

Notice how the records in the master (`titles`) table are restricted with the country condition in its master (`publishers`); WAM takes care of automatically building the underlying outer join. Clicking one of the upwards arrows picks a record back into the new Sale record:

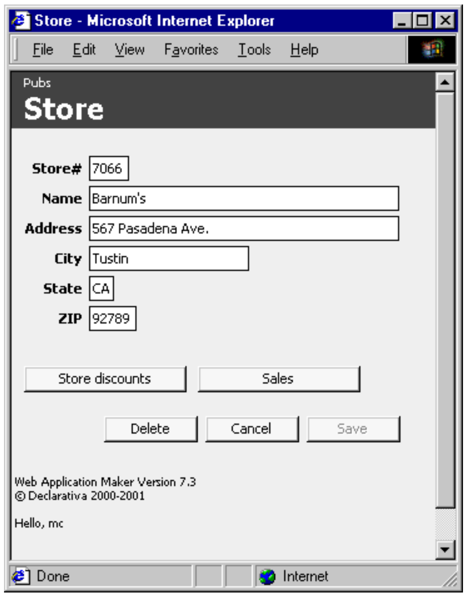

Sale - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Sale
Cancel
Save
Done
Internet

Store	7066	Store	Barnum's
Order	xpto11	Date	02-05-2001 12:17:30
Qty	1		
Terms	NET 30		
Price	7.99		
Title#	Emotional Security: A New Algorithm		
	USA		

Web Application Maker Version 8
© Declarativa 2000-2001

3.3.4 Add a (deep) detail list

We'll now see how to create a detail list for a row, by adding a detail list of titles to the existing store row, which initially got only navigation to its (immediate) detail tables, `discounts` and `sales`. This detail list will actually be a button with the same functionality of “Store Discounts” and “Sales” but related with Titles.


Store - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Store
Store discounts
Sales
Delete
Cancel
Save
Done
Internet

Store#	7066
Name	Barnum's
Address	567 Pasadena Ave.
City	Tustin
State	CA
ZIP	92789

Web Application Maker Version 7.3
© Declarativa 2000-2001

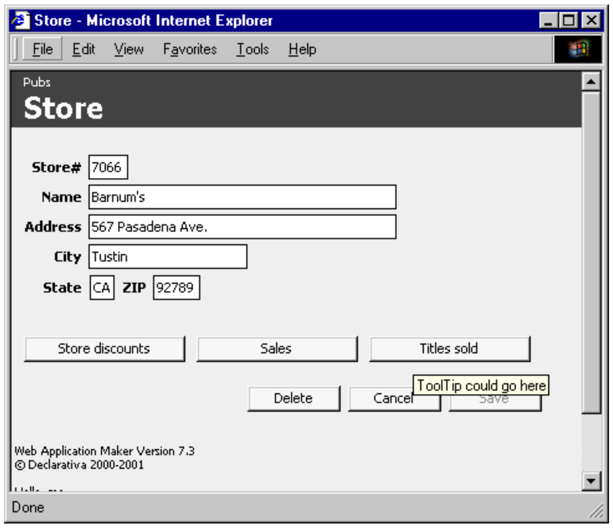
Hello, mc

For this we'll again resort to the concept of foreign key path, and change the WAMmodel:

```
-- define a detail list based on the 2 endpoints of the FK path, linked to an editing row
-- for titles, and showing list search filters:
INSERT INTO WAM_LIST
VALUES('dbo', 'dbo', 'FK_sales_stores,dbo.FK_sales_titles', 'dbo', 'titles', 1, null, 0, 1, 0)
-- define a string for the button and list title:
INSERT INTO WAM_PRESENTATION
VALUES('en', 'LIST', 'dbo', 'FK_sales_stores,dbo.FK_sales_titles', 'Titles sold', 'ToolTip could
go here', 0, '')
```

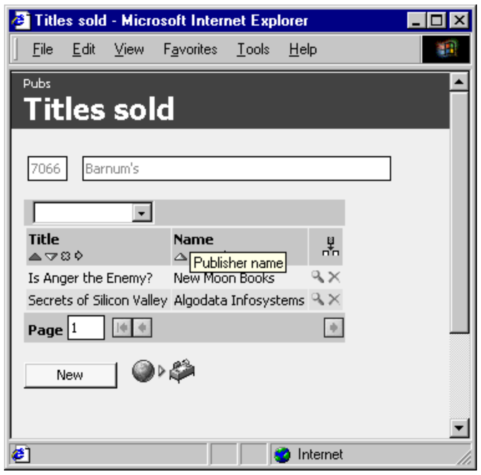
```
-- provide more context to the detail list:
INSERT INTO WAM_LOOKUP_COLUMN
VALUES('dbo', 'FK_sales_stores,dbo.FK_sales_titles', 'dbo', 'FK_sales_stores', 'stor_name', 1, 0,
1)
-- while we're at it, put the zip code next to the city:
UPDATE WAM_PRESENTATION SET keep_with_next=1
WHERE  language='en' AND type='ROW_COLUMN' AND schema='dbo' AND name='stores.state'
```

Here's the result, when the mouse is hovering over the new button:


Store - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Store

Store#	7066
Name	Barnum's
Address	567 Pasadena Ave.
City	Tustin
State	CA ZIP 92789

Clicking the button displays the new list, already customized by the user who added a titles column and a related column from publishers (the mouse is hovering over it, hence the displayed ToolTip):


Titles sold - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Titles sold
7066
Barnum's
Publisher name
Page 1
New
Internet

Title	Name
Is Anger the Enemy?	New Moon Books
Secrets of Silicon Valley	Algodata Infosystems

Were captions desired near the store description (lookup), it would be necessary to add just one or two records to `WAM_PRESENTATION`.

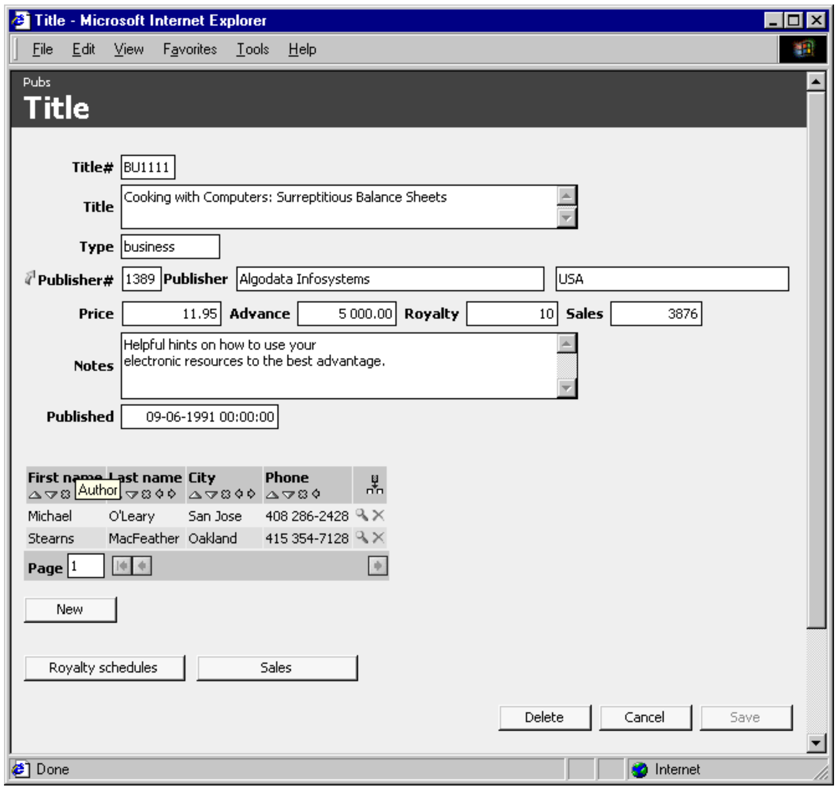
3.3.5 Make a detail list embedded in a row

So far we've seen **standalone detail lists**: detail lists invoked from a row, but which appear in a separate page.

Many times it's useful to embed a list in a row; all that's involved to make an existing detail list embedded in the row for its master table is removing the correspondent records from the WAM_LOOKUP_COLUMN table, and optionally hiding its search filter popup menu:

```
DELETE FROM WAM_LOOKUP_COLUMN
WHERE table_schema='dbo' AND table_name='FK_titleauthor_title'
UPDATE WAM_LIST set show_criterion=0
WHERE user_id='dbo' AND table_schema='dbo' AND table_name = 'FK_titleauthor_title'
```

On reopening a title row, and after the user picked the 4 columns shown in the list, here's the result, a row with an **embedded detail list**:


Title - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Title

Title#	BU1111
Title	Cooking with Computers: Surreptitious Balance Sheets
Type	business
Publisher#	1389
Publisher	Algodata Infosystems
	USA
Price	11.95
Advance	5 000.00
Royalty	10
Sales	3876
Notes	Helpful hints on how to use your electronic resources to the best advantage.
Published	09-06-1991 00:00:00

First name	Last name	City	Phone
Michael	O'Leary	San Jose	408 286-2428
Stearns	MacFeather	Oakland	415 354-7128

Notice that `titleauthor` is a "link" table implementing a N-N relationship between authors and titles; although it was the base for defining the detail list, the WAM automatic join and user preference mechanisms allowed an user to actually tailor the list to effectively become "authors for a title", and not just "titleauthor link records for a title".

At this stage we have an application that is capable of displaying and editing all 'pubs' data in a nontrivial manner, including type checking in the browser layer, database layer error handling, and other features. The application at this stage consists just on the generic WAMLibrary runtime system together with the default WAMmodel, which was obtained automatically from the database, as well as some tuning in the WAM Admin area.

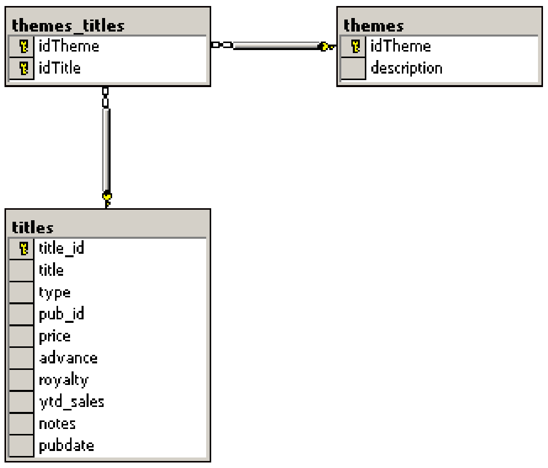
In the next chapters we will improve it by performing some database improvements and ASP scripting.

3.4 Improving the database

We'll now make some changes to the database, enhancing the interface but still without ASP scripting.

3.4.1 Adding tables

The first issue that could occur during implementation phase of a database based web application is the changing of the database itself. In its most simple case, this would mean adding/removing tables. Will now add a table called ‘themes’, where we’ll store possible themes for each title, and associate it with the ‘titles’ table. As each title may have several themes, and each theme can be subject of a single title, we’ll have a connection table in between:



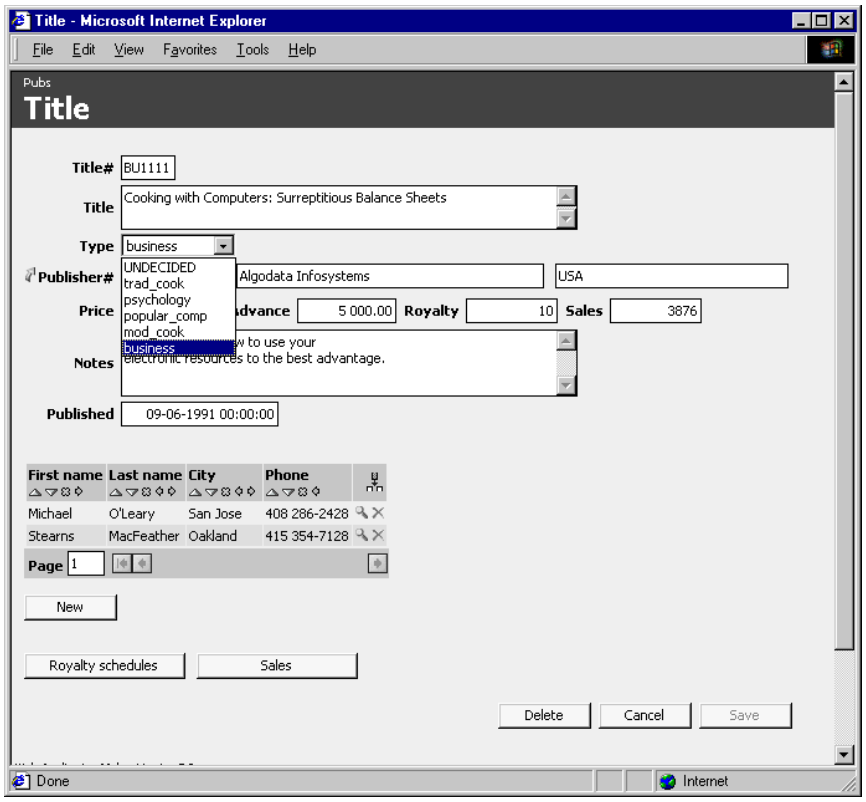
(…)

3.4.2 Add a check constraint to a table

Table constraints allow a declarative style for specifying some restrictions on data. Let's suppose that we wish to constrain the field `titles.type` to the set of values present in the example database:

```
ALTER TABLE dbo.titles ADD CONSTRAINT CK_titles
CHECK ([type] = 'UNDECIDED' or [type] = 'trad_cook' or [type] = 'psychology' or [type] =
'popular_comp' or [type] = 'mod_cook' or [type] = 'business')
```

WAM detects this particular form of table check constraints at runtime, to automatically improve the GUI with a popup menu:


Title - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Title

Title#	BU1111
Title	Cooking with Computers: Surreptitious Balance Sheets
Type	business
Publisher#	Algodata Infosystems USA
Price / Advance / Royalty / Sales	5 000.00 / 10 / 3876
Notes	…w to use your electronic resources to the best advantage.
Published	09-06-1991 00:00:00

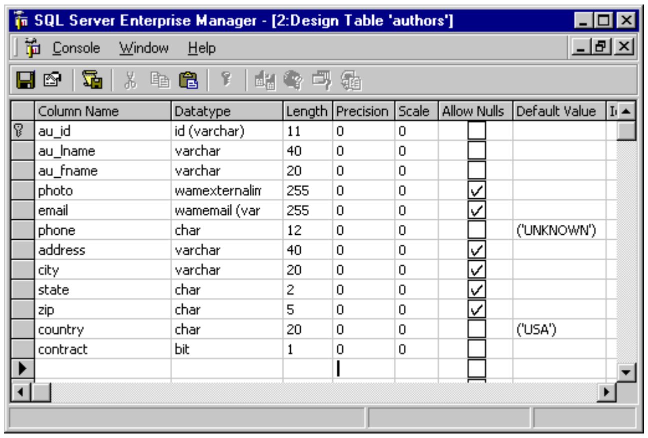
UNDECIDED
trad_cook
psychology
popular_comp
mod_cook
business

First name	Last name	City	Phone
Michael	O'Leary	San Jose	408 286-2428
Stearns	MacFeather	Oakland	415 354-7128

3.4.3 Add fields to a table

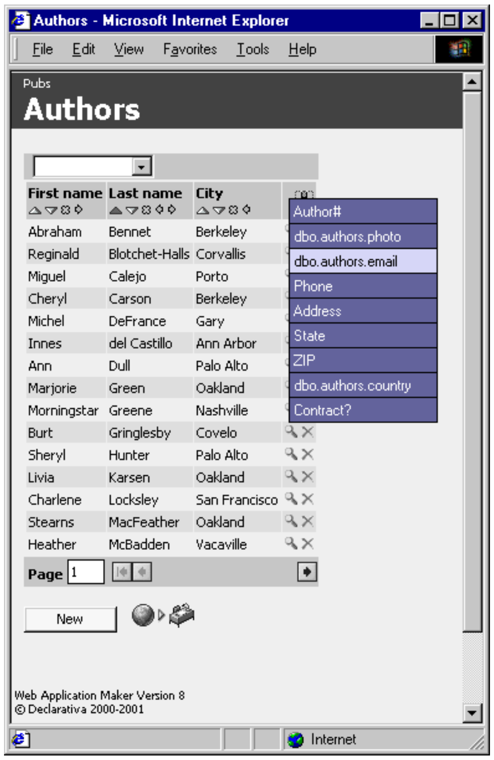
WAM includes a few "user-defined data types" providing additional validation, formatting or editing functionality in the GUI (See *Predefined user data types*).

We'll now add an external image field to the authors table, an email and a mandatory country field (with default 'USA'), putting them in the logical positions within the table:



Column Name	Datatype	Length	Precision	Scale	Allow Nulls	Default Value
au_id	id (varchar)	11	0	0		
au_lname	varchar	40	0	0		
au_fname	varchar	20	0	0		
photo	wamexternalir	255	0	0	✓	
email	wamemail (var	255	0	0	✓	
phone	char	12	0	0		('UNKNOWN')
address	varchar	40	0	0	✓	
city	varchar	20	0	0	✓	
state	char	2	0	0	✓	
zip	char	5	0	0	✓	
country	char	20	0	0		('USA')
contract	bit	1	0	0		

The Authors list immediately reflects the change in its column popup menu:

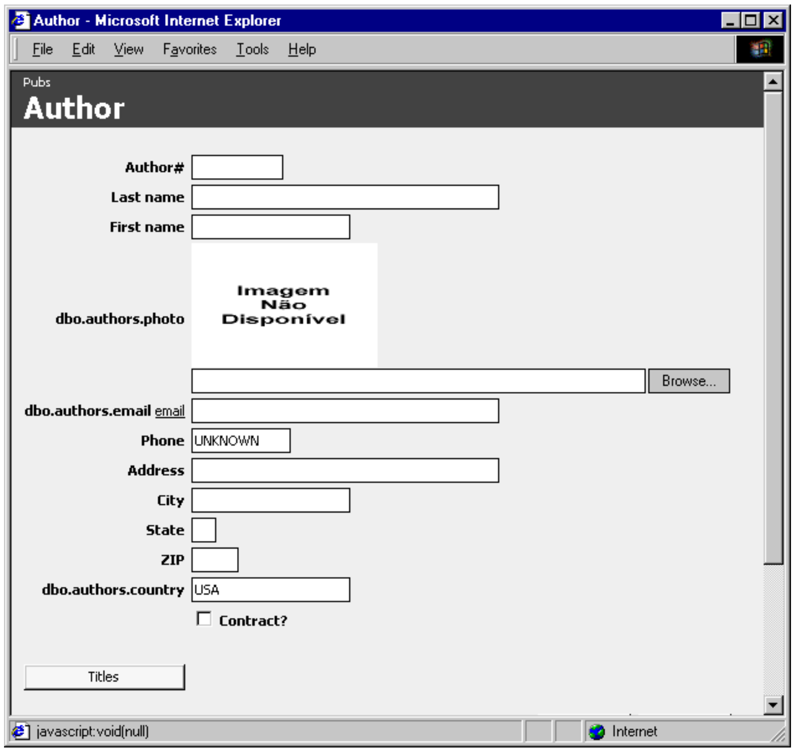

Authors - Microsoft Internet Explorer

Pubs
Authors

First name	Last name	City
Abraham	Bennet	Berkeley
Reginald	Blotchet-Halls	Corvallis
Miguel	Calejo	Porto
Cheryl	Carson	Berkeley
Michel	DeFrance	Gary
Innes	del Castillo	Ann Arbor
Ann	Dull	Palo Alto
Marjorie	Green	Oakland
Morningstar	Greene	Nashville
Burt	Gringlesby	Covelo
Sheryl	Hunter	Palo Alto
Livia	Karsen	Oakland
Charlene	Locksley	San Francisco
Stearns	MacFeather	Oakland
Heather	McBadden	Vacaville

Web Application Maker Version 8
© Declarativa 2000-2001

Clicking the "New" button opens up an improved Author row:

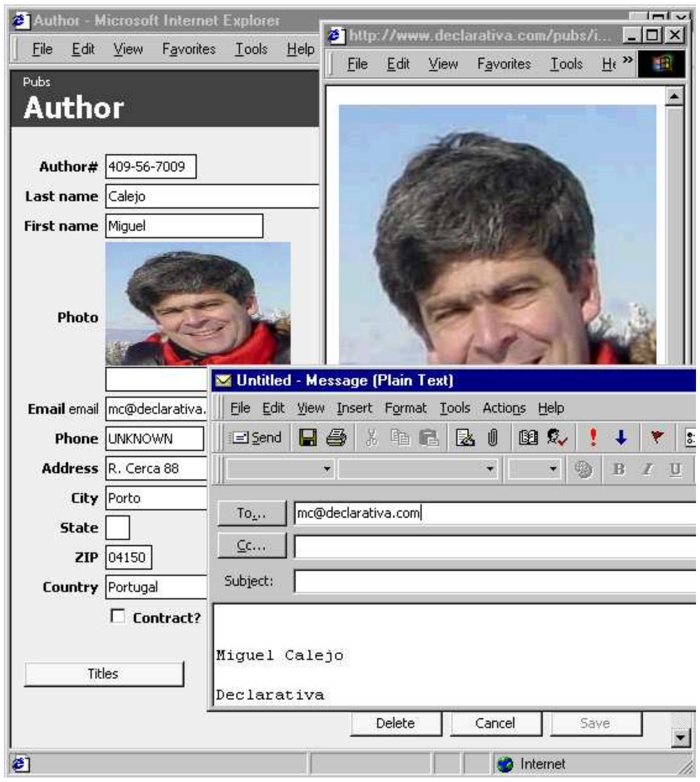

Author - Microsoft Internet Explorer

Pubs
Author

Author#	
Last name	
First name	
dbo.authors.photo	Imagem Não Disponível
dbo.authors.email email	
Phone	UNKNOWN
Address	
City	
State	
ZIP	
dbo.authors.country	USA
	☐ Contract?

We left the new columns (`photo, email, country`) with database name captions, to demonstrate how a minimal change to the application (database) reflects into extra functionality; defining better captions implies adding 3 records to `WAM_PRESENTATION`. *Imagem Não Disponível* is Portuguese for *image not available*; this GIF will be replaced by a language-neutral variant in a future WAM version.

The "Browse…" button picks a local file in the user's drive, uploads it to the server, and stores it in a server directory. After filling out the above page and reopening it from the list:


Author - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Author
http://www.declarativa.com/pubs/i...
Untitled - Message (Plain Text)
File Edit View Insert Format Tools Actions Help
Send
To...
Cc...
Subject:
Titles
Delete
Cancel
Save
Internet

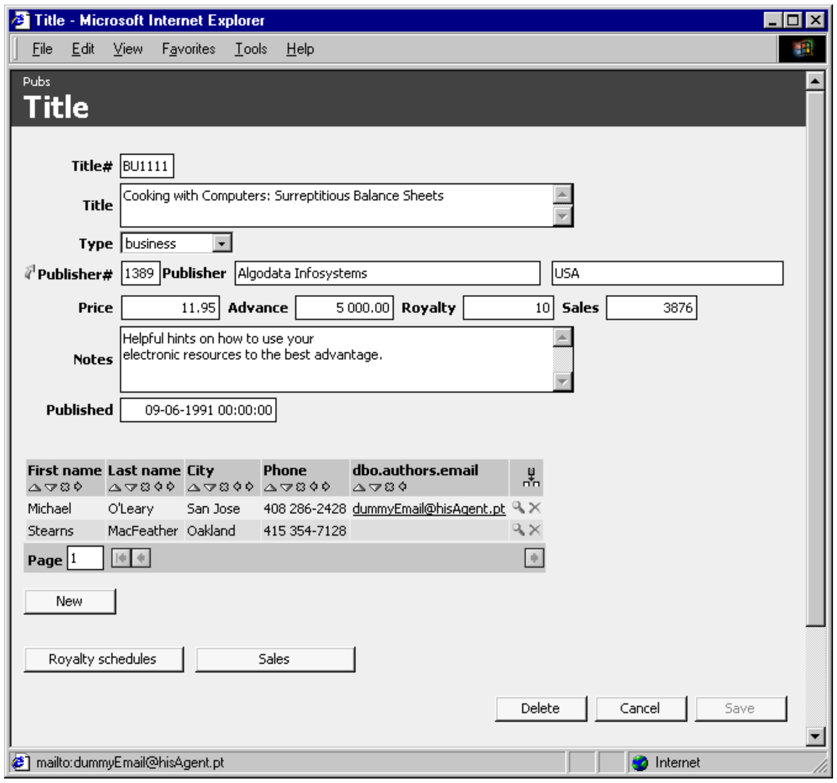
Author#	409-56-7009
Last name	Calejo
First name	Miguel
Photo	
Email email	mc@declarativa.
Phone	UNKNOWN
Address	R. Cerca 88
City	Porto
State	
ZIP	04150
Country	Portugal
Contract?	☐

To: mc@declarativa.com

```
Miguel Calejo

Declarativa
```

The user double-clicked the reduced image to see the image in original size, and clicked the email link to invoke his standard email application.

The `authors` change naturally reflects all over the application. Let's fill a couple of author emails, go back to a title record and add the email column to the embedded list. Here's the result:


Title - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Title

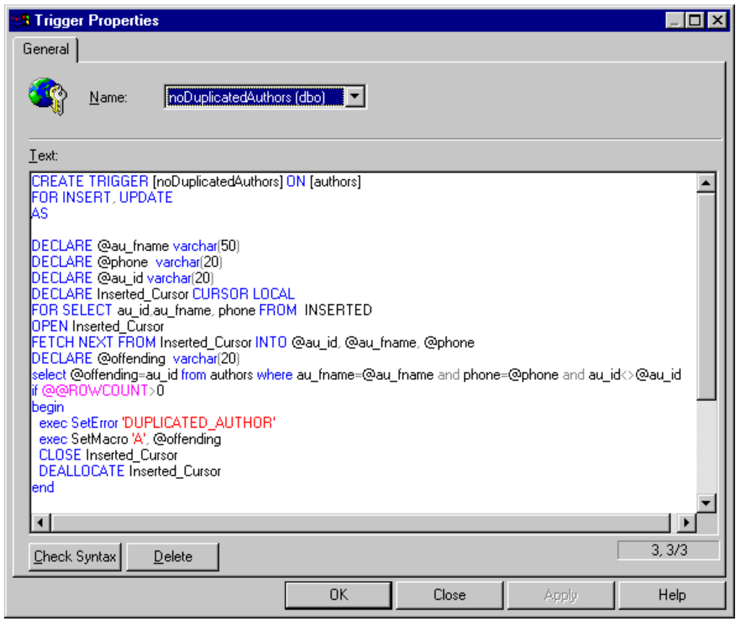
Title#	BU1111
Title	Cooking with Computers: Surreptitious Balance Sheets
Type	business
Publisher#	1389
Publisher	Algodata Infosystems
	USA
Price	11.95
Advance	5 000.00
Royalty	10
Sales	3876
Notes	Helpful hints on how to use your electronic resources to the best advantage.
Published	09-06-1991 00:00:00

First name	Last name	City	Phone	dbo.authors.email
Michael	O'Leary	San Jose	408 286-2428	dummyEmail@hisAgent.pt
Stearns	MacFeather	Oakland	415 354-7128	

Page 1

3.4.4 Add a trigger with parameterized error message

Let's suppose that 2 authors cannot have the same phone and first name, and that we wish to enforce this rule as a trigger:



Name: noDuplicatedAuthors (dbo)

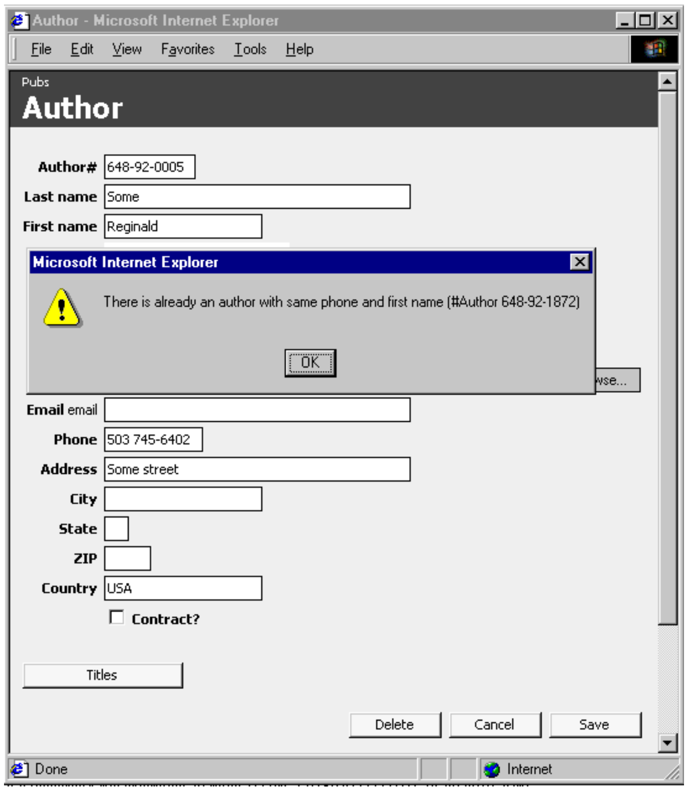
Text:

```
CREATE TRIGGER [noDuplicatedAuthors] ON [authors]
FOR INSERT, UPDATE
AS

DECLARE @au_fname varchar(50)
DECLARE @phone  varchar(20)
DECLARE @au_id varchar(20)
DECLARE Inserted_Cursor CURSOR LOCAL
FOR SELECT au_id,au_fname, phone FROM  INSERTED
OPEN Inserted_Cursor
FETCH NEXT FROM Inserted_Cursor INTO @au_id, @au_fname, @phone
DECLARE @offending  varchar(20)
select @offending=au_id from authors where au_fname=@au_fname and phone=@phone and au_id<>@au_id
if @@ROWCOUNT>0
begin
  exec SetError 'DUPLICATED_AUTHOR'
  exec SetMacro 'A', @offending
  CLOSE Inserted_Cursor
  DEALLOCATE Inserted_Cursor
end
```

WAM procedure SetError raises a SQL server error, referring a message in WAM_PRESENTATION; SetMacro passes a value that will be put in the named message placeholder. The error message can be defined (for English) with:

```
  INSERT INTO WAM_PRESENTATION
  ('en','ERROR','dbo','DUPLICATED_AUTHOR',
   'There is already an author with same phone and first name (#Author @A)', 0, '')
```

Here's an user attempt to insert an author violating the trigger:


Author - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Author

Author#	648-92-0005
Last name	Some
First name	Reginald

Microsoft Internet Explorer

There is already an author with same phone and first name (#Author 648-92-1872)

Email email	
Phone	503 745-6402
Address	Some street
City	
State	
ZIP	
Country	USA
	☐ Contract?

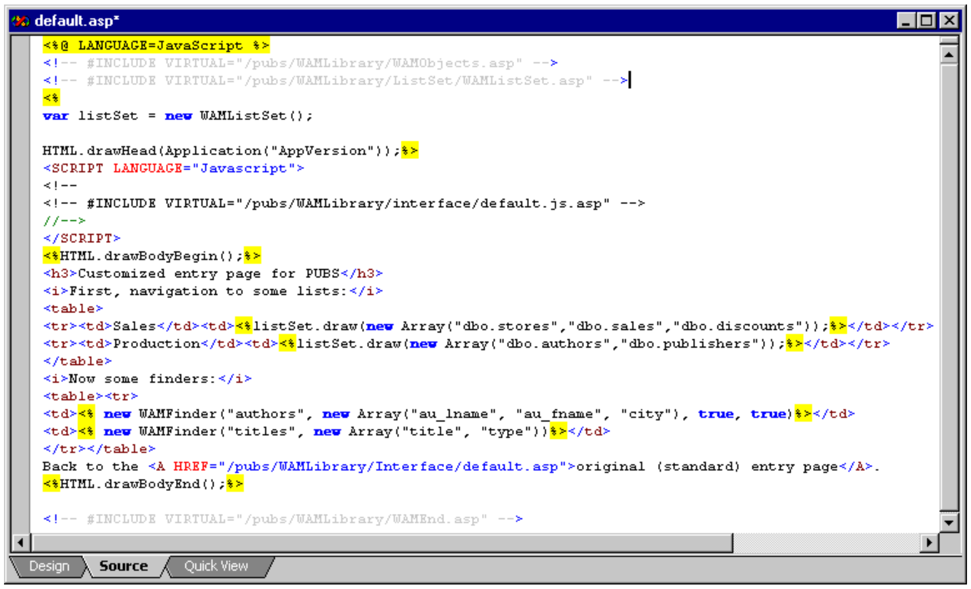
Titles
Delete
Cancel
Save
Done
Internet

3.5 ASP scripting

No matter how good a GUI generator may be, real applications tend to demand customized code… In general, coding customizations collides with declarative *model-driven (code-less) development*. WAM allows pure model-driven development through its GUI generator as seen above, but it also provides *model-supported customization*, by exposing its GUI generator API to the programmer.

3.5.1 Improve the entry page

The original entry page was produced by /WAMLibrary/Interface/default.asp, which provides a “default” entry page for any application. We'll now build another `default.asp` page, showing list buttons grouped by functional area, and with a couple of specific finders:



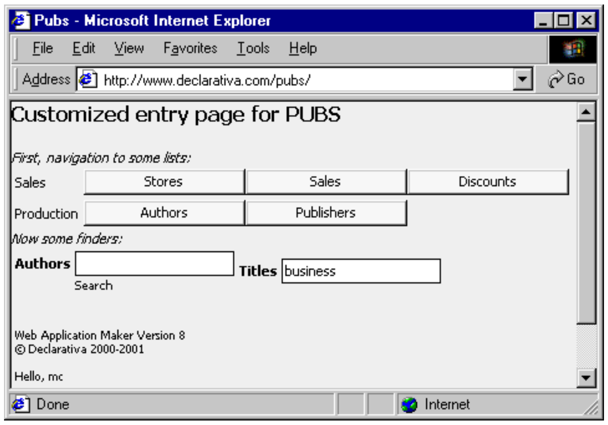
```
<%@ LANGUAGE=JavaScript %>
<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/WAMObjects.asp" -->
<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/ListSet/WAMListSet.asp" -->
<%
var listSet = new WAMListSet();

HTML.drawHead(Application("AppVersion"));%>
<SCRIPT LANGUAGE="Javascript">
<!--
<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/interface/default.js.asp" -->
//-->
</SCRIPT>
<%HTML.drawBodyBegin();%>
<h3>Customized entry page for PUBS</h3>
<i>First, navigation to some lists:</i>
<table>
<tr><td>Sales</td><td><%listSet.draw(new Array("dbo.stores","dbo.sales","dbo.discounts"));%></td></tr>
<tr><td>Production</td><td><%listSet.draw(new Array("dbo.authors","dbo.publishers"));%></td></tr>
</table>
<i>Now some finders:</i>
<table><tr>
<td><% new WAMFinder("authors", new Array("au_lname", "au_fname", "city"), true, true)%></td>
<td><% new WAMFinder("titles", new Array("title", "type"))%></td>
</tr></table>
Back to the <A HREF="/pubs/WAMLibrary/Interface/default.asp">original (standard) entry page</A>.
<%HTML.drawBodyEnd();%>

<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/WAMEnd.asp" -->
```

Note: another approach would be to add records to WAM_TABLE_GROUP (see section) and stick to WAM's default.asp

The `WAMListSet` object fetches enough information to produce buttons for all application lists. It's `draw` method has a variant accepting an array of list names, to make it easy constructing groups of related navigation buttons.

The `WAMFinder` object (which was indirectly invoked by `WAMFinders` in the original entry page) builds a simple navigation and search entry point for a table or view, optionally with a direct link to its list search filter page, and is provided with an array of columns to be searched in sequence – unless they are of different types, in which case the sequence is prunned at runtime depending on the type of the value (string, number) the user enters.

Here's the new entry page:


Pubs - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Address
http://www.declarativa.com/pubs/
Go
Done
Internet

Customized entry page for PUBS

First, navigation to some lists:

Sales	Stores	Sales	Discounts
Production	Authors	Publishers	

Now some finders:

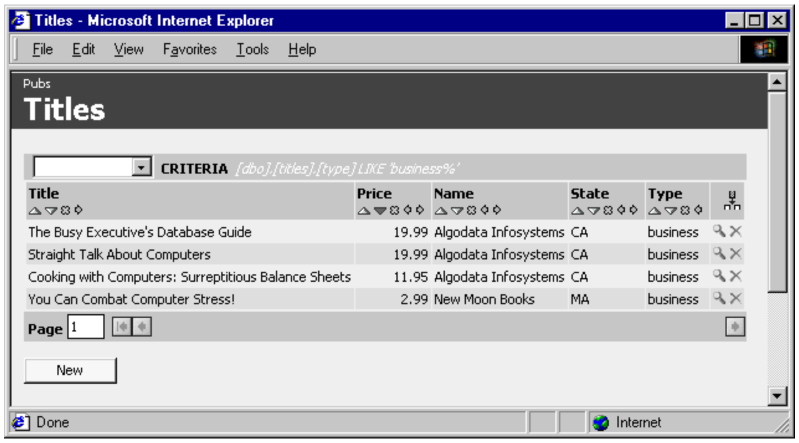
Authors [ ] Search **Titles** business

Web Application Maker Version 8
© Declarativa 2000-2001

Hello, mc

(The GUI finder labels and tips were specified by 2 additional records in WAM_PRESENTATION; an upcoming WAM version will build those automatically from existing row column information)

After the user hits enter on the titles finder field:


Titles - Microsoft Internet Explorer

Pubs
Titles

CRITERIA [dbo].[titles].[type] LIKE 'business%'

Title	Price	Name	State	Type
The Busy Executive's Database Guide	19.99	Algodata Infosystems	CA	business
Straight Talk About Computers	19.99	Algodata Infosystems	CA	business
Cooking with Computers: Surreptitious Balance Sheets	11.95	Algodata Infosystems	CA	business
You Can Combat Computer Stress!	2.99	New Moon Books	MA	business

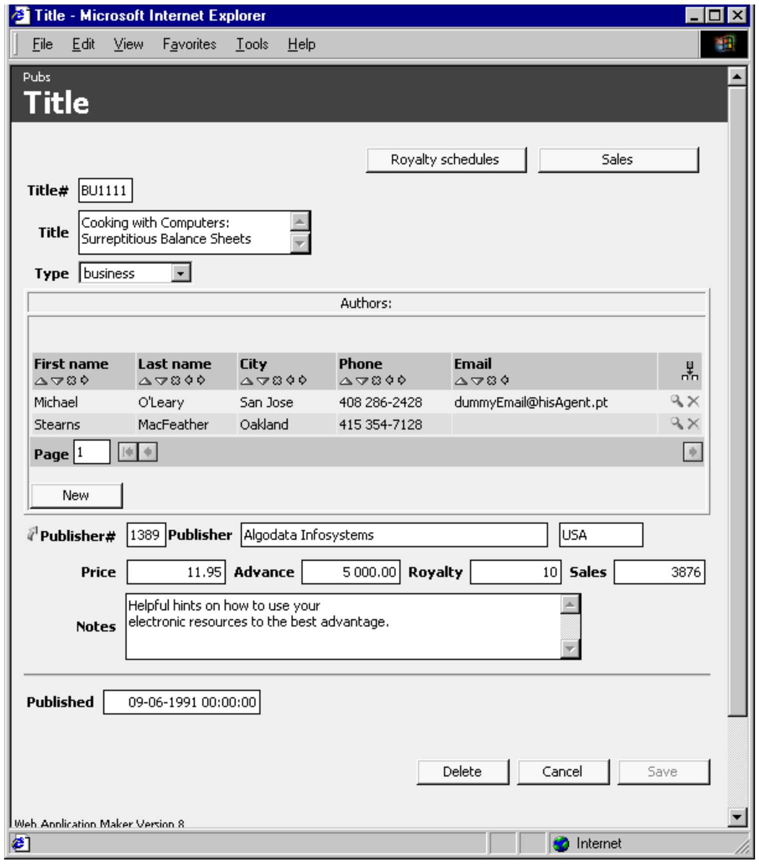
Page 1

New

Since no book has `title` "business" that search fails, and the next search (on `type`) took place, invoking the titles list with a specific search filter. If the finder field had instead a value determining a single record it would open its row page instead.

3.5.2 Customize a row page

Now we'll customize the row for titles, changing the layout so that the detail buttons appear at the top of the page, the authors embedded list is closer to the book title, both the title and (looked-up) publisher country fields are smaller, and the published date appears after an HTML separator:


Title - Microsoft Internet Explorer

Pubs
Title

Royalty schedules | Sales

Title# BU1111
Title Cooking with Computers: Surreptitious Balance Sheets
Type business

Authors:

First name	Last name	City	Phone	Email
Michael	O'Leary	San Jose	408 286-2428	dummyEmail@hisAgent.pt
Stearns	MacFeather	Oakland	415 354-7128	

Page 1

New

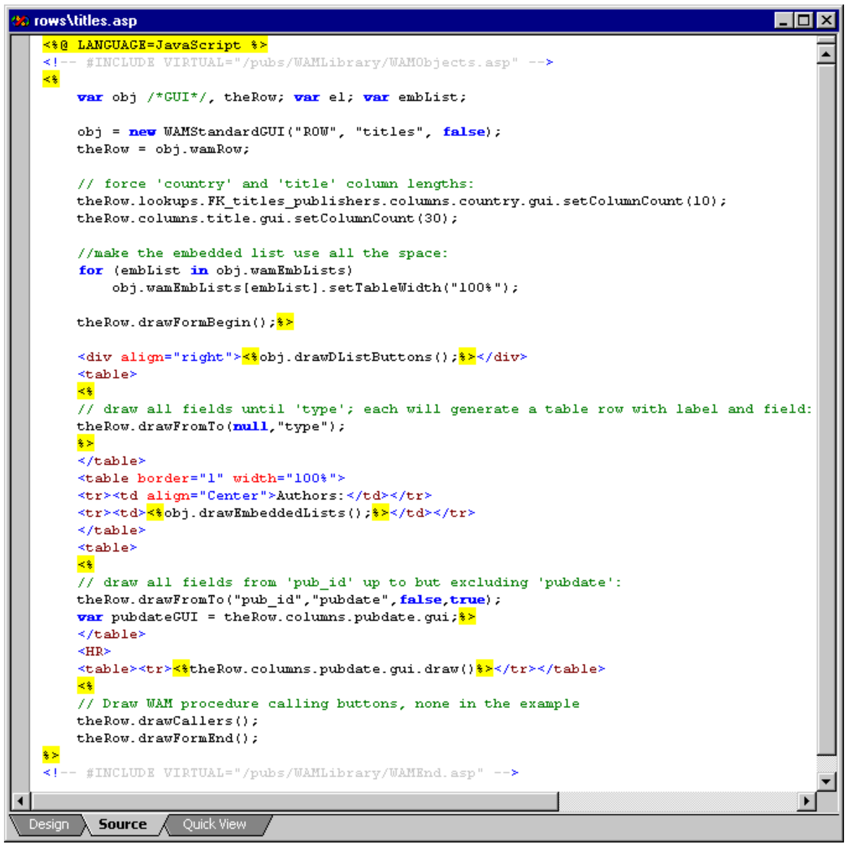
Publisher# 1389 Publisher Algodata Infosystems USA
Price 11.95 Advance 5 000.00 Royalty 10 Sales 3876
Notes Helpful hints on how to use your electronic resources to the best advantage.

Published 09-06-1991 00:00:00

Delete | Cancel | Save

Web Application Maker Version 8

For this we had to create the following Active Server Page in the `rows` directory (so far all lists and rows were displayed by WAM's `standard.asp`), named after the table/view it serves. Notice how the `WAMStandardGUI` object constructor initializes a GUI (row) generator at the beginning, and how generation is requested incrementally afterwards as we interleave customized HTML code with model-based specification:

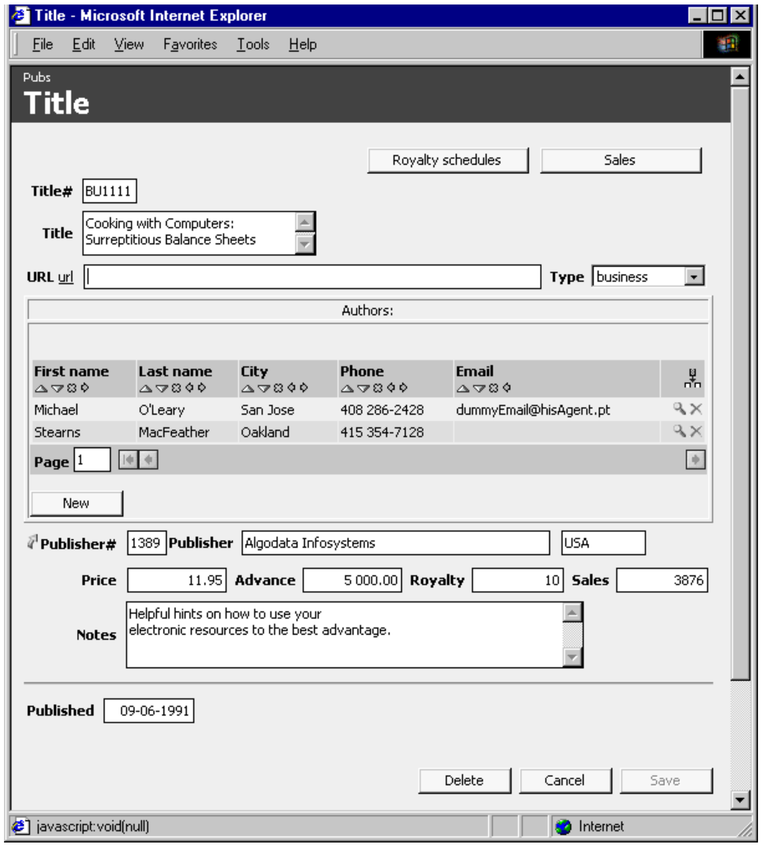

rows\titles.asp

```
<%@ LANGUAGE=JavaScript %>
<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/WAMObjects.asp" -->
<%
    var obj /*GUI*/, theRow; var el; var embList;

    obj = new WAMStandardGUI("ROW", "titles", false);
    theRow = obj.wamRow;

    // force 'country' and 'title' column lengths:
    theRow.lookups.FK_titles_publishers.columns.country.gui.setColumnCount(10);
    theRow.columns.title.gui.setColumnCount(30);

    //make the embedded list use all the space:
    for (embList in obj.wamEmbLists)
        obj.wamEmbLists[embList].setTableWidth("100%");

    theRow.drawFormBegin();%>

    <div align="right"><%obj.drawDListButtons();%></div>
    <table>
    <%
    // draw all fields until 'type'; each will generate a table row with label and field:
    theRow.drawFromTo(null,"type");
    %>
    </table>
    <table border="1" width="100%">
    <tr><td align="Center">Authors:</td></tr>
    <tr><td><%obj.drawEmbeddedLists();%></td></tr>
    </table>
    <table>
    <%
    // draw all fields from 'pub_id' up to but excluding 'pubdate':
    theRow.drawFromTo("pub_id","pubdate",false,true);
    var pubdateGUI = theRow.columns.pubdate.gui;%>
    </table>
    <HR>
    <table><tr><%theRow.columns.pubdate.gui.draw()%></tr></table>
    <%
    // Draw WAM procedure calling buttons, none in the example
    theRow.drawCallers();
    theRow.drawFormEnd();
%>
<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/WAMEnd.asp" -->
```

This ASP refers just the following database objects:

- The `titles` table (but not its columns), to generate a row
- The titles/publishers foreign key `FK_titles_publishers` and the column `publishers.country`, to change its lookup field size
- The columns `titles.type` and `titles.pub_id`, to define the position where the embedded list will appear
- The column `titles.pubdate`, so a HTML element could be added before it

The above ASP script has no more database dependencies! Other columns can be added or deleted from database tables, data types can change, and the row will still behave as intended, and reflect changes appropriately. For example, let's declare the `pubdate` to have the data type WAMDate so we abstract the time in the GUI, and add a URL field after the title, keeping together with the next field (`WAM_PRESENTATION.keep_with_next=1`). Here's the result:



Title - Microsoft Internet Explorer

Pubs

Title

Title# BU1111

Title: Cooking with Computers: Surreptitious Balance Sheets

URL url

Type: business

Authors:

First name	Last name	City	Phone	Email
Michael	O'Leary	San Jose	408 286-2428	dummyEmail@hisAgent.pt
Stearns	MacFeather	Oakland	415 354-7128	

Page 1

Publisher# 1389 Publisher Algodata Infosystems USA

Price 11.95 Advance 5 000.00 Royalty 10 Sales 3876

Notes: Helpful hints on how to use your electronic resources to the best advantage.

Published 09-06-1991

javascript:void(null)

3.5.3 *** Enhancing appearance with CSS

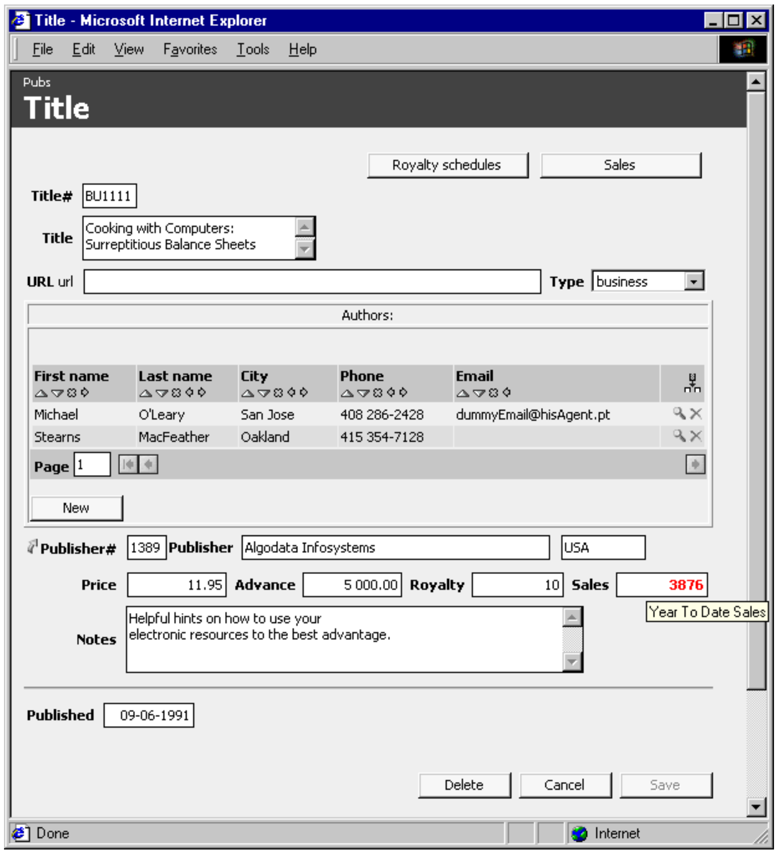
*** [IMAGEM PARA LISTA TITLES COM COLUNA NOTES]

By default, WAM does not wrap around the text of alphanumeric data type columns in lists. As you can see the above list is exceeding the window limits and causing some usability problems. To avoid this let’s add the following line of code to the customized CSS application page (cf. 5.3.1): `COL.cssdbotitlesnotes{width:auto;}`.

Other appearance details can be enhanced, just add to your CSS page the elements you want to change (cf. 5.6.2) and feel free to enhance the application to your needs.

3.5.4 Add a browser client script

Let's now see how browser scripts can be attached to WAM generated fields. Let's call the user's attention to the sales field whenever its value is below the advance field, suggesting something is wrong managing this book, by putting it in **red and bold**:



Title - Microsoft Internet Explorer

Pubs

Title

Royalty schedules | Sales

Title# BU1111

Title Cooking with Computers: Surreptitious Balance Sheets

URL url

Type business

Authors:

First name	Last name	City	Phone	Email
Michael	O'Leary	San Jose	408 286-2428	dummyEmail@hisAgent.pt
Stearns	MacFeather	Oakland	415 354-7128	

Page 1

New

Publisher# 1389 **Publisher** Algodata Infosystems USA

Price 11.95 **Advance** 5 000.00 **Royalty** 10 **Sales** 3876

Year To Date Sales

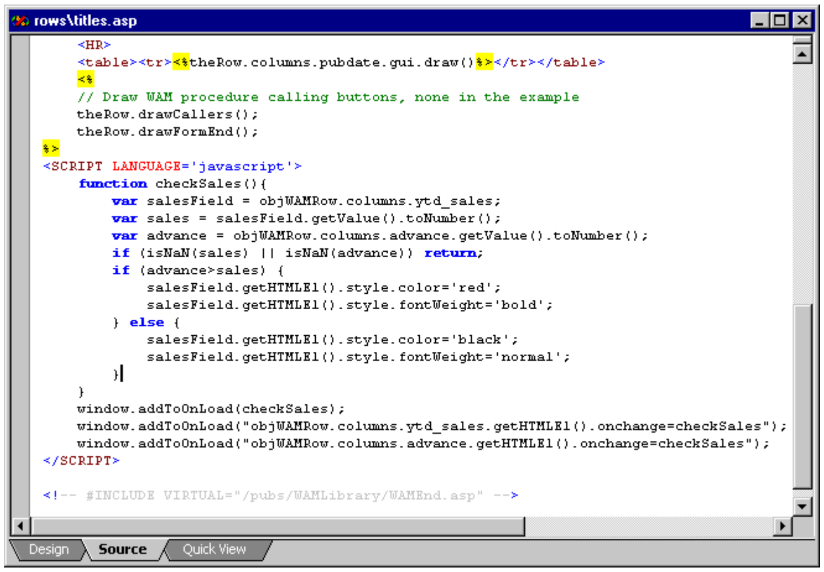
Notes Helpful hints on how to use your electronic resources to the best advantage.

Published 09-06-1991

Delete | Cancel | Save

Done | Internet

When the user changes either the sales or advance field the sales aspect changes. Here's the previous ASP, with a single JavaScript block added at the end:



rows\titles.asp

```
    <HR>
    <table><tr><%theRow.columns.pubdate.gui.draw()%></tr></table>
    <%
    // Draw WAM procedure calling buttons, none in the example
    theRow.drawCallers();
    theRow.drawFormEnd();
%>
<SCRIPT LANGUAGE='javascript'>
    function checkSales(){
        var salesField = objWAMRow.columns.ytd_sales;
        var sales = salesField.getValue().toNumber();
        var advance = objWAMRow.columns.advance.getValue().toNumber();
        if (isNaN(sales) || isNaN(advance)) return;
        if (advance>sales) {
            salesField.getHTMLEl().style.color='red';
            salesField.getHTMLEl().style.fontWeight='bold';
        } else {
            salesField.getHTMLEl().style.color='black';
            salesField.getHTMLEl().style.fontWeight='normal';
        }
    }
    window.addToOnLoad(checkSales);
    window.addToOnLoad("objWAMRow.columns.ytd_sales.getHTMLEl().onchange=checkSales");
    window.addToOnLoad("objWAMRow.columns.advance.getHTMLEl().onchange=checkSales");
</SCRIPT>

<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/WAMEnd.asp" -->
```

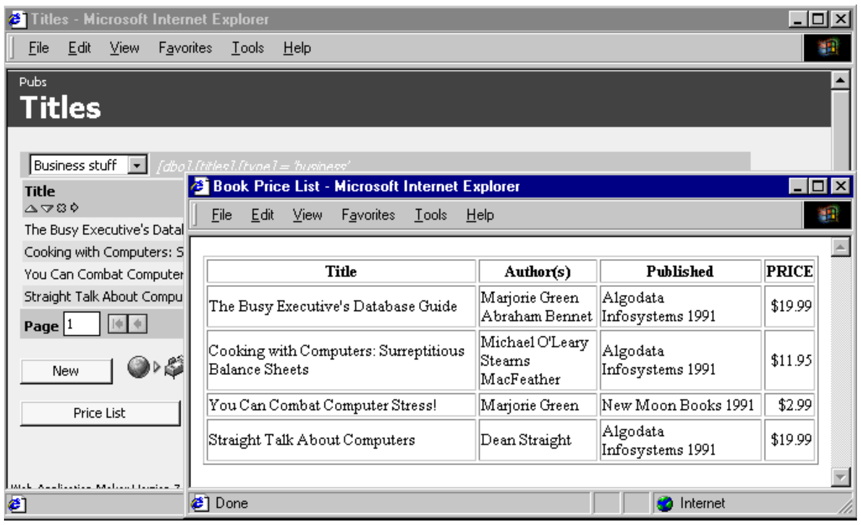
The `addToOnLoad` method stacks an instruction for execution after the page is loaded and WAM objects are setup. The first applies the condition when the row page opens, the other two make sure it is applied again when either field changes.

3.5.5 Add an external ASP

Let's now see how a WAM generated-page can invoke other ("hand-made") ASPs. Say a book price list is needed, to be invoked from the titles list: priceList.asp, a simple "printer-friendly" HTML table. Here are the necessary WAMmodel additions, assuming that the priceList.asp file will be placed in the standard procedures directory:

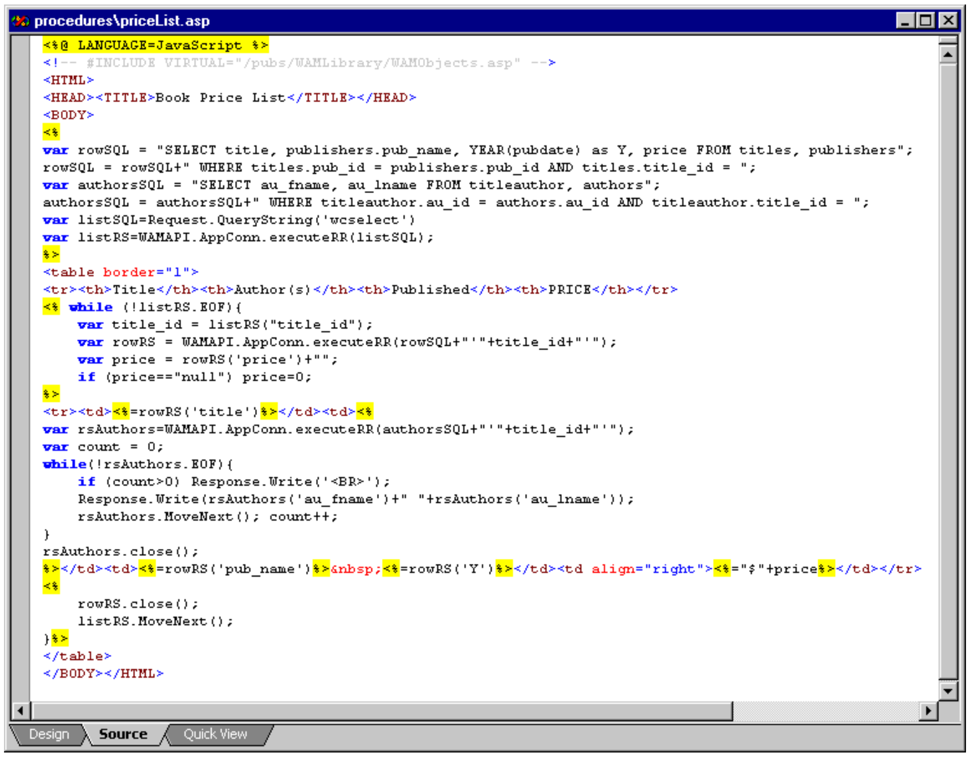
```
-- rather than a SQL stored procedure we're invoking an ASP, so procedure_schema=''
INSERT INTO WAM_PROCEDURE_CALL
VALUES('', 'priceList.asp', 'LIST', 'dbo', 'titles',0)
-- specify English aspect for the button
INSERT INTO WAM_PRESENTATION
VALUES('en','CALLER','priceList.asp','Price List','Show a price list',0,'')
```

Here's the new Titles list, after the user clicked the new button:


Titles - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Pubs
Titles
Business stuff
[dbo].[titles].[type] = 'business'
Title
The Busy Executive's Datal
Cooking with Computers: S
You Can Combat Computer
Straight Talk About Compu
Page 1
New
Price List
Book Price List - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Done
Internet

Title	Author(s)	Published	PRICE
The Busy Executive's Database Guide	Marjorie Green Abraham Bennet	Algodata Infosystems 1991	$19.99
Cooking with Computers: Surreptitious Balance Sheets	Michael O'Leary Stearns MacFeather	Algodata Infosystems 1991	$11.95
You Can Combat Computer Stress!	Marjorie Green	New Moon Books 1991	$2.99
Straight Talk About Computers	Dean Straight	Algodata Infosystems 1991	$19.99

Here's the full ASP:



```
<%@ LANGUAGE=JavaScript %>
<!-- #INCLUDE VIRTUAL="/pubs/WAMLibrary/WAMObjects.asp" -->
<HTML>
<HEAD><TITLE>Book Price List</TITLE></HEAD>
<BODY>
<%
var rowSQL = "SELECT title, publishers.pub_name, YEAR(pubdate) as Y, price FROM titles, publishers";
rowSQL = rowSQL+" WHERE titles.pub_id = publishers.pub_id AND titles.title_id = ";
var authorsSQL = "SELECT au_fname, au_lname FROM titleauthor, authors";
authorsSQL = authorsSQL+" WHERE titleauthor.au_id = authors.au_id AND titleauthor.title_id = ";
var listSQL=Request.QueryString('wcselect')
var listRS=WAMAPI.AppConn.executeRR(listSQL);
%>
<table border="1">
<tr><th>Title</th><th>Author(s)</th><th>Published</th><th>PRICE</th></tr>
<% while (!listRS.EOF){
    var title_id = listRS("title_id");
    var rowRS = WAMAPI.AppConn.executeRR(rowSQL+"'"+title_id+"'");
    var price = rowRS('price')+"";
    if (price=="null") price=0;
%>
<tr><td><%=rowRS('title')%></td><td><%
var rsAuthors=WAMAPI.AppConn.executeRR(authorsSQL+"'"+title_id+"'");
var count = 0;
while(!rsAuthors.EOF){
    if (count>0) Response.Write('<BR>');
    Response.Write(rsAuthors('au_fname')+" "+rsAuthors('au_lname'));
    rsAuthors.MoveNext(); count++;
}
rsAuthors.close();
%></td><td><%=rowRS('pub_name')%>&nbsp;<%=rowRS('Y')%></td><td align="right"><%="$"+price%></td></tr>
<%
    rowRS.close();
    listRS.MoveNext();
}%>
</table>
</BODY></HTML>
```

ASPs should perform their function only on the selected data submitted by the list, as specified by its search filter, hence WAM passes the current list SQL statement in the QueryString's `wcselect` variable. However since lists have unknown runtime-defined joins per user, in general the ASP developer should assume that the SQL statement provides only the primary key values for the list base table, and get the additional columns by executing additional SELECTs.

Were the Titles list *rigid* for an user (meaning, if the user would be unable to add/remove columns) priceList.asp might avoid the need for the first SELECT statement, by assuming certain columns to be always present in the list SQL statement. Rigid lists for a group of users can be realized with a trivial permissions policy regarding WAM_LIST_COLUMN, a WAMmodel table.

3.5.6 Tour end - architecture overview

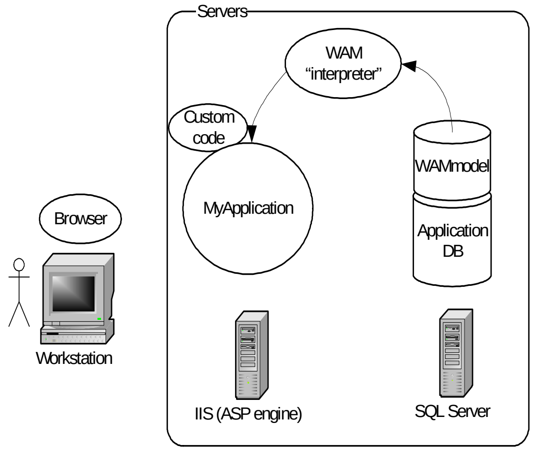
You might want to recap section 2.8 above. Bringing back WAM's interface equation:

Web interface = wam(DB Meta Information + WAMmodel) + Custom pages using wam objects + Other pages

As illustrated in the tour typical, a WAM application project will typically start from its database alone, without any (developer written) web page, relying on the WAMLibrary's generic pages (namely default.asp and standard.asp, cf. section 5.1) to implement full data navigation and editing.

When WAMmodel improvements aren't enough, custom rows and custom lists will probably be added (cf. section 5.2): ASPs using WAM objects. Along the way, the database design will probably be improved too.

The next picture, a more infrastructural counterpart to the diagrams in section 2.8, depicts the main system components involved:

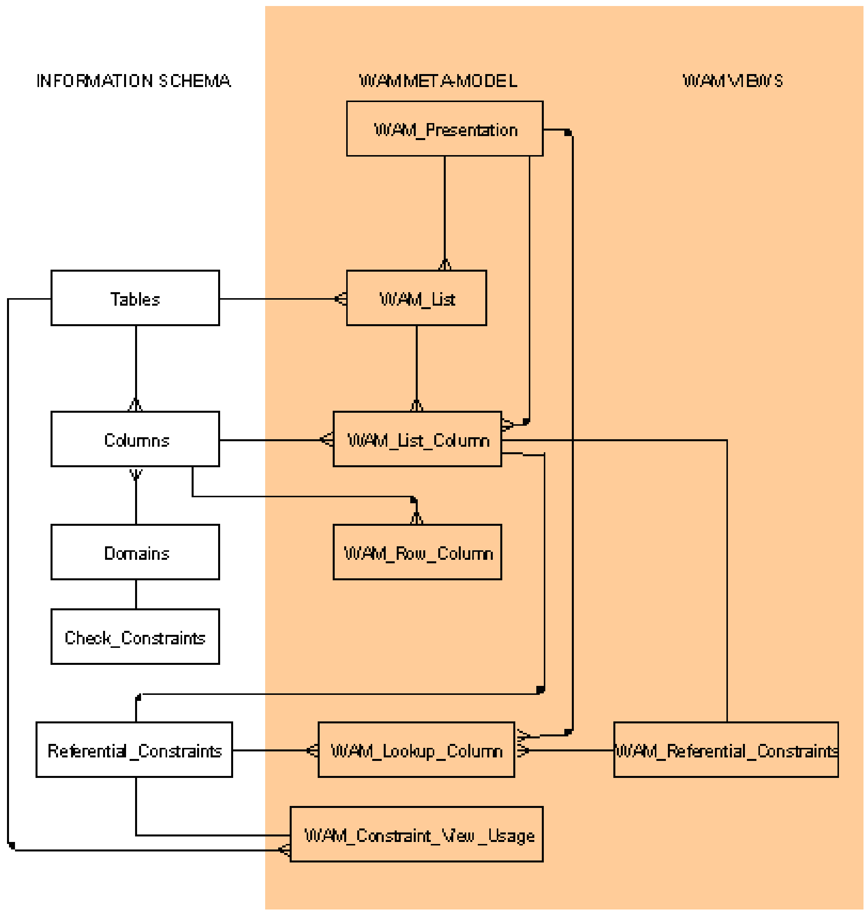


We'll next approach WAM from two perspectives: WAMmodel first, web layers afterwards.

4 Database server layer: meta information and WAMmodel

WAM is based on a set of VIEWs accessing the SQL engine intrinsic meta information for the application database and associated native objects (stored procedures etc.), plus a set of WAM tables with additional information – the "WAMmodel". The application may use more than one database schema.

Here's a simplified diagram (some WAMmodel tables omitted):



The 5 views on the left reflect the SQL Server intrinsic meta-information, the rest are WAM tables that must be at least partially filled out for the particular application.

The integrity of the relationships shown is only partially enforced with triggers, due to the impossibility of having the database engine enforce relationships to its meta (INFORMATION_SCHEMA) objects. A consistency checking procedure, including integrated inconsistency fixing, is available through WAMAdmin, cf. section 7.5.

4.1 WAMmodel: the WAM tables

***Missing: `exec dbo.WAMFillWAMDataSize`

Together with database meta information, the WAMmodel fuels WAM. The WAMmodel design pursued minimalism and no redundancy vs. built-in database engine meta information. We fought hard to keep the WAMmodel small, but real and reasonable requirements from customer projects determined its current composition:

WAMmodel table	**Why we could not avoid it**
WAM_PRESENTATION	Database engines do not provide a place for multi-lingual strings attached to database objects. Without records here WAM still functions, but with much less power.
WAM_ROW_COLUMN	Although the database knows its table columns, sometimes it's necessary to state something as hidden, not persistent, etc. – otherwise no records need to go in here This table is usually filled for less than 5% of the data elements in an application.
WAM_ROW_GROUP	Some applications have tables with many fields and detail tables, which must be structured for the user
WAM_LOOKUP_COLUMN	Although foreign keys determine the basic mechanics of looking up values in related tables, it is necessary to specify what columns are relevant
WAM_LIST	Need a place to store current search criterion/filter for each user, and a couple other list attributes
WAM_LIST_COLUMN	Need to (let the user indirectly) specify the SELECT behind the list
WAM_CRITERION	Need to persist list search criteria/filters
WAM_PROCEDURE_CALL	The database and the file system already know what are the procedures, but we need to specify they're actually callable from somewhere
WAM_CONSTRAINT_VIEW_USAGE	Foreign keys and foreign key paths are so useful for interface generation that we couldn't live without simulating them among VIEWs too; and every dba loves the flexibility and control of VIEWs
WAM_TABLE_GROUP	Sometimes users get confused by too many choices, this provides (optional) structuring of tables into groups or "application areas"
WAM_TREE	Not all reflexive relations in tables are to be displayed as trees; a record here specifies a tree.
WAM_HELP	Provides higher level abstractions, with more structure to ease learning and to audit users
WAM_PREFERENCES	Project after project, application users demand distinct entry pages
WAM_FINDER	Finders are too important to require recurrent trivial scripting

Each table has only juicy, no redundant information; WAM introspects other items from the database meta information once, and caches it automatically for efficiency.

Following are detailed descriptions for the WAMmodel tables. Primary key columns are prefixed with "#". In the column descriptions the following meta constructs will be used: A|B denotes A or B, [C] means optional C and regular parentheses are to be interpreted as meta language.

WAMmodel tables are editable with any SQL client, including the WAM application itself through edition links activated by WAMAdmin (see 7).

4.1.1 WAM_PRESENTATION

This is the most important WAMmodel table, as it contains all user-visible strings and also embeds the default navigational structure (see "Default Navigational Structure").

Column name	***Description***
#language	Language code (cf. for example http://www.oasis-open.org/cover/iso639a.html)
#type	Type of GUI object (see below)
#schema_name	Database schema, or '' – empty string – for external objects, such as CALLERs to custom ASPs
#name	Name of database object or external procedure (see below)
Caption	String to be used as title for row and lookup field, list column, row, list, procedure call button, etc
Tip	ToolTip help text for the GUI object
keep_with_next	Bit indicating whether this GUI object should (1) or not (0) stay in the same line as the next one to be laid out. Default is no. Inspired in a Microsoft Word paragraph attribute, this bit allows some "visual grouping" of consecutive objects.
Comment	Slot for programmer comments

The name and type WAM_PRESENTATION columns can have the following related values:

Type	***Name***
LIST	table_name | view_name | FK_name | FK_name1,FK_schema2.FK_name2, ... FK_schemaN.FK_nameN
LIST_COLUMN	LIST_Name+schema_name.table_name.column_name[+required_join]
ROW	table_name | view_name
ROW_COLUMN	ROW_Name.column_name
LOOKUP_COLUMN	(LIST_Name | ROW_Name)+FK_name+schema_name.table_name.column_name[+required_join]
CALLER	SQL Stored Procedure name | ASP path note: ASPs for callers must be placed in a "procedures" directory, under the root directory for the application
CALLER_SUCCESS	Message to show at the end of a Stored Procedure (SP) execution, started by a caller action, when it finishes without errors; if this record is absent no message is shown. This message can contain placeholders referring SP output variable names, that will be replaced with their correspondent values
ERROR	Error message name; caption will contain the error string, which can have macro placeholders (see Error Handling)
CUSTOM	Convenience for user-defined presentations outside WAM's realm, for example to decorate buttons with nonstandard functions
GUIELEMENT_FIND	WAM standard strings for finders
GUIELEMENT_LIST, MESSAGE_LIST	WAM standard strings for lists
CONSTRAINT, GUIELEMENT_CRIT	WAM standard strings for criteria (search filters)
GUIELEMENT_ROW, MESSAGE_ROW	WAM standard strings for rows
MESSAGE_CALLER	WAM standard strings for callers
CHECK_CONSTRAINT	TableName.ColumnName+ConstraintValue

4.1.1.1 Multi-lingual support

Supporting of multiple user languages is facilitated by the architecture, which centralizes all user-visible strings in the WAMmodel's WAM_PRESENTATION table, whose primary key includes a `language` column.

All GUI fragments and error messages are generated for a language, according to the preferred browser language; if this is unavailable the WAMLibrary will search WAM_PRESENTATION for the default language (defined at application startup).

4.1.2 WAM_ROW_COLUMN

Optional properties for a column in a row. Although the column value will always be present in the GUI's invisible object data model, if the column is mentioned in WAM_ROW_COLUMN it may have different behavior. Omitting a column in WAM_ROW_COLUMN assigns it default behavior.

Column name	***Description***
#table_schema	Row owner
#table_name	Row name: table name or view name
#column_name	Table column name

Enabled	Bit indicating whether users can (1) or can not (0) edit the column value in the GUI; default is yes. Disabling may be useful to display computed values.
Visible	Bit indicating whether the column will (1) or will not (0) be visible in the GUI; default is yes. Making a column invisible may be useful to hide a foreign key feeding a lookup containing (more meaningful) looked-up fields
persistent	Bit indicating whether the column value will be included in the INSERT/UPDATE statements; default is yes. Disabling persistence may be useful for columns computed by database triggers.

4.1.3 WAM_LOOKUP_COLUMN

A row for a table or view can contain column fields belonging to other tables or views, as long as there are FK paths to them. The same applies for standalone detail lists, which may require the display of master-related fields in other tables or views. Those columns are the lookup columns, and each one is specified by a record in this table:

Column name	***Description***
#table_schema	Table owner or FK constraint owner
#table_name	Row name or Detail List name: table_name or view_name or FK_name or FK_path
#constraint_schema	FK constraint owner
#constraint_name	Name of FK constraint at the beginning of the FK path
#column_name	Name of the looked-up column
#required_join	If the FK path to the looked-up column has just one FK (the column is in the master table), this field is left empty; otherwise it will contain the FK path
column_order	Relative position of the column among looked-up columns
Enabled	Same meaning as for WAM_ROW_FIELD
Visible	Same meaning as for WAM_ROW_FIELD

There are several standard navigational and data-entry mechanisms associated to lookups. For more details see the WAMLookup object.

}

4.1.4 WAM_LIST

This table must be populated with a record for each list and detail list for user 'dbo'. The first time another user attempts to visualize a list its record is cloned and added to WAM_LIST; this is so each user has his/her own current search filter applied to the list.

A list is based in a single table or view. A detail list is based in a foreign key constraint or FK path (FK1,scheme2.FK2,…, where FK1

Column name	***Description***
#user_id	User name for this preferences
#table_schema	Table owner or View owner or FK owner
#table_name	Table name or View name or FK name or FK path
edit_table_schema	Owner for edit table
edit_table_name	Table or view whose row will be used to edit
show_criterion	Bit indicating whether the criterion (search filter) display and navigation will (1) or not(0) be included in the list; default is yes
current_criterion	Name, given by the user, of the current search filter applied to the list
distinct_row_set	Bit indicating if the DISTINCT option will (1) or not (0) be included in the SELECT statement.
auto_refresh	Bit indicating if the list will (1) or not (0) be refreshed automatically after column structure, search filter or (associated row) data changes. If 0 the list will appear with a “Show Data” button so the user can explicitly trigger the refresh.
inline_editing	If true this list (for this user) will allow inline editing of base table columns, except for foreign keys. Notice that when the user exits a list cell this persists immediately and there's no undo mechanism, so this should be reserved for non naïve users.
allow_aggregates	If true, list columns will be able to specify an aggregate function

4.1.5 WAM_LIST_COLUMN

Columns to show in a list/detail list. *This table is entirely maintained by the user through the standard list controls.*

A column can belong to the list's base table/view, or it can be an external column. The table where the column belongs is identified by table_column_schema + table_column_name. For external columns with ambiguous retrieval paths we must specify, in requiredJoin, the foreign key name or foreign key path to use when looking up the column.

Column name	***Description***
#user_id	User name for this preferences
#table_list_schema	List owner
#table_list_name	List name, cf. WAM_LIST
#table_column_schema	Owner of the table containing the column
#table_column_name	Name of table or view containing the column
#column_name	Name of the column
#requiredJoin	Usually this will be filled with an empty string. Whenever there's more than one FK PATH between table_list_name and table_column_name or table_column_name is not directly related to table_list_name this must be filled out with the full FK PATH to use.
column_width	Column width in pixels
column_order	Relative order of column in list
column_sort	asc: sort data ascending desc: sort data descending nulll: no sort for the column
aggregate_function	Can specify a single aggregate function ('AVG', 'COUNT', 'MAX', 'MIN' or 'SUM'), whose result will appear below the column values. The value will be for all values in the record set, not just for those shown in the current page.

4.1.6 WAM_CRITERION

Criteria (search filter) filtering data shown in lists. *This table is entirely maintained by each user through the standard list and filter controls*.

Column name	***Description***
#user_id	User owning this search filter
#table_schema	Table owner
#table_name	List name, cf. WAM_LIST
#criterion_description	Name the user has given to identify the search filter
Sql_condition	Condition for the search filter, in the form of an SQL WHERE clause. Search conditions in the WHERE clause are linked with conjunctions (AND).
huge_set	IN condition for SQL WHERE clause to apply to a column when the user wants to restrict the column value to a huge set of possible values. It is used to allow easy specification of a list of values by pasting it from somewhere, typically a file.

4.1.7 WAM_PROCEDURE_CALL

Specify buttons to call SQL stored procedures or to invoke custom ASPs. The "caller" is the GUI entity where the button will be placed. Its aspect will be determined by a CALLER entry in the WAM_PRESENTATION table.

Column name	***Description***
#procedure_schema	Owner for the SQL stored procedure, when caller executes one, or empty '', when it invokes an ASP
#procedure_name	SQL stored procedure name or relative URL of custom ASP
#caller_type	Type of caller (cf. WAM_PRESENTATION). Can be ROW or LIST.
#caller_schema	Owner
#caller_name	Name of caller (cf. WAM_PRESENTATION)
show_confirm	Bit indicating whether a confirmation dialog should be shown before the procedure is called The text for confirm message that appears in dialog is defined in table WAM_PRESENTATION, column caption for type = 'MESSAGE_CALLER', schema_name = 'dbo', name='confirm' and can be defined for several languages
show_warning_at	Maximum number of records that a list can have without forcing a confirmation dialog too appear when this procedure is invoked. Specifying a value for this field prevents unexpected long waits by the user. NULL means do not show this dialog (although show_confirm may cause a dialog to appear)

See WAMCaller object for more details.

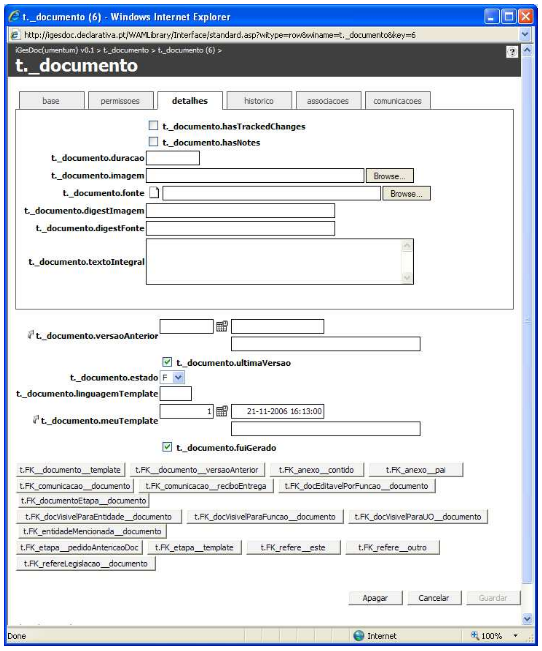
4.1.8 WAM_ROW_GROUP

This allows interface grouping of row elements (fields – including lookups, detail lists and callers) into tabbed panes.

Group names can have captions in WAM_PRESENTATION.

Column name	***Description***
#TABLE_SCHEMA	Table or View schema
#TABLE_NAME	Table or View name
#GROUP_NAME	Name of group in row
#ELEMENT_NAME	Name of column, caller or detail list
ELEMENT_TYPE	COLUMN, LIST or CALLER

The following row has 6 groups defined, with some elements outside groups:


t._documento (6) - Windows Internet Explorer
http://igesdoc.declarativa.pt/WAMLibrary/Interface/standard.asp?witype=row&winame=t._documento&key=6
iGesDoc(umentum) v0.1 > t._documento > t._documento (6) >
t._documento
base
permissoes
detalhes
historico
associacoes
comunicacoes
Browse...
Browse...
t.FK__documento__template
t.FK__documento__versaoAnterior
t.FK_anexo__contido
t.FK_anexo__pai
t.FK_comunicacao__documento
t.FK_comunicacao__reciboEntrega
t.FK_docEditavelPorFuncao__documento
t.FK_documentoEtapa__documento
t.FK_docVisivelParaEntidade__documento
t.FK_docVisivelParaFuncao__documento
t.FK_docVisivelParaUO__documento
t.FK_entidadeMencionada__documento
t.FK_etapa__pedidoAntencaoDoc
t.FK_etapa__template
t.FK_refere__este
t.FK_refere__outro
t.FK_refereLegislacao__documento
Apagar
Cancelar
Guardar
Done
Internet
100%

☐ t._documento.hasTrackedChanges
☐ t._documento.hasNotes
t._documento.duracao
t._documento.imagem
t._documento.fonte
t._documento.digestImagem
t._documento.digestFonte
t._documento.textoIntegral
t._documento.versaoAnterior
✓ t._documento.ultimaVersao
t._documento.estado F
t._documento.linguagemTemplate
t._documento.meuTemplate 1 21-11-2006 16:13:00
✓ t._documento.fuiGerado

An element can not belong to more than one group. Elements without group (not belonging to any) appear outside tabbed panes. If the first column has no group, all elements without group will appear above the tabs, otherwise below.

4.1.9 WAM_TABLE_GROUP

This specifies table/view grouping in the interface. It identifies the groups for the application and enumerates all tables that belong to each group. Group names can have captions defined in WAM_PRESENTATION.

A table may belong to multiple groups. Groups have direct impact in the interface: items in list popup menus are organized in submenus, an item for each group is created in the menu's first level, and each group item opens a submenu with tables contained in correspondent group. In addition, the Interface\default.asp page (see below) will group list buttons.

Column name	***Description***
#TABLE_GROUP	Name given by the user to identify group
#TABLE_SCHEMA	Table or View schema
#TABLE_NAME	Table or View name

4.1.10 WAM_TREE

This table specifies hierarchical interfaces to binary relations. Each record represents a table that defines a tree structure through a binary relation between tree nodes. This information is used to support hierarchical viewing and querying of trees. Any table may represent a tree, as long as it obeys the following conditions:
- A tree node is represented by a table tuple
- Its primary key is a single string field, and contains the values to be displayed in the tree
- It has an alternate key "node_key", an integer
- It has a field referring the parent node key, "parent", an integer
- It has a field with the minimum of all node keys under the node, "left", an integer
- It has a field with the maximum of all node keys under the node, "right", an integer
- It may or not have other fields

The four integer fields are independent of the tree node values, and can be computed from a simpler <node,parent> relation table. WAM provides a sample SQL stored procedure ("fillSubClass.sql") that creates and fills a particular tree table for the <node,parent> relation table, and which can be adapted.

Column name	***Description***
#TABLE_SCHEMA	Schema of the tree table
#TABLE_NAME	Name of the tree table
NODE_COLUMN_NAME	Name of the "node_key" column
LEFT_COLUMN_NAME	Name of the "left" column
RIGHT_COLUMN_NAME	Name of the "right" column
PARENT_COLUMN_NAME	Name of the "parent" column

Notice that there's no indication of which is the field to display in the tree nodes, because it must be the primary key of the tree table, so WAM simply introspects it.
See WAMTree object in 5.4.9.5

4.1.11 WAM_CONSTRAINT_VIEW_USAGE

This table was created to complement the basic database engine ability to represent relationships, by allowing us to represent relations between views or a table and a view, instead of only relations between tables. This allows the use of "simple" VIEWS (see "Support for VIEWS") as first class citizens in what concerns rows, lists, lookups, etc. Neverthless a FK must exist between the view base tables.

Column name	***Description***
#WAM_CONSTRAINT_SCHEMA	FK wam constraint owner
#WAM_CONSTRAINT_NAME	Name given by the user to complementary relation
CONSTRAINT_SCHEMA	FK constraint schema; typically same as DETAIL_TABLE_SCHEMA
CONSTRAINT_NAME	Name of FK constraint that will serve from support to the new relation evolving views
DETAIL_TABLE_SCHEMA	Owner of detail table or detail view
DETAIL_TABLE_NAME	Name of detail table name or detail view name in the relation
MASTER_TABLE_SCHEMA	Owner of master table or master view
MASTER_TABLE_NAME	Name of master table name or master view name in the relation

Notice that this table is to be filled only with records for FKs involving at least one view; a FK between two tables is simply stated to the database engine.

4.1.12 WAM_HELP

This table contains information to generate online help:

Column name	Description
#language	
#type	
#name	
title	

description	
table_group	
app_status	Is this feature already developed?
help_status	Is this feature already documented?
precedents	
usage_sql	

See 9.5.2 for more information.

4.1.13 WAM_PREFERENCES

Each record in this table contains the preferences of an individual user:

Column name	Description
#user_id	User name
interface_default	XML document with entry page preferences

See 5.1.1.1 for more details.

4.1.14 WAM_FINDER

Each record in this table specifies a finder field object, typically present in the application entry page:

Column name	Description
#table_schema	
#table_name	
#finder_columns	The columns to be considered when searching (separated by commas)

See 5.4.9.3 for more details.

4.2 *Treating VIEWS (almost) as tables*

Although WAM treats tables and views similarly, in the current version only "simple VIEWS" are supported in what concerns the GUI: VIEWS defined by a projection over a single table, or over a table join, in which case the first table enumerated in its FROM clause is considered as the VIEW base table. Furthermore, the base table primary key must be included in the VIEW columns.

Views can be created to specify subsets for a table, restricted by one or more column values. Those columns act as filters and their possible values must be explicitly declared in the WHERE clause respecting one of the following formats: WHERE [table_name].[column_name] = column_value1 OR [table_name].[column_name]=column_value2 … AND …; or WHERE [table_name].[column_name] IN (column_value1, column_value2,…) AND ….

If no WAM_PRESENTATION information exists for a VIEW but it exists for its base table, it is used.

In addition to the above, VIEWs can have "foreign keys" defined between them, thus extending the relational GUI generation capabilities for tables schema to view schema as well (namely lookups, detail lists and user-customizable joins in lists); since this is not supported by the underlying SQL engine, it requires explicit support in the WAMmodel, see section 4.1.11.

4.2.1 VIEW updating

Given VIEW S.V in schema S, if there are stored procedures S.wam_update_V and S.wam_delete_V, they'll be used to persist VIEW "records"; the stored procedures are responsible for implementing any update/delete cascades explicitly. As a matter of fact, this also applies to tables.

4.3 SQL and system permissions and how they are used

WAM introspects the standard SQL and operating system permissions and generates the GUI accordingly. This means that buttons, links or menu options, which are known to cause a security violation, are hidden; so are tables and fields. The same principle is applied to those WAMmodel tables that can be updateable by the user, as when configuring a list.

So the WAM GUI effectively adapts to the underlying permissions for the current user:

Situation	**What WAM objects do**
user belongs to wam_criterionreader role	Show list filters' combo box, allowing user to apply existing search filters
user belongs to wam_criterionwriter	Add two options to the list filters' combo box: New Filter; Edit Current; effectively allows user to create search filters
user belongs to wam_listcolchange	Show four buttons in each column heading: sort in ascending order; sort in descending order; move column to left; move column to right;
wam_listcolwriter	Show two buttons: remove column from list; add columns to list; effectively allows the user to define joins
column without SELECT permission	Do not show the column in lists, rows, search filters and list's menu
table without SELECT permission	Do not show access to table (lists, rows, search filters, detail lists' buttons, lists' menu)
table without INSERT permission	Do not show buttons to create new records
column without UPDATE permission	The corresponding row's field is disabled
table without UPDATE permission	All row's fields are disabled and the row's save button is not shown
table without DELETE permission	do not show both row and list delete buttons
stored procedure without EXECUTE permission	The corresponding WAMCaller button is not shown
external ASP file without Windows 'execute' permission	The corresponding WAMCaller button is not shown

4.4 Predefined SQL user data types

Some SQL domains/user types are predefined by WAM so an application can specify additional functionality at both the SQL server and the GUI levels, simply by setting the SQL type of a table column.

4.4.1 wamdate, wamtime and wammoment

wamdate and wamtime are fragments extracted from the data type 'datetime'; wamdate denotes the calendar date part, and wamtime the time part.

wammoment was created to allow the storage of vague dates, so when the user doesn't know a precise date he can for instance type just the month, year or decade.

4.4.2 wamhierarchical1 and wamhierarchical2

wamhierarchical1 and wamhierarchical2 where created to support tables with hierarchical values coded as hierarchical keys in a single field. They differ in the number of chars used to specify each level, so wamhierarchical1 uses one char and wamhierarchical2 uses two chars per level. When generating the interface WAM allows navigation through the hierarchy.

4.4.3 wamphone

A phone number. In the current implementation it is just treated as a varchar, except in the WAP interface, where a “dial this number” link will appear if the mobile phone supports this WTA feature.

4.4.4 wamemail, wamfax and wamurl

These support format validation for emails, fax numbers or url, respectively. When generating the interface WAM will recognize them and provide the expected navigation, from both rows and lists: email/fax message header setup and URL page opening.

4.4.5 wamimage

Supports the use of images stored on the web, referred by absolute URLs. The user interface will displays the images just like wamexternalimage, but without support for uploading.

4.4.6 wamexternalimage

Supports the use of images stored outside the database. When WAM detects a column using wamexternalimage data type it generates all the necessary GUI support to allow the upload (in rows) and display (in rows and lists) of images. The uploaded image is saved in a file named after the PK value for the record, in directory <web_application_directory>/images/<table_name>, where <table_name> is the name for the table that has the column.

File uploading is done using Persit Software’s ASPUpload server component[17].

4.4.7 wamexternalfile

Columns defined with this user defined data type have the same behavior as wamexternalimage columns. But instead of displaying the file, WAM generates a link to download the resource. And instead of saving the file in the ‘images’ subdirectory, WAM saves it in the <web_application_directory>/files/<table_name>, subdirectory.

File uploading is done using Persit Software’s ASPUpload server component[18].

4.4.8 wamcustomresource (for customized file upload and download)

Same behavior as for wamexternalfile, except it allows the application to save the uploaded files in a different location and, if necessary, using a different user account; and the access to the uploaded file can be done through an intermediate ASP, e.g. to implement secure access[19].

Requires a Javascript handler functions to be declared the global.asa (5.5) file, through the following function call:

```
  WAM.setCustomResource(fileSpecs, details)
```

`fileSpecs` is a path to the file that containing the functions to be used when WAM manages the file. `details` is an object providing access to those functions, through its following properties:

- `fileServerBaseLocation` – the name of the function that returns the location to where the files will be saved. It can be in a different server;

[17] Evaluation copy available at http://www.aspupload.com/download.html
[18] Evaluation copy available at http://www.aspupload.com/download.html
[19] As in an application providing row-level (data dependent) access permissions

- `extraPath` – the name of the function that returns an extra directory path;
- `resourceName` – the name of the function that returns the name of the new file uploaded;
- `logonUser` – the name of the function that must return an object with domain, user and password values. It will be used by the ASPUpload component to change the user that will save the file. Return the null value if you want to use the current user credentials;
- `url` – the name of the function that returns an absolute URL to the ASP page that will allow the user to download the file (this page must be implemented by the application developer);

All the functions will receive two arguments, first is a WAMRowColumn object that represents the wamcustomresource column, and the second argument is the WAMAPI.AppConn object[20] with the current user connection.

see WAM.setCustomResource in 5.5.

4.4.9 wamcolor (for color pickers)

*** This supports format validation for html colors and enhances the input field with a color control. Colors are stored as hexadecimal RGB strings.

4.4.10 wamrange

User data type based on ‘varchar’ data type which values must be on the form “N of MaxN”, where N and MaxN must be positive integers. The value is shown in lists as a horizontal bar with two segments (N and MaxN-N) and a toolTip “N of MaxN”. wamrange has no special impact in WAMRow, it is handled as a normal ‘varchar’, without validation for the format “N of MaxN”.

4.4.11 wamglatlng (for Google Maps)

*** This supports format validation for Google Maps latitude/longitude coordinates (decimal format). It also enhances the input field with a control that opens a WAM customized Google Maps page. This page shows the current coordinate and allows you to pickup a new one. The lists that has columns of this user defined data type are also enhanced, WAM adds a link in the column value to the WAM customized Google Maps page and adds a button to open the same WAM customized Google Maps page with markers to all coordinates in the current list.

For GoogleMaps to display you need to get a key from Google at http://www.google.com/apis/maps , and declare it to WAM in the global.asa file (see 5.5).

4.4.12 wamclass (for entity subclassing support)

See section 4.6

*4.5 *** Support for free text search*[21]

Esta será a função que irá complementar a view documento nas pesquisas full-text:

```
CREATE function [t].[wam_freetext_TABLENAME](
        @freetext_string varchar(256)
) returns table
return (select v.id from t.documento as v inner join t._documento as d on d.id = v.id where freetext(d.textoIntegral,
@freetext_string))
GO
```

[20] Of class DBConnect
[21] Requires the installation of word breakers in SQL Server, cf. http://support.microsoft.com/kb/908441/en-us

View support, for views using nondeterministic functions (such as when it depends on the current user…): Finders and list filters look for functions "wam_freetext_«viewname»" and "wam_contains_«viewname»", resp., ….

***Se a view estiver indexada, as funções não são necessárias porque a pesquisa será aplicada directamente na view.

4.6 Support for object-oriented modeling

Relational databases can represent some object structures, even without having specific object oriented support in the engine - conceptual objects must still persist in real world databases! WAM includes support for a simple class/subclass (inheritance) representation schema, where an object instance persists into (1-1) related table tuples; each table will contain a special meta information field storing the (most specific) "class" of the instance.

Suppose that Entity, Organization and Individual are SQL tables. WAM will assume that Organization is a subtype of Entity if:

- Both Entity and Organization have primary keys with the same name, and data type.
- Both Entity and Organization have some field T with SQL (user) data type 'wamclass'; this field will not appear to the user
- Organization's primary key is also a foreign key to Entity

Suppose similar properties hold for table Individual too. This being the case, the WAM generated GUI will have the following differences (vs. inheritance not present):

- The rows for Individual and Organization will include all ("inherited") fields from Entity
- When opening a row from an Entity list, the row for the most specific class of the record will be opened, e.g. Individual or Organization (or Entity)
- When creating a row from an Entity list, a popup menu will ask the user whether he desires to create an Individual, an Organization or an Entity
- When saving a row for an Organization or Individual, both that table record and a related record in Entity will be saved; each field T will contain the “class” of the tuple (e.g. ‘Organization’ or ‘Individual’, resp.)

This mechanism applies to multi-level class-subclass chains; the hierarchy must be a tree though (no multiple inheritance). Along an inheritance path (e.g. Individual, Entity, Thing, …) tables can NOT have fields with identical names (besides the primary key and the 'wamclass' field): column names must be unique in the path.

Notice that defining subclasses assumes that these are disjoint; in the example above, an Organization cannot be simultaneously an Individual.

***Note**: in the current implementation it is necessary to call the stored procedure dbo.WAMInheritance after each database structure change relevant to the above, which will build a WAM table with inheritance information.*

4.7 Initial WAMmodels to start development

For starting up application development either of two WAMmodels is provided, by the WAM Installer or the SQL scripts referred in “Manual installation”.

4.7.1 Empty WAMmodel

An **empty WAMmodel** has no information specific to the database it is stored in; all its tables will have no records except for WAM_PRESENTATION, which will have only generic WAM captions and error messages.

4.7.2 Default WAMmodel

A **Default WAMmodel** for a database is an empty WAMmodel plus records that specify the following “default user interface”:

- One list and row for each table and view
- All lists have one or two pre-configured columns, selected using a WAM heuristic based on database meta information
- A lookup (in the detail row) and a stand-alone detail list (in the master row) for each foreign key
- An entry page allowing navigation to all lists, including a few WAMFinders providing navigation to primary key values, has determined by an heuristic based on the foreign key graph

The WAM installer optionally generates a default WAMmodel for the application database.

4.8 Better delete/update cascading for SQL Server

WAM includes Transact-SQL stored procedures and (additional) WAMmodel tables that augment SQL Server with the ability to *better* deal with cascading updates and deletes. Although SQL Server 2000/2005 already support cascading, it does so following a different (to our view worse) strategy: for example, cascading deletes are executed by first deleting master tables and then detail tables, leaving these temporarily inconsistent and breaking its trigger logic – for example, if a detail table trigger maintains a field in the master table. The WAM cascading logic deletes detail tables first, and adds the “NULIFY” (set null) action.

The current implementation adopts a dynamic strategy, where model information is introspected at runtime to process the cascading logic.

In order to coexist with SQL Server’s foreign key constraints these must be relaxed during cascading updates, which requires ALTER TABLE permissions on the current user, potentially causing a security problem.

4.8.1 The two additional WAMmodel tables

4.8.1.1 WAM_DELETE_RULES

This table must be filled for those FK-PK relations whose delete rule is not supported by SQL Server: cascade.

Column Name	Datatype	Description
#CONSTRAINT_CATALOG	nvarchar(128)	Constraint qualifier (FK)
#CONSTRAINT_SCHEMA	nvarchar(128)	Constraint owner (FK)
#CONSTRAINT_NAME	nvarchar(128)	Constraint name (FK)
#UNIQUE_CONSTRAINT_CATALOG	nvarchar(128)	Unique constraint qualifier (PK)
#UNIQUE_CONSTRAINT_SCHEMA	nvarchar(128)	Unique constraint owner (PK)
#UNIQUE_CONSTRAINT_NAME	nvarchar(128)	Unique constraint name (PK)
DELETE_RULE	char(1)	Rule to apply to details when deleting master: C – Cascade D – Set Default N – Set Null

4.8.1.2 WAM_UPDATE_RULES

This table must be filled for those FK-PK relations whose update rule is not supported by SQL Server: cascade.

Column Name	Datatype	Description
#CONSTRAINT_CATALOG	nvarchar(128)	Constraint qualifier (FK)
#CONSTRAINT_SCHEMA	nvarchar(128)	Constraint owner (FK)
#CONSTRAINT_NAME	nvarchar(128)	Constraint name (FK)
#UNIQUE_CONSTRAINT_CATALOG	nvarchar(128)	Unique constraint qualifier (PK)
#UNIQUE_CONSTRAINT_SCHEMA	nvarchar(128)	Unique constraint owner (PK)
#UNIQUE_CONSTRAINT_NAME	nvarchar(128)	Unique constraint name (PK)
UPDATE_RULE	char(1)	Rule to apply to details when updating master: C – Cascade D – Set Default

		N – Set Null

4.8.2 Data manipulation procedures

WAM_Delete and WAM_Update are the stored procedures that allow you to execute delete and update actions, respectively, applying the rules you defined in the above WAMmodel tables. For other relations WAM stored procedures simply apply the operating rule for SQL Server.

When executed, both stored procedures verify referential integrity, returning error messages to the user when it is violated.

You must be careful when writing the statements to execute delete or update WAM procedures and duplicate all quotes for those field values that requires quotes, as you can see in the examples below. The string passed to the WAM procedure that contains field values, is treated inside it to verify that all fields have their values in a correct format.

4.8.2.1 WAM_Delete

This procedure deletes the row identified in @THEpk argument from the table identified in @THEtable argument.

Parameter	Data Type	Description
@THEtable	varchar(128)	Table name where to delete the row
@THEpk	varchar(8000)	String with PK columns and values which identifies the row to be deleted
Using WAM_Delete		
exec WAM_Delete ‘Employees’, ’EmployeeID=5’ exec WAM_Delete ‘EmployeeTerritories’, ‘EmployeeID=5,TerritoryID=’’10019’’’		

4.8.2.2 WAM_Update

This stored procedure updates the column values for the row which has the PK mentioned in @OLDpk with new values referred in @NEWfields, and (optionally) @NEWtextFieldValue_1 … @NEWtextFieldValue_10

Parameter	Data Type	Description
@THEtable	varchar(128)	Table name where the update is to be made
@OLDpk	varchar(8000)	Column names and values identifying the PK for the row to be updated
@NEWpk	varchar(8000)	Column names and values for the new PK – the same as OLDpk if PK doesn’t change
@NEWfields	varchar(8000)	All column names and values, including those for the PK, for the row to be updated. Remarks: columns with text data type must be supplied separately using the parameters below.
@NEWtextFieldName_1	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_1	Text	Value for the column with text data type – optional
@NEWtextFieldName_2	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_2	Text	Value for the column with text data type – optional
@NEWtextFieldName_3	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_3	Text	Value for the column with text data type – optional
@NEWtextFieldName_4	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_4	Text	Value for the column with text data type – optional
@NEWtextFieldName_5	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_5	Text	Value for the column with text data type – optional
@NEWtextFieldName_6	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_6	Text	Value for the column with text data type – optional
@NEWtextFieldName_7	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_7	Text	Value for the column with text data type – optional
@NEWtextFieldName_8	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_8	Text	Value for the column with text data type – optional
@NEWtextFieldName_9	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_9	Text	Value for the column with text data type – optional
@NEWtextFieldName_10	varchar(128)	Name for the column with text data type – optional
@NEWtextFieldValue_10	Text	Value for the column with text data type – optional

Using WAM_Update
exec WAM_Update 'Category', 'CategoryID=4', 'CategoryID=4', 'CategoryID=4,CategoryName='' Dairy Products'',Description='' Cheeses and milk'''

4.9 About SQL engine built-in meta information

4.9.1 SQL-92's INFORMATION_SCHEMA

WAM assumes the database schema to be designed and maintained with some external tool, such as Microsoft's Enterprise Manager or any other. At runtime the data model is gleaned from the standard INFORMATION_SCHEMA (http://msdn.microsoft.com/library/psdk/sql/ia-iz_10.htm). In particular WAM uses the information about tables, views, columns, domains, primary keys, foreign keys, users and their privileges.

Some domains (custom user types) are used to implement abstract data types known to WAM objects on the GUI side, such as weight, vague dates, hierarchical codes, URL, email, etc

4.9.2 Additional VIEWs and tables

In order to complement the INFORMATION_SCHEMA some additional VIEWS are defined, accessing SQL Server system tables and WAMmodel tables to get the following information:

4.9.2.1 WAM_DETAIL_LISTS

NOTE: on the latest WAM version this view is implemented as a cache (maintained by WAMmodel triggers) in a table for efficiency.

This view exposes the relationship between rows and their correspondent detail lists. Detail lists related to a row are determined by detecting FKs for the table in which the row is based, and, for each FK, finding lists in the WAM_LIST table that uses that FK as a start point in their FK paths. Details lists will be either standalone or embedded, as they have or not the caption field filled in WAM_PRESENTATION.

Column name	*Description*
Listtype	"embList" for embbeded, "saDList" for standalone
Row_name	Row's name
Language	Language for detail list in WAM_PRESENTATION
Type	"list", type in WAM_PRESENTATION for detail list
Name	Detail list's name
Caption	Empty '' when listtype=embList, label to use when drawing button in row when listtype= saDList
Tip	Tip for the button when listtype= saDList
Keep_with_next	See WAM_PRESENTATION table
Comments	See WAM_PRESENTATION table
Edit_table_schema	Owner for edit table
Edite_table_name	Table or view whose row will be used to edit items in the detail table

4.9.2.2 WAM_REFERENTIAL_CONSTRAINTS

This is a complementary view to the REFERENTIAL_CONSTRAINTS view of INFORMATION SCHEMA. It gives us more detailed information about how two tables are linked by means of a FK, indicating which FK field, from the detail table, links to which PK (primary key) field in the master table.

Column name	*Description*
constraint_schema	FK constraint owner
constraint_name	FK constraint name
constraint_table_name	Detail table name
constraint_column_name	FK column's name in detail table
unique_constraint_schema	Master table owner
unique_constraint_table_name	Master table name
unique_constraint_column_name	PK column's name in master table
ordinal_position	Column identification number

4.9.2.3 WAM_STORED_PROCEDURES

Reflects information about SQL Stored Procedures and their parameters.

Column name	*Description*
sp_schema	Stored procedure owner
sp_name	Stored procedure name
Parameter	Stored procedure parameter name
Domain	User-defined data type
data_type	System-supplied data type
Isoutput	Bit indicating if the parameter is an output parameter (1) or not (0)

4.9.2.4 WAM_AUDIT_LOG

This table contains an abstraction of the HTTP log, and describes the user's actions related to GUI objects. It's created and maintained by a task that copies IIS logs to WAM_INET_LOG, and then extracts the juice of WAM_INET_LOG to WAM_AUDIT_LOG. 22

Column name	Description
#id	Unique automatic identification
access_date	Date and time of user operation
user_name	User's account
object_type	Type of GUI object
object_name	Name of GUI object
user_action	Type of action produced
context_url	Relative path of URL
context_value	Query string of context_url
time_taken	The duration of time, in milliseconds, that the action consumed. (same as IIS log)
server_id	The identification of the server on which the log entry was generated.

This table allows extensive logging of user actions (including data access through the application human interface) as well as application tuning: the time_taken field reveals SQL optimization opportunities.

4.10 WAMmodel SQL utilities

The following SQLstored procedures provide some bulk operations on the WAMmodel.

[22] WAM_INET_LOG_ERROR is in development to maintain the errors occurred in this process

4.10.1 Copying list preferences between users

```
WAMSetDefaultPreferences (destinationuser,listschema,listname,usermodel,delfilters)
```

Deletes or creates (default) values for a specific destinationuser. This procedure acts in 3 tables,WAM_CRITERION, WAM_LIST and WAM_LIST_COLUMN. If the list_name parameter is null it sets all values to default otherwise if list_name is given it only changes the values related to that list_name. The delfilters parameter allows to control if is the user values in WAM_CRITERION (his search filters) are to be deleted.

```
WAMCopyCriterion (user_id,table_schema,table_name,criterion_description)
```

Given a user_id, table_schema,table_name and criterion_description, this procedures copies all search filters that match table_schema,table_name and criterion_description of user_id to all others users in WAM_LIST, for the same table_name and table_schema.

```
WAMApplySharedCriterion(user_from,user_to,table_schema,table_name,criterion_description)
```

Copy a search filter from a user to another of a specific table by creating new records in the WAM_CRITERION table and updating the current_criterion field in WAM_LIST table for the “receiving user” user_to.

4.10.2 Copying WAMmodel definitions for a table to a view

```
WAMviewObjects (table_schema, table_name, view_schema, view_name,
   master_constraints, detail_constraints, wam_row_columns, wam_lookup_columns, wam_lists)
```

VIEWs automatically share its underlying table WAM_PRESENTATION records for columns. But they do not share lookup defininitions, detail lists, etc. This procedure copies WAMmodel records for a (table) row to a (view) row; it will copy (or not) each of the data referred by the boolean arguments (last 5 above).

4.10.3 Editing foreign keys in a WAMmodel

```
WAMreplaceFkconstraints(oldSchema,oldName,newSchema,newName)
```

When you need to change a foreign key constraint (fk) name in the database and if the WAMmodel already refers it, it will become inconsistent. This situation can be overcome by using WAMAdmin’s inconsistency detector (section 7.5); but it can also be prevented by using the above procedure, which changes de fk name in the WAMmodel.

5 Web server and browser layers: the WAMLibrary objects

Given database meta information plus WAMmodel at runtime, it is possible to derive from it all or substantial fragments of the GUI, using either URLs or Javascript (server) objects defined in the WAMLibrary, the WAM Classic ASP generic runtime. Those objects are in fact a higher level layer over Microsoft ADO objects: since they know about the application meta-model, and since they support many of the GUI fragments a user expects, it is possible to write or customize large parts of the application GUI with little effort.

In the following sections some examples will refer the Northwind database[23].

5.1 Dispensing with ASP programming: the standard pages

In order to prototype (or even develop) an application without any ASP scripting two WAMmodel-driven pages are available, which act as runtime “GUI interpreters” for the model, implementing “The default interface navigational structure”. See the “WAM development tour” for examples.

5.1.1 A standard entry page: default.asp

This page provides a button to navigate for each of all (non-detail) lists, and finders for the “most interesting”[24] tables. It can be accessed at <application_root>/WAMLibrary/Interface/default.asp

In order to customize this page (meaning, to replace it by a custom page) see 5.4.10. However there's also support for user personalization, allowing each user to have a different set of finders and list buttons, see 5.1.1.1.

(See Interface\default.asp)

5.1.1.1 Customizing the entry page

The entry page can be customized to be unique for each user. Customization mode is available by clicking on “Personalize”, and allows to:

- Show, hide or move groups in the entry page.
- Show or hide filters in the entry page.
- Show or hide lists in the entry page.

If a user doesn’t have a customized page, the user receives the dbo’s preferences, and starts customizing using those preferences.

After finish, the user should click in “Finish personalization”, and is able to continue any time later.

The current personalization for an user is kept in a record in WAM_PREFERENCES.

5.1.1.1.1 Customizing groups

The user can customize groups using the buttons: .

The button allow the user to add a group in a specific position. Clicking in the button, we will get a menu containing the groups that are hidden and that can be added to the entry page.

The button and , allow the user to change the group position order.

[23] http://www.microsoft.com/downloads/details.aspx?familyid=06616212-0356-46a0-8da2-eebc53a68034&displaylang=en

[24] Using the heuristics in Finder Groups, see 5.4.9.4

Finally, the button ✖, allow the user to remove a group.

A group is not actually deleted from the user preferences, but just hidden. This feature allows a user to restore a deleted group, keeping the previous lists and filters preferences.

5.1.1.1.2 Customizing lists

The “Add list” link, on the left top of each group, allows making visible a list. The link produces a menu containing the lists bellowing to the current group, that are hidden and that can be added.

The buttons ✖over the lists, allows removing the respective list.

5.1.1.1.3 Customizing finders

Customizing finders is very similar to customize lists. The “Add finder”, on right top of each group, allows making visible a finder, and the buttons ✖over the finder, allows removing the respective finder.

5.1.2 All other pages: standard.asp

This implements a page with a row (possibly with embedded lists and/or buttons to standalone detail lists), a list or a detail list. Query string variables are passed to request either of the three page types:

type	**Winame**	**further variables**
list	list name	
sadlist	detail list name	foreign key value to relate to a master record, as key_column=value ennumeration
row	row name	primary key value (key), default value (wrdf_fieldName), groupname (wig)

Notice that the current list search filter is not passed in the query string, nor is it POSTed: it’s persistent in the WAM_LIST table for the current user.

When displaying a row, standard.asp will disable a detail list button if the list is empty and the user has no SQL permission to INSERT.

The standard.asp functionality is available for scripting via the WAMStandardGUI object.

In order to customize this page (meaning, to replace it by a custom page for a particular list or row) see 5.4.2

(See Interface\standard.asp)

5.2 The default interface navigational structure

Assuming that we are not using ASP scripting, and that the default WAMmodel for the database was generated, we’ll have five types of nodes in the navigation graph: start page, list, row, stand-alone list and external page nodes. Additionally, we also have the following types of (directed) navigational edges: DISPLAY_LIST, EDIT, ZOOM, EDIT_RELATED, BACK, DISPLAY_DETAIL, and CALL. Embedded lists are considered part of its embedding row, and search filter pages are considered part of their list owners.

The graph can be thus defined as follows, for now ignoring SQL permissions for simplicity:

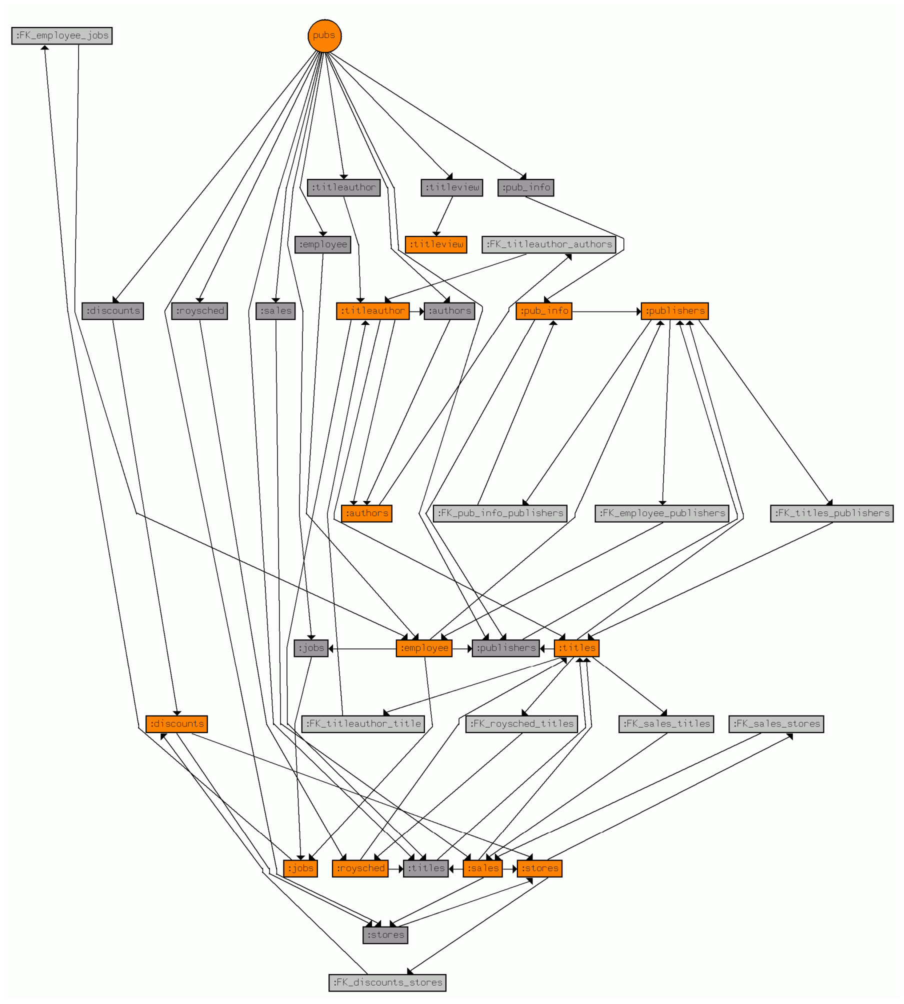
- The start node is in the graph
- For each table or view, there is a list node, with a DISPLAY_LIST edge connected to it from the start node
- For each list node, there is an EDIT edge to a row node for the same table/row (sometimes referred as “drilldown” navigation), and an inverse BACK edge
- For each of the “most interesting tables” (therefore with finders in the start page) there’s a link from the start node to the finder table list and another to the row, and a BACK edge from the row to the start node
- If a row node has an embedded (detail) list whose edit_table is T, there’s an EDIT edge to the row node for T, and in inverse BACK edge

- For each lookup in a row, based on a FK to a table/view M, there will be a ZOOM edge to the list node for M (and an inverse BACK edge) and an EDIT_RELATED edge to the row node for M (and an inverse BACK edge)
- For each (standalone) detail list of a row, there is a DISPLAY_DETAIL edge from the row node to the stand-alone list node
- For each WAMCaller present in a row/list page, there's a CALL edge to an external page node, representing an external ASP instance

The graph is "built" as needed while using the application, as follows:

- The user starts in a node, typically the start page, and follows a link
- Entering a node through other than a BACK link originate a new separate browser window
- Leaving a node through a BACK link destroys the node window
- Destroying a node window transitively destroys all dependent nodes (those that were created as a direct or indirect consequence of following links from the node)

The previous graph for the 'pubs' database example can be represented like this (as shown by aiSee, a graph displaying application [AbsInt 2001] based on WAMAdmin output, cf. section 7.6):



The top round node represents the entry page. Dark gray rectangles are lists, light gray ones are standalone detail lists, and orange rectangles are rows.

Notice that this structure does not necessarily materialize in its entirely, it is generated lazily by the application as the user navigates between pages; also, the structure will grow over a user session because invoking the same page repeatedly creates new page instances, following the web interaction style. And the actual application navigation graph may of course later include other pages and links due to ASP scripting.

The navigational graph is actually **filtered by the current user permissions**; the above definition should read instead “for each X for which the current user has SELECT permission...".

Finally, the graph drawing produced with WAMAdmin does not include BACK or CALL links for clarity, nor does it take permissions into account.

5.3 About custom pages and WAM directories

5.3.1 File directory structure for a WAM application

At the application root level we must have a global.asa file. The global.asa defines the connections strings used to connect to the database, as well as other application variables (cf. 5.4.10). Typically there will also be a default.asp page, as the main entry to the application; it may contain custom HTML, or simply a redirect to the WAM default entry page (see 5.1.1).

Lists and rows that are entirely specified by the WAMmodel dispense with specific ASPs, they’ll simply be accessed through WAM’s standard.asp (see 5.1.1.1). But customized lists and rows require ASPs, that should be located respectively in the **lists** and **rows** application root subdirectories, and whose filenames must match the WAMmodel list and row names (e.g. table or view name, FK path). They entail no navigation graph change per se (unless it’s coded in the ASP): the WAMLibrary will find at runtime that the row/list ASP exists and will use it rather than the generic standard.asp

If the application’s database has wamexternalimage or wamexternalfile user defined data type columns (cf. 4.4), WAM saves the uploaded files in subdirectories, **images/*table_name*** and **files/*table_name***, respectively. Where *table_name* is the name of the table where the column belongs.

Security notice: The images and files directories must not have (ASP) execute permissions.

The default appearance of the WAM generated pages can be changed, in some aspects, using style sheets, which must be placed in subdirectory **css** (see 5.6.2).

All the subdirectories described above must be created in the application root directory.

5.3.2 WAM internals: WAMLibrary directory structure

5.3.2.1 Files in root directory

_MFAQM - contains the sql queries used by WAM, distinct files for each database server.

_SMQM - contains the sql queries used by WAM,distinct files for each database server.

DBConnect - provides all methods for database interactions such as establishing connections, transaction and query execution...

DBDatatypes - defines the datatypes to be used by WAM. This file is essential because each database server has its own data types.

JScriptFuncs - mostly data conversion functions.

WAMAppGlobal.asa - sets WAM's global variables. e.g. version,connection string,default database schema...

WAMCache - by storing information which is likely to be reaccessed in (web server) memory, this speeds up the app.

WAMField - stores data related to a WAM object's graphical representation (such as its label, tip,....) and methods to its easy retrieval (HTML-formatted, datatype aware,...).

WAMInit - WAM initialization objects.

WAMList - one of the main files, controlling all lists and related items. This file works together with WAMList_GetWAMModel.

WAMRow - another main file, with functions and methods for row control; works together with WAMROW_GetWAMModel.

WAMRow_DML – methods for insert,delete and update of rows.

WAMError - Error handling.

WAMEventHandler – event handling

WAMLB - balances the server load by checking the server network configuration and distributing jobs between servers.

WAMList_EditRow - ?????? when user is viewing a wam list and asks for the edition of a specfic row, it checks the existence of such row and then allows to edit it (if exists).

WAMList_SaveColumn – updates a wamcolumn's value in the database.

WAMList_ZoomRow – allow to see the row of a list's column master table.

WAMMenu - ??

WAMMisc - contains miscellaneous functions used by wam (wampermission, dbobject,....)

WAMObjects - set of “include” statements for the WAMLibrary's most commonly used files.

WAMQueryMap - loads and maps all the required sql queries from _SMQM and _MFAQM files so that they can be easily retrieved providing only a name.

5.3.2.2 WAMLibrary subdirectories

Admin -

api	- WAM api
code	-
dbdependencies	- check WAM dependencies from CUSTOM rows,lists.
dbinfo	- gives information of the database, such as row,tables,procedures
default	- default interface for admin use.
dependecies	- function that allows WAM to check dependencies.
documenter	- displays in a form of a HTML document, an enumeration of the lists and rows created for each table or view used by the application.
garbagecollector	- file that checks for unmatched (referenced tables) wamexternalimage and wamextarnalfile data types and provides the tools for removing them.
graph	- creates a graphical representation of the database and WAMmodel by providing a gdl file to current user.
statistics	- shows the database statistics, base table number, views,etc.
WAMmodelConsistency	– contains functions to check the consistency of the database.
WAMmodelEditmode	– turns edition mode on and off
Cache/default	– turns wam cache on and off.

ExportImport/default – interface for export,import tools.

[ExportImport]/FileGenerator – functions to create the export file.

ExportImport/export.prolog - database export tools and functions. The export file follows a prolog syntax.

ExportImport/import.prolog - database import tools and functions. The import is made from a prolog file.

Setup - in this folder we have all files needed to install WAMmodel.

Criterion -

WAMCriterion – functions to manage search criteria defined to or by user.

criterion - creates criterion and execute it.

Huge set - methods for creating and controlling huge sets defined by user

Dialog -

WAMDialog - displays a dialog box.

Finder -

finder – creates a finder object and executes it.

WAMFinder –?? defines finders objects for WAM search functionalities. This works in tables (views??) and search fields are through primary keys columns.

GoogleMaps

default -

WAMGoogleMaps -

Group

WAMListGroup - functions to manage list that belong to a group

WAMTableGroup - contains methods to organize the application tables,views when groups exists or are created.

[Images]

ShowImage – displays an image in the browser

[Interface]

Login – displays a form so that the user can log in

Help – displays context-aware help

WAMStandardGUI – WAMRow and WAMList standard graphical user interface generator

Standard – handles the html requests and instantiates a new instance of WAMStandardGUI with the parameters received (witype and winame)

Default.js – contains the methods which open a requested row or list in a new window, wether by calling standard.asp with the requested values or by opening a custom made WAMList/WAMRow.

Default – responsible for displaying the WAM's initial page

[Interface].[Help] – contains files that are responsible for the WAM Help

[LiveLookup]

livelookup – creates and executes an lookup object.

LiveLookup – contains all functions for wam lookup columns like add a lookup field, removing,changing lookup column value...

ZoomRowexist – contains the tool functions to that allow to select from a column master table.

[WAP]

This folder contains WAM for WAP support.

5.4 Programming ASPs with WAM objects

We'll now review the ASP objects supporting WAM's own `standard.asp` and `default.asp` generic pages, and which can be used in customized ASPs.

You'll notice frequent messages 'draw' being sent to WAM objects; `draw` essencially means "generate HTML and send it to the browser".

5.4.1 Including the WAMLibrary

WAM objects are defined in a few JavaScript files that need to be included, usually at the beginning and end of the ASP file:

```
<!-- #INCLUDE VIRTUAL="/MyApp/WAMLibrary/WAMObjects.asp" -->
…your code here…
<!-- #INCLUDE VIRTUAL="/MyApp/WAMLibrary/WAMEnd.asp" -->
```

5.4.2 Using the WAMStandardGUI object

WAMStandardGUI is used to generate standard GUI elements for a Row, a List or a Details List, allowing the user to access WAMRow, WAMList or WAMStandaloneDetailList objects respectively. It's the top level object to use for most customized pages.

When used to generate a row, WAMStandardGUI gives access not only to the WAMRow object, but also to the embedded detail list, if exists, and a collection of buttons to invoke standalone detail lists related to that row.

5.4.2.1 Implementing a page for a row, without customization

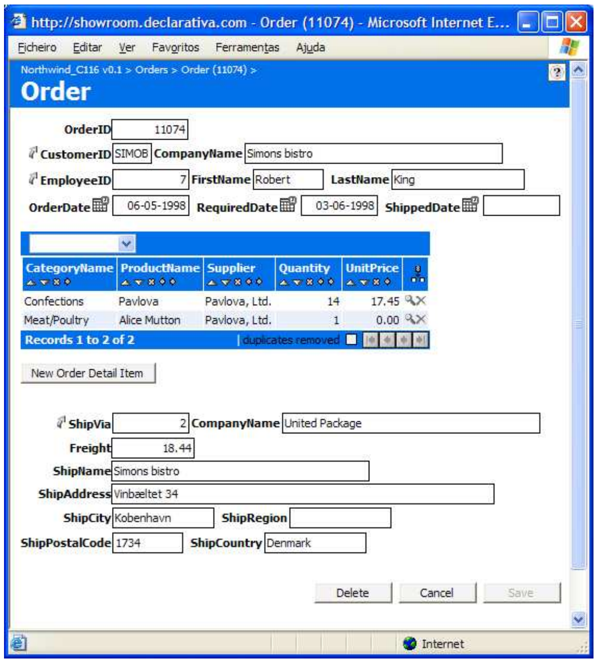
```
var std = new WAMStandardGUI("ROW", "my_row", false);
std.draw(); // draw it as if no custom code existed
```

5.4.2.2 A row with some customization

```
var std = new WAMStandardGUI("ROW", "my_row", false);
std.wamRow.drawFormBegin();
std.drawDListButtons(5); // place buttons at top, 5 per line rather than 4 (default)
std.wamRow.draw(false); // draw the bulk of the row after the buttons…
std.drawEmbeddedLists(); // embedded list…
std.wamRow.drawFormEnd();
```

5.4.2.3 A row with embedded list in the middle

WAM row pages draw fields first, and its embedded list (if any) at the end. How can we obtain the following Orders row in the Northwind application, with embedded list after the ShippedDate field?


http://showroom.declarativa.com - Order (11074) - Microsoft Internet E...
Ficheiro Editar Ver Favoritos Ferramentas Ajuda
Northwind_C116 v0.1 > Orders > Order (11074) >
New Order Detail Item
Delete
Cancel
Save
Internet

Order

OrderID	11074				
CustomerID	SIMOB	CompanyName	Simons bistro		
EmployeeID	7	FirstName	Robert	LastName	King
OrderDate	06-05-1998	RequiredDate	03-06-1998	ShippedDate	

CategoryName	ProductName	Supplier	Quantity	UnitPrice
Confections	Pavlova	Pavlova, Ltd.	14	17.45
Meat/Poultry	Alice Mutton	Pavlova, Ltd.	1	0.00

Records 1 to 2 of 2 | duplicates removed

ShipVia	2	CompanyName	United Package
Freight	18.44		
ShipName	Simons bistro		
ShipAddress	Vinbæltet 34		
ShipCity	Kobenhavn	ShipRegion	
ShipPostalCode	1734	ShipCountry	Denmark

The following does it:

```
<%
var rowOrders = new WAMStandardGUI("ROW", "Orders", false);
rowOrders.wamRow.drawFormBegin();
%>
<table cellpadding='0' cellspacing='0'>
<%  rowOrders.wamRow.drawFromTo(null, "ShippedDate"); %>
<tr><td colspan='2'><%  rowOrders.drawEmbeddedLists(); %></td></tr>
<%  rowOrders.wamRow.drawFromTo("ShippedDate", null, true); %>
</table>
<%  rowOrders.wamRow.drawFormEnd(); %>
```

5.4.2.4 A page for a standalone list, without customization

```
var std = new WAMStandardGUI("LIST", "my_list", false);
std.draw(); // draw it as if no custom code existed
```

5.4.2.5 A page for a standalone list, with some customization

```
var std = new WAMStandardGUI("LIST", "my_list", false);
std.wamList.drawFormBegin();
std.wamList.drawCallers(); // caller buttons will be over the list, rather than below
std.wamList.draw(false, false);
std.wamList.drawFormEnd();
```

5.4.2.6 A page for a standalone detail list, without customization

```
var std = new WAMStandardGUI("LIST", "my_sdlist", false);
std.wamList.draw(); // draw it as if no custom code existed
```

5.4.2.7 A page for a standalone detail list, with user customization

```
var std = new WAMStandardGUI("SADLIST", “my_sdlist", false);
std.wamList.setTableWidth("100%");
std.wamList.drawFormBegin();
std.wamList.drawCallers(); // caller buttons will be over the list, rather than below
std.wamList.draw(false, false);
std.wamList.drawFormEnd();
```

5.4.3 Client Javascripting

In a customized page using WAM, several Javascript objects are made available for scripting in the browser, as well as high-level data events. So customization is not restricted to the web server layer: it is possible to run Javascript in the browser with direct access to (browser client representations) of WAM objects[25].

If you need to add JavaScript code to your custom page, that needs to be executed after the page loads (after all WAM objects are created), use `window.addToOnLoad(MyJavaScriptCall).` This method creates a stack of JavaScript calls to be executed after the page loads. You can add a reference to a function or a string code to be executed (eval).

5.4.3.1 Predefined variables in a (customized) row

WAM.row (of class WAMRow) is the main object generated by the server for the requested row.

This object has two main collections, `columns` and `lookups`. The indexes values of these main collections are the names of the table's columns and the table's foreign key constraints, respectively.

The `columns` collection contains the javascript objects (of class WAMRowColumn), that represent the html input fields for columns (e.g. WAM.row.columns["column_name"]).

The `lookups` collection contains the javascript objects (of class WAMLookup) that represents each lookup (e.g. WAM.row.lookups["FK_master_detail"]).

Each lookup has a collection of `lkColumns`, containing javascript objects (of class WAMLookupColumn) that represents the html input fields for each looked up column. In this collection, the index value has the following syntax: *table_column_name*+*required_join* (e.g. WAM.row.lookups["FK_master_detail"].lkColumns["column_name+dbo.FK_master_x_detail_y"]).

For each standalone (non embedded) **detail list**, there will be a button in the row. The buttons (html elements) generated have the following names: "btn*fk_path*", where fk_path represents the name of the list defined in the WAM_LIST table; spaces, dots and commas are replaced by the character underscore (_) (e.g. btndbo_FK_titles_publishers).

For each **caller**, WAM generates a HTML button with name objCaller*_procedure_schema_procedure_name*. When generating names, spaces and dots are replaced by the character underscore (_). When the caller represents an (external) ASP page the procedure_schema is omitted.

The callers are available in rows and lists.

5.4.3.2 Predefined variables in a (customized) list

WAM.list (of class WAMList) is the main object generated by the server for the requested list.

This object has one main collection named `rows`. This is a JavaScript array object with primary key information for each line in the list. Each element of the array has a property (`pk`) with another collection, the primary key values. (e.g. WAM.list.rows[0].pk[n].value).

Caller buttons are available, with the same names as for rows.

5.4.3.3 WAM high-level events

At the client side, we have these events available from each (browser Javascript) WAM object:

Object	Event	Description
WAMRow	WAMROW_ONBEFOREDELETE	Fires before sending the request to the web server to delete the record.
	WAMROW_ONBEFOREINSERT	Fires before sending the request to the web server to insert the record.
	WAMROW_ONBEFOREUPDATE	Fires before sending the request to the web server to update the record.
WAMLookup	WAMLOOKUP_ONAFTEREXECUTE	Fires after successfully executing the lookup.

[25] You can see some more details about these WAM objects by using the API help in WAMAdmin, see 7.2

	WAMLOOKUP_ONAFTERCLEAR	Fires after cleaning the lookup.
WAMCaller	WAMCALLER_ONBEFOREEXECUTE	Fires before executing the caller.
	WAMCALLER_ONAFTEREXECUTE	Fires after successfully executing the caller.

In order to wire a Javascript function to these events, use the set method of the source's events object; for each WAM object type above:

```
WAM.row.events.set(event_name, function_name);
WAM.row.lookups["LookupName"].events.set(event_name, function_name);
objcallername.events.set(event_name, function_name);
```

Lookup names as in the WAMmodel. For objcallernames see 5.4.3.1

5.4.4 WAM events for server Javascript

At the server side, only the WAMRow object has events.

Event	**Description**
WAMROW_ONAFTERDELETE	Fires after successfully deleting the record from the database. It is executed within an open database transaction using the WAMAPI.AppConn object.
WAMROW_ONAFTEREXECUTE	This is the last event to fire. It fires when the row finished execution. No open database transaction is available.
WAMROW_ONAFTERINSERT	Fires after successfully inserting the record in the database. It is executed within an open database transaction using the WAMAPI.AppConn object.
WAMROW_ONAFTERUPDATE	Fires after successfully updating the record in the database. It is executed within an open database transaction using the WAMAPI.AppConn object.
WAMROW_ONBEFOREEXECUTE	This is the first event to fire. It fires when the row start execution. No open database transaction is available.
WAMROW_ONBEFOREDELETE	Fires before deleting the record from the database. It is executed within an open database transaction using the WAMAPI.AppConn object.
WAMROW_ONBEFOREINSERT	Fires before inserting the record in the database. It is executed within an open database transaction using the WAMAPI.AppConn object.
WAMROW_ONBEFOREUPDATE	Fires before updating the record in the database. It is executed within an open database transaction using the WAMAPI.AppConn object.

In order to wire a (server) Javascript function to one of these events, get a reference to the WAMRow object and set its events:

```
var std = new WAMStandardGUI("ROW", "my_row", false);
var myRow = std.wamRow;
myRow.events.set(event_name, function_name);
std.draw();
```

5.4.5 Row objects

5.4.5.1 WAMRow

`WAMRow` is used to generate a row based in a table or view, and which has the capacity to insert, update, delete, and visualize table or view records. The row has two main collections: columns that belong to the base table and lookups that refer columns in other tables – each having a FK path from the base table. `WAMRow` interacts with two other objects, `WAMRowColumn` and `WAMRowLookup`.

(See WAMRow.asp for more information)

5.4.5.2 WAMRowColumn

`WAMRowColumn` is typically used in the row context and has all the information about a column, not only that gathered from the database builtin meta model but also the extra information from the WAMmodel tables. This is used to generate the GUI element for the column, according to its data type, size, etc., formatting and validating it accordingly.

(See WAMRow.asp)

5.4.5.3 WAMRowLookup

`WAMRowLookup` is used with rows to both retrieve related data from other tables, given a foreign key value, or to fill foreign key values based on a user choice over related data. Each lookup has two navigation links, one to a master table row and the other to a list for the master table.

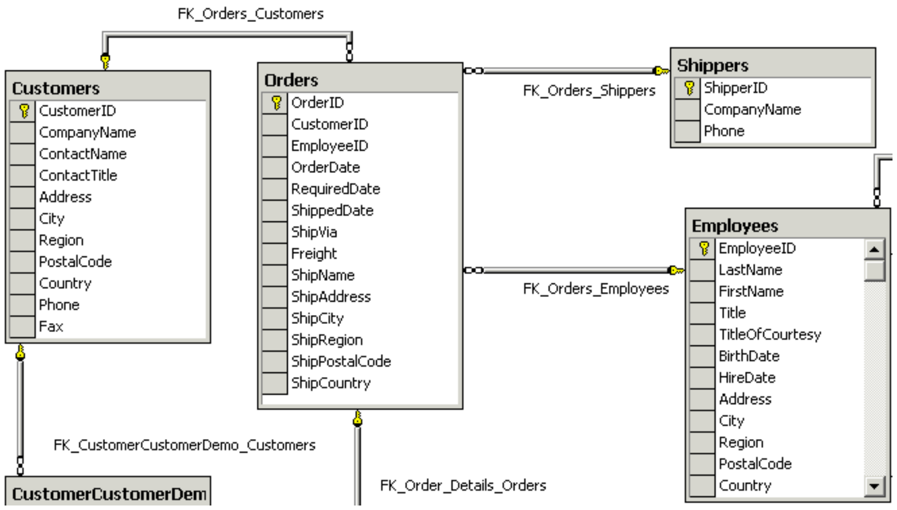
This object has a collection of `WAMRowColumns` for all looked-up columns.

(See WAMRow.asp)

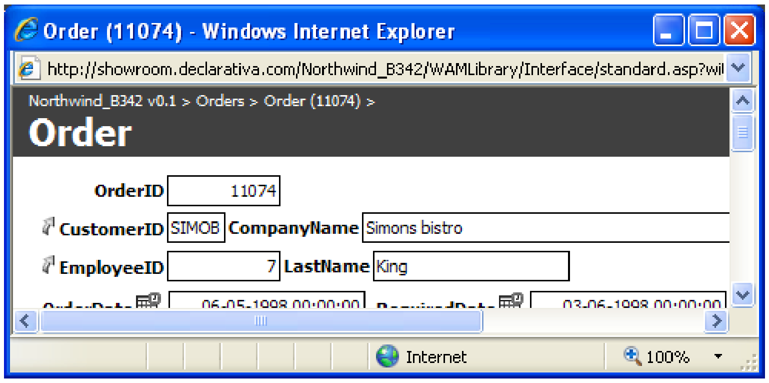
5.4.5.3.1 How to make conditional lookups

Sometimes it's useful to restrict the records in a lookup using one or more column values from the row which contains the lookup. It can be done using either row base columns or other looked up columns.

Let's consider the row Orders for this example, assuming that we want to apply a filter (a value from table Customers) on the lookup to the Employees table:



So when the user tries to find an employee by (say) typing a partial last name in the following page:


Order (11074) - Windows Internet Explorer

http://showroom.declarativa.com/Northwind_B342/WAMLibrary/Interface/standard.asp?wi

Northwind_B342 v0.1 > Orders > Order (11074) >

Order

OrderID 11074

CustomerID SIMOB CompanyName Simons bistro

EmployeeID 7 LastName King

OrderDate 06-05-1998 00:00:00 RequiredDate 03-06-1998 00:00:00

…the lookup (and the disambiguation list, if it appears) should reject employees that are not from the same country of the customer[26]. In order to do this, the lookup filter will be applied to the column Country (Employees table), using the value from row lookup column Country in Customers.

[26] May sound a bit weird, but we wanted to stick to the Northwind example;-)

For that you'll need to customize the row for table Orders as shown bellows.

```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMObjects.asp" -->
<% var rowOrders = new WAMStandardGUI("ROW", "Orders"); %>
<SCRIPT LANGUAGE=javascript>
<!--
function setLookupFilter(){
 WAM.row.lookups.FK_Orders_Employees.addFilterColumn(
  new WAMLookupColumn('Employees', 'Country', WAMDataType.dtnvarchar, null, null, null, null, ''),
  WAM.row.lookups.FK_Orders_Customers.lkColumns.Country
 );
}
window.addToOnLoad(setLookupFilter);
//-->
</SCRIPT>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMEnd.asp" -->
```

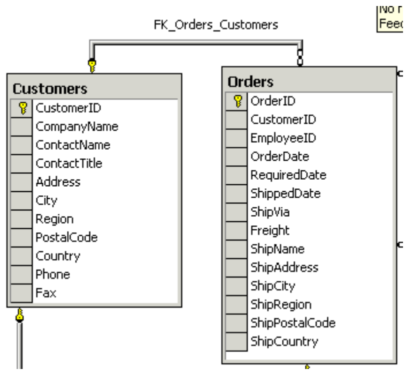
In the function `setLookupFilter` the method `addFilterColumn` (from the WAMLookup object) is used to specify the filter to apply to the lookup. The first argument refers to the column that will be used to filter the lookup and the second argument is the row column that has the value to apply to the filter column (first argument).

5.4.5.3.2 How to get dynamic default values for columns in a row using lookups

This example shows how to customize a row, having one or more lookups, to fill by default some columns using additional looked up columns.

Let's take the row Orders as an example.

We will show how to automatically fill the Shipping columns (ShipName, ShipAddress,…,ShipCountry) in the row Orders, using hidden looked-up columns to the Customer table:



Two steps are needed. First, insert[27] some records into the WAM table `WAM_LOOKUP_COLUMN`, to add the Shipping columns to the list of looked-up columns for the lookup to Customers table:

```
-- Add columns Address, City, Region, Code and Country
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'Orders', 'dbo', 'FK_Orders_Customers', 'Address', ' ', 2, 1, 0 )
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'Orders', 'dbo', 'FK_Orders_Customers', 'City', ' ', 3, 1, 0 )
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'Orders', 'dbo', 'FK_Orders_Customers', 'Region', ' ', 4, 1, 0 )
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'Orders', 'dbo', 'FK_Orders_Customers', 'Code', ' ', 5, 1, 0 )
INSERT INTO WAM_LOOKUP_COLUMN
VALUES( 'dbo', 'Orders', 'dbo', 'FK_Orders_Customers', 'Country', ' ', 6, 1, 0 )
```

Second, customize the row Orders:

[27] This can also be done using the WAM Admin interface

```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/WAMLibrary/WAMObjects.asp" -->
<% var rowOrders = new WAMStandardGUI("ROW", "Orders"); %>
<% if (rowOrders.wamRow.isNew()){ %>
<SCRIPT LANGUAGE=javascript>
<!--
function fillShipColumns(){
      var lkc = WAM.row.lookups.FK_Orders_Customers.lkColumns;
      var rc = WAM.row.columns;
      rc.ShipName.setValue(lkc.CompanyName.getValue());
      rc.ShipAddress.setValue(lkc.Address.getValue());
      rc.ShipCity.setValue(lkc.City.getValue());
      rc.ShipRegion.setValue(lkc.Region.getValue());
      rc.ShipPostalCode.setValue(lkc.PostalCode.getValue());
      rc.ShipCountry.setValue(lkc.Country.getValue());
}
function setLookupHandlers(){
      WAM.row.lookups.FK_Orders_Customers.event.set(WAMLOOKUP_ONAFTEREXECUTE, fillShipColumns);
}
window.addToOnLoad(setLookupHandlers);
//-->
</SCRIPT>
<% } %>
<!-- #INCLUDE VIRTUAL="/WAMLibrary/WAMEnd.asp" -->
```

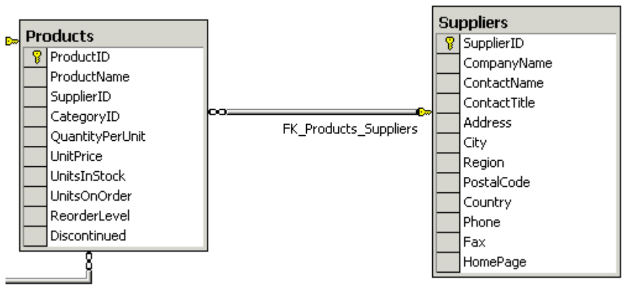
The function `setLookupHandlers` defines the `fillShipColumns` as the function to be triggered after lookup's execution (lookup event `ONAFTEREXECUTE`).

The function `fillShipColumns` initializes the values for the Shipping columns with the values retrieved by looked-up columns.

5.4.5.4 How to execute lazy lookups

Typically in a WAM application a lookup is triggered after the user fills one of the looked up columns, but if a lookup has more than one looked up column, it could be useful to use a lazy lookup. A lazy lookup allows the user to choose the moment when to execute the lookup, allowing the use of more than one looked up column in the execution of the reverse lookup.

Let's take the row Products for this example and the lookup to the table Suppliers:



We will add the column Country to the lookup for table Suppliers, by inserting a new record into the WAM_LOOKUP_COLUMN table (this could also be done with WAMAdmin):

```
INSERT INTO dbo.WAM_LOOKUP_COLUMN VALUES('dbo', 'Products', 'dbo', 'FK_Products_Suppliers',
'Country', ' ', 2, 1, 1)
```

Then we need to customize the row Products to implement the lazy lookup:

```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMObjects.asp" -->
<%
var products = new WAMStandardGUI("ROW", "Products", false);
```

```
// --- Before drawing the row define the lookup to the table Suppliers as lazy
products.wamRow.lookups.FK_Products_Suppliers.setLazy();
products.draw();
%>
<SCRIPT LANGUAGE=javascript>
<!—
function executeLookup(){
      // --- Execute reverse lookup
      WAM.row.lookups.FK_Products_Suppliers.reverseExecute();
}
// Create a button in the interface to execute the reverse lookup when clicked
// The button will be displayed next to the last looked up column
// It could be specified in HTML in some other way… the following patches WAM generated HTML:
function createButton(){
      // --- Retrieve the container of the last looked up column’s input field
      var el = WAM.row.lookups.FK_Products_Suppliers.lkColumns.Country.getHTMLEl().parentElement;
      var btn = window.document.createElement("INPUT");
      btn.type = "BUTTON";
      btn.value = "Supplier ...";
      btn.onclick = executeLookup; // this "wires" the event to execute the lazy lookup
      // --- WAMButton look and feel
      btn.className = "cssWAMstandardButton";
      el.appendChild(btn);
}
window.addToOnLoad(createButton);
//-->
</SCRIPT>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMEnd.asp" -->
```

At this moment you are able to test your lazy lookup. Fill any or both of the looked up columns CompanyName and Country, and press the button “Supplier…” to execute the reverse lookup.

5.4.6 Operations: WAMCaller

`WAMCaller` is the object that allows the user to execute a SQL stored procedure (SP) or to invoke an ASP, by clicking the button generated inside a row or a list by `WAMCaller`. The object has a collection of arguments automatically determined by the caller type where the caller button is placed, so for “row” it creates arguments for PK columns and for “list” it creates an argument with the current SQL statement used in the list, including the current search filter. A confirmation dialog can be requested before the execution of the SP or ASP with a bit in the WAMmodel (cf. `WAM_PROCEDURE_CALL.show_confirm`).

If a value is specified for show_warning_at, in table `WAM_PROCEDURE_CALL`, a test is made before de procedure is executed or the ASP is called in order to find if the number of records to be processed exceeds that value. When this happens, an alert is shown to the user warning him to the possible long wait he probably will face if he chooses to continue with the procedure execution.

After the execution of a SP a success message is shown, if defined in `WAM_PRESENTATION` (by a record with `type=CALLER_SUCCESS`).

In addition to the implict context provided by its row or list, a caller can require further input from the user before executing; this is supported for stored procedures, by using a "WAMA" table.

(See ProcedureCaller\WAMaller.asp)

5.4.6.1 WAM caller user arguments: WAMA

In order to require further input from the user before executing a stored procedure (caller) P, it is sufficient to have in the database a table named "WAMA_P", with the columns identical (in name and type) to the stored procedure arguments. After the user clicks the caller button, a new "modal" page appears with a "ROW" for WAMA_P.

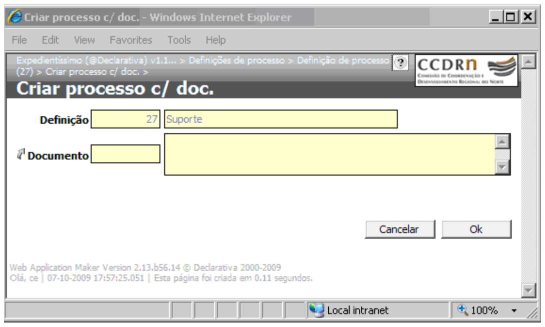
The WAMA_P table should have no records. It is used merely for its metadata.

Example:

create table dbo.WAMA_teste (numero int primary key, obs varchar(32));

(include permissions for SELECT and INSERT)

The WAMA page can be improved by extending its WAMmodel (namely with captions and lookup columns, although no detail lists - because there never will be a record in WAMA_P). Here's an example for a WAMA for a stored procedure invoked from a ROW with a simple primary key (the first field), an additional argument (a foreign key to another table), while providing the user with lookup fields, the one above to show some ROW context, and the second to ease data input:


Criar processo c/ doc. - Windows Internet Explorer
File Edit View Favorites Tools Help
CCDRn
Cancelar
Ok
Local intranet
100%

Expedientíssimo (@Declarativa) v1.1... > Definições de processo > Definição de processo (27) > Criar processo c/ doc. >

Criar processo c/ doc.

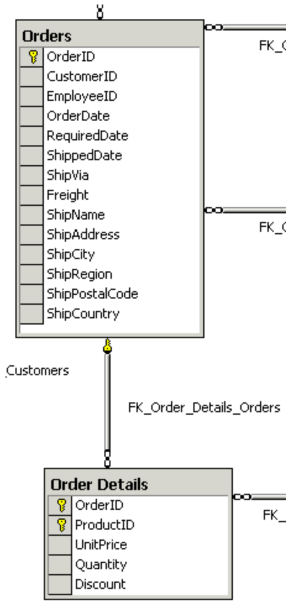
Definição	27	Suporte
Documento		

Web Application Maker Version 2.13.b56.14 © Declarativa 2000-2009
Olá, ce | 07-10-2009 17:57:25.051 | Esta página foi criada em 0.11 segundos.

The above covers most situations; for further control (e.g. client-side Javascripting, etc.), a WAMA ROW can be customized, as for ordinary tables.

5.4.6.2 How to control the enabled state of the WAM callers' buttons (enabled/disabled)

Sometimes it's necessary to control the enabled state of the WAM callers' buttons. We'll use the row Orders for an example: a button which calculates the order total, but which should be enabled only if the order exists in the database (meaning that the user is looking at it *and* it has already been inserted).



First, we define the stored procedure we wish to call, something to compute the order total from its detail lines:

```
CREATE PROCEDURE [dbo].[OrderTotal]
      @OrderID int,
      @Total money output -- watch this name referred below…
AS
SELECT SUM(ROUND(CONVERT(money, Quantity * (1 - Discount) * UnitPrice), 2)) AS '@total'
FROM dbo.[Order Details]
WHERE OrderID = @OrderID
```

After creating the stored procedure in the SQLServer, add it to the WAM_PROCEDURE_CALL table (possibly with WAMAdmin):

```
INSERT INTO dbo.WAM_PROCEDURE_CALL VALUES(‘dbo’, ‘OrderTotal’, ‘ROW’, ‘dbo’, ‘Orders’, 0, null);
```

The last step is the customization of the row Orders:

```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMObjects.asp" -->
<%
var rowOrders = new WAMStandardGUI("ROW", "Orders", false);
// Before drawing the row Orders check if the record being edited exists in the database
if (!rowOrders.wamRow.exists()){
      // --- Disable the button for the WAMCaller dbo.OrderTotal
      rowOrders.wamRow.callers["dbo.OrderTotal"].button.setEnabled(false);
}
// Now, the row can be drawn:
rowOrders.draw();
%>
<SCRIPT LANGUAGE=javascript>
<!—
// --- Displays the total of the order
function showOrderTotal(){
      // --- @Total is the output argument of the dbo.OrderTotal stored procedure
      alert(objCaller_dbo_OrderTotal.outputArgs["@Total"].getValue());
}
// --- Define caller events
function setCallerEvents(){
      // --- Defines the function to be called after the WAMCaller execution
      objCaller_dbo_OrderTotal.event.set(WAMCALLER_ONAFTEREXECUTE, showOrderTotal);
}
window.addToOnLoad(setCallerEvents);
//-->
</SCRIPT>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMEnd.asp" -->
```

5.4.6.3 How to execute WAMCallers with more arguments, beyond primary key columns

WAMCallers in a row assume that the stored procedure will take as arguments the primary key values for the current row. Now we'll enhance the previous example (5.4.6.2) by adding a new input parameter (client discount) to calculate the total of the order with the current client discount applied to it.

First, let's alter the stored procedure, to add a new input parameter Discount:

```
ALTER  PROCEDURE [dbo].[OrderTotal]
      @OrderID int,
      @Discount int = 0,
      @Total money output
AS
SELECT ((100.0-@Discount)/100.0)*SUM(ROUND(CONVERT(money, Quantity * (1 - Discount) * UnitPrice),
2)) AS '@total'
FROM dbo.[Order Details]
WHERE OrderID = @OrderID
```

Some changes need also to be made in the customized row Orders:

- add the next javascript lines to the script block of the row Orders:

```
      function setCallerArguments(){
        // --- Add the new argument Discount to the WAMCaller OrderTotal’s arguments collection
        objCaller_dbo_OrderTotal.add(new WAMArgument("Discount"));
```

```
}
window.addToOnLoad(setCallerArguments);
```

- add the next javascript function to the script block of the row Orders

```
function setCallerDiscount(){
        var discount = prompt("Please enter the customer discount:", 0);
        if (discount == null || isNaN(discount)) discount = 0;
        // --- Set the entered value to Discount argument
        objCaller_dbo_OrderTotal.arguments.Discount.setValue(discount);
}
```

- add this line to the function `setCallerEvents`:

```
objCaller_dbo_OrderTotal.event.set(WAMCALLER_ONAFTEREXECUTE, setCallerDiscount);
```

5.4.6.4 How to obtain results from a WAMCaller, and to open a window depending on them

Add an event handler myeventHandler, and fetch a single caller result. In the following example, the ROW caller calls a stored procedure named t.instanciaProcesso:

```
<%@ LANGUAGE='JScript' %>
<!-- #INCLUDE VIRTUAL="/WAMLibrary/WAMObjects.asp" -->
<% var gui = new WAMStandardGUI("ROW"); // generate the normal page %>
<script charset='iso-8859-1' type='text/javascript' language='javascript'>
<!--
function myEventHandler(){
        var objCaller = objCaller_t_instanciaProcesso;
        // a future WAM version may avoid this textual reference by passing the caller as argument
        if (objCaller.getReturnStatus() != _WAMCaller.STATUS.FAIL){
                var result = objCaller.outputArgs["MYRESULT"].getValue();
                // assumes SELECT … as 'MYRESULT' at the end of the stored procedure
                window.de.add( // uses the window "button deactivator" object
                        $("btnt_instanciaProcesso"), // shortcut for getElementById…
                        window.childWindows[ // now the new window id:
                          window.openChild(
                            "<%= WAMAPI.Page.baseURL('WAMLibrary/Interface/standard.asp', 'witype',
'row', 'winame', 't.processo', 'key', '')%>"+ result)
                        ]);
                // opens new ROW, disabling the caller button until the new ROW window loads
        }
}
function initialize(){
        objCaller_t_instanciaProcesso.event.set(_WAMCaller.EVENTS.ONAFTEREXECUTE, myEventHandler);

}
window.addToOnLoad(initialize);
//-->
</script>
<!-- #INCLUDE VIRTUAL="/WAMLibrary/WAMEnd.asp" -->
```

5.4.7 Lists: WAMList

`WAMList` is used to generate lists for tables. Columns in a list can be either from the list's base table or from any table directly or indirectly related to it by a FK path. The list generated can have a record navigation bar and a search filter definition for filtering rows. From the list's GUI it's also possible to delete a record or to open its correspondent row for edition.

Note: A customized web page can contain only one WAMList object

The user can personalize the lists, with operations to add, change or remove search filters, as well as to add, remove, change sorting or change position of list columns in list. The result is kept as user preferences in the WAMmodel (database).

By default a list shows 15 records by page, but this can be configured for all lists in the global.asa file (see 5.5).

By default lists do not show its total number of records. This information can be made visible for each list by setting its WAM_LIST.**recordCountVisible** bit. Notice that this may degrade performance, as it implies an additional SELECT COUNT(*) statement to be executed. Furthermore, the user must have SELECT permission for all table columns involved, otherwise SELECT COUNT(*) fails due to a permission error.

A list can be opened without showing its data; instead, it will contain a button to show the data. This occurs only for those lists that are defined in the WAMList table with the **auto-refresh** option turned off (auto_refresh column has the value 0). The user can then execute all the operations to customize the list (column adding/removing, ordering, search filters) without re-fetching data from the database, and when he/she is done finally see the data, according to the customization made. The no refresh option is useful for lists with heavy demands on the database server.

There are two exceptions when the auto-refresh option is not considered: lists opened in zoom mode and lists opened in consequence of a finder action. In both these cases lists are always opened in auto-refresh mode.

A list will usually allow the user to open a row only for the base table, by clicking the magnifying glass button; by calling **WAM.setListToMasterNavigation**(true) in the Application_OnStart function in global.asa all lists will also allow the user to open a row for any table with a visible column, by providing an hyperlink in it. For example, in a list with customer names and invoice numbers a click in an invoice number will open one (of many) invoice of the customer, whereas a click on any of instances of a customer will open it.

A list may display duplicated information, say if the user has hidden some columns. There is however a list toggle button, invisible by default, which adds a DISTINCT qualifier to the list SELECT statement; for this button to be visible, the following function should be invoked in the global.asa file, by the Application_OnStart function: **WAM.setListDistinctRowSet**(true).

By accessing the list's hierarchical popup menu the user can add or remove the columns visible in a list. The columns presented in the menu are the columns from the list's base table plus columns from tables/views that have a relation with it through a foreign key constraint or a foreign key path. By default only columns from master tables/views relations are shown in the menu, but this can be configured in order to allow also the addition of columns from detail tables/views relations, by invoking the following function in the global.asa file, from the Application_OnStart function: **WAM.setMenuDetailNavigation** (true). Notice that this may degrade performance, as users get the power to specify queries with more records than the list base table.

A list filter can be easily shared among users if **WAM.setAllowListFilterSharing(true)** is called in global.asa; this will make all lists display a link to copy the current list URL (with filter applied) so the user can give it to someone else.

If **WAM_TABLE_GROUP** records exist for the related tables/views, the list hierarchical popup menu will group tables under their group names.

A list is invoked in either of 3 modes: normal, zoom (to pick a value for a row lookup) and search filter (to disambiguate a finder search).

By default lists will display (scaled-down versions of) images in WAMimage and WAMExternalImage columns. This can have a negative impact on performance, even on intranets, and so there's a method to declare columns with (large) images to be ommited in all lists of the application: **WAM.hideImagesInLists**("owner.tableOrView.column"), to be called by Application_OnStart in global.asa ; for example WAM.hideImagesInLists("dbo.fotoVinha.foto").

A (customized) list can adopt a different strategy for alternatively highlighting its rows. By messaging the list object before drawing the list with

list.**setBackgroundStyle**(WAMLIST_BACKGROUNDSTYLE_GROUPED);

the row background colors will change following data value changes in the leftmost ordered column. The default strategy can be explicitly specified with list.setBackgroundStyle(WAMLIST_BACKGROUNDSTYLE_ALTERNATED);

Finally, any list column can have an **aggregate** value at the bottom. There's currently no user gadget to specify this, it must be specified in WAM_LIST and WAM_LIST_COLUMN in the WAMmodel. Hence this is personalizable per user. As usual, the 'dbo' records function as prototypes for other users.

(See WAMList.asp)

5.4.7.1 How to create a customized List

A list can be easily customized to add some extra functionality.

In the next lines we will show you a simple customization of the list Customers. We will add a link to column CompanyName to show a menu (for the example, with only one option) which will allow the user to create a new Order for the selected Customer.

```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMObjects.asp" -->
<%
var customers = new WAMStandardGUI("LIST", "Customers", false);
customers.wamList.columns["Customers+dbo.Customers.CompanyName"].
 setLink("NW.setCurrentRow(this); NW.menu.open(event);");
customers.draw();
%>
<SCRIPT LANGUAGE=javascript>
<!—
// --- Defines global object
var NW = {
      // --- Menu object
      menu:null,
      current:null,
      // --- Sets the row index of the table cell clicked
      setCurrentRow:function(el){
        // --- The argument el returns the HTML A object
        // --- Get index of line clicked by checking the rowIndex of the HTML TR object
        this.current = el.parentElement.parentElement.rowIndex - 1;
      },
      // --- Open a new (child) window to row Orders
      // --- It uses the primary key values as row defaults
      createNewOrder:function(){
        // --- Close menu
        this.menu.close();
        // --- Open row Orders with a default value in the CustomerID column
        window.openChild(
          "../rows/Orders.asp?wrdf_CustomerID=" + WAM.list.rows[this.current].pk[0].value
        );
      }
};
// --- Creates the menu object
window.addToOnLoad(__createMenu);
function __createMenu(){
      NW.menu = new MAmenu(__initializeMenu);
}
// --- Menu items are defined here
function __initializeMenu(){
      NW.menu.add(new MAmenuItem("New Order", "Create new order for this customers",
"javascript:NW.createNewOrder();"))
}
//-->
</SCRIPT>
<!-- #INCLUDE VIRTUAL="/Northwind/WAMLibrary/WAMEnd.asp" -->
```

The variable `customers` is an instance of the `WAMStandardGUI` object for the list Customers. The `setLink` method turns the column CompanyName into a link column and associates javascript code (`NW.setCurrentRow(this); NW.menu.open(event);`) to the event "click" of that link.

5.4.7.2 How to build lists referring tables from external DBS

Suppose that you have a WAM application based on database A and you want to list records from a table Tb in database B. How to do that?

- In database A, create a table (let's call it table Ta) with the same number of columns that compose the primary key of table Tb from database B. The columns don't need to have the same names, but must be of the same data type.

- In database A, create a view (V) referring the primary key columns and all the columns you want to show from table Tb, with an outer join to table Ta, to show all records from table Tb. All the Tb columns will be considered by WAM as expressions since they don't belong to the base table, so they cannot be edited.
- In the WAMmodel, create a WAMlist record to the view V, which will show the records from table Tb.

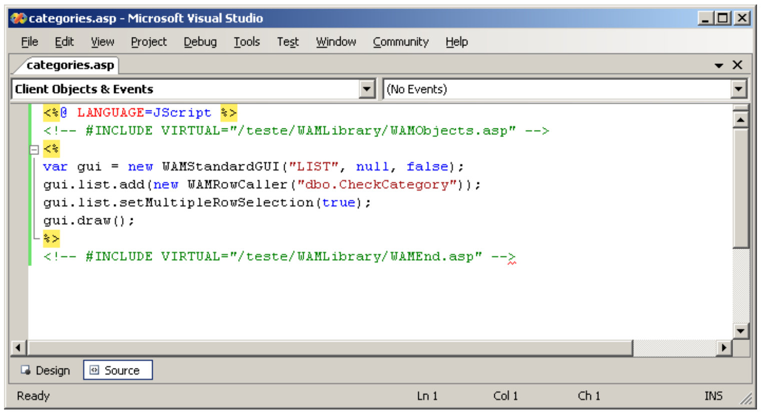
Although we've previously mentioned that the Tb columns cannot be edited, there's a way you can use to workaround that situation. For that you need to create the table Ta as a copy of table Tb (only the schema, not the data). Having this implemented you only need to make the view V editable. You can see how to do that in the (*** future!) section "How to make a view editable" (*** procurar no meu email!!!!).

5.4.7.3 Allowing multiple selection

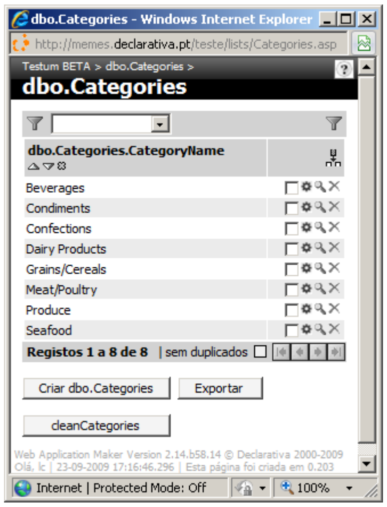
Sometimes it's useful to allow the user to select a subset of the visible list items, by clicking. To insert the multiple selection option we need to create a customized list ASP, and insert in it something like:

gui.list.setMultipleRowSelection(true);

Here's an example:


categories.asp - Microsoft Visual Studio
File Edit View Project Debug Tools Test Window Community Help
categories.asp
Client Objects & Events
(No Events)
Design
Source
Ready Ln 1 Col 1 Ch 1 INS

```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/teste/WAMLibrary/WAMObjects.asp" -->
<%
var gui = new WAMStandardGUI("LIST", null, false);
gui.list.add(new WAMRowCaller("dbo.CheckCategory"));
gui.list.setMultipleRowSelection(true);
gui.draw();
%>
<!-- #INCLUDE VIRTUAL="/teste/WAMLibrary/WAMEnd.asp" -->
```


dbo.Categories - Windows Internet Explorer
http://memes.declarativa.pt/teste/lists/Categories.asp
Testum BETA > dbo.Categories >
dbo.Categories
Criar dbo.Categories
Exportar
cleanCategories
Internet | Protected Mode: Off
100%

dbo.Categories.CategoryName	
Beverages	☐
Condiments	☐
Confections	☐
Dairy Products	☐
Grains/Cereals	☐
Meat/Poultry	☐
Produce	☐
Seafood	☐

Registos 1 a 8 de 8 | sem duplicados ☐

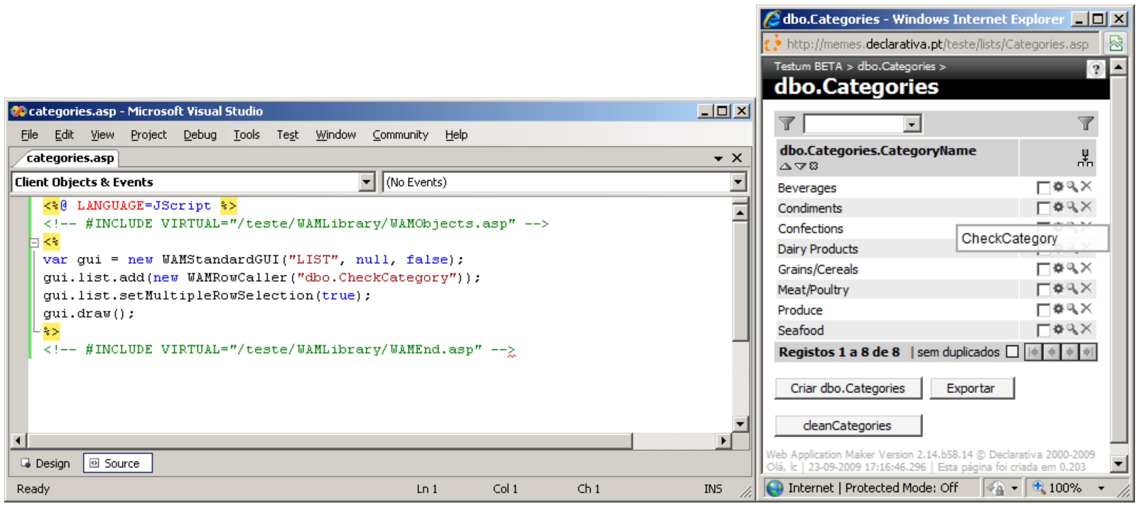
Web Application Maker Version 2.14.b58.14 © Declarativa 2000-2009
Olá, lc | 23-09-2009 17:16:46.296 | Esta página foi criada em 0.203

*** show example of stored procedure processing selected items

5.4.7.4 Invoking row callers from a list

In order to improve usability, it is possible to allow the user to invoke row callers from a list, without opening the row page. Row callers may be accessible from a list by the button ⚙ (right next to the "magnifying glass"), which produces a menu containing the row callers available.

To access a row caller from a list, we need to create a customized list. Here is an example:



```
<%@ LANGUAGE=JScript %>
<!-- #INCLUDE VIRTUAL="/teste/WAMLibrary/WAMObjects.asp" -->
<%
var gui = new WAMStandardGUI("LIST", null, false);
gui.list.add(new WAMRowCaller("dbo.CheckCategory"));
gui.list.setMultipleRowSelection(true);
gui.draw();
%>
<!-- #INCLUDE VIRTUAL="/teste/WAMLibrary/WAMEnd.asp" -->
```

dbo.Categories - Windows Internet Explorer

http://memes.declarativa.pt/teste/lists/Categories.asp

Testum BETA > dbo.Categories >

dbo.Categories

dbo.Categories.CategoryName	
Beverages	
Condiments	
Confections	CheckCategory
Dairy Products	
Grains/Cereals	
Meat/Poultry	
Produce	
Seafood	

Registos 1 a 8 de 8 | sem duplicados

Web Application Maker Version 2.14.b58.14 © Declarativa 2000-2009
Olá, lc | 23-09-2009 17:16:46.296 | Esta página foi criada em 0.203

Row callers must be explicitly added as shown above[28].

5.4.8 Other list objects

5.4.8.1 WAMListColumn

Used inside a list to support list columns, and so a collection is kept in `WAMList`. Each column is generated according to its data type and presentation (align, width, etc) and can have a sorting order defined by the user. An aggregate value can appear at the bottom, cf. 4.1.5.

List columns can have specific HTML style information, by adding styles in the application Cascading Style Sheets, eg in /css/WAMcss.win.css . The style should be named with "css" + OWNER + BASE_TABLE_OR_VIEW + COLUMN_NAME, for example the following makes column description in list dbo.workDone use word wrap:

```
COL.cssdboworkDonedescription{
        width: auto;
}
```

(See WAMList.asp)

5.4.8.1.1 How to prepare a list (defining default columns for users)

The columns that appear in a list for each user can be easily configured, following the steps below:

- access the list you want to configure and add/remove the desired columns using the popup menu
- you can set your list preferences to be used as default by the new users that will access that list

```
exec WAMSetDefaultPreferences
  @user='dbo', @listShema='dbo', @listName='Orders', @user_model='YOUR_ACCOUNT_NAME'
```

If you want to refresh the list preferences for all users, before executing the above you need to delete those preferences:

```
delete from WAM_LIST_COLUMN where table_list_schema = 'dbo' and table_list_name = 'Orders'
delete from WAM_LIST where table_schema 'dbo' = and table_name = 'Orders'
```

[28] Some row callers may have JavaScript dependencies (e.g., interface objects), therefore WAM can not assume that all row callers are callable from the list page.

5.4.8.2 WAMListCriterion (search filter)

(See WAMCriterion.asp)

For each list in the application the user can define search criteria for filtering results, implemented as restricted SQL WHERE clause fragments.

Search filters are supported by this object that handles a collection of search filters for each list, created by the current user in the application. When viewing a list the user can edit the current search filter or create a new one, naming it, applying it to the list, or else removing it from the collection. When creating a new search filter the columns that appear as default are those ones that belong to list's edit table and are visible to the user, but the user can add more columns (only) from the list edit table.

For each column in the search filter there is a menu accessible to the user to help him define the correspondent restriction.
List search filters know about tree node fields (the primary keys of tree tables,). Whenever such a field is added to the filter, an additional operator is available: "_", denoting "under this node".
Lists whose filters include "under a tree node" sub conditions do an extra short SELECT first, to fetch the left/right values for the node, and then add a "node_key BETWEEN left AND right" condition to the WHERE clause of the list SELECT; for this purpose WAM adds a join condition with the tree table to the SELECT statement.

5.4.8.2.1 Conditions involving the current user

The constant "me" ("eu" in Portuguese) denotes (for SQL Server) `SUSER_SNAME(),` the currently authenticated user account.

5.4.8.2.2 Relative time search filter conditions

List search criteria can have time interval conditions relative to "now" in a field, by using the syntax operator, value and unit;

- **unit** can be: year, quarter, month, day, dayofyear, week, minute, second, millisecond. It suffices to write the initial letters of the unit
- **operator** can be any of the relational operators: >, <, >=, …, etc
- **value** must be an integer

Example: "-2month … -1month" means (this date field value) is in the interval between two and one months ago from today

Search filters can also use two constants which can be useful to specify "current week" intervals: **lastsaturday** and **nextfriday**, which denote respectively 0h00 of the previous saturday, and 24:00 of the next friday. For example, a "current week" filter can be defined with lastsaturday...nextfriday.

Finally, there's also the constant 'today', denoting the current date/time.

5.4.8.3 WAMEmbeddedDetailList

There are situations where you want to draw a list embedded in a row, to show records from a detail table related to a row's base table record; that's the purpose of `WAMEmbeddedDetailList`. `WAMEmbeddedDetailList` automatically filters records from the detail table using the FK that connects the two tables (or views).

This is a particular case of `WAMList` with the same functionality; the difference is that instead of being based on a table/view, a WAMEmbeddedDetailList is based in a FK - when the detail table is directly connected to the row's base table - or in a FK path - when more than one FK is needed to connect the two tables or views. It is always connected to a single row.

(See WAMList.asp)

5.4.8.4 WAMStandaloneDetailList

Similarly to `WAMEmbeddedDetailList`, `WAMStandaloneDetailList` is also a particular case for a `WAMList` without loosing functionality, and it's also based on a FK or FK path. Instead of a list embedded in a row, this object generates a standalone detail list, in other words, a list for the detail table and a lookup to the master table to provide context. Typically access to these detail lists is made through buttons drawn in a master table (or view) row.

(See WAMList.asp)

5.4.9 Other objects

5.4.9.1 WAMTableGroup

WAMTableGroup generates an HTML table with buttons and fast entry points (Finders) to the lists for the tables contained in a given group. If no group is specified in its constructor, WAMTableGroup generates tables for all the groups specified in WAM_TABLE_GROUP table, and an HTML table with all remaining tables that doesn't belong to any group. This object is useful for application entry pages.

(See group\WAMTableGroup.asp)

5.4.9.2 WAMListGroup

WAMListGroup generates (just) buttons to access lists within a group, if a group name is specified, or all lists in the application, if no group is specified. Similar to WAMTableGroup without the Finders.

(See group\WAMListGroup.asp)

5.4.9.3 WAMFinder

`WAMFinder` creates, for a given table and a set of columns ordered by inquiry importance, a fast entry point - search - to access a list or a row, depending on the number of results it finds for that search. Successive SELECT statements are executed, looking for the value typed by the user in each column, considering only columns with compatible type (numbers for a number, strings for a string).

Creating a WAMFinder using its Javascript constructor ignores the information in WAM_FINDER (see 4.1.14) for that finder.

See 2.5.1.

(See Finder\WAMFinder.asp)

5.4.9.4 WAMFinderGroup

WAMFinderGroup creates, for a given table group, WAMFinders for the 3 most "interesting" tables in the group, considering only the primary key column (*** and only the first one if the key as more than one, right?). The most "interesting" tables are those with more masters.

Records in WAM_FINDER (see 4.1.14) can add finders to the set determined by the heuristic; they also determine which columns (other than the primary key default above) are considered.

Within the context of WAM's user customizable entry page (default.asp, see 5.1.1.1), the precise finder set to appear is determined by the user.

(See Finder\WAMFinder.asp)

5.4.9.5 WAMTree

WAMTree generates a web page fragment, based in a tree table, with that tree in expandable outline form. Clicking a tree node pops up a menu with all (non-embedded) detail lists defined in the WAMmodel that have that tree table as master. By choosing a menu option the user can view a detail list with records “under <clicked node >”. “Under query” defines a BETWEEN query sub term, encoding the tree hierarchy based on the left/right keys defined in the table tree.

KNOWN ISSUES in the current version:

- There's what we consider a buglet on the tree: when expanding a node, any previously opened detail lists will be closed.
- Tree node menu items (that navigate to detail lists) are never disabled
- The tree title is currently obtained from the WAM_PRESENTATION for type LIST; we could introduce a new type 'TREE' so that a list and tree can have different titles, but this might be overkill.
- Specifying a nonexistent tree node in a search filter causes the list to display with zero tuples, but no error message.
- In lists and search filter pages, the add field menus displays the 4 tree key fields in tree tables, in addition to the tree node field.

5.4.9.6 WAMDialog

WAMDialog allows the developer to implement a client dialog box with a window title, message title and body (which can be composed by multiple lines), and buttons. The dialog box generated has a predefined size (width=300px; height=300px) that can be changed .

The dialog box can be invoked in either of the three following forms:

- Alert: A dialog box with an alert message and an ok button that waits for the user to click the button to close the window.

- Show: A dialog box with a message and a set of buttons, defined by the developer, that waits for the user to click one of the buttons, and returns a value indicating which button the user clicked or undefined if the user closes the window.

- Prompt: A dialog box that prompts the user with a message, an input field and two buttons: OK and Cancel. It returns the value the user entered in the input field, when the user chooses OK button, null if the user chooses the Cancel button or undefined if he closes the window. The developer can define a value to appear has default in the input field.

The following lines must be included in pages that refer WAMDialog object (unless HTML.drawBodyEnd method is invoked, as happens in any page containing a WAM list or row):

```
<script LANGUAGE='JavaScript' SRC='/my_Application/WAMLibrary/Dialog/WAMDialog.js'>
</script>
<script LANGUAGE='JavaScript' SRC='/my_Application/WAMLibrary/JScriptFuncs.js'>
</script>
```

Implementing an “alert dialog”

```
  var objDialog = new WAMDialog("window title", "Message title");
  //Composing the message
  objDialog.addLineBr("Line 1 with a breakspace");
  objDialog.addEmptyLine();
  objDialog.addLine("Last Line for the message");
  //Changing dialog box width and height
  objDialog.setWidth("220px");
  objDialog.setHeight("200px");
  objDialog.alert();
```

Implementing a “show dialog”

```
  var objDialog = new WAMDialog("window title", "Message title", "Message body");
  //adding buttons to show in the dialog
  objDialog.addButton(new WAMButton("button1 caption"));
  objDialog.addButton(new WAMButton("button2 caption"));
  objDialog.addButton(new WAMButton("button3 caption"));
  //getting button selected
  var option = objDialog.show();
  switch (option+””){
    case "undefined":
    //no button was clicked
    break;
    case "button1 caption":
    //button1 was clicked
    break;
    case "button2 caption":
    //button2 was clicked
    break;
    case "button3 caption":
    //button3 was clicked
    break;
  }
```

Implementing a "prompt dialog"

```
  var objDialog = new WAMDialog("window title", "Message title", "Message for the input field");
  //prompts a message with a default value in the input field
  var data = objDialog.prompt("default value for the input field");
```

5.4.9.7 The WAMAPI object

WAMAPI aggregates several global utility objects, created when the page starts executing client code:

- `Page`. Includes methods that simplify the handling of values received by the ASP from URL query string or HTML form variables. It also contains relevant information about the web application mainly used for writing URLs. (See WAMMisc.asp)
- `HTML`. Includes methods to write html code blocks that must be included in every ASP page that uses WAM objects; it's also a preliminary step for abstracting WAM from HTML
- `AppConn`. This object handles a connection to the database with the User connection string, allowing the execution of SQL statements, either returning a RecordSet (`executeRR`) or not (`execute`). Essencially it's a wrapper for an ADO `Connection` object, adding WAM-specific error handling. (See DBConnect.asp)
- `WAMConn`. Same as `AppCon`, but for the WAM connection string (cf. 5.5)
- `Error` (see 6.1.1)
- `User` - Keeps the information about user that is logged in (authenticated by Windows) the application. Through this object the developer can get the name – in applications without IIS authentication the name returned is "Anonymous"- and preferred language(s); it can also draw the user photo image (with `WAMAPI.User.drawPhoto()`), if it is placed in the "images/users/" directory, under the root directory for the application, with the file name "*user_name*.gif".

5.4.10 How to customize the default entry page

Note*: as an alternative to the following type of customization, consider simply relying on user-driven personalization of the entry page, see5.1.1.1*

The simplest way to build a custom version of /default.asp is to copy the WAM default.asp page from YourApplication/WAMLibrary/Interface to the YourApplication root directory. Then, you need to edit the include directives (change from FILE to VIRTUAL, adjusting paths, just as when customizing rows and lists, see 5.4.1. You now have a new application default entry page.

For example, to add a new WAMFinder (cf. 5.4.9.3) to the Customers table in the "Sales" group in the Northwind application you need to add the following line of JavaScript after creating the groupSet object (and before it is drawn!):

```
  var F = new WAMFinder('dbo.Customers', ['CustomerID', 'CompanyName'], false);
  groupSet.items.Sales.finders.add(F);
```

The second argument of the WAMFinder constructor is an array specifying the columns to search (in order).

To delete/hide an undesirable WAMFinder, just remove it from the finders array of its table group, for example:

```
  delete groupSet.items.Sales.finders.items['dbo.Orders'];
```

5.5 Global WAM variables: the global.asa file

WAM requires the ASP global.asa file to initialize some application variables, by invoking the functions above, and optionally others.

- WAM.setWAMConnectionString(`connectionString`) – connection string (as documented in the ADO `Connection` object) for an SQL Server authenticated login, whose SQL permissions include the "System Administrators" role; this is used by generic WAM code
- WAM.setAppConnectionString(`connectionString`) – user connection string, for either a SQL Server or Windows authenticated login; this is used for all user interaction with the database
- WAM.setAppVersion(`version`) – an application name, that will appear as title in WAM-generated pages
- WAM.setDefaultLanguage(`language`) – a default language, determining the language if the browser has no language preference
- WAM.setDefaultDatabaseSchema(`schema`) – a default database schema
- WAM.setDecorType(`type`) – the navigational control decoration type, either 'IMG' to use GIF buttons or anything else to use text links with a WAM font (in order to optimize page loading in the browser)
- WAM.setAppInfo(`info`) – an application footer note
- WAM.setListPageSize(`size`) – number of records, by page, to show in lists
- WAM.setCache(`flag`) – a boolean that indicates, if true, that the application caches WAM Model information, if false, the application doesn't cache information
- WAM.setCloseOnSave(`flag`) – a boolean that indicates, if true, that rows are closed after its data is saved by the user in database; otherwise rows remain opened.

The following functions can be called from Application_OnStart and affect lists (see WAMList object, 5.4.6):

- WAM.setMenuDetailNavigation(true);
- WAM.setRecordCountVisible(true);
- WAM.setListDistinctRowSet(true);
- WAM.setListToMasterNavigation(true);
- WAM.setAllowListFilterSharing(true)

The following functions can be called from Application_OnStart and affect the list and row columns of types wamemail or wamfax resp.:

- WAM.setWAMEmailBCC(`EmailAddressForBCC`) – an email address to be placed in BCC email field when the user clicks the email link, in wamemail columns. This can be used for example in conjunction with an archiving agent
- WAM.setWAMFaxGateway (`EmailAddessForFaxGateway`) – an email address for the fax gateway to be used when the user clicks the fax link, in wamfax columns; the mailto: link is setup for "FAX_NUMBER <EmailAddessForFaxGateway >. This allows automatic setup of the header for an email message to a fax gateway such as GFI's (http://www.gfi.com) FaxMaker

Here's a typical global.asa, catching the ASP `Application_OnStart` event and invoking WAM's initialization functions:

```
<!-- #INCLUDE VIRTUAL="/Pubs/WAMLibrary/WAMAppGlobal.asa.asp" -->
<SCRIPT LANGUAGE="JavaScript" RUNAT="Server">
function Application_OnStart(){
    // WAM account in SQL Server:
    WAM.setWAMConnectionString("Provider=SQLOLEDB.1;User ID=WAMaccount;PASSWORD=WAMpassword;Initial
Catalog=MyDB;Data Source=MyServer;");
    // User account, in this example delegating to Windows's authentication
    WAM.setAppConnectionString("Provider=SQLOLEDB.1;Integrated Security=SSPI;Initial
Catalog=MyDB;Data Source=MyServer;");
    WAM.setAppVersion("MyApp");
    WAM.setDefaultLanguage("en");
    WAM.setDefaulSchema("dbo");
    WAM.setDecorType("IMG");
    WAM.setAppInfo("Some footnote");
    WAM.setListPageSize(30);
    WAM.setMenuDetailNavigation(true);
    WAM.setRecordCountVisible(true);
    WAM.setListDistinctRowSet(true);
}
</SCRIPT>
```

The following two functions set global formatting for all numeric and date fields; defaults are day-month-year and 999 999.00:

- `WAM.setDateFormat("dmy", "-");`
- `WAM.setMoneyFormat(".", 2, " ");`

The following function changes the default window autosizing strategy, forcing all windows to maximize when they open:

- `WAM.setWindowMaximized(false);`

The following makes calendar controls appear near datetime, WAMDate and WAMTime fields:

- `WAM.setCalendar(true);`

The following configures the functions required for `wamcustomresource` columns:

`WAM.setCustomResource(fileSpecs, details)` – `fileSpecs` is a path to the file that containing the functions to be used when WAM manages the file. `details` is an object providing access to those functions. See 4.4.8

The following provides information necessary for GoogleMaps column fields (see 4.4.11):

```
WAM.setGoogleMapsAPIKey(googleKey, defaultLatitude, defaultLongitude,
defaultZoom)
```

The following forbids access to all users, displaying a warning; can be used during application maintenance:

```
Application("App_HavingAMassage") = true; // false would make the app available
```

5.6 Global WAM styles and GIFs

WAM uses a set of CSSs and GIF icons in the generated HTML fragments, which can be redefined for an application.

5.6.1 GIF icons

The following are in WAMLibrary/images:

Icon	**Context**	**Description**
menu-more.gif	LIST	used on menu to indicate a sub-menu
no-image.gif	ROW	image not available
uc.gif	at page bottoms	under construction
nav-firstpage.gif	LIST	goto first page
nav-nextpage.gif	LIST	goto next page
nav-previouspage.gif	LIST	goto previous page
nav-lastpage.gif	LIST	goto last page
nav-firstpage-d.gif, nav-nextpage-d.gif, nav-previouspage-d.gif, nav-lastpage-d.gif	LIST	same as previous 3, but for when the buttons are disabled
move-column-left.gif	LIST	move column one position to left
move-column-right.gif	LIST	move column one position to right
order-asc.gif	LIST	sort list by ... in ascending order
order-desc.gif	LIST	sort list by ... in descending order
order-asc-p.gif	LIST	list column is sorted ascending
order-desc-p.gif	LIST	list column is sorted descending
remove-column.gif	LIST	remove column from list
add-column.gif	LIST/CRITERION	open menu to add a new column
row-edit.gif	LIST	edit record
row-delete.gif	LIST	delete record

arrow-top-right.gif	ROW	lookup - find a record
arrow-top-left.gif	LIST	lookup/zoom - return a record
export.gif	LIST	export list
father.gif	ROW	open father's row (wamhierarchical columns)
descendants.gif	ROW	open descendants' list (wamhierarchical columns)
bit-checked.gif	LIST	checkbox checked, for bit columns
bit-unchecked.gif	LIST	checkbox unchecked, for bit columns
criterion-remove-column.gif	CRITERION	remove column from search filter

5.6.2 CSS styles

Styles are defined in the WAMLibrary, which contains several CSS files. Additional (app-dependent) files can be placed in /css (or css directory at the application root), using the same names below, allowing styles to be added or redefined.

A style sheet is used for Windows browsers (`WAMLibrary/WAMcss.win.css.txt`), and another for Macintosh browsers (`WAMLibrary/WAMcss.mac.css.txt`):

Style	**Purpose**
.cssWAMstandardButton	Buttons used in the application
.cssWAMstandardEdit	Row INPUT elements
TEXTAREA.cssWAMstandardEditTEXTAREA	Row TEXTAREA elements
SELECT.cssWAMstandardEdit	Row SELECT (popup) elements
A	All links used in the application
A.cssWAMzoom	Links to open a row in "zoom" mode (lookup)
A.cssWAMdecor	Link that uses font decoration (decoration types: GIF, FONT)
.cssWAMcritDescription	SQL WHERE clause that appears in list when using a search filter
.cssWAMcritPopup	List search filter's popup (HTML SELECT element)
TD.cssWAMwinTitle	Page header
SPAN.cssWAMwinTitleTop	Application's label
SPAN.cssWAMwinTitleBottom	Row or list name
TD.cssWAMlistDark	Top and bottom of the list
TR.cssWAMlistLight	Even list lines
TR.cssWAMlistSelected	List's selected line (tables with wamhierarchical columns)
SPAN.cssWAMfinderSearch	Finder's advanced search link
TD	Global TD
BODY	BODY
TD.cssWAMlistCell	List’s cell
TD.cssWAMlistRangeBar	Segment for N value (wam_range columns)
TABLE.cssWAMlistRangeBar	Segment for MaxN-N value (wam_range columns)

There's also a way to associate styles to specific list columns, see WAMListColumn object.

5.7 *WAM cache and its freshness vs. the database*

WAM has a meta-information cache, to minimize access to the WAMmodel in the database. In older versions it was stored in IIS Session variables; it is now implemented in IIS Application variables. The current implementation therefore allows each user's cache to persist between sessions, speeding up (comparatively to the old Session approach) the first visualization of most application pages in all but the first day.

Therefore when developing an application and changing its WAMModel or database structure, it is usually necessary to restart IIS to see the effect of the changes in the application, because WAM caches some WAMmodel information in application variables. For this either (a) UNLOAD the application, to force a cleaning up of all user caches (with the IIS Admin application open up the properties for the application)... or (b) simply save `global.asa`, which makes IIS UNLOAD and restart the application

`WAMAdmin` is aware of this, and so it disables the *current user's* cache automatically when editing; it can also disable or enable (all) other user caches (referred as "Global WAM Cache").

For other scenarios WAM provides an URL so the user/developer can clean his own cache without restarting IIS. To close a session the developer should invoke the following URL:

`ApplicationRoot/WAMLibrary/RefreshWAMmodel.asp.` A user shortcut is also available: by clicking the user name appearing in the footer (in the application entry page) the session is terminated.

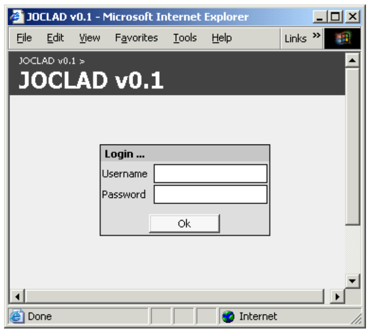
5.8 Custom user authentication

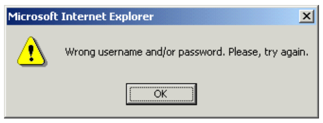
So far it has been assumed that the aplication user is authenticated by either the (web server running on the) Windows operating system and/or the associated data server. But in many situations it is more convenient to authenticate using some application-specific approach - for example the application may prefer to manage (potentially unlimited) user accounts in its own database.

The WAM login feature addresses this need. All you need to provide is an authentication (ASP) script fragment, placed somewhere in the web structure, and declare it on Application_OnStart().

```
WAMsetLoginValidationPage("authenticate.asp"); // placed at the application root directory
```

The ASP fragment should validate the user/password, which it must get from two POSTed form fields, and on success set two Session variables, WAM_UserLoggedIn (boolean) and WAM_UserNameLoggedIn (username). When an user attempts to access any WAM-based page (e.g. an ASP including the WAMObjects.asp file) he/she will be redirected to a standard login page:


JOCLAD v0.1 - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links
JOCLAD v0.1 >
JOCLAD v0.1
Login ...
Username
Password
Ok
Done
Internet



Wrong username and/or password. Please, try again.

(*"JOCLAD" above being the name of an existing application*). After the user submits the name/password the authentication fragment ('/authenticate.asp' above) is executed with a Server.Execute command; if it sets `Session("WAM_UserLoggedIn")` to `true`, the user is redirected to his original (WAM application) URL request. If not an error dialog appears.

Here's an example for an authentication script fragment:

```
<%@ Language='JavaScript' %>
<%
if(Request.Form.Count>0&&
 (Request.Form("userName")+"")=='aUser'&&
 (Request.Form("password")+"")=='aPassword') // a real app might compare to values in its database
 {
    Session("WAM_UserLoggedIn") = true;
    Session("WAM_UserNameLoggedIn") = "aUser";
 }
%>
```

So after login success the validation script must update these WAM Session variables: WAM_UserLoggedIn and WAM_User NameLoggedIn. The possible values for the WAM_UserLoggetIn session variable are either true or false. WAM_UserNameLoggedIn should have the name of the user logged in.

In order to obtain the user identity on the data server side (e.g. in a VIEW, as when defining data-dependent user access policies) the following user functions are available:

- `dbo.GetUser()`, for SQL Server
- `informix.GetUser()`, for Informix IDS

5.9 WAP support – the WAM Mobile interface

WAM includes a preliminary version of a WAP interface, which automatically offers any WAM application an alternative mobile phone interface, based on the same WAMmodel behind WAM applications for web browsers.

The WAP interface for an application is invoked through the URL http://APPLICATION_URL/WAMLibrary/WAP/ For this to be active WAP MIME types must be defined in the web server; this is documented in the WAM Installer.

Most WAM aspects such as a multi-lingual interface, rows (single record forms), field validation and other features are implemented based on the same WAMmodel used in applications for browsers, except for the following current limitations: no inverse lookups; no immediate field validation due to the lack of WML events vis a vis web browsers; no ability to create list search filters (users get to reuse filters created through a web browser); SQL user permissions and table constraints are not visually reflected in the interface, they will simply originate database permission errors later.

The WAM Mobile interface is compatible with Wireless Application Protocol 1.2 or later devices.

6 Error and warning Handling

WAM includes the ability to report parameterized multilingual error messages (either database server, web server or web client raised) and (except for database server errors) to focus on a relevant GUI object. Each error situation is defined in the WAMmodel, as a record in WAM_PRESENTATION with type=ERROR. The 'name' is used to raise the error, and 'caption' contains the error message, which may include value placeholders to enrich the message with context.

An analogous mechanism exists for warning the user before an operation concludes, rolling it back if the user does not confirm it.

6.1 Raising and handling errors in the SQL layer

To handle errors SQL Server has the @@ERROR system function, that can be executed after each Transact-SQL statement to test if the statement generated an error. The @@ERROR function returns 0 if the statement executed with success or the error number if it failed.

WAM adds two stored procedures to improve this mechanism: SetError and SetMacro. Working together with @@ERROR this allows the user to return more information about the error that occurred inside a trigger or stored procedure to the ASP.

SetError raises an error indicating the “error name” in WAM_PRESENTATION. SetMacro indicates a placeholder and its correspondent value for the last error raised using SetError, so that each placeholder in the error description can be replaced by a parametrized value. Here's an example:

```
if SomeErrorCondition
begin
  exec SetError 'MY_SQL_ERROR_NAME'
  exec SetMacro 'X', SomeSQLExpressionToEnrichTheErrorMessage
  exec SetMacro 'Y', AnotherSQLExpression
end
```

All WAM objects in the ASP layer handle SQL layer errors automaticaly, using the facilities in the next section.

6.1.1 Raising and handling warnings in the SQL layer

Similar to the above error handling procedures, the above cause a modal dialog box to appear:

- exec dbo.SetWarning '_WARNING_CODE_HERE_'

- exec dbo.SetWMacro 'M', '_MACRO_VALUE_HERE_'

Several warnings can be shown along the same execution path; WAM keeps track of the user confirmations; the SQL code is executed each time, but is only committed when all warnings are confirmed (ignored) by the user.

6.2 Raising and handling errors in the ASP (web server) layer

WAM includes the object WAMError to help the programmer dealing with errors inside an ASP. You can use this object to raise your own errors inside an ASP, or to handle errors returned by the connection after executing a SQL statement - either from the engine or from raising by WAM's own SetError and SetMacro stored procedures.

As in SQL layer handling, errors are assumed to have a correspondent entry in table WAM_PRESENTATION, and optionally placeholders to enrich the message. So in order to show errors with the correct message it is necessary to fetch it from database.

Here's a typical code snippet example to handle database server errors:

```
// Server-side JavaSript:
try{
  WAMAPI.AppConn.execute(MySQLstatement);
}
catch(e){
  //Collect errors generated in stored procedure
  WAMAPI.Error.clear(); // clear the error collection
  WAMAPI.Error.catchErrors(WAMAPI.AppConn); // fetch messages and context values from db
}
…
if (WAMAPI.Error.hasErrors()) {
  // might do something if there were errors in the current collection
}
```

And here's a code snippet to raise an error in the ASP:

```
// Server-side JavaSript:
if (SomeErrorCondition){
  WAMAPI.Error.SetError('MY_ASP_ERROR_NAME')
  WAMAPI.Error.SetMacro('A', 'Some meaningful value for message placeholder @A')
  WAMAPI.Error.SetFocusTo('SomeHTMLelement'); // string with client JavaScript expression
  WAMAPI.Error.catchErrors(); // fetch messages and context values from db
}
```

In both cases errors caught in the web server layer are automatically displayed in a modal dialog, when the page is loaded in the web client browser.

(See WAMError.asp for more information)

6.3 Raising and handling errors in the web client layer

Similarly to the web server layer there's also an error object, always present in pages generated with WAM. With this you can raise errors, having their messages and placeholder values fetched from WAM_PRESENTATION. Here's an example:

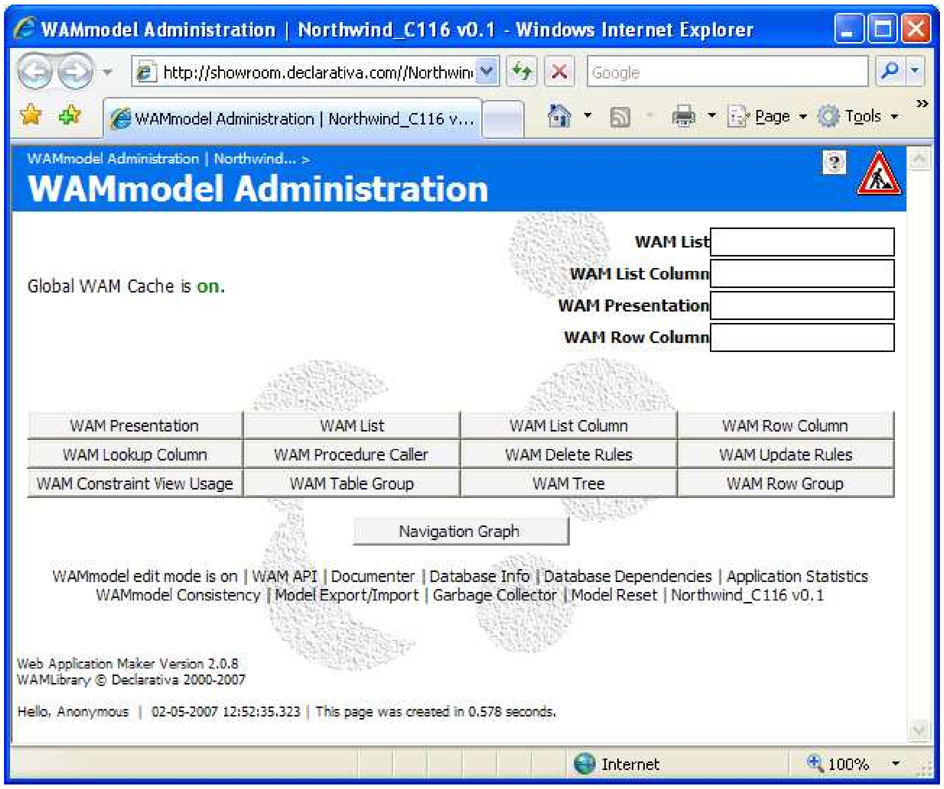
```
// Client(browser)-side JavaSript:
if (SomeErrorCondition){
  Error.SetError('MY_BROWSER_ERROR_NAME')
  Error.SetMacro('A', 'Some meaningful value for message placeholder @A')
  Error.SetFocusTo(SomeHTMLElement) // such as a form field
  Error.catchErrors(); // fetch messages and context values from db
}
```

An IFRAME is used to access the database within `catchErrors`

(See WAMError.js for more information)

7 WAMAdmin

WAM provides a WAMmodel administration utility, built with WAM itself:


WAMmodel Administration | Northwind_C116 v0.1 - Windows Internet Explorer
http://showroom.declarativa.com//Northwin
Google
WAMmodel Administration | Northwind_C116 v...
Page
Tools
WAMmodel Administration | Northwind... >
WAMmodel Administration
WAM List
WAM List Column
WAM Presentation
WAM Row Column
WAM Presentation
WAM List
WAM List Column
WAM Row Column
WAM Lookup Column
WAM Procedure Caller
WAM Delete Rules
WAM Update Rules
WAM Constraint View Usage
WAM Table Group
WAM Tree
WAM Row Group
Navigation Graph
WAMmodel edit mode is on | WAM API | Documenter | Database Info | Database Dependencies | Application Statistics
WAMmodel Consistency | Model Export/Import | Garbage Collector | Model Reset | Northwind_C116 v0.1
Internet
100%

Global WAM Cache is on.

Web Application Maker Version 2.0.8
WAMLibrary © Declarativa 2000-2007

Hello, Anonymous | 02-05-2007 12:52:35.323 | This page was created in 0.578 seconds.

7.1 *Browser-based WAMmodel editing*

Edition of any WAM tables can be done with an SQL client, such as Microsoft's Enterprise Manager, Visual Interdev or other tools.

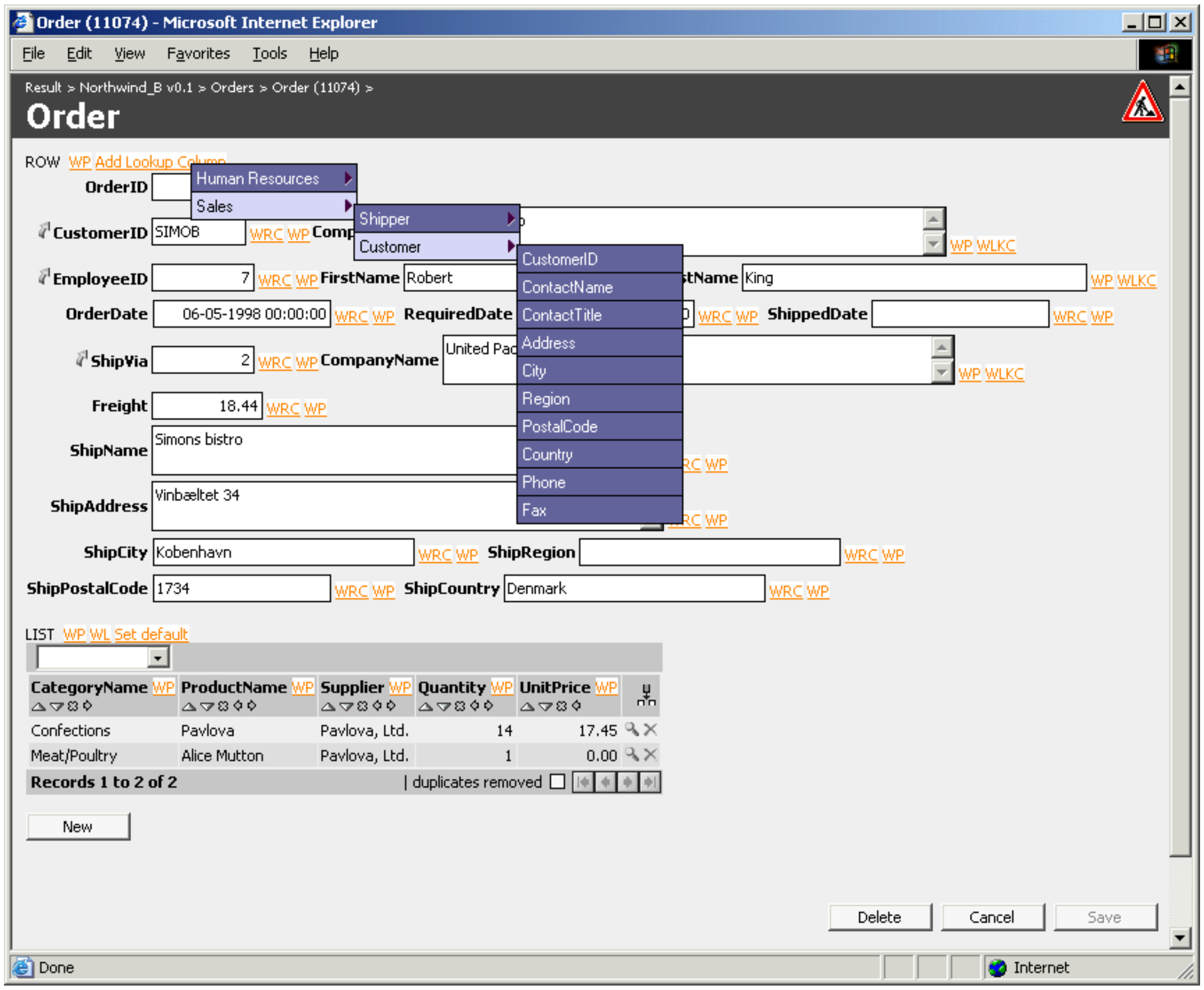
But there are two direct WAM-based ways to perform editing:

- Application **lists** and **search filters** allow normal (authorized) users to edit the related WAMmodel
- **WAMAdmin**

WAMAdmin is a WAM-based (hence browser based) front-end to all WAMmodel tables, and is installed together with the WAMLibrary; it is invoked with the URL `ApplicationRoot/WamLibrary/Admin`. In production environments the Admin directory should have access limited to administrators/developers.

By setting the “WAMmodel edit mode” to ON (bottom left of screen), the whole application transmutes into “edit mode”, displaying an “under construction sign” and showing orange links to relevant WAMmodel records, and allowing interactive picking of lookup columns. This mode is turned on by default when the WAMAdmin entry page is displayed the first time, and can be turned off by toggling “WAMmodel edit mode”,

For example the following is a Northwind “Order” record shown in WAMAdmin edit mode while the developer is choosing additional related columns to add to WAM_LOOKUP_COLUMN:



Result > Northwind_B v0.1 > Orders > Order (11074) >

Order

ROW WP Add Lookup Column

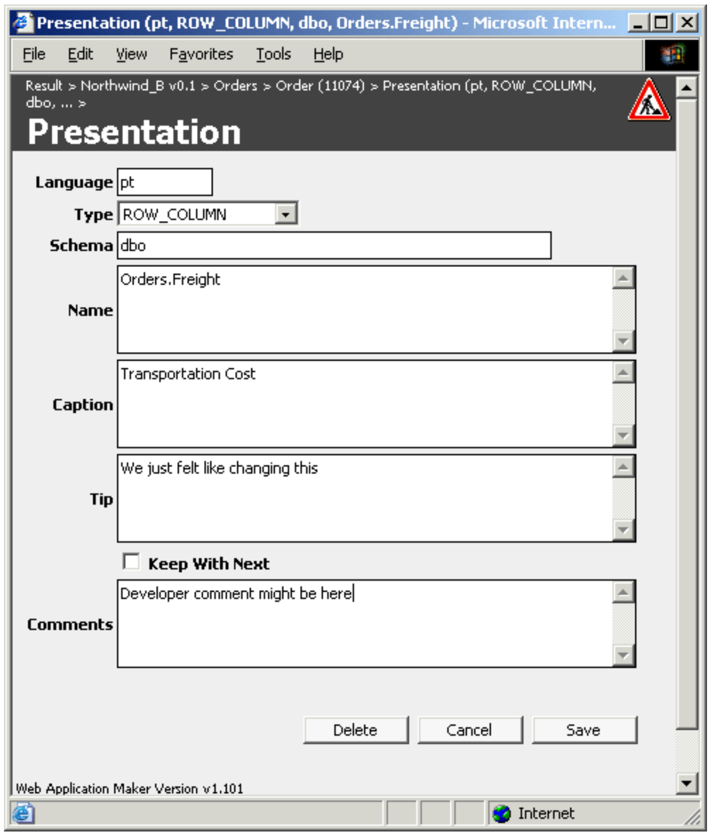
OrderID					
CustomerID	SIMOB	WRC WP	**Comp**		WP WLKC
EmployeeID	7	WRC WP	**FirstName**	Robert	**tName** King WP WLKC
OrderDate	06-05-1998 00:00:00	WRC WP	**RequiredDate**	WRC WP	**ShippedDate** WRC WP
ShipVia	2	WRC WP	**CompanyName**	United Pac	WP WLKC
Freight	18.44	WRC WP			
ShipName	Simons bistro	RC WP			
ShipAddress	Vinbæltet 34	RC WP			
ShipCity	Kobenhavn	WRC WP	**ShipRegion**		WRC WP
ShipPostalCode	1734	WRC WP	**ShipCountry**	Denmark	WRC WP

LIST WP WL Set default

CategoryName WP	ProductName WP	Supplier WP	Quantity WP	UnitPrice WP
Confections	Pavlova	Pavlova, Ltd.	14	17.45
Meat/Poultry	Alice Mutton	Pavlova, Ltd.	1	0.00

Records 1 to 2 of 2 | duplicates removed

Following the orange links opens the WAMmodel tuple for edition with WAMAdmin; for example, to change the “Freight” caption the developer would follow the “WP” link and change the (meta) data in the following page:


Presentation (pt, ROW_COLUMN, dbo, Orders.Freight) - Microsoft Intern...

Result > Northwind_B v0.1 > Orders > Order (11074) > Presentation (pt, ROW_COLUMN, dbo, ... >

Presentation

Language	pt
Type	ROW_COLUMN
Schema	dbo
Name	Orders.Freight
Caption	Transportation Cost
Tip	We just felt like changing this
	☐ Keep With Next
Comments	Developer comment might be here

Web Application Maker Version v1.101

The following WAMmodel edition links are made available in edit mode:

Link	Purpose
WP	Edit (possibly creating) WAM_PRESENTATION record
WRC	Edit (possibly creating) WAM_ROW_COLUMN record
WLKC	Edit WAM_LOOKUP_COLUMN record.
Add Lookup Column	Add a new WAM_LOOKUP_COLUMN defining a lookup on the present row, using a hierarchical menu similar to that appearing during end user list join building
WL	Edit WAM_LIST record
Set default	Make the current list configuration for the current user define what the 'dbo' user list should look like (and hence what all users will get), in terms of WAM_LIST_COLUMNs and WAM_CRITERION. The rationale is for the developer to configure the list as an end user would, and then click this link.
WPC	Edit WAM_PROCEDURE_CALL record

In addition to lists and rows, WAM table group objects (e.g. in the default.asp page) also feature some WAMmodel edition links.

7.2 WAM API Reference

The WAM objects, both client and server, are documented in the WAM API Reference, automatically generated invoking URL `ApplicationRoot/WamLibrary/Admin/api.asp,` or following the **WAM API** link in the **WAMAdmin** entry page. For each object the following information is supplied: constructor, methods, properties and collections.

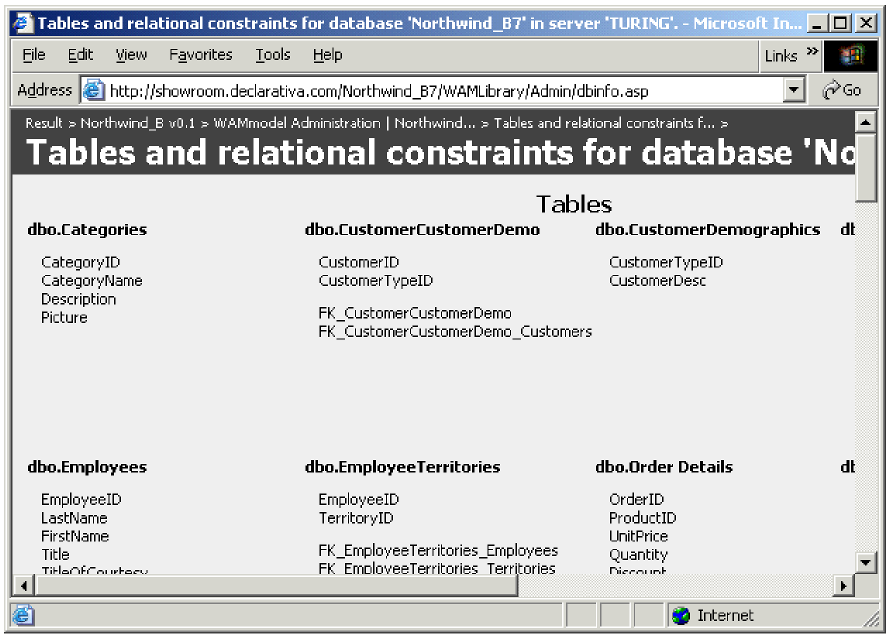
7.3 Developer Documentation

WAM provides a tool, intended to be used by developers, which shows, in a form of a HTML document, an enumeration of the lists and rows created for each table or view used by the application, and a brief description for each one (see column comments in the WAM_PRESENTATION table). The document also represents the relation between rows and lists as well as their properties, such as columns, captions, etc.

The document can be generated invoking the following URL: `ApplicationRoot/WamLibrary/Admin/documenter.asp,` or following the **Documenter** link in the **WAMAdmin** entry page. Adding the parameter ?wdschema=X restricts documentation to objects in SQSL schema X.

7.4 Database structure Information

Sometimes it may be convenient to introspect the database structure through the browser, for example when a direct connection to the database server is not possible. The following page (partially shown below) displays all tables, columns, foreign keys and stored procedures:


Tables and relational constraints for database 'Northwind_B7' in server 'TURING'. - Microsoft In...
File
Edit
View
Favorites
Tools
Help
Links
Address
http://showroom.declarativa.com/Northwind_B7/WAMLibrary/Admin/dbinfo.asp
Go
Internet

Result > Northwind_B v0.1 > WAMmodel Administration | Northwind... > Tables and relational constraints f... >

Tables and relational constraints for database 'No

Tables

dbo.Categories

CategoryID
CategoryName
Description
Picture

dbo.CustomerCustomerDemo

CustomerID
CustomerTypeID

FK_CustomerCustomerDemo
FK_CustomerCustomerDemo_Customers

dbo.CustomerDemographics

CustomerTypeID
CustomerDesc

dbo.Employees

EmployeeID
LastName
FirstName
Title
TitleOfCourtesy

dbo.EmployeeTerritories

EmployeeID
TerritoryID

FK_EmployeeTerritories_Employees
FK_EmployeeTerritories_Territories

dbo.Order Details

OrderID
ProductID
UnitPrice
Quantity
Discount

It can can be generated invoking the following URL: `ApplicationRoot/WamLibrary/Admin/dbinfo.asp,` or following the **Database Info** link in the **WAMAdmin** entry page.

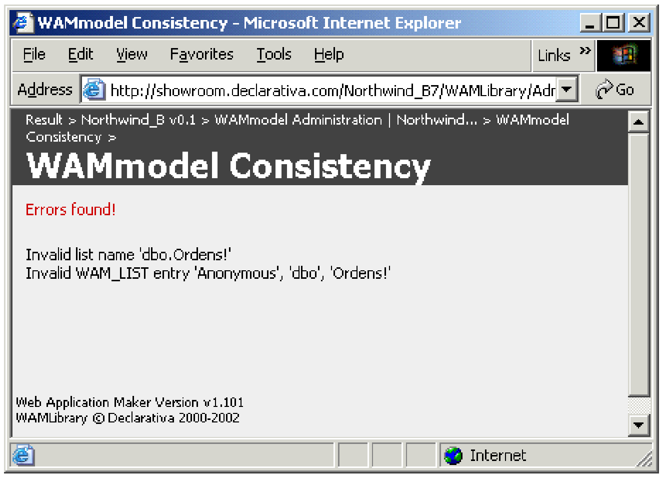
7.5 WAMmodel consistency with database structure

WAMmodel tables refer database objects, such as tables, views, foreign key paths, stored procedures, etc. In order for the WAMmodel to be correct, it is necessary that all referred objects exist.

WAMmodel consistency triggers, defined in the WAMmodel tables, normally enforce this. If for any reason it is necessary to disable these triggers (say because you're doing some bulk loading or multi-step processing on WAMmodel tables), disable them with the "Set WAMmodel consistency OFF", and re-enable them later with "Set WAM consistency ON", on WAMAdmin's entry page.

Independently of the incremental consistency checking on WAMmodel's side by its triggers being active or not, on the other side the SQL database schema may change unannounced.

To deal with all these possibilities WAMAdmin includes a WAMmodel consistency checker, callable by following the "Wammodel Consistency" link in the WAMAdmin entry page. For example, the following is the result after a record in WAM_PRESENTATION for the Northwind example was changed erroneously to "Ordens" instead of "Orders":


WAMmodel Consistency - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links
Address http://showroom.declarativa.com/Northwind_B7/WAMLibrary/Adr
Go
Result > Northwind_B v0.1 > WAMmodel Administration | Northwind... > WAMmodel Consistency >

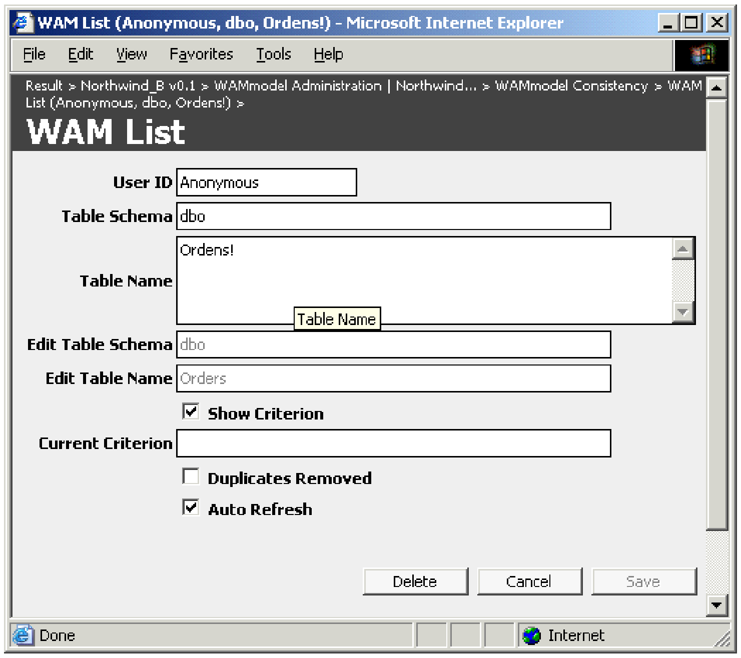
WAMmodel Consistency

Errors found!

Invalid list name 'dbo.Ordens!'
Invalid WAM_LIST entry 'Anonymous', 'dbo', 'Ordens!'

Web Application Maker Version v1.101
WAMLibrary © Declarativa 2000-2002

Each error message provides a link to the offending WAMmodel record. Following the first link above leads the developer to the following WAMAdmin page, to fix the problem:


WAM List (Anonymous, dbo, Ordens!) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Result > Northwind_B v0.1 > WAMmodel Administration | Northwind... > WAMmodel Consistency > WAM List (Anonymous, dbo, Ordens!) >

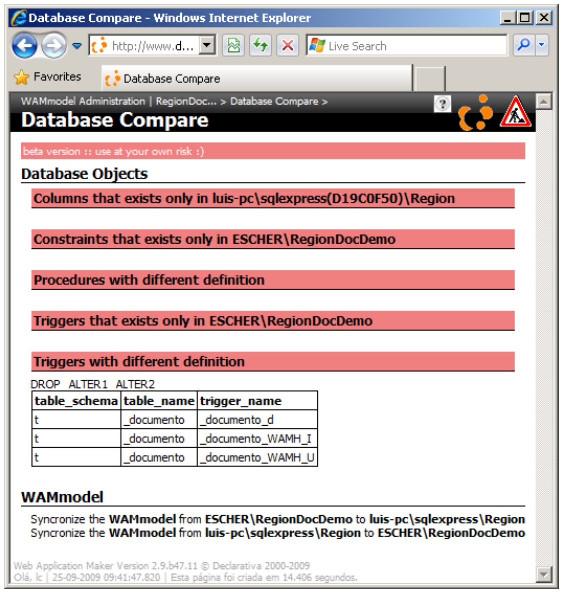
WAM List

User ID	Anonymous
Table Schema	dbo
Table Name	Ordens!
Edit Table Schema	dbo
Edit Table Name	Orders
	☑ Show Criterion
Current Criterion	
	☐ Duplicates Removed
	☑ Auto Refresh

Table Name
Delete
Cancel
Save
Done
Internet

7.6 Database Compare (dbCompare)

WAMAdmin provides “database compare", a tool to simplify the synchronization between two databases. This tool, intended to be used by developers, compares WAMmodel and database schemas, and allows the developer to make specific changes from one side to the other.


Database Compare - Windows Internet Explorer
http://www.d...
Live Search
Favorites
Database Compare
WAMmodel Administration | RegionDoc... > Database Compare >

Database Compare

beta version :: use at your own risk :)

Database Objects

Columns that exists only in luis-pc\sqlexpress(D19C0F50)\Region

Constraints that exists only in ESCHER\RegionDocDemo

Procedures with different definition

Triggers that exists only in ESCHER\RegionDocDemo

Triggers with different definition

DROP ALTER1 ALTER2

table_schema	table_name	trigger_name
t	_documento	_documento_d
t	_documento	_documento_WAMH_I
t	_documento	_documento_WAMH_U

WAMmodel

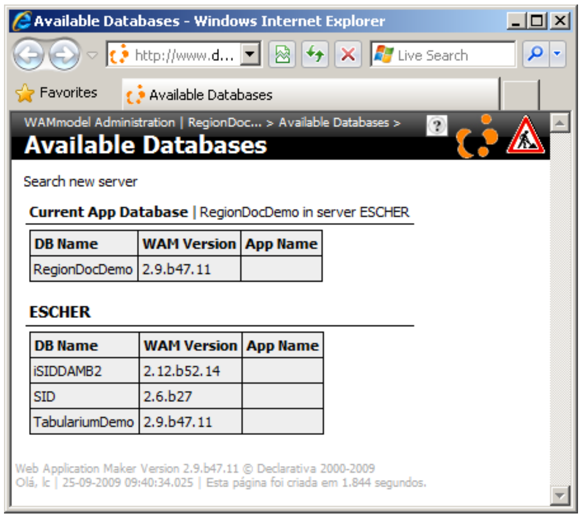
Syncronize the **WAMmodel** from **ESCHER\RegionDocDemo** to **luis-pc\sqlexpress\Region**
Syncronize the **WAMmodel** from **luis-pc\sqlexpress\Region** to **ESCHER\RegionDocDemo**

Web Application Maker Version 2.9.b47.11 © Declarativa 2000-2009
Olá, lc | 25-09-2009 09:41:47.820 | Esta página foi criada em 14.406 segundos.

7.7 Available databases (dbInspector)

WAMAdmin provides “available databases”, a tool to know all databases present on a server, as well as WAM version and application name.

By default, this tool lists all databases present in the current application server, but other servers can be accessed.


Available Databases - Windows Internet Explorer
http://www.d...
Live Search
Favorites
Available Databases
WAMmodel Administration | RegionDoc... > Available Databases >

Available Databases

Search new server

Current App Database | RegionDocDemo in server ESCHER

DB Name	WAM Version	App Name
RegionDocDemo	2.9.b47.11	

ESCHER

DB Name	WAM Version	App Name
iSIDDAMB2	2.12.b52.14	
SID	2.6.b27	
TabulariumDemo	2.9.b47.11	

Web Application Maker Version 2.9.b47.11 © Declarativa 2000-2009
Olá, lc | 25-09-2009 09:40:34.025 | Esta página foi criada em 1.844 segundos.

7.8 *** Other WAMAdmin features

- Navigation graph generation (obsolete)
- Export/Import
- DB dependencies: shows ASPs with database object dependencies in their source code
- Global cache control

8 WAM installation and application setup

Two major steps are involved: setting up the WAMmodel on the database and configuring the web server application. The following assumes familiarity with the relevant Microsoft tools.

8.1 Using the installer

Just double-click the setup.wsf file.

At the end you should follow the suggestion and review the WAMLibrary/Admin file permissions, to forbid arbitrary WAMmodel edition by users through WAMAdmin. Add WAP MIME types if WAP access is desired.

8.2 Manual installation

8.2.1 WAMmodel setup

To create the WAMmodel execute two scripts, `WAMCascader.sql` that creates stored procedures and tables for trigger cascading, and `WAMmodel.sql` that creates tables and stored procedures for the WAMmodel. Afterwards create a default WAMmodel for the database, by executing the following script sequence: `fill_WAMList.sql`, `fill_WAMListColumn.sql`, `fill_WAMList_DetailLists.sql`, `fill_WAMLookupColumn.sql` and `fill_WAMPresentation.sql`.

Finally add some generic WAM records into tables WAM_LIST, WAM_LIST_COLUMN and WAM_PRESENTATION, by importing the 3 .txt files (say) with the Enterprise Manager Import wizard or with Query Analyser:

```
BULK INSERT WAM_PRESENTATION FROM 'MyPath\WAM_PRESENTATION.txt'
BULK INSERT WAM_LIST FROM 'MyPath\WAM_LIST.txt'
BULK INSERT WAM_LIST_COLUMN FROM 'MyPath\WAM_LIST_COLUMN.txt'
```

(see WAMmodel.zip)

8.2.2 Web application setup

- Create a web application in Internet Information Server (4.0 or 5.0), in a new virtual directory; you must enforce the following properties:

```
Directory > Application Settings > Configuration > App Options
       [x] Enable buffering (checked)
       [x] Enable parent paths (checked)
       Default ASP language = JavaScript
```

- Copy the WAMLibrary files from WAMLibrary.zip to a directory named WAMLibrary situated at the root of the web application.
- Prepare a global.asa file (see section 5.4.10 above)

The WAM application front-end can be invoked by http://APPLICATION_URL/WAMLibrary/Interface, and the WAMmodel administration facility by http://APPLICATION_URL/WAMLibrary/Admin.

8.2.3 Removing WAM

In order to remove all WAMmodel tables and WAM stored procedures etc. from a database, you can call

```
  WAMuninstall (selfdestroy)
```

If the argument is 1, the procedure will destroy itself too.

8.3 Browser limitations: Firefox and Safari

WAM's reference browser is Internet Explorer. Firefox and Safari are also supported, with some minor limitations:

- Selection of finder fields to be used by the finder is not available.
- Shortcut to copy URL is not available.
- Multiple selection option is not available.

9 Misc

9.1 Hands-on workshop template

Following is a possible structure for an introductory hands-on WAM session, for half to a full day depending on the audience's proficiency with (specially) SQL Server and ASPs. Most items refer relevant manual sections.

9.1.1 Basics

- The existing software infrastructure that WAM will add to: SQL Server, SQL Management Studio, database restore; IIS, web sites, ASPs, global.asa, your favorite text editor
- Manual copying/installing of an existing WAM application (NorthwindC): copy folder, share as web directory F, edit global.asa with database name and user credentials, access http://localhost/F
- WAM application user interface overview (section 2.)
- First part of section 2.8

9.1.2 Installing WAM and simple WAMmodel tweaking

- Install WAM over existing Northwind DB (3.1);
- Compare new app with NorthwindC
- Second part of 2.8
- Improving the Orders row (3.3.1, 3.3.2)
  - Adding a field title, the hard way
  - WAMAdmin: adding another title, disable a field, keep with next
  - Grouping some elements in the row (4.1.8)
- Grouping some tables (4.1.8)
- My list is the best for all
- WAMmodel tables overview (4.1)
- A button to an external ASP (3.5.5, 5.4.6)

9.1.3 FK path wonders

- FK graphs and paths (2.9, 2.10, 2.11)
- Adding a lookup to the Orders row, the hard way (4.1.3)
- Adding lots of lookups
- To be embedded or to stand alone, that is the question (3.3.5)
- Deeper lists (3.3.4):
  - Orders handled in a territory
  - Customers buying stuff from a supplier
- Grouping row elements in Supplier (4.1.8)

9.1.4 Better databases

- Back to the development process (2.8)
- Nicer error messages (3.4.4, 6.1)
  - please don't kill the customer
- Dates without time (4.4.1)
- Storing files "in" database records: wamexternalfile (4.4.7)
- Adding a CustomerType table
  - A colored field (4.4.9)
  - Relating it to Customers
  - More lookups in Customers, Orders
- About CHECKing constraints: adding sex to Employees (3.4.2)
- WAMmodel vs. database (meta)editing
  - Inconsistency problems (7.5)
  - WAMAdmin reviewed
- To probe further: do your GoogleMap with wamgmlatlng (4.4.11)

9.1.5 Finally, some web application code!

- So far, WAM's default.asp and standard.asp (5.1)
- A better entry page: adding a finder for Customers (5.4.10)
- About custom rows and lists (5.3.1, 5.4)
  - The Orders row in NorthwindC (5.4.2.3)

9.2 WAM model-customization continuum and project methodology

As could be seen in "WAM development tour", WAM allows a balanced use of modeling and coding:

- Modeling alone allows the application to have significant GUI functionality, cf. 3.3 above
- An application with no scripting outside the database server will use the WAM-provided standard ASPs, and will reflect improvements to the database and/or the WAMmodel, see for example "Add fields to a table" and "Add a trigger with parameterized error message"
- ASP scripting using WAM objects should delegate as much as possible GUI generation, to make code less brittle to database changes; see for example "Customize a row page" and "Add a browser client script"
- Scripting outside of WAM-generated pages can be invoked with WAMmodel specified links, see for example "Add an external ASP"

There are other important aspects not shown, namely the impact of user permissions on the multiple GUI profiles, obtained automatically by WAM at runtime.

So WAM-based development does not fall in the extreme scenarios (1) "use the model to generate the first version and then forget it because we need to change the code!" or (2) "use an enormous model so we can always parameterize rather than code!" Instead WAM supports a hybrid development style, fulfilling a continuum from one to the other extreme: the *model-customization continuum*, made possible by exposing to the developer the GUI generation facilities at runtime. This of course marries very well with the dynamic nature of the web's GUI.

WAM is methodology neutral; it mainly speeds up the web GUI construction, and may be integrated into existing project approaches. And it doesn't introduce significant dependencies – future incarnations of the developed system may reuse the investment in the SQL layer (database structure, user roles, triggers, …) and replace the WAMLibrary by something else, even reusing all linguistic and other declarative information from the WAMmodel.

In our customer projects with WAM we've been following approximately the following project steps:

1. Team Setup
    - 1.1. Tool and workstation configuration
    - 1.2. MS SQL Server, Active Server Pages / JavaScript, and Visual InterDev training
    - 1.3. WAM training
2. Put together and discuss informal requirements report
3. First database structure, including full data import for testing and validation
4. Development of most general web interface
    - 4.1. Get first WAM interface (automatic)
    - 4.2. Tune WAMmodel – table groups, captions, lists, lookups
    - 4.3. Development of external ASPs for customized documents, reports etc.
    - 4.4. Development of automatic communication templates
5. Data validations, procedural operations
    - 5.1. Triggers and check constraints
    - 5.2. Stored procedures
    - 5.3. Other external ASPs (beyond documents and reports)
    - 5.4. Conditions for automatic communication agent
6. More specific web interfaces
    - 6.1. Definition of SQL users and roles,
    - 6.2. Development of customized WAM web pages for more intense or frequent user interactions
    - 6.3. Permission and additional VIEW coding, automatically conditioning the web interface
    - 6.4. Validation of the multiple interfaces
7. Pre-flight
    - 7.1. Customer server installation: data, IIS, automatic communicator, printing agent
    - 7.2. User demos and early testing
    - 7.3. Full data import
    - 7.4. Use in parallel with old system
    - 7.5. Postmortem review, fixing
8. Production
    - 8.1. Final data import
    - 8.2. Running it
    - 8.3. Basic user training and baby-sitting
    - 8.4. Documentation
    - 8.5. Stabilizing

The above steps mention automatic communication and (web page) printing Declarativa agents, which are currently not part of WAM.

9.3 Comparison with other tools

Note: This section is outdated, please check http://www.declarativa.com/wam

This section attempts to position WAM regarding other tools. For a good overview of existing related tools see [Fraternali 99].

We're unaware of GUI generators pursuing such a close integration between their input (GUI specifications) and the back-end database applications they serve; to our knowledge major commercial tools require human form/web page code revision if as little as a new single field is added to the database. Most provide (development time) wizards to generate code that can later be customized to a certain extent, but which is fragile against database schema changes.

Such is the case for example for Macromedia Cold Fusion, Oracle Forms Services and Microsoft Access. ASP-DB [MMS 2001], a product whose purpose intersects WAM's, has the same problem, lacking any connection to the database engine meta-model. The same applies for "single-shot" code wizards such as Microsoft Visual InterDev's

Most research in the area proposes the use of higher-level conceptual models, as basis for application generation, in complement or preceding the relational model. Such is the case for Araneus [U.Roma 2001], Autoweb [Fraternali et al. 2001], and Strudel [AT&T 2001], see [Griffiths et al 1998] [Silva 2000] for surveys. Given our lack of resources and focus on quickly developing practical solutions for customers it seemed preferable to embrace a pragmatic lower level model, closer to industry practice. Although our WAMmodel defines a few tables that must be filled "by hand", its core is simply the SQL DBMS own built-in Meta structure, always present and which may even be the output of higher level modeling processes, independently of WAM.

The software community has lately shown concern with "cross-dependencies", practical manifestations of higher-order concepts in the programmer's mind that are not properly captured into available programming abstractions such as functions, objects or rules. This is one of the main motivations for both Aspect Oriented Programming [Xerox 2001] and Intentional Programming [Microsoft 2001b]. WAM's pragmatic implementation of model-based web programming addresses a particular class of such cross-dependencies, namely those stemming from the multi layered nature of web program executors and the innate web structure dependencies from database models. And in fact we found thinking similar to ours in the description of a proposed application for Intentional Programming, cf. the "Scribble" example in [Simonyi 2001]. You can take WAM as providing a "web GUI front-end defining intention" for Transact-SQL programmers.

9.4 ***Performance considerations

Recap expected admissible server load, refer WAMLoadBalancer.

Refer WAM_AUDIT_LOG and how to use it for performance

9.5 Help system

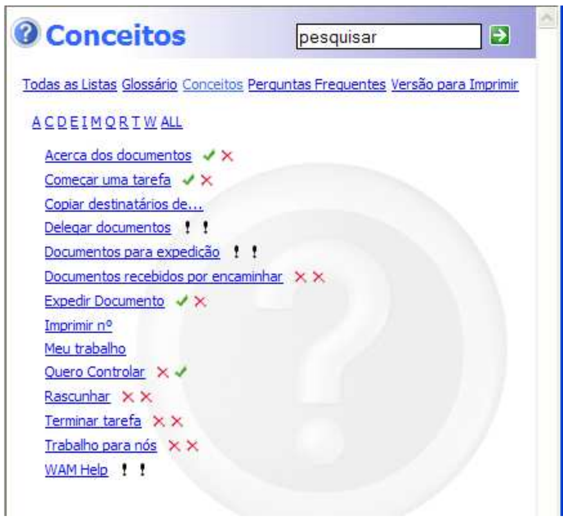
All WAM-generated pages provide online help.

9.5.1 User perspective

Tooltips are shown directy on the application page, whenever the mouse cursor flies over a field/column.

The help panel is available by pressing F1 or clicking the ? icon. In addition to navigation links and a search field, it displays 4 main kinds of help content, exemplified next:

- Concepts ("Conceitos") necessary to understand the application
- Frequently asked questions ("Perguntas"), task-oriented
- Descriptive help about all lists in the application, as well as about each column/field on a row;
- Glossary ("Glossário")

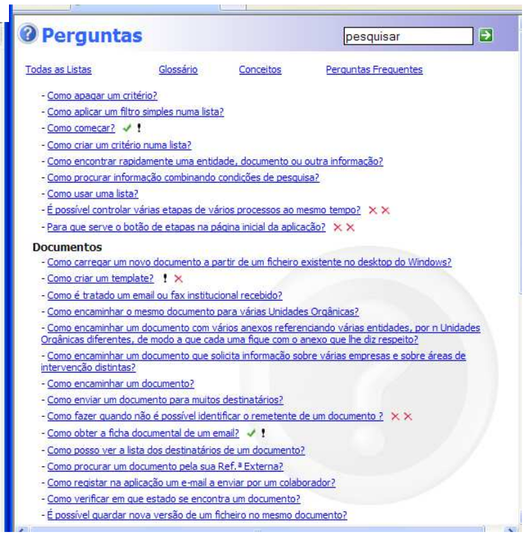


Conceitos pesquisar

Todas as Listas Glossário Conceitos Perguntas Frequentes Versão para Imprimir

A C D E I M Q R T W ALL

- Acerca dos documentos ✓ ✗
- Começar uma tarefa ✓ ✗
- Copiar destinatários de…
- Delegar documentos ! !
- Documentos para expedição ! !
- Documentos recebidos por encaminhar ✗ ✗
- Expedir Documento ✓ ✗
- Imprimir nº
- Meu trabalho
- Quero Controlar ✗ ✓
- Rascunhar ✗ ✗
- Terminar tarefa ✗ ✗
- Trabalho para nós ✗ ✗
- WAM Help ! !



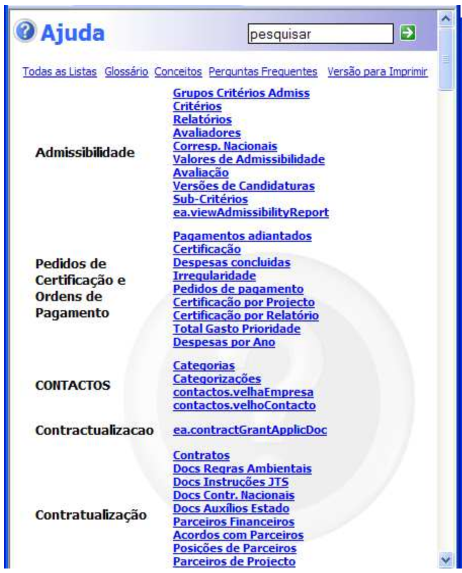
Perguntas pesquisar

Todas as Listas Glossário Conceitos Perguntas Frequentes

- Como apagar um critério?
- Como aplicar um filtro simples numa lista?
- Como começar? ✓ !
- Como criar um critério numa lista?
- Como encontrar rapidamente uma entidade, documento ou outra informação?
- Como procurar informação combinando condições de pesquisa?
- Como usar uma lista?
- É possível controlar várias etapas de vários processos ao mesmo tempo? ✗ ✗
- Para que serve o botão de etapas na página inicial da aplicação? ✗ ✗

Documentos

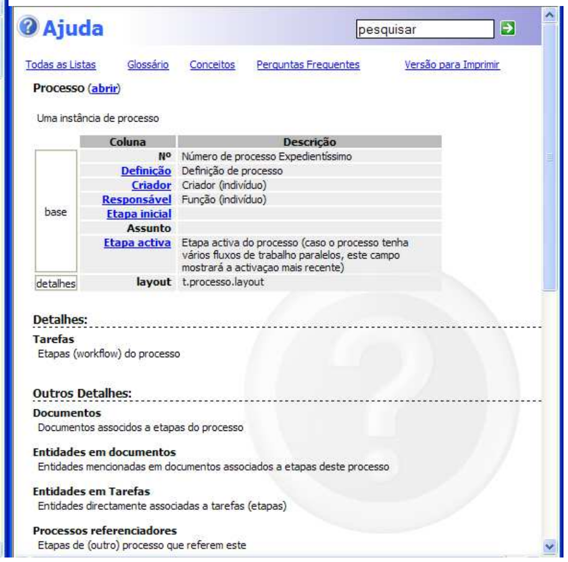
- Como carregar um novo documento a partir de um ficheiro existente no desktop do Windows?
- Como criar um template? ! ✗
- Como é tratado um email ou fax institucional recebido?
- Como encaminhar o mesmo documento para várias Unidades Orgânicas?
- Como encaminhar um documento com vários anexos referenciando várias entidades, por n Unidades Orgânicas diferentes, de modo a que cada uma fique com o anexo que lhe diz respeito?
- Como encaminhar um documento que solicita informação sobre várias empresas e sobre áreas de intervenção distintas?
- Como encaminhar um documento?
- Como enviar um documento para muitos destinatários?
- Como fazer quando não é possível identificar o remetente de um documento ? ✗ ✗
- Como obter a ficha documental de um email? ✓ !
- Como posso ver a lista dos destinatários de um documento?
- Como procurar um documento pela sua Ref.ª Externa?
- Como registar na aplicação um e-mail a enviar por um colaborador?
- Como verificar em que estado se encontra um documento?
- É possível guardar nova versão de um ficheiro no mesmo documento?



Ajuda pesquisar

Todas as Listas Glossário Conceitos Perguntas Frequentes Versão para Imprimir

Admissibilidade	Grupos Critérios Admiss Critérios Relatórios Avaliadores Corresp. Nacionais Valores de Admissibilidade Avaliação Versões de Candidaturas Sub-Critérios ea.viewAdmissibilityReport
Pedidos de Certificação e Ordens de Pagamento	Pagamentos adiantados Certificação Despesas concluídas Irregularidade Pedidos de pagamento Certificação por Projecto Certificação por Relatório Total Gasto Prioridade Despesas por Ano
CONTACTOS	Categorias Categorizações contactos.velhaEmpresa contactos.velhoContacto
Contractualizacao	ea.contractGrantApplicDoc
Contratualização	Contratos Docs Regras Ambientais Docs Instruções JTS Docs Contr. Nacionais Docs Auxílios Estado Parceiros Financeiros Acordos com Parceiros Posições de Parceiros Parceiros de Projecto



Ajuda pesquisar

Todas as Listas Glossário Conceitos Perguntas Frequentes Versão para Imprimir

Processo (abrir)

Uma instância de processo

	Coluna	Descrição
base	**Nº**	Número de processo Expedientíssimo
	Definição	Definição de processo
	Criador	Criador (indivíduo)
	Responsável	Função (indivíduo)
	Etapa inicial	
	Assunto	
	Etapa activa	Etapa activa do processo (caso o processo tenha vários fluxos de trabalho paralelos, este campo mostrará a activaçao mais recente)
detalhes	**layout**	t.processo.layout

Detalhes:

Tarefas
Etapas (workflow) do processo

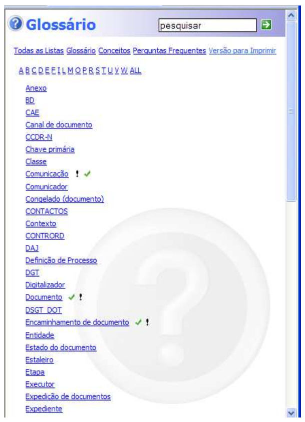
Outros Detalhes:

Documentos
Documentos associdos a etapas do processo

Entidades em documentos
Entidades mencionadas em documentos associados a etapas deste processo

Entidades em Tarefas
Entidades directamente associadas a tarefas (etapas)

Processos referenciadores
Etapas de (outro) processo que referem este



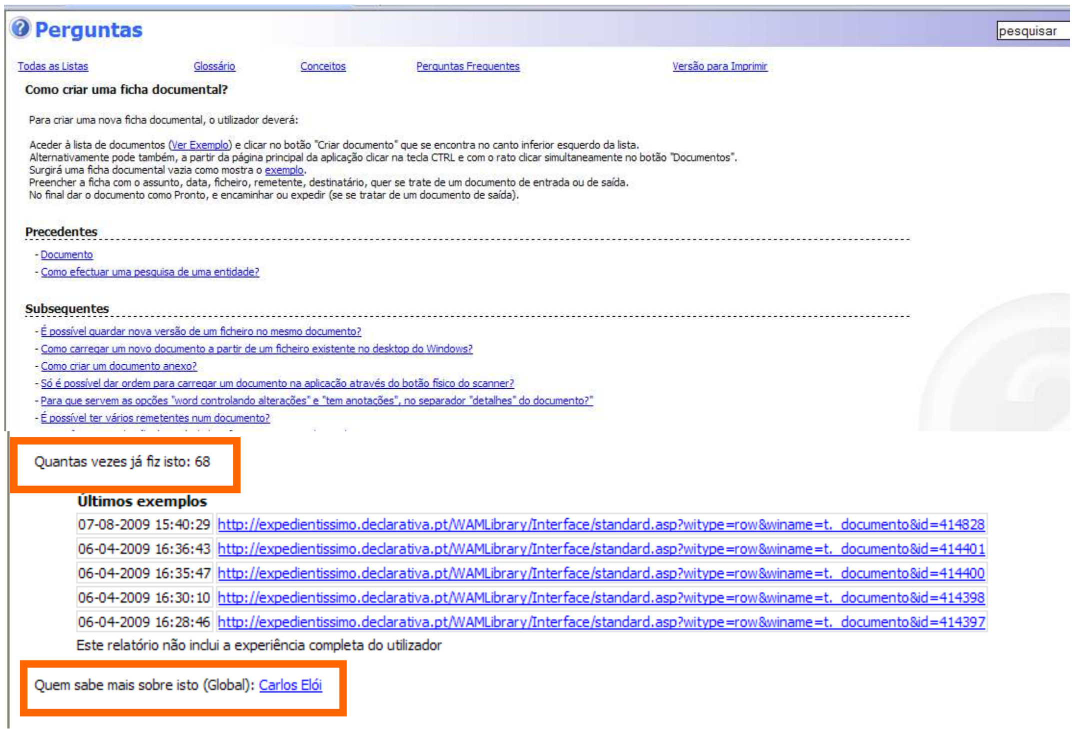
Glossário pesquisar

Todas as Listas Glossário Conceitos Perguntas Frequentes Versão para Imprimir

A B C D E F I L M O P R S T U V W ALL

- Anexo
- BD
- CAE
- Canal de documento
- CCDR-N
- Chave primária
- Classe
- Comunicação ! ✓
- Comunicador
- Congelado (documento)
- CONTACTOS
- Contexto
- CONTRORD
- DAJ
- Definição de Processo
- DGT
- Digitalizador
- Documento ✓ !
- DSGT_DOT
- Encaminhamento de documento ✓ !
- Entidade
- Estado do documento
- Estaleiro
- Etapa
- Executor
- Expedição de documentos
- Expediente

Frequently Asked Questions, being the source for more pragmatic, task-oriented help, are supported with additional information. Next is the help panel for "How to create a document", in a document management application:



Perguntas

pesquisar

Todas as Listas | Glossário | Conceitos | Perguntas Frequentes | Versão para Imprimir

Como criar uma ficha documental?

Para criar uma nova ficha documental, o utilizador deverá:

Aceder à lista de documentos (Ver Exemplo) e clicar no botão "Criar documento" que se encontra no canto inferior esquerdo da lista.
Alternativamente pode também, a partir da página principal da aplicação clicar na tecla CTRL e com o rato clicar simultaneamente no botão "Documentos".
Surgirá uma ficha documental vazia como mostra o exemplo.
Preencher a ficha com o assunto, data, ficheiro, remetente, destinatário, quer se trate de um documento de entrada ou de saída.
No final dar o documento como Pronto, e encaminhar ou expedir (se se tratar de um documento de saída).

Precedentes

- Documento
- Como efectuar uma pesquisa de uma entidade?

Subsequentes

- É possível guardar nova versão de um ficheiro no mesmo documento?
- Como carregar um novo documento a partir de um ficheiro existente no desktop do Windows?
- Como criar um documento anexo?
- Só é possível dar ordem para carregar um documento na aplicação através do botão fisico do scanner?
- Para que servem as opções "word controlando alterações" e "tem anotações", no separador "detalhes" do documento?"
- É possível ter vários remetentes num documento?

Quantas vezes já fiz isto: 68

Últimos exemplos

07-08-2009 15:40:29	http://expedientissimo.declarativa.pt/WAMLibrary/Interface/standard.asp?witype=row&winame=t._documento&id=414828
06-04-2009 16:36:43	http://expedientissimo.declarativa.pt/WAMLibrary/Interface/standard.asp?witype=row&winame=t._documento&id=414401
06-04-2009 16:35:47	http://expedientissimo.declarativa.pt/WAMLibrary/Interface/standard.asp?witype=row&winame=t._documento&id=414400
06-04-2009 16:30:10	http://expedientissimo.declarativa.pt/WAMLibrary/Interface/standard.asp?witype=row&winame=t._documento&id=414398
06-04-2009 16:28:46	http://expedientissimo.declarativa.pt/WAMLibrary/Interface/standard.asp?witype=row&winame=t._documento&id=414397

Este relatório não inclui a experiência completa do utilizador

Quem sabe mais sobre isto (Global): Carlos Elói

In addition to the answer to the question, the user is shown precedent and subsequent questions - questions whose answers the user should know before asking the present question, or which he may with to read, respectively. This order among help items defines the best learning paths for users.

Further more, since a question maps into an application operation (typically the invocation of a caller button, or saving or deleting or consulting something), the help panel tells the user how many times he has executed this operation… and provides links to review the last 5.

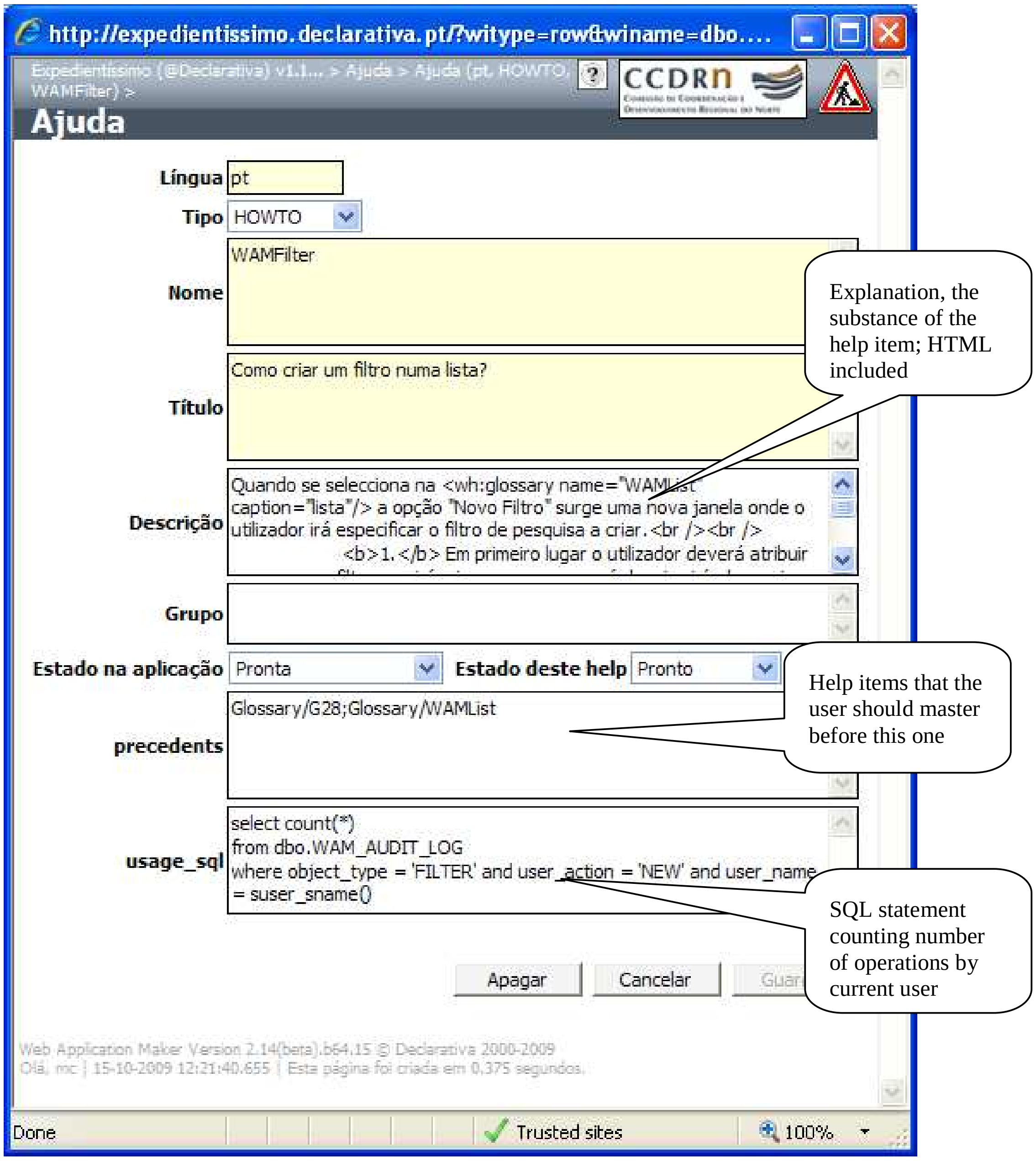
(***Note: this paragraph currently applies only to RegionDoc/Tabularium WAM applications***) The help panel also identifies the other user in the same organizational unit who executed the operation more times, suggesting him/her as a source for human help on this particular application feature.

9.5.2 Developer perspective: adding help content

Descriptive help about lists, rows and columns/fields is entered in WAM_PRESENTATION (captions and tooltips, which in adition to being shown to the user are used to generate the help panel).

But most online help material resides in WAM_HELP (see 4.1.12), which can be edited with WAMAdmin while running the application, so all changes propagate live to users.

The following figure shows a help ("Ajuda") item, one record in WAM_HELP:


http://expedientissimo.declarativa.pt/?witype=row&winame=dbo....

Expedientíssimo (@Declarativa) v1.1... > Ajuda > Ajuda (pt, HOWTO, WAMFilter) >

Ajuda

Língua	pt
Tipo	HOWTO
Nome	WAMFilter
Título	Como criar um filtro numa lista?
Descrição	Quando se selecciona na <wh:glossary name="WAMList" caption="lista"/> a opção "Novo Filtro" surge uma nova janela onde o utilizador irá especificar o filtro de pesquisa a criar.<br /><br /> <b>1.</b> Em primeiro lugar o utilizador deverá atribuir
Grupo	
Estado na aplicação	Pronta
Estado deste help	Pronto
precedents	Glossary/G28;Glossary/WAMList
usage_sql	select count(*) from dbo.WAM_AUDIT_LOG where object_type = 'FILTER' and user_action = 'NEW' and user_name = suser_sname()

Explanation, the substance of the help item; HTML included

Help items that the user should master before this one

SQL statement counting number of operations by current user

Apagar
Cancelar
Guar
Web Application Maker Version 2.14(beta).b64.15 © Declarativa 2000-2009
Olá, mc | 15-10-2009 12:21:40.655 | Esta página foi criada em 0.375 segundos.
Done
Trusted sites
100%

In addition to the language (“Língua”) of the item (thus one record is needed for each language supported by the application), the help designer specifies the type of the topic (if it is a “how to” topic, a term of glossary, a concept) he/she wants to create.

Then name, title, group and description of the new help topic. Help topics can be split in groups (groups of FAQ’s within the same thematic, for instance), so it is easier for users to find them. The name of the topic is an internal expression used by developers (acting as part of the table key). The title is the question that will appear to users. The description field is the answer.

'Precedents' (optionally) indicates help items whose reading by the user should precede the current one, because they provide knowledge required to understand the current one. It's a list of Help_group/HelpItemName, separated by semicolons.

9.5.2.1 WAM_HELP Description tags

To use (application data) examples or to refer other help items, it is necessary to insert specific tags in the “Description” field. Here is the list of those specific tags that can be used by developers when they are writing a description for a help item:

9.5.2.1.1 WH:ROW (<wh:row/>)

This tag is used to create a link to a row.
Use Cases:

<wh:row table="<schema.name>"/>	Open an empty row
<wh:row table="<schema.name>" key="<key>(,<key>...)"/>	Open a row with a specific primary key
<wh:row table="<schema.name>" defaults="<key>=<value>"/>	*WAMA* Rows

Optional Attributes:

caption="<string>" String which contains the link

9.5.2.1.2 WH:LIST (<wh:list/>)

Create a link to a list.
Use Cases:

<wh:list table="<schema.table>"/>	Open the defined list in *table* attribute
<wh:list table="<schema.table>" filterColumn="<schema.table.column>(,<schema.table.column>...)"> filterValue="<key>(,<key>...)"/>	Open the list *table* applying the filter *filterColumn=filterValue*
<wh:list fkPath="<esquema.fkPath>" key="<chave>" />	Open the detail list *fkPath* which contains the *key*

Optional Attributes:

caption="<string>" String which contains the link

9.5.2.1.3 WH:GLOSSARY / WH:CONCEPT / WH:HOWTO (<wh:glossary/> / <wh:concept/> / <wh:howto/>)

<wh:glossary name="<name_WAM_HELP>"/ >	Link to the glossary item defined by *name*
The same is applied to the other two cases	

Optional Attributes:

caption="<string>" String which contains the link

9.6 Charts for lists

**** elaborate*

WAM includes an external ASP with a generic charts (bidimensional business graphics) generator for lists, which based on the current list configuration provides further configuration options to the user:

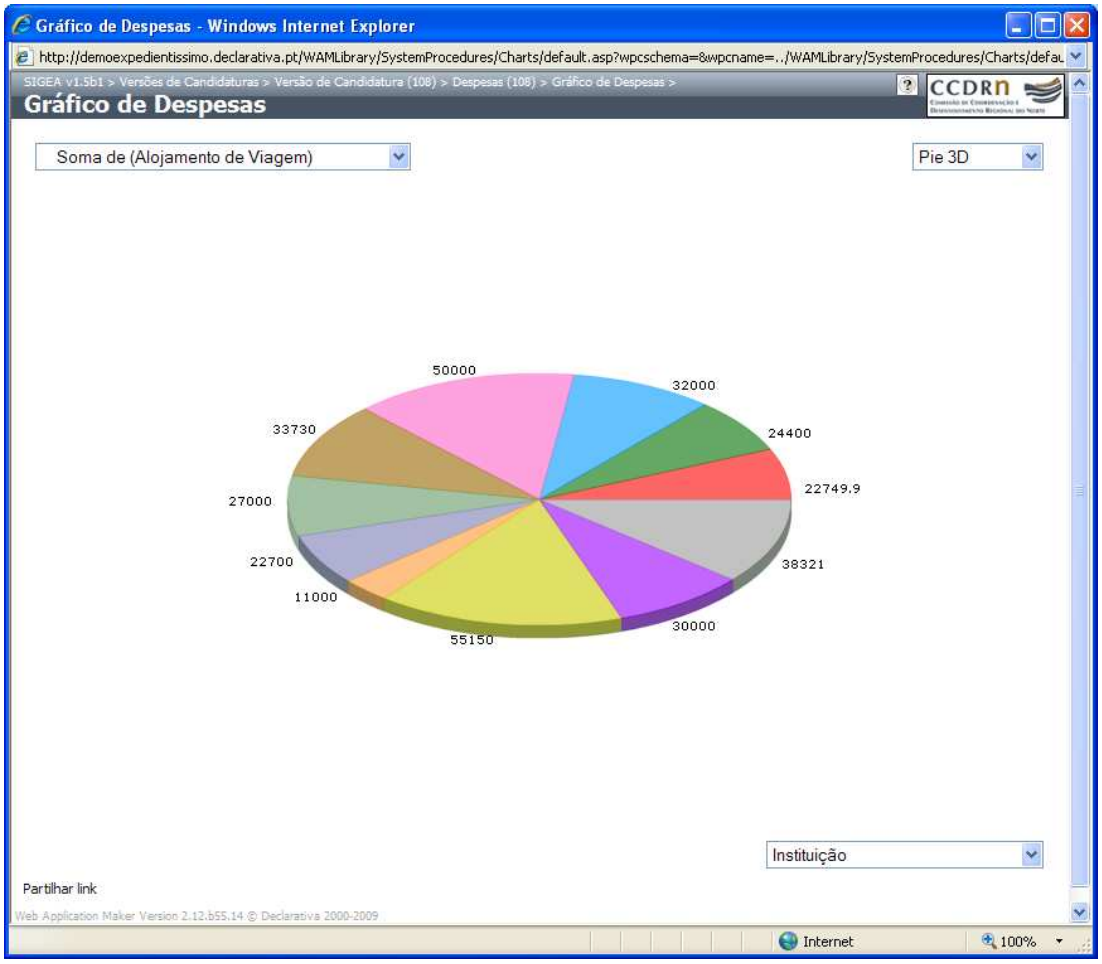


Gráfico de Despesas - Windows Internet Explorer
http://demoexpedientissimo.declarativa.pt/WAMLibrary/SystemProcedures/Charts/default.asp?wpcschema=&wpcname=../WAMLibrary/SystemProcedures/Charts/defau
SIGEA v1.5b1 > Versões de Candidaturas > Versão de Candidatura (108) > Despesas (108) > Gráfico de Despesas >
Gráfico de Despesas
CCDRN
Soma de (Alojamento de Viagem)
Pie 3D
50000
32000
24400
22749.9
38321
30000
55150
11000
22700
27000
33730
Instituição
Partilhar link
Web Application Maker Version 2.12.b55.14 © Declarativa 2000-2009
Internet
100%

The above was generated from a list "Despesas"; the bottom right combo box shows electable columns for "independent variable"; the top left combo show columns candidate for "dependent variables", including some simple aggregations as above ("Soma de" - sum of).

To use it in any list simply insert a caller definition:

INSERT INTO dbo.WAM_PROCEDURE_CALL VALUES(' ', '../WAMLibrary/SystemProcedures/Charts/default.asp', 'LIST', '<list_schema>', '<list_name>', 0, null).

9.7 References

AbsInt, "aiSee - Graph Visualization", http://www.absint.com/aisee/, May 29, 2001

Alferes, J., Leite, J., Pereira, L., Przymusinska, H., Przymusinski, T., "Dynamic Updates of Non-Monotonic Knowledge Bases", The Journal of Logic Programming 45(1-3): 43-70, 2000.

AT&T Labs-Research, "Strudel Web-site Management System", http://www.research.att.com/sw/tools/strudel/, May 29, 2001

Coram, T., Lee, J., "Experiences -- A Pattern Language for User Interface Design", http://www.maplefish.com/todd/papers/experiences/Experiences.html, May 29, 2001

Fraternali, P., "Tools and Approaches for Developing Data-Intensive Web Applications: A Survey", ACM Computing Surveys, Vol. 31, No. 3, September 1999

Fraternali et al., "Autoweb site", http://www.ing.unico.it/autoweb/, May 2001

Gamboa, T., "Towards the Development of Information Systems in Portuguese", MSc Thesis, Universidade Portucalense, Portugal, 1998

Griffiths, T., McKirdy, J., Forrester, G., Paton, N., Kennedy, J., Barclay, P., Cooper, R., Goble, C., & Gray, P., "Exploiting Model-Based Techniques for User Interfaces to Databases", in Proceedings of VDB-4, Chapman & Hall, London. pp. 21-46. 1998, http://citeseer.nj.nec.com/griffiths98exploiting.html

Lamma, E., Riguzzi, F., Pereira, L., "Strategies in Combined Learning via Logic Programs", Machine Learning 38(1/2): 63-87, January 2000

Major Micro Systems, "ASP-db, The Ultimate Web Database Tool", http://www.majormicro.com/asp-db7.html, May 29, 2001

Microsoft Corporation, "Active Server Pages", http://windows.microsoft.com/windows2000/en/server/iis/htm/asp/iiwawelc.htm, May 29, 2001

Microsoft Corporation, "Intentional Programming", http://research.microsoft.com/ip/, May 29, 2001

Patterns Home Page, http://www.hillside.net/patterns/patterns.html, May 29, 2001

Ramanathan, S., Hodges, J, "Reverse Engineering Relational Schemas to Object-Oriented Schemas", Technical Report MSU-960701, Mississipi State University, USA, 1996

Silva, P., " User Interface Declarative Models and Development Environments: A Survey", http://citeseer.nj.nec.com/359639.html, June 29, 2001

Simonyi, C., "Intentional Programming", http://research.microsoft.com/ip/mar99_files/frame.htm, May 29, 2001

U. di Roma Tre, "The ARANEUS Project", http://www.dia.uniroma3.it/Araneus/, May 29, 2001

Xerox PARC, "Aspect-Oriented Programming", http://www.parc.xerox.com/csl/projects/aop/, May 29, 2001